Institut für Informatik
Service und Software Engineering

Durchgängige Verfolgbarkeit im Vorfeld der Softwareentwicklung von E-Government-Anwendungen

Ein ontologiebasierter und modellgetriebener Ansatz am Beispiel von Bürgerdiensten

Dissertation
zur Erlangung des akademischen Grades
"doctor rerum naturalium"
(Dr. rer. nat.)
in der Wissenschaftsdisziplin Informatik

eingereicht an der
Mathematisch-Naturwissenschaftlichen Fakultät
der Universität Potsdam

von
Thomas Off

Potsdam, den 22. August 2011

Online veröffentlicht auf dem
Publikationsserver der Universität Potsdam:
URL http://opus.kobv.de/ubp/volltexte/2012/5747/
URN urn:nbn:de:kobv:517-opus-57478
http://nbn-resolving.de/urn:nbn:de:kobv:517-opus-57478

Inhaltsverzeichnis

Erklärung

Ich, Thomas Off, erkläre hiermit, dass die Arbeit bisher an keiner anderen Hochschule eingereicht worden ist sowie selbständig und ausschließlich mit den angegebenen Mitteln angefertigt wurde.

Potsdam, den 22. August 2011

_____

Thomas Off

Dank

An dieser Stelle möchte ich mich bei all jenen bedanken, die diese Arbeit unterstützt und begleitet haben. Den Anstoß zu dieser Arbeit gab Frau Prof. Dr.-Ing. habil. Erika Horn, die auch die Betreuung übernahm. An ihre wertvollen, richtungweisenden Anregungen und ihre offene Art erinnere mich gern. Mit ihrem Tod im Jahr 2009 verlor die deutsche Informatik eine wichtige Forscherin und große Persönlichkeit.

Frau Prof. Dr.-Ing. Margaria-Steffen übernahm die Betreuung der Arbeit in der schwierigen letzten Phase. Bei ihr möchte ich mich ganz besonders bedanken, denn die wunderbare Zusammenarbeit und die konstruktiven, stets lösungsorientierten Diskussionen haben maßgeblich zum fachlichen Gelingen und zur Fertigstellung dieser Arbeit beigetragen.

Die Anregungen und Hinweise von Herrn Prof. em. Dr. Klaus Lenk weiteten meinen Blick und gaben der Arbeit aus Sicht der Verwaltungsinformatik ihre Richtung. Darüber hinaus waren es diese Anregungen und Hinweise, die mir in zeitweise schwierigen Phasen des Entstehungsprozesses die notwendige Zuversicht gaben. Hierfür möchte ich mich herzlich bedanken.

Mein besonderer Dank gilt Prof. Dr. Tino Schuppan, der durch seinen konstruktiv-kritischen Blick auf die Informatik für diese Arbeit wichtige fachliche Fragen aufgeworfen hat, die mich stets vorangebracht haben.

Während der Entstehungszeit dieser Arbeit unterstützten mich zahlreiche Personen mit Diskussionen und fachlichen Hinweisen. Aus diesem Kreis möchte ich mich herzlich bei Frau Jutta Bratz bedanken, die mir mit ihren Hinweisen und Anregungen eine große Unterstützung gewesen ist.

Ohne den Rückhalt, den mir meine Familie gegeben hat, wäre diese Arbeit nicht möglich gewesen. Ihr gilt mein ganz besonderer, persönlicher Dank.

1 Einleitung

Einleitend wird der Hintergrund (Abschnitt 1.1) des Themas dieser Arbeit vorgestellt sowie die Motivation (Abschnitt 1.2) zur Auseinandersetzung mit dem Thema dargelegt. Hintergrund und Motivation münden in die Fragestellung (Abschnitt 1.3), für deren Bearbeitung eine Vorgehensweise vorgestellt wird. An der Vorgehensweise ist die weitere Gliederung der Arbeit (Abschnitt 1.4) orientiert. Durch ein fortlaufendes Beispiel wird der erarbeitete Ansatz illustriert. Der Kontext dieses Beispiels wird deshalb in der Einleitung vorgestellt (Abschnitt 1.5). Die Einleitung schließt mit Lesehinweisen zur Arbeit (Abschnitt 1.6).

1.1 Hintergrund

Die primäre Aufgabe der öffentlichen Verwaltung[1] liegt im Vollzug von Gesetzen und Rechtsvorschriften, zu deren Unterstützung sie seit den 1960er Jahren Datenverarbeitungs- und Informationstechnik einsetzt, um insbesondere Effizienz und Qualität des Verwaltungshandelns zu gewährleisten. In der Frühzeit beschränkte sich diese Technik auf die reine Datenverarbeitung, wobei nur solche Teile des Verwaltungshandelns unterstützt wurden, die durch eine hohe Aufgabendeterminiertheit gekennzeichnet waren und deshalb organisatorisch zu so genannten Wesen (z.B. Meldewesen oder Kfz-Wesen) zusammengefasst wurden.[2] Mittlerweile findet zunehmend die Durchdringung der Verwaltung mit moderner Informationstechnik statt, die mit dem Begriff Electronic Government (E-Government) bezeichnet wird. Dadurch verstärkte und verbreiterte sich der Zusammenhang zwischen den durch Gesetze und Rechtsvorschriften mehr oder weniger stark determinierten bzw. aus ihnen abgeleiteten Aufgaben einerseits und der Aufgabenunterstützung durch die Informationstechnik andererseits. Heute besteht eine „immense Abhängigkeit der öffentlichen Verwaltung“[3] von der eingesetzten Informationstechnik. Ursachen hierfür liegen zum einen in den neuen technischen Möglichkeiten moderner Informations- und Kommunikationstechnik, die nicht mehr nur Automatisierung, sondern auch intensive Unterstützung bei der Aufgabenerledigung ermöglichen. Auch der veränderte Charakter der Gesetzgebung trägt dazu bei, denn das „Gesetz regelt nicht mehr nur das grundlegend Wichtige und Wesentliche, sondern enthält auch bis ins Einzelne gehende Detailregelungen, die die Einzelfallentscheidung oftmals nicht mehr nur determinieren, sondern gewissermaßen selbst vorwegnehmen und den Gesetzesanwender zum mechanistischen Vollstrecker ohne weiteren Differenzierungsbedarf degradieren.“[4] Der Zusammenhang zwischen Gesetzen und eingesetzter Informationstechnik wirkt sich jedoch nicht auf jeden Bestandteil eines informationstechnischen Systems in gleichem Maße aus. Neben Hardware und Kommunikationsinfrastruktur ist Software in Form von Systemsoftware und Anwendungssoftware der Hauptbestandteil moderner informationstechnischer Systeme. Während die Systemsoftware universell einsetzbar ist, ist die Anwendungssoftware immer auf eine spezielle fachliche Aufgabenerledigung ausgerichtet.[5] Determinieren also Gesetze und Rechtsvorschriften die Aufgaben und die Aufgabenerledigung in der öffentlichen Verwaltung, dann bestimmen sie dadurch aus Sicht

[1] Dem Begriff „öffentliche Verwaltung“ liegt im allgemeinen Sprachgebrauch bereits ein intuitives Verständnis zugrunde, das für die einleitenden Ausführungen ausreichend ist und in Abschnitt 2.5 konkretisiert wird.
[2] [Kübler, 1987], S. 37.
[3] [Wulff, 2010], S. 43.
[4] [Burghart, 1996], S. 24.
[5] Der Begriff „Anwendungssoftware“ wird in Anlehnung an Balzert ([Balzert, 1996], S. 23) verstanden als eine Software, die Aufgaben des Anwenders lösen hilft und die Systemsoftware der zugrunde liegenden Hardware verwendet. Damit wird der Betriff der Anwendungssoftware hier weitgehend synonym zum Begriff des Anwendungssystem im engeren Sinne verwendet (vgl. Informatik-Begriffsnetz, Gesellschaft für Informatik, online: http://public.beuth-hochschule.de/~giak/, letzter Aufruf am 18.08.2011).

der Softwaretechnik vor allem die Anforderungen an die Prozesse und die Anwendungssoftware, mit denen die Aufgabenerledigung ablauforganisatorisch und informationstechnisch durchgeführt wird.

In der Informatik sind Fragestellungen nach den allgemeinen Zusammenhängen zwischen Anforderungen und Softwaresystemen unter dem Begriff der Verfolgbarkeit (engl. Traceability) bereits ausführlich erörtert worden. Entsprechende Erfahrungen und Beobachtungen aus der praktischen Anwendung der Softwaretechnik liegen vor. Insbesondere gibt es qualitative Einschätzungen, nach denen der Nutzen solcher Ansätze den Aufwand zu ihrer Einführung und Anwendung überwiegt. Allerdings stellt sich der Hauptnutzen zeitlich erst sehr viel später als die eigentliche Dokumentation der notwendigen Verfolgbarkeitsinformationen (in Wartung, späteren Iterationen oder Inkrementen des Systems) ein.[6] Verfolgbarkeit gilt als wichtige Grundvoraussetzung, um die Anwendungssoftware effizient entwickeln, betreiben und warten zu können. Von den existierenden Ansätzen werden die Elemente einer Anforderungsspezifikation verwendet und auf unterschiedliche Art bis zu ihrer Realisierung in Softwaresystemen verfolgbar gemacht. Diese Art der Verfolgbarkeit gilt in der Forschung als abschließend bearbeitet und wird in der Praxis bereits gut beherrscht. Aus Sicht der Softwaretechnik existiert aber ein bekanntes und nach wie vor nicht abschließend gelöstes Problem hinsichtlich der generellen Verfolgbarkeit im Vorfeld der Anforderungsspezifikation. Aus Sicht der Wissenschaft besteht hier Forschungsbedarf und in der Praxis wird Verfolgbarkeit im Vorfeld der Anforderungsspezifikation nur unzureichend beherrscht. Der zuvor eingeführte Zusammenhang zwischen Gesetzen und Anwendungssoftware zur Unterstützung des Vollzugs dieser Gesetze betrifft das Vorfeld der Anforderungsspezifikation. Für solche Aspekte einer Anforderungsspezifikation, die sich aus Gesetzen ableiten lassen, kann der im Rahmen dieser Arbeit neu entwickelte Ansatz das Problem der Verfolgbarkeit im Vorfeld der Anforderungsspezifikation lösen. Dadurch wird es möglich, gezielt, schnell und damit auch effizienter und qualitativ besser eine Reaktion auf Änderungen in Gesetzen und Rechtsvorschriften durch Anpassung der Anwendungssoftware zu ermöglichen. Die Verfolgbarkeit ist unter anderem Voraussetzung, um die Rechtmäßigkeit des Verwaltungshandelns auf der Basis einer softwaretechnischen Umsetzung der Aufgabenunterstützung nachvollziehen und überprüfen zu können. Im Falle einer Gesetzesänderung ist es von Vorteil, wenn die betroffenen Elemente der Anwendungssoftware durch Verfolgen des Zusammenhangs zwischen Gesetz und Anwendungssoftware systematisch abgeleitet und dann modifiziert werden können.

Darüber hinausgehend wurde in der Verwaltungspraxis und der Wirtschaft gefordert, dass bei einer beabsichtigten Gesetzesänderung die Folgen für die eingesetzte Informationstechnik Berücksichtigung finden sollten.[7] Mit dem „Gesetz zur Änderung des Gesetzes zur Einsetzung eines Nationalen Normenkontrollrates (NKRGÄndG)“ vom 16. März 2011 ist verbindlich geregelt worden, dass eine als Erfüllungsaufwand bezeichnete Auswirkungsabschätzung von Zeit und Kosten, „die durch die Befolgung einer bundesrechtlichen Vorschrift bei Bürgerinnen und Bürgern, Wirtschaft sowie der öffentlichen Verwaltung entstehen“[8], im Gesetzgebungsverfahren zu erfolgen hat. Dabei sind insbesondere solche Aufwände abzuschätzen, die für die Beschaffung, Verfügbarmachung und Übermittlung von Daten und sonstigen Informationen an Behörden und Dritte entstehen würden.[9] Aus dieser Verpflichtung zur Berücksichtigung des Erfüllungsaufwands ergibt sich, dass zukünftig auch die Auswirkungen von Gesetzesänderungen auf die Anwendungssoftware der öffentlichen Verwaltung im Vorfeld der Gesetzgebung berücksichtigt werden könnten, wenn (noch)

[6] Der Nutzen der Verfolgbarkeit stellt sich sehr viel später nach Aufzeichnung der notwendigen Verfolgbarkeits-Informationen (in Wartung, späteren Iterationen oder Inkrementen des Systems) ein (vgl. [IEEE 830, 1998], S.8; übereinstimmend [Torkar et al., 2011]).

[7] Vgl. [Wulff, 2010].

[8] Gesetz zur Einsetzung eines Nationalen Normenkontrollrates vom 14. August 2006 (BGBl. I S. 1866), das durch Artikel 1 des Gesetzes vom 16. März 2011 (BGBl. I S. 420) geändert worden ist (im Folgenden NKRG), §2 (1).

[9] Vgl. NKRG, §2 (2).

Spielraum bei der Ausgestaltung der Änderung besteht. Hierzu ist es notwendig, den Zusammenhang zwischen Gesetz und Anwendungssoftware verfolgen zu können.

Auch während der Entwicklung der Anwendungssoftware ist es für die Arbeit der beteiligten Personen und für zu treffende Entscheidungen vorteilhaft, wenn nachvollzogen werden kann, welche Elemente welche rechtlichen Anforderungen umsetzen. Aus Sicht der Verwaltungs- und Rechtswissenschaften würden sich weiterhin Möglichkeiten bieten, die Wirkungen von Gesetzesänderungen auf die für den Vollzug eingesetzte Anwendungssoftware abschätzen zu können. Dies ist insbesondere dann sinnvoll, wenn Gesetzesvorhaben inhärent abhängig vom Funktionieren komplexer Softwaresysteme sind. Ohne Kenntnis der Zusammenhänge könnten Änderungen an Details der zugrunde liegenden Gesetze versehentlich erheblichen Anpassungsaufwand (und damit verbundene Kosten) verursachen. Es wäre von Vorteil, diesen Änderungsaufwand für stark softwareabhängige Gesetzesimplementationen im Vorfeld zu ermitteln und, sofern mehrere Vorhabensalternativen mit der beabsichtigten Wirkung zur Wahl stehen, diejenige zu wählen, die mit geringerem Aufwand verbunden ist. Der Einsatz offener Standards ist notwendig, um diese Zusammenhänge flexibel, zukunfts- und investitionssicher (insbesondere unabhängig von konkreten Werkzeugimplementierungen) abzubilden.

Die Untersuchung des Zusammenhangs zwischen Elementen in Gesetzen und Rechtsvorschriften einerseits sowie den Anforderungen an Prozesse und Anwendungssoftware zu deren Vollzug andererseits hat, wie im Vorangegangenen ausgeführt, grundsätzlich Relevanz, zum einen aus Sicht der Softwaretechnik und zum anderen aus Sicht der Verwaltungs- und Rechtswissenschaften. Der Versuch, diesen Zusammenhang zu untersuchen, ist daher auch keineswegs neu. Bereits seit der Frühzeit der Informationstechnik Ende der 1960er, Anfang der 1970er Jahre wurden in loser Folge entsprechende Themenstellungen im Hinblick auf die jeweils aktuellen Techniken, Methoden und Werkzeuge (und die ihnen inhärenten Limitierungen) bearbeitet. Die Ansätze der 1960er und 1970er Jahre verfolgten beispielsweise das Ziel, aus Gesetzen und Rechtsvorschriften direkt die Programmierung eines Systems oder programmiernahe Vorgaben abzuleiten. Anstelle der erwarteten umfassenden Erfolge mündeten diese Ansätze in der Forderung nach „Automationsgerechter Gesetzgebung“[10], als ihren Vertretern bewusst wurde, dass den Gesetzen und Rechtsvorschriften inhärente Eigenschaften eine direkte Umsetzung in einen Programmcode unmöglich machen, weil der Zusammenhang nicht derart ausgeprägt ist, dass eine direkte programmtechnische Umsetzung erfolgen kann. Spätere Ansätze, wie die der *Künstlichen Intelligenz*, der *Regelbasierten Systeme* und die Repräsentation der Gesetzesinhalte in sonstigen wissensbasierten Systemen konnten die mit ihnen bei Aufkommen der Technologie verbundenen hohen Erwartungen zunächst nicht erfüllen.[11] Durch den erfolgreichen Einsatz in geeigneten Anwendungsbereichen[12] und die kontinuierliche Weiterentwicklung der Technologie in Forschungsprojekten hat sie nun einen hohen Reifegrad erreicht.[13] Heute werden in vielen Behörden der öffentlichen Verwaltung wissensbasierte Systeme eingesetzt. Darunter befinden sich auch Vertreter mit einer sehr großen Regelbasis. Ein herausragendes Beispiel ist das von der Niederländischen Einwanderungsbehörde entwickelte und eingesetzte System, dass den gesamten Verwaltungsvollzug durch formalisierte Regeln unterstützt.

Bei diesen Ansätzen steht die Abbildung der Inhalte von Gesetzestexten in technischen Softwarestrukturen oder in implementierbaren Regeln im Vordergrund. In der Softwareentwicklung von Anwendungen, auch solchen für den Vollzug von Rechtsvorschriften, hat

[10] Vgl. u.a. [Oertzen, 1970], [Oertzen, 1972], [Berg, 1968], [Fiedler, 1973], [BMI, 1973].
[11] Vgl. [Traunmüller, 2003], S. 85.
[12] Beispiele für derartige Anwendungsgebiete sieht Traunmüller „im Steuerwesen, in den Sozialversicherungen und im Sicherheitswesen“ ([Traunmüller, 2003], S. 85).
[13] Hier und im Folgenden: Freundliche Auskunft von Thomas F. Gordon, Fraunhofer-Institut für Offene Kommunikationssysteme (FOKUS) in Berlin und Prof. für Argumentationstechnologie am Institut für Informatik der Universität Potsdam (persönliches Gespräch, Berlin, 24.06.2011).

sich im Gegensatz dazu eine zusätzliche Ebene etabliert, die Gesetze mit Technik verbindet, indem sie zwischen der Softwareentwicklung und den Verwaltungswissenschaften liegt. Auf dieser Ebene findet die Anforderungsanalyse statt, die als Ergebnis fachliche Anforderungen als Bestandteil einer Anforderungsspezifikation produziert.

1.2 Motivation

Im Rahmen dieser Arbeit wurden vier relevante Entwicklungen der vergangenen Jahre vor dem Hintergrund der Verfolgbarkeit von Anforderungen identifiziert. Aufgrund dieser Entwicklungen wird hier angenommen, dass eine neuerliche Auseinandersetzung mit dem Zusammenhang von Gesetzen bzw. Rechtsvorschriften (im Folgenden kurz: Rechtsvorschriften) und den Anforderungen an Anwendungssoftware zu ihrem Vollzug zu neuen Erkenntnissen und neuen Ansätzen führen kann. Für die Formulierung einer Arbeitshypothese wird davon ausgegangen, dass die Erkenntnisse und Ansätze durch Kombination ihrer einzelnen Elemente in einen gemeinsamen Anwendungsbereich genutzt werden können (Abbildung 1).

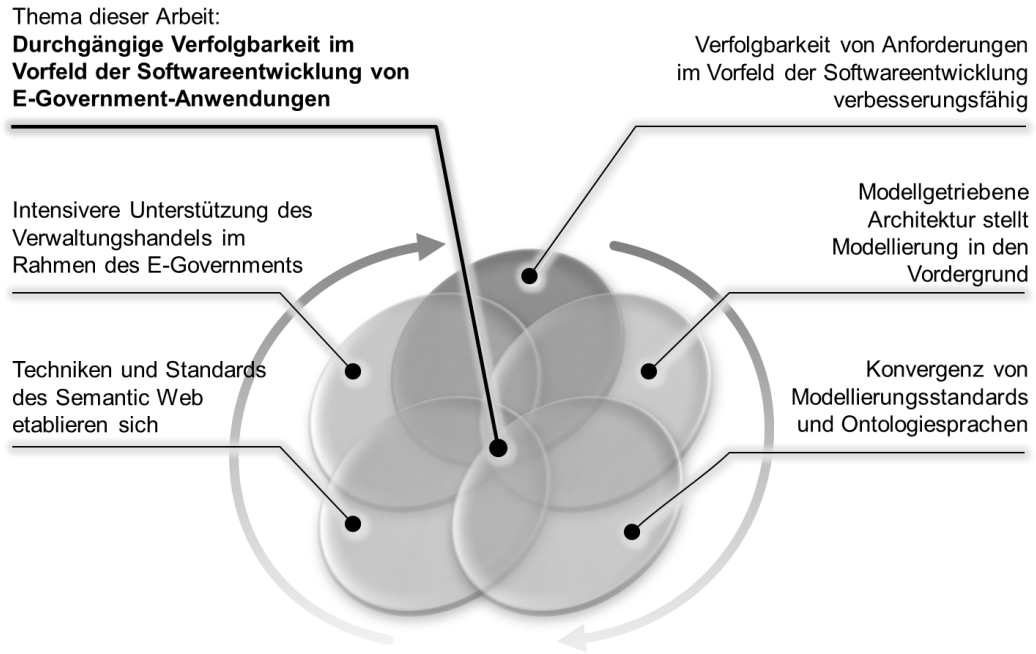


Abbildung 1 – Fokus der Arbeit als gemeinsamer Anwendungsbereich von vier aktuellen Entwicklungen

Die erste Entwicklung zeigt sich darin, dass in den vergangenen Jahren die Bedeutung von Modellen und die Bedeutung der damit verbundenen Modellierungsaktivitäten für alle Phasen der Softwareentwicklung stark zugenommen hat, so dass heute von modellgetriebener Softwareentwicklung oder Architektur gesprochen wird.[14] Dies betrifft nicht mehr nur die technische Modellierung (Datenbankschemata, technischer Entwurf), sondern vor allem auch die Darstellung, Formalisierung und Beschreibung von Anforderungen, z.B. in Form von Geschäftsprozess- und Anwendungsfallmodellen. Diese Modelle beschreiben eine auf ein bestimmtes Ziel ausgerichtete vereinfachte Sicht auf die Realität oder auf eine Vorstellung von der Realität, die sie mit Modellelementen und

[14] In dem Konzept der modellgetriebenen Architektur (Model Driven Architekture) wird diese neue Sichtweise besonders deutlich. Sie formalisiert erstmals wichtige Zusammenhänge zwischen den bereits etablierten Modelltypen und führt die schrittweise Transformation von Modellen der einzelnen Typen ineinander entsprechend definierter Transformationsregeln ein. Mit der MDA und ihrem transformationsbasierten Ansatz für Modelle steht jetzt ein formalisiertes Konzept zur Verfügung, das Potential für ein großes Einsatzspektrum hat.

Beziehungen zwischen den Elementen abbildet.[15] In der modellgetriebenen Architektur werden derartige Modelle zugrunde gelegt und die Idee von Transformationsprozessen zwischen Modellen besonders betont. In diesen Transformationsprozessen werden die Elemente von Ausgangsmodellen in Elemente von Zielmodellen gemäß definierter Regeln transformiert. Durch die Aufzeichnung von Informationen, die während der Durchführung einer solchen Transformation anfallen, können Verfolgbarkeitsbeziehungen zwischen den transformierten Elementen konstruiert werden. Für jedes Element eines Zielmodells ist dann entlang der Beziehung verfolgbar, aus welchem Element bzw. welchen Elementen des Ausgangsmodells es hervorgegangen ist. Zusätzlich zur Aufzeichnung dieser Beziehungen kann die Aufzeichnung der jeweils eingesetzten Regeln Aufschluss darüber geben, warum eine Transformation des Ausgangselements in ein oder mehrere Zielelemente in dieser Weise stattgefunden hat. Innerhalb der typischen Phasen des modellgetriebenen Softwareentwicklungsprozesses kann Verfolgbarkeit mit modernen Werkzeugen und Methoden als gegeben betrachtet werden. Diese grundlegenden Mechanismen der modellbasierten/-getriebenen Ansätze bieten für sich genommen noch keine Lösung, mit der die Verfolgbarkeit von Anforderungen zu ihren Ursprüngen und auch zu Ursprüngen in Rechtsvorschriften sichergestellt werden kann. Dies vor allem deshalb, weil sowohl der etablierte Standard in diesem Bereich, die *Model Driven Architecture (MDA[16])*, als auch andere modellgetriebene Techniken ihren Schwerpunkt auf die Softwareentwicklung im engeren Sinne, d.h. auf Analyse, Design und Implementierung legen. Andere Phasen des Lebenszyklus eines Softwaresystems, wie beispielsweise die Einführung und der Betrieb bleiben ebenso unberücksichtigt wie die frühen Phasen und das Vorfeld der Anforderungsanalyse. Deutlich wird dies auch dadurch, dass die MDA immer von der Existenz eines Modells ausgeht, das es zu transformieren gilt. Wie aber dieses initiale Modell erzeugt werden kann, wird nicht behandelt.

Die aktuell konvergierenden Modellierungsstandards im Bereich der Softwareentwicklung und die eingesetzten Standards zur Definition von Ontologien, wie sie beispielsweise seit langem im Bereich der künstlichen Intelligenz, der Wissensbasierten Systeme oder heute auch für das Semantic Web verwendet werden, bilden die dritte für diese Arbeit relevante Entwicklung. Der Ontologie-Begriff wird in diesem Kontext als eine explizite Spezifikation einer Konzeptionalisierung verwendet.[17] Unter einer Konzeptionalisierung wird die abstrakte Zusammenfassung von Typen von Dingen, Begriffen und Konzepten der realen Welt bzw. eines Ausschnittes der realen Welt unabhängig von einer konkreten Sprache oder Notation verstanden. Die explizite Spezifikation gibt dieser abstrakten Konzeptionalisierung eine konkrete Form, indem sie die Konzeptionalisierung mit (Symbolen) einer Sprache (oder auch mehrerer Sprachen) zusammen bringt. Die Ontologie nutzt diese Sprache zur Benennung von Begriffen/Konzepten und zur Beschreibung ihrer Bedeutung(en). Das Ziel der Ontologie ist es, die Zahl der möglichen Bedeutungen für ein sprachliches Symbol idealer Weise auf genau eine zu reduzieren. Voraussetzung ist, dass zur Beschreibung nur solche Elemente genutzt werden, die zuvor ausdrücklich definiert wurden. Lange Zeit wurden hierfür Sprachen aus dem Forschungsbereich der Künstlichen Intelligenz (z.B. ONTOLINGUA, LOOM, OCML, FLogic) verwendet. Für den Einsatz im *World Wide Web* des Internet (WWW) hat das für diesen Bereich zuständige *World Wide Web Consortium (W3C)[18]* spezielle Ontologie Markup-Sprachen (z.B. RDF/RDFS, OWL) standardisiert. Die bei der Standardisierung der Modellierung in der Softwaretechnik führende *Object Management*

[15] Im Fall von Geschäftsprozessmodellen werden beispielsweise Prozessaktivitäten als Modellelemente verwendet; die Anwendungsfallmodelle verwenden Anwendungsfälle und Akteure als Modellelemente.
[16] Vgl. [OMG MDA, 2000], [OMG MDA, 2001], [OMG MDA, 2003a], [OMG MDA, 2003b].
[17] Vgl. hier und im Folgenden [Gruber, 1993].
[18] Das W3C ist eine internationale Community, deren Mitgliedsorganisationen und Mitarbeiter gemeinsam mit der Öffentlichkeit an Standards für das World Wide Web arbeiten (vgl. http://www.w3.org/Consortium/, letzter Aufruf: 21.08.2011). Die W3C-Standards werden, basierend auf breitem Konsens, in Form technischer Spezifikationen oder Empfehlungen veröffentlicht (vgl. http://www.w3.org/standards/, letzter Aufruf: 21.08.2011).

Group (OMG)[19] hat das *Ontology Definition Metamodel (ODM)* standardisiert, so dass Modellierungsstandards (z.B. UML, MOF) einerseits durch definierte Abbildungsregeln auf RDFS und OWL im bisherigen Einsatzbereich der Ontologien für die Wissensrepräsentation, konzeptionelle Modellierung und für die Entwicklung formaler Taxonomien anwendbar werden. Anderseits können die Abbildungsregeln auch verwendet werden, um mit Ontologien formalisierte Konzepte für den Bereich der Modellierung zugänglich zu machen.

Mit der Entwicklung des *Semantic Web* ist die Erwartung verbunden, dass durch den Einsatz von Ontologien innerhalb existierender und neu zu schaffender Inhalte des WWW die maschinelle Verarbeitung dieser Inhalte erleichtert wird. Die zuvor eingeführten Ontologie Markup-Sprachen werden verwendet, um einzelne Elemente dieser Inhalte mit einer maschinell verarbeitbaren Semantik zu versehen. Rechtsvorschriften können in diesem Sinne auch als (potenzielle) Inhalte des WWW angesehen werden. Bereits heute liegen viele Rechtsvorschriften im Internet öffentlich verfügbar vor. Für die Sachbearbeitung innerhalb von Verwaltungsträgern löst die Bereitstellung von Rechtstexten und Vorschriften im verwaltungsinternen Intranet, das auf den Techniken des WWW basiert, die verbreiteten Loseblatt-Sammlungen ab. Dadurch können die Techniken und Methoden des Semantic Web sowohl auf öffentlich zugängliche Rechtsvorschriften als auch auf verwaltungsinterne Vorschriften und Regelungen angewendet werden. Um sie für die maschinelle Verarbeitung zu erschließen, muss zusätzlich eine geeignete Ontologie zugrunde gelegt werden, die es ermöglicht, innerhalb der Texte die Elemente mit einer Semantik zu versehen, die eine maschinelle Verarbeitung ermöglicht.

Heute ist das ursprüngliche Leitbild der Automatisierung von Verwaltungshandeln, das die frühen Ansätze prägte, nicht mehr adäquat. Vielmehr steht aktuell die Frage nach einer intensiven Unterstützung des Verwaltungshandelns im Sinne des *E-Governments* im Vordergrund.[20] Mit E-Government wird die „Durchführung von Prozessen der öffentlichen Willensbildung, der Entscheidung und der Leistungserstellung in Politik, Staat und Verwaltung unter sehr intensiver Nutzung der Informationstechnik“[21] bezeichnet. Durch diese Definition wird besonders deutlich, dass sowohl unterstützende Anwendungssysteme als auch die zu unterstützenden Prozesse der Verwaltung eine integrale Einheit bilden, womit die Informationstechnik noch intensiver als bisher auf die Aufgabenerledigung wirkt. Dieser Zusammenhang darf aber weder „ausschließlich mit der Jura- oder Informatik-Brille“[22] gesehen werden. Zum einen ist die Ableitung der Programmierung eines Systems oder programmiernaher Vorgaben aus Gesetzestexten mit dem Zweck der Automatisierung des Gesetzesvollzugs offensichtlich nicht mehr angemessen. Die frühere Ansätze dominierende Suche nach Übereinstimmung von Recht und deren Umsetzung in Anwendungssoftware darf „in ihrer Tragweite nicht überschätzt werden“[23]. Insbesondere der fehlende Detaillierungsgrad von Rechtsvorschriften, die fehlende Darstellung der sachlichen bzw. zeitlichen Dimensionen, die Fokussierung auf das Endergebnis und die gleichzeitige Vernachlässigung des Weges dorthin sowie die fehlende Berechenbarkeit erweisen sich als Probleme.[24] Zum anderen erschöpft sich Verwaltungshandeln nicht in der rein juristischen Prüfung der Rechtmäßigkeit einer Rechtsfolge anhand eines Vorschriftentextes. Beispielsweise wählt die Verwaltung von mehreren möglichen und rechtlich zulässigen Rechtsfolgen diejenige, die den größten Beitrag zur Erreichung gesellschaftlicher oder politischer Ziele liefert. Durch das E-Government rücken derartige Unterschiede zwischen Verwaltungshandeln und Rechtsanwendung in den Gestaltungsspielraum der Informations-

[19] Die OMG ist ein Zusammenschluss zahlreicher Unternehmen und Institutionen. Ihre Aufgabe ist u. a. die Definition von Modellierungsstandards. Die Homepage der OMG ist unter der folgenden Adresse zu erreichen: http://www.omg.org/ (letzter Aufruf: 21.08.2011).
[20] Vgl. [GI&ITG, 2000].
[21] [GI&ITG, 2000], S. 3.
[22] [Schuppan, 2011], S. 19.
[23] Vgl. [Luhmann, 1997], S. 45.
[24] Vgl. [Luhmann, 1997], S. 49 ff.

und Kommunikationstechnik. Doch auch die unreflektierte Anwendung dieser Technik im E-Government kann die gewünschte Wirkung verfehlen, wenn beispielsweise Informatiker durch ihr Tun Dinge in einer Form entstehen lassen[25], „die aus verwaltungspolitischer Sicht kaum wünschenswert sind."[26] Entsprechend müssen innerhalb des Zusammenhangs zwischen Elementen in Rechtsvorschriften einerseits und den Anforderungen an Prozesse und Anwendungssoftware eines E-Government-Systems andererseits die Gestaltungsspielräume der Verwaltungswissenschaften berücksichtigt werden. Das kann durch den Einsatz semantischer Technologien erreicht werden, die eine stärkere Einsicht in die fachlichen Zusammenhänge des Einsatzbereiches bringen.[27]

1.3 Fragestellung und Vorgehensweise

Vor dem dargestellten Hintergrund und ausgehend von den aufgezeigten Entwicklungen lautet die Fragestellung dieser Arbeit:

> **Wie kann im Vorfeld der Softwareentwicklung die Verfolgbarkeit der Ursprünge von Anforderungen in Rechtsvorschriften an E-Government-Anwendungen verbessert werden?**

Die zuvor dargestellten Entwicklungen bieten jeweils für sich genommen keine Lösung, mit der die Verfolgbarkeit von Anforderungen zu ihren Ursprüngen und auch nicht zu Ursprüngen in Rechtsvorschriften sichergestellt werden kann. Ein möglicher Ansatz muss Folgendes bieten:

1. Er muss unmittelbar auf die Texte von Rechtsvorschriften anwendbar sein, ohne dass von der Existenz eines initialen Modells ausgegangen wird.
2. Er muss in der Auseinandersetzung mit dem Text ein Ergebnis produzieren, das Verfolgbarkeit zu den Textelementen sicherstellt und als Ausgangsbasis für die modellgetriebene Softwareentwicklung verwendet werden kann.
3. Er muss das Einfließen verwaltungsspezifischer Aspekte ermöglichen, die nicht oder nicht vollständig in den Texten von Rechtsvorschriften berücksichtigt sind, um die Voraussetzung dafür zu schaffen, dass statt Automatisierung des Vollzugs die intensive Unterstützung im Sinne des E-Governments ermöglicht wird.
4. Er muss den nahtlosen Übergang zum standardisierten modellgetriebenen Softwareentwicklungsprozess bieten, um die durchgängige Verfolgbarkeit von den Ursprüngen einer Anforderung bis zum ausführbaren Anwendungssystem zu ermöglichen.
5. Er muss mit offenen Standards implementiert werden, um unabhängig von proprietären[28] Implementierungen flexibel einsetzbar zu sein.

Im Rahmen dieser Arbeit wird von der Existenz eines neuartigen Ansatzes ausgegangen, der auf der Kombination ausgewählter Elemente der zuvor vorgestellten Entwicklungen basiert und den Bedingungen genügt. Deshalb wird vor diesem Hintergrund folgende Arbeitshypothese formuliert:

---

[25] Vgl. [Schuppan, 2011], S. 19.
[26] [Schuppan, 2011], S. 19.
[27] Vgl. [Vitvar et al., 2010], S. 4.
[28] Das Wort wird im Rahmen dieser Arbeit in seiner allgemeinen, weit verbreiteten Bedeutung für die Informations- und Kommunikationstechnik benutzt. Es bezeichnet hier solche Technologien, Standards und Software, die abhängig von konkreten Herstellern sind und von Außenstehenden nicht oder nicht vollständig nachvollzogen werden können. Das Wort proprietär soll insbesondere den Gegensatz zu offenen Standards (z.B. Modellierungsnotationen/-methoden, Dateiformate, Kommunikationsprotokolle) und zu freier, quelloffener Software (Open Source) ausdrücken.

Werden die Techniken des Semantic Web auf die Texte von Rechtsvorschriften angewendet, können Ursprünge von Anforderungen, basierend auf einer geeigneten Ontologie, mit einer definierten Semantik versehen werden, so dass unter Berücksichtigung verwaltungsspezifischer Aspekte und unter Nutzung der Abbildungsregeln des ODM-Standards der Anschluss an die MDA-basierte, modellgetriebene Softwareentwicklung von E-Government-Anwendungen gefunden wird, der die Aufzeichnung von Verfolgbarkeitsinformationen und deren Nutzung im Sinne der Fragestellung, basierend auf offenen Standards, ermöglicht.

Es gibt zum aktuellen Zeitpunkt keinen Ansatz, der den zuvor definierten Anforderungen genügt und für die Verfolgbarkeit von Elementen in Rechtsvorschriften und den daraus resultierenden Anforderungen an Prozesse und Anwendungssoftware im Rahmen des E-Government eingesetzt werden kann. Ziel dieser Arbeit ist die Entwicklung eines Ansatzes, der die Arbeitshypothese belegt und die an ihn gestellten Anforderungen erfüllt. Zu diesem Zweck wird in den folgenden Schritten vorgegangen:

In einem ersten Schritt werden die aufgezeigten Entwicklungen detailliert betrachtet, um eine Einordnung in das Thema dieser Arbeit zu ermöglichen und die relevanten Aspekte und Elemente dieser Ansätze zu identifizieren. Der Schritt endet mit einer Bestandsaufnahme, die die relevanten Elemente der einzelnen Entwicklungen aufzeigt, Lücken identifiziert und einen möglichen Ansatz skizziert. Ohne dem Ergebnis dieser Untersuchung vorzugreifen, kann an dieser Stelle auf die folgenden drei Aspekte hingewiesen werden, die in der Arbeitshypothese zum Tragen kommen und eine intensive Auseinandersetzung erfordern.

Es wird eine Ontologie entwickelt, die mit den Techniken, Methoden und Werkzeugen des Semantic Web eingesetzt wird, um in Texten von Rechtsvorschriften relevante Ursprünge von Anforderungen durch Annotationen mit einer definierten Semantik zu versehen. Dann wird eine Erweiterung der MDA vorgeschlagen, die unter Nutzung des ODM eine Lösung für den Übergang von annotierten Rechtsvorschriften zu MDA-konformen Modellen darstellt. Die exemplarische Verwendung dieser Erweiterung in Form eines Modells für die Modellierung der Ursprünge von Anforderungen und deren Ergänzung um verwaltungsspezifische Besonderheiten wird dann gezeigt. Darauf aufbauend wird der Übergang vom initialen Modell zum Modell einer Anforderungsspezifikation durch MDA-konforme Transformation definiert und dabei insbesondere die Lücke geschlossen, die aus der unterschiedlichen Semantik beider Modelle resultiert. Abschließend werden die bei der Durchführung der Transformation aufgezeichneten Verfolgbarkeitsinformationen zu einem Gesamtansatz kombiniert und die sich daraus ergebenden Möglichkeiten für die Verfolgbarkeit vorgestellt.

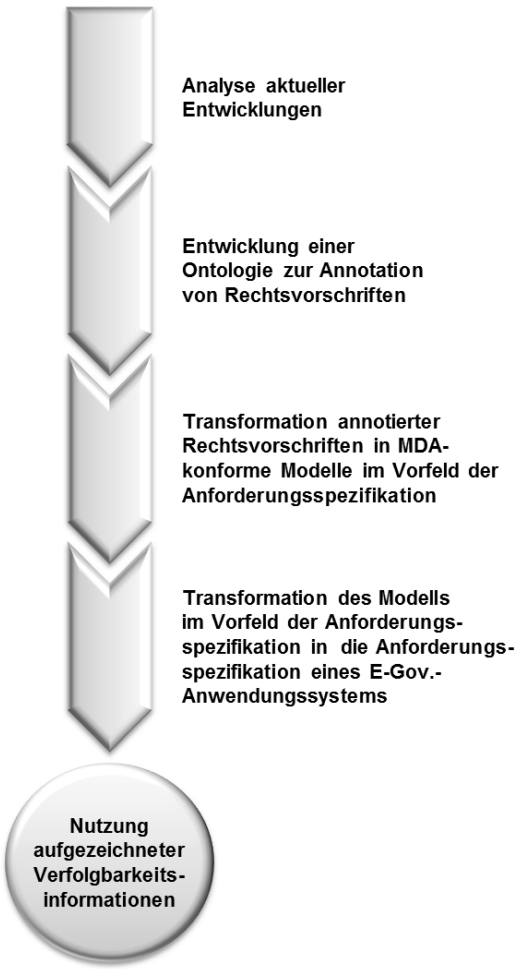


Abbildung 2 – Illustration der Vorgehensweise

Es ist nicht möglich, alle für die Anforderungsanalyse notwendigen Informationen aus Rechtsvorschriften abzuleiten. Es wird zum einen immer berechtigte Anforderungen an ein E-Government-Anwendungssystem geben, die sich nicht auf diese Elemente zurückführen lassen. Umgekehrt wird es auch immer Elemente in Rechtsvorschriften geben, die sich nicht in Anforderungen niederschlagen. Deshalb wird explizit zwischen Ursprüngen von Anforderungen in Rechtsvorschriften und fixierten Anforderungen unterschieden.

1.4 Gliederung der Arbeit

Das Kapitel 1 umfasst die Einleitung der Arbeit, die in den Hintergrund einführt, die Motivation darlegt, anschließend die Fragestellung ableitet und die Vorgehensweise dieser Arbeit skizziert. In der Einleitung werden auch das fortlaufende Beispiel vorstellt und wichtige Lesehinweise gegeben.

An die Einleitung schließt sich das Kapitel 2 an, in dem die von diesem Ansatz genutzten grundlegenden Entwicklungen vorgestellt werden. Es werden die relevanten Forschungsarbeiten und Ansätze dargestellt und die Fragestellung dieser Arbeit eingeordnet. Im Ergebnis des Kapitels zeigt die Bestandsaufnahme einen möglichen Lösungsansatz, der weitergehend präzisiert und abgegrenzt wird. Dieser Lösungsansatz basiert auf der Annotation von Rechtsvorschriften unter Verwendung einer geeigneten Ontologie.

Der folgende Kapitel 3 stellt daher die im Rahmen dieser Arbeit entwickelte Ontologie des Verwaltungshandelns vor. Zunächst werden die relevanten Konzepte der Ontologie innerhalb des Anwendungsbereichs identifiziert. Sie bilden die Konzeptionalisierung der Ontologie. Anschließend erfolgt die werkzeugunterstützte Formalisierung dieser Konzepte in einer Ontologie-Markup Sprache. Dann werden die Einsatzmöglichkeiten der Ontologie zur Annotation von Rechtsvorschriften am Beispiel vorgestellt. Dabei wird insbesondere die Verfolgbarkeit des Zusammenhangs von Annotation und annotierter Textpassage untersucht. Um die Annotationen für die Modellierung verwenden zu können, werden im Rahmen dieser Arbeit entsprechende Transformationen und eine Erweiterung der MDA entwickelt.

Im Kapitel 4 wird zunächst die MDA-Erweiterung eingeführt. Dann werden sowohl für die Ontologie als auch für die Annotationen der Rechtsvorschriften die Transformationen vorgestellt, mit denen die Annotationen in entsprechende Modelle der MDA überführt werden können. Für die so erstellten Modelle wird gezeigt, wie sie im Vorfeld einer Anforderungsspezifikation benutzt werden können, um Besonderheiten des Verwaltungshandelns in der Modellierung zu berücksichtigen. Das Kapitel schließt mit einer Einschätzung zur Verfolgbarkeit der durchgeführten Transformation und Modellierung.

Kapitel 5 stellt die im Rahmen dieser Arbeit entwickelten Transformationen vor, mit denen die Modelle im Vorfeld der Anforderungsspezifikation in informationstechnikunabhängige Modelle einer Anforderungsspezifikation für E-Government-Anwendungen überführt werden können. Dazu ist der Einsatz eines zusätzlichen Referenzmodells notwendig, dessen Einsatz ebenfalls beschrieben wird. Im Ergebnis des Kapitels werden die vorliegenden Verfolgbarkeitsinformationen zusammengefasst und eine Einschätzung zur Verfolgbarkeit der Transformation und Modellierung gegeben.

Die zunächst nur bezogen auf einzelne Transformationen, Modellierungs- und Annotationsaktivitäten erarbeitete Einschätzung zur Verfolgbarkeit wird in Kapitel 6 zu einem Gesamtansatz kombiniert. Seine Bestandteile, technische Umsetzung und Möglichkeiten werden beschrieben.

Die Arbeit schließt in Kapitel 7 mit der Prüfung der zugrunde liegende Arbeitshypothese anhand der erzielten Ergebnisse. Basierend auf dem Prüfergebnis wird zusammenfassend eine Antwort auf die Fragestellung dieser Arbeit gegeben.

Im Anhang der Arbeit sind die Code-Darstellungen (Listings) vollständig dargestellt sowie das Abbildungs-, Tabellen- und Literaturverzeichnis enthalten.

1.5 Fortlaufendes Beispiel

Die Arbeit wird durch ein fortlaufendes Beispiel aus dem Bereich *Straßenverkehr* begleitet, auf das in den späteren Abschnitten Bezug genommen wird. Es dient der Illustration der vorgestellten Inhalte und wird entlang der Struktur dieser Arbeit weiterentwickelt. Aus dem Bereich Straßenverkehr wurde das *Bewohnerparken* (vormals Anwohnerparken) für das fortlaufende Beispiel gewählt, da es einen abgrenzbaren Regelungsbereich bildet, keine umfangreiche Einführung in weitere Zusammenhänge und Hintergründe erfordert, intuitiv nachvollziehbar ist und die Regelungen wenig komplex sind, aber gleichzeitig alle Elemente umfassen, die zur Illustration des in dieser Arbeit entwickelten Ansatzes benötigt werden.

Mit Regelungen zum Bewohnerparken soll es den Bewohnern städtischer Quartiere mit erheblichem Parkraummangel ermöglicht werden, in fußläufiger Entfernung zur Wohnung ihr Fahrzeug abzustellen. Das Bewohnerparken umfasst sowohl die Kennzeichnung von Parkmöglichkeiten, als auch die Erteilung von Berechtigungen für das Abstellen eines Fahrzeugs in den gekennzeichneten Parkflächen durch die Straßenverkehrsbehörden.

Abbildung 3 – Beispiel eines Bewohnerparkausweises und der Kennzeichnung von Parkflächen für Bewohner

Die Berechtigung wird dem Bürger nach erfolgreicher Prüfung aller Bedingungen in Form eines Bescheides mitgeteilt. Gleichzeitig wird ihm ein Bewohnerparkausweis ausgehändigt, der gut sichtbar im Fahrzeug ausgelegt werden muss. Grundlage für dieses Verwaltungshandeln ist im Wesentlichen die „Allgemeine Verwaltungsvorschrift zur Straßenverkehrs-Ordnung (VwV-StVO)“[29], in der es heißt:

> „Allgemeine Verwaltungsvorschrift zur Straßenverkehrs-Ordnung (VwV-StVO)“
>
> […] Zu § 45:
>
> […] Bewohnerparkausweise werden auf Antrag ausgegeben. Einen Anspruch auf Erteilung hat, wer in dem Bereich meldebehördlich registriert ist und dort tatsächlich wohnt. Je nach örtlichen Verhältnissen kann die angemeldete Nebenwohnung ausreichen. Die Entscheidung darüber trifft die Straßenverkehrsbehörde ebenfalls im Einvernehmen mit der Stadt. Jeder Bewohner erhält nur einen Parkausweis für ein auf ihn als Halter zugelassenes oder nachweislich von ihm dauerhaft genutztes Kraftfahrzeug. Nur in begründeten Einzelfällen können mehrere Kennzeichen in den Parkausweis eingetragen oder der Eintrag "wechselnde Fahrzeuge" vorgenommen werden. Ist der Bewohner Mitglied einer Car-Sharing-Organisation, wird deren Name im Kennzeichenfeld des Parkausweises eingetragen. Das Bewohnerparkvorrecht gilt dann nur für das Parken eines von

[29] Allgemeine Verwaltungsvorschrift zur Straßenverkehrs-Ordnung (VwV-StVO) Vom 22. Oktober 1998 In der Fassung vom 17. Juli 2009 (im Folgenden kurz: VwV-StVO).

außen deutlich erkennbaren Fahrzeugs dieser Organisation (Aufschrift, Aufkleber am Fahrzeug); darauf ist der Antragsteller schriftlich hinzuweisen. […]“

Beispiel 1 - Relevanter Auszug aus der VwV-StVO

Die vorliegende Arbeit beschränkt sich in ihrem Beispiel auf Verwaltungsleistungen, die unmittelbar für Bürger in einem konkreten Einzelfall erbracht werden. Daher wird hier lediglich die Erteilung von Berechtigungen betrachtet und andere Aspekte außer Acht gelassen (z.B. die Kennzeichnung von Parkflächen durch entsprechende Schilder).

1.6 Lesehinweise

Diese Arbeit verwendet als Standardschriftart für den Fließtext Garamond. Wird ein feststehender Begriff oder Name erstmalig neu eingeführt, ist er durch *Kursivschrift* hervorgehoben. Wenn innerhalb des Textes Begriffe einer Programmiersprache, Programmbefehle usw. verwendet werden, so sind diese als `Courier New` formatiert. Umfangreiche Code-Beispiele werden als eigene Absätze grau unterlegt und mit einer einfachen Linie am linken Seitenrand dargestellt. Besteht die Notwendigkeit sie durch Auslassungen zu kürzen, so enthalten die Zeilen „…“, das auch innerhalb eines Kommentars eingefasst sein kann (z.B. im XML-Code als `<!-- ... -->` bzw. im QVT-Code als `-- ...`).

Für Bezeichner von Klassen, Attributen, Variablen in einer Programmiersprache oder in einem Modell wird die als „camelCase“ bzw. „PascalCase“ bekannte Schreibweise genutzt, um auch längere Bezeichner ohne Leerzeichen übersichtlich darstellen zu können. In Bezeichnern wird auf die Verwendung von Umlauten aus Kompatibiltätsgründen verzichtet (z. B. wird ö als oe, Ä als Ae und ß als ss dargestellt).

Das fortlaufende Beispiel ist grau unterlegt und kann durch die doppelte Linie am linken Seitenrand von den Code-Beispielen unterschieden werden. Um sehr große Modelle oder Screenshots (Bildschirmfotos) in Abbildungen darzustellen, werden diese in ihrer Ausrichtung dem Seitenlayout angepasst. Sind nur Ausschnitte des Modells oder Screenshots relevant, so wird der entsprechende Ausschnitt in der Abbildung mit einem Schatten unterlegt.

2 Grundlagen und Einordnung

Der Abschnitt beginnt mit einer Fundierung des bisher informell verwendeten Begriffs der Verfolgbarkeit. Um die zuvor formulierte Arbeitshypothese durch Entwicklung eines entsprechenden Ansatzes bestätigen zu können, werden dann die Forschungsarbeiten und Ansätze zur modellgetriebenen Architektur (Model Driven Architecture), zu Ontologien und ihrer Konvergenz mit etablierten Modellierungsstandards, zum Semantic Web sowie zum E-Government untersucht. Die erarbeiteten Erkenntnisse werden in Form einer Bestandsaufnahme zusammengefasst. Die Bestandsaufnahme dient einerseits dazu, den Lösungsansatz dieser Arbeit zu konkretisieren. Andererseits wird vor dem Hintergrund des konkretisierten Lösungsansatzes die Abgrenzung zu verwandten Ansätzen vorgenommen.

2.1 Verfolgbarkeit in der Softwareentwicklung

Den Ausgangspunkt jeder ingenieursmäßigen, professionellen Softwareentwicklung – auch der für die öffentliche Verwaltung – bildet die Ermittlung von Anforderungen an das zu erstellende oder weiterzuentwickelnde Softwaresystem. Die Forschungsrichtung *Requirements Traceability* (Verfolgbarkeit von Anforderungen) hat in der Informatik u.a. die Untersuchung und Erklärung des Zusammenhangs zwischen fachlichen Anforderungen und Bestandteilen eines Softwaresystems zum Gegenstand.

Unter einer Anforderung wird allgemein eine Bedingung verstanden, die ein System erfüllen oder eine Fähigkeit bzw. ein Leistungsmerkmal, das ein System aufweisen muss. Anforderungen können aus verschiedenen Quellen stammen, unterschiedliche Struktur und unterschiedlichen Detaillierungsgrad besitzen. In der Praxis zeigt sich, dass Anforderungen beispielsweise aus Ideen und Meinungen der Auftraggeber und sonstigen Beteiligten eines Softwareentwicklungsprojektes, aus abstrakten Zielen, aus vorhandenen oder neu gestalteten Arbeitsabläufen der Organisation oder aus Leistungsmerkmalen eines abzulösenden Altsystems stammen können. Anforderungen werden im Rahmen der ingenieursmäßigen Softwareentwicklung in einer Anforderungsspezifikation dokumentiert. Sie ist eine Dokumentation aller Anforderungen, die ein System verständlich, korrekt, eindeutig, vollständig, widerspruchsfrei, umsetzbar und verifizierbar beschreiben.[30] Es werden generell zwei Arten von Anforderungen unterschieden: funktionale Anforderungen und nicht-funktionale Anforderungen. Funktionale Anforderungen beziehen sich auf das Verhalten oder eine Funktion des Systems und beschreiben, was das System leisten soll. Nicht-funktionale Anforderungen beziehen sich in der Regel auf Eigenschaften, die wesentliche Teile des Systems oder das System als Ganzes während des Betriebs oder über den Lebenszyklus aufweisen sollen. Anforderungen an die Usability, die Sicherheit, den Schutz, die Performance, die Skalierbarkeit, die Portabilität und die Zielplattform sind Beispiele für nicht-funktionale Anforderungen.

Es gibt für unterschiedliche Einsatzgebiete von Anwendungssoftware auch unterschiedliche Formen der Anforderungsspezifikation. Im Bereich der wirtschafts- und verwaltungsorientierten Anwendungen haben sich Anforderungsspezifikationen auf Basis von Modellen, hier speziell auf Basis von Prozess-, Anwendungsfall- und Klassendiagrammen (siehe unten, Seite 24) durchgesetzt, die um weitere Informationen (z.B. als Beschreibung in Form von Fließtext oder Tabellen oder spezielle Diagramme) ergänzt werden. Diese Modelle beschreiben primär funktionale Anforderungen. Er ist aber auch möglich, nicht-funktionale Anforderungen in diese Modelle zu integrieren. Darüber hinaus ist vor allem die Erstellung einer separaten Dokumentation in Form von strukturiertem Fließtext verbreitet.

Aufbauend auf der Anforderungsspezifikation wird das System entwickelt bzw. weiterentwickelt, so dass es den aus den Anforderungen resultierenden Bedingungen genügt und die durch sie geforderten Fähigkeiten bzw. Leistungsmerkmale aufweist. Während des

[30] Vgl. [IEEE 830, 1998], S. 4.

Entwicklungsprozesses werden die Anforderungen zu diesem Zweck schrittweise verfeinert und konkretisiert. Aus ihnen werden neue Elemente abgeleitet, die dann direkt oder indirekt die geforderten Fähigkeiten und Bedingungen sicherstellen. Auch diese Elemente werden verfeinert und konkretisiert, bis ein ausführbares Softwaresystem vorliegt.

Nicht nur die Anforderungen, sondern auch die aus ihnen abgeleiteten „neuen" Elemente werden dokumentiert. So entstehen Arbeitsergebnisse (z.B. Modelle, Textdokumente, Grafiken), die sich in Darstellungsform, in Struktur und im Abstraktionsniveau unterscheiden, aber in der Regel inhaltlich aufeinander aufbauen. Es ergibt sich naturgemäß ein Zusammenhang zwischen diesen aufeinander aufbauenden und inhaltlich zusammenhängenden Elementen der einzelnen Arbeitsergebnisse (z.B. Anforderungen, Arbeitsergebnissen des Entwicklungsprozesses und Bestandteilen des Softwaresystems).[31]

Bezogen auf den Zusammenhang zwischen den Elementen einer Anforderungsspezifikation und anderen Arbeitsergebnissen wird der Begriff der Verfolgbarkeit verwendet und im Rahmen dieser Arbeit zunächst definiert als die Möglichkeit, den Lebenszyklus einer Anforderung beschreiben und verfolgen zu können. Diese Definition verwendet den Begriff des Lebenszyklus um deutlich zu machen, dass Anforderungen bestimmte Entwicklungsschritte durchlaufen. Unabhängig von der konkreten Ausprägung der Entwicklungsschritte kann die Verfolgbarkeit in vertikale und horizontale Verfolgbarkeit unterschieden werden, die hier wie folgt verwendet werden:[32]

- Die vertikale Verfolgbarkeit beschreibt die Beziehungen, die Elemente der gleichen Phase bzw. Entwicklungstätigkeit haben können. Derartige Beziehungen können durch Verfeinerung oder Präzisierung einer Anforderung entstehen. Die vertikale Verfolgbarkeit umfasst aber insbesondere auch die (zeitlichen) Beziehungen, die im Sinne von Versionen eines Elements existieren.[33] Diese zeitlichen Beziehungen werden von Brcina als Spezialisierung der vertikalen Verfolgbarkeit unter dem Begriff evolutionäre Verfolgbarkeit zusammengefasst.[34] Für das Thema dieser Arbeit ist diese weitergehende Unterscheidung jedoch nicht notwendig.

- Die horizontale Verfolgbarkeit umfasst die Beziehungen, die zwischen Elementen unterschiedlicher Phasen bzw. Entwicklungstätigkeiten bestehen. Sie ermöglicht beispielsweise die Umsetzung einer Anforderung in der Design-Phase und im implementierten System verfolgen zu können.

Horizontale Verfolgbarkeit kann weiterhin hinsichtlich der Richtung, die durch die Verfolgbarkeit beschrieben wird, in zwei Arten unterschieden werden:[35]

- Vorwärtsgerichtete Verfolgbarkeit (Forward Traceability) umfasst die Beziehungen, die sich aus dem Ablauf von Entwicklungstätigkeiten und Phasen ergeben und von einem Element zu den Elementen führen, die aus diesem hervorgegangen sind. Beispiel hierfür sind die Beziehungen, die ein Anwendungsfall (Use Case) zu Klassen hat, die aufgrund der Analyse der zugehörigen Anwendungsfallbeschreibung aus ihm hervorgegangen sind.

- Rückwärtsgerichtete Verfolgbarkeit (Backward Traceability) umfasst alle Beziehungen, die von einem Element ausgehen und zurück zu allen Elementen führen, aus denen es hervorgegangen ist bzw. auf denen es aufbaut. Ein Beispiel für rückwärtsgerichtete Verfolgbarkeit wäre die Beziehung einer Klasse im Programmcode zu der Design-Klasse, aus der sie hervorgegangen ist.

---

[31] Vgl. u.a. [Pinheiro, 1996], S. 4 f., übereinstimmend [Jacobson et al., 1999], S. 10.

[32] In der Literatur besteht Einigkeit über diese Formen der Verfolgbarkeit, allerdings werden ihre Bezeichnungen nicht einheitlich verwendet, weil die zur Orientierung verwendeten Achsen in den jeweiligen Darstellungen wechseln (vgl. [Brcina, 2006], S. 4).

[33] Vgl. [Winter, 1999], S. 148.

[34] Vgl. [Brcina, 2006], S. 4.

[35] Vgl. [IEEE 830, 1998], S. 8

Eine andere Unterscheidung in funktionale und nicht-funktionale Verfolgbarkeit führt Pinheiro ein.[36] Während sich funktionale Verfolgbarkeit auf das Verfolgen funktionaler Anforderungen bezieht, deckt die nicht-funktionale Verfolgbarkeit seiner Ansicht nach über das Verfolgen nicht-funktionaler Anforderungen hinausgehende Aspekte ab (z.B. Gründe, Kontext und Entscheidungen). Pinheiro ist der Auffassung, dass die aus ihnen resultierende nicht-funktionale Verfolgbarkeit nicht direkt handhabbar ist und deshalb in die Form funktionaler Verfolgbarkeit überführt werden müsse: „Non-functional traces are not amenable to direct reference. [...] A non-functional trace has to be re-expressed in terms of functional ones."[37] Für Aspekte, die sich auf nicht-funktionale Anforderungen beziehen, wird im Rahmen dieser Arbeit eine andere Auffassung vertreten. Nicht-funktionale Anforderungen werden nach dem Stand der Technik in geeigneter Form als Bestandteil der Anforderungsspezifikation formalisiert. Es existieren spezielle Verfolgbarkeitsansätze, die ohne eine Abbildung auf funktionale Verfolgbarkeit auskommen.[38] Die anderen angeführten Aspekte nicht-funktionaler Verfolgbarkeit (Gründe, Kontext und Entscheidungen) sind für diese Arbeit nur soweit relevant, wie sie ihre Ursprünge in Rechtsvorschriften haben. Die generelle Unterscheidung in funktionale und nicht-funktionale Verfolgbarkeit macht deutlich, dass im Rahmen dieser Arbeit sowohl funktionale, als auch nicht-funktionale Anforderungen und deren Ursprünge berücksichtigt werden müssen.

Verfolgbarkeitsbeziehungen zwischen Elementen können beispielsweise durch Matrizen, als Diagramme in Modellen, als Querverweise oder in verlinkten Hypertext-Dokumenten dargestellt werden.[39] Wobei Brcina Vorteile bei Hypertext-Dokumenten sieht, da diese strukturelle Beziehungen leichter ersichtlich und wieder auffindbar machen.[40]

Prinzipiell ist von den folgenden drei Verwendungsmöglichkeiten von umfangreichen Verfolgbarkeitsinformationen auszugehen:[41]

- Selektives Nachvollziehen ermöglicht anhand definierter Kriterien eine Eingrenzung der vorliegenden Informationen (z.B. anhand des Typs von Ausgangselementen, Beziehungen oder Zielelementen), um verschiedene Sichtweisen auf die Verfolgbarkeitsinformationen zu extrahieren.

- Interaktives Nachvollziehen: Die Verfolgbarkeitsinformationen werden in einer Form bereitgestellt, dass interaktiv entlang des Beziehungspfades zwischen den Elementen vorwärts und rückwärts navigiert werden kann. Die rückwärtsgerichtete Navigation ermöglicht es hierbei, zu einem bereits betrachteten Elemente zurückzukehren (Browsing).

- Ungeführtes Nachvollziehen: Die Verfolgbarkeitsinformationen ermöglichen es, von genau einem Element auszugehen, es zu untersuchen und dann zum nächsten Element zu gehen. Für ein explizit ausgewähltes Element können dessen Kontext, Details und Beziehungen eingesehen werden.

Durch diese und vergleichbare spezielle Verwendungsmöglichkeiten helfen Ansätze zum Verfolgen von Anforderungen beim Verstehen der Zusammenhänge innerhalb von Anforderungen selbst und übergreifend zwischen Anforderungen und Elementen anderer Phasen der Systementwicklung.[42] Die Verfolgbarkeit von Anforderungen ist wesentliche Voraussetzung dafür, dass die Ergebnisse einer Systementwicklung die Anforderungen der Bedarfsträger (z.B. des Auftraggebers bzw. Kunden) abdecken.[43] Notwendige Änderungen und Fehlverhalten eines System können aus fachlicher Sicht in der Regel viel einfacher

[36] Vgl. [Pinheiro, 2004], S. 96 f.
[37] [Pinheiro, 2004], S. 99.
[38] Vgl. [Kassab et al., 2009] für einen Überblick und einen metamodellbasierten Ansatz.
[39] Vgl. [Brcina, 2006], S. 5.
[40] Vgl. [Brcina, 2006], S. 7 f.
[41] Vgl. hier und im Folgenden [Pinheiro, 1996], S. 24.
[42] Vgl. [Palmer, 1997], S. 412; übereinstimmend [Jacobson et al., 1999], S. 10.
[43] Vgl. [Palmer, 1997], S. 412.

anhand der Anforderungen an Systemfunktionen benannt werden als anhand der Implementierung dieser Funktionen (z.B. im Programmcode einer Programmiersprache). Damit ermöglicht Verfolgbarkeit die Betrachtung der Geschichte bzw. Vergangenheit eines Leistungsmerkmals des Systems entlang des Lebenszyklus.[44]

Um den Verfolgbarkeitsansatz dieser Arbeit in die etablierten Begrifflichkeiten einordnen zu können, müssen die durch ihn in Beziehung gesetzten Elemente der gleichen oder einer anderen Phase bzw. Entwicklungstätigkeit zugeordnet werden können. Wie sich Gesetze und Rechtsvorschriften im Verhältnis zur Anforderungsspezifikation in den Lebenszyklus einordnen, wird durch die bisherige Definition nicht klar bestimmt. Deshalb wird hier weitergehend die IEEE Norm 830 für die Verfolgbarkeit einer ganzen Anforderungsspezifikation verwendet, durch die zwei Aspekte (a und b) deutlich werden:

> **„A software requirements specification is traceable if [a] the origin of each of it's requirements is clear and if [b] it facilitates the referencing of each requirement in future development or enhancement documentation."[45]**

Zum einen umfasst Verfolgbarkeit die Möglichkeit, die Beziehungen zwischen Anforderungen untereinander und/oder Ergebnissen nachfolgender Aktivitäten (z.B. dem Design bzw. der Implementierung) herstellen zu können. Als weiteren Aspekt umfasst Nachvollziehbarkeit auch die Möglichkeit, den Ursprung (engl. origin) jeder Anforderung klären zu können. Auf dieser Basis haben Gotel und Finkelstein eine Unterscheidung der Verfolgbarkeit in zwei Klassen herausgearbeitet, indem sie die Anforderungsspezifikation in den Mittelpunkt stellen und die Verfolgbarkeit von Anforderungen relativ zur Anforderungsspezifikation definieren.[46] Die so genannte *pre-requirements specification (Pre-RS) tracebility* (Verfolgbarkeit im Vorfeld der Anforderungsspezifikation) betrifft die Aspekte der Verfolgbarkeit einer Anforderung, die relevant sind, bevor diese in eine Anforderungsspezifikation (engl. requirements specification) aufgenommen wird. Die *post-requirements specification (Post-RS) tracebility* (Verfolgbarkeit im Nachgang der Anforderungsspezifikation) behandelt alle Aspekte in Zusammenhang mit der Verfolgbarkeit einer Anforderung, die bereits in die Anforderungsspezifikation aufgenommen wurde und in folgenden Phasen und/oder Arbeitsergebnissen (S1 … Sn) transformiert wird (vgl. Abbildung 4).[47] Offen lassen Gotel und Finkelstein dabei die Frage der Verfolgbarkeit von Elementen, die nicht auf direktem Wege aus den Köpfen von Beteiligten in Anforderungen einer Anforderungsspezifikation resultieren.

---

[44] Vgl. [Ramesh&Jarke, 2001], S. 62.

[45] [IEEE 830, 1998], S. 8.

[46] Vgl. hier und im Folgenden [Gotel&Finkelstein, 1994].

[47] Als Erweiterung der Verfolgbarkeit im Vorfeld der Anforderungsspezifikation schlägt Pinheiro (in [Pinheiro, 1995], S. 10 f.), basierend auf Arbeiten von Gotel (unter Verweis auf [Gotel, 1995]), Environmental Tracing (Verfolgbarkeit im Umfeld) vor. Hierunter wird die Verfolgbarkeit von Einflüssen aus dem sozialen Umfeld, innerhalb dessen die Entwicklung des Systems durchgeführt wird, auf die Elemente des Systems verstanden. Als Beispiele führt Pinheiro Beteiligte, Richtlinien/Verfahrensweisen, kulturelle und organisatorische Aspekte an. Diese Einflüsse wirken nicht nur in der Anforderungsanalyse, sondern in allen Phasen der Systementwicklung. Für den Ansatz dieser Arbeit ist die Verfolgbarkeit im Umfeld nur relevant bezogen auf Umfeldeinflüsse, die sich aus Rechtsvorschriften ableiten lassen. Daher ist die Verfolgbarkeit im Umfeld aus Sicht dieser Arbeit eine Spezialisierung der Verfolgbarkeit im Vorfeld der Anforderungsspezifikation.

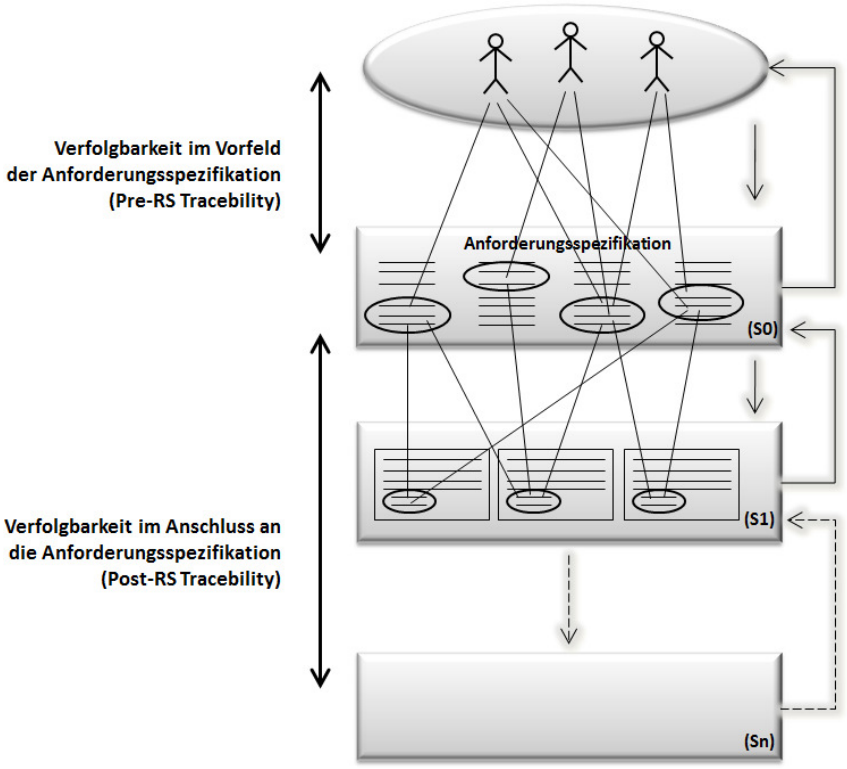


Abbildung 4 – Zusammenhang zwischen Pre-RS- und Post-RS-Traceability[48]

Gotel und Finkelstein stellen als Ergebnis ihrer Analyse fest[49], dass Verfolgbarkeit im Nachgang der Anforderungsspezifikation bereits gut beherrscht wird und dass die in diesem Zusammenhang verbliebenen Probleme nicht unüberwindlich sind. Im Gegensatz dazu sind die Aspekte, mit denen sich die Verfolgbarkeit im Vorfeld der Anforderungsspezifikation befasst, weder gut verstanden, noch umfassend unterstützt.[50] Die Mehrzahl der in der Praxis auftretenden Probleme resultiert im Wesentlichen aus fehlender oder ungeeigneter Verfolgbarkeit im Vorfeld und nicht im Nachgang der Anforderungsspezifikation. Insbesondere mangelt es an Techniken, die es ermöglichen, Informationen aufzuzeichnen und nachvollziehbar zu machen, die in Zusammenhang mit der Erstellung der Anforderungsspezifikation stehen.[51] Auch Ramesh und Jarke kommen hinsichtlich dieses Aspektes der Verfolgbarkeit von Anforderungen zu einer vergleichbaren Schlussfolgerung: „(…) a major challenge is the linking of rationales and sources to the requirements.“[52] Das bestehende Problem und die damit verbundene Herausforderung im Umgang mit der Verfolgbarkeit im Vorfeld der Anforderungsspezifikation kann seit den Arbeiten von Gotel und Finkelstein Mitte der 1990er Jahre als weithin bekannt angesehen werden. Eine systematische Untersuchung und Industriestudie kommt allerdings auch im Jahr 2011 noch zu dem Schluss, dass vorhandene und weit verbreitete Ansätze und Techniken zur Verfolgbarkeit die bestehende Problematik noch immer nicht lösen.[53] Die Thematik bleibt damit weiterhin relevant.

Durch die Betrachtung vorhandener Ansätze zur Verfolgbarkeit können Erkenntnisse darüber gewonnen werden, welche Defizite durch den Ansatz dieser Arbeit verbessert werden müssen. Dazu werden exemplarische Ansätze untersucht, die ihren Schwerpunkt in der Konzeptionalisierung haben, auf methodische Schritte ausgerichtet oder in Form eines

[48] Eigene Abbildung nach [Gotel&Finkelstein, 1994], S. 11.
[49] Vgl. hier und im Folgenden [Gotel&Finkelstein, 1994].
[50] [Gotel&Finkelstein, 1994], S. 13.
[51] Vgl. [Gotel&Finkelstein, 1994], S. 12f.
[52] [Ramesh&Jarke, 2001], S. 5.
[53] Vgl. [Torkar et al., 2011], S. 34.

Werkzeugs verfügbar sind. Ansätze mit dem Schwerpunkt auf Konzeptionalisierung entwickeln Modelle (oder vergleichbare Formalisierungen) auf einem übergeordneten Abstraktionsniveau, z. B. Metamodelle. Ihre Elemente werden bei Verwendung in einem konkreten Kontext in Form von Modellen instanziiert, um die relevanten Informationen zu formalisieren. Vertreter dieser Ansätze sind beispielsweise das *Comprehensive Traceability Model*[54] und das *Reference Model for Requirements Traceability*[55], die sehr umfangreich sind und einen universellen Ansatz verfolgen, allerdings in der praktischen Anwendung kaum von Bedeutung sind. Auch die *Unified Modelling Language* (siehe Abschnitt 2.2, Seite 23) bietet in ihrem Metamodell auf Basis der Metaklasse `Dependency` und deren Spezialisierungen universelle Möglichkeiten zur Dokumentation von Verfolgbarkeitsbeziehungen (z. B. mit den Stereotypen «`trace`» oder «`refine`»). Andere Ansätze gehen über die reine Formalisierung mittels Metamodellen hinaus, indem sie methodische Schritte und Regeln ihrer Anwendung definieren. Der in Deutschland für alle Bundesbehörden verbindliche *Entwicklungsstandard für IT-Systeme des Bundes (EStdIT)*, der in der aktuellen Version als *V-Modell XT*[56] bezeichnet wird, sieht die Verfolgbarkeit ausgehend von Anwenderanforderungen innerhalb der Anforderungsspezifikation (Lastenheft) zu Architekturaspekten, Hardware- und Softwareeinheiten vor. Eine Pre-RS Traceability zu den Ursprüngen einer Anforderung ist im V-Modell nicht vorgesehen.[57] Ein weiterer bekannter Vertreter sind die auf dem *Rational Unified Process (RUP)* basierenden *Traceability-Strategien für das Anforderungsmanagement mit Anwendungsfällen*[58]. Sie formalisieren allgemeingültige Konzepte (z.B. Bedürfnisse, Leistungsmerkmale) in einer Art Ontologie, um sie für die Nutzung im RUP handhabbar zu machen. Diese Konzepte sind teilweise dem Vorfeld der Anforderungsspezifikation zuzuordnen, so dass eine Pre-RS Traceability basierend auf diesen allgemein gültigen Konzepten möglich wäre. Die *RADIX* Methode[59] basiert auf Dokumenten, in denen Textpassagen mit Markierungen versehen werden, um sie mit anderen Textpassagen des gleichen oder anderer Dokumente in Beziehung zu setzen. Die dabei verwendeten Marken sind vordefiniert und entsprechen z.B. Anforderungen, Schlüsselworten und Erläuterungen. Die Methode besteht aus mehreren Aktivitäten, die zur Erzeugung, Nutzung und Verifikation der Dokumente durchlaufen werden. Weitere Ansätze sind vor allem durch ihre Implementierung innerhalb eines Werkzeugs zum Requirements-Engineering (z.B. *IBM Rational DOORS*, *Borland CaliberRM*, *Polarion Requirements*) bekannt. Von ihnen ist auch *IBM Rational Requisit Pro* ein repräsentativer Vertreter[60], mit dem zwischen Anforderungen, die aus Textpassagen in Dokumenten hervorgegangen sind, Verfolgbarkeitsbeziehungen zu Use Cases in *IBM Rational Rose* erzeugt werden können. Requisit Pro nutzt standardmäßig die Elemente der oben vorgestellten Traceability-Strategien des Rational Unified Process, ermöglicht aber auch weitere benutzerdefinierte Verfolgbarkeitsbeziehungen. Darüber hinaus bieten CASE-Werkzeuge (z.B. *MID Innovator*, *microTool objectiF*, *BOC Adonis*) Möglichkeiten, innerhalb ihres Metamodells die Verfolgbarkeitsbeziehungen zwischen den mit ihnen modellierten Elementen (auch werkzeugübergreifend, wie objectiF und Adonis zeigen) zu verwalten. Werden durch Werkzeuge bestimmte Vorgehensmodelle unterstützt, so kann die Aufzeichnung von Informationen, die bei der Durchführung von Aktivitäten des Vorgehensmodells entstehen, Rückschlüsse auf Zusammenhänge zwischen den bearbeiteten Arbeitsergebnissen

[54] Vgl. [Toranzo&Castro, 1999].
[55] Vgl. [Ramesh&Jarke, 2001].
[56] [V-Modell XT, 2006], [V-Modell XT Bund, 2009].
[57] Das V-Modell in der Version aus dem Jahr 1997 hatte grundsätzliche Mängel hinsichtlich der Anforderungsanalyse und -spezifikation. Ein Gutachten, das dem Bundesministerium für Verteidigung vorliegt, kommt zu dem Schluss, dass das Requirements Engineering im V-Modell 97 nicht ausreichend berücksichtigt ist (vgl. [Äko, 2001], S. 8). Das aktuelle und in diesem Punkt verbesserte V-Modell XT sieht insbesondere für die Sicherstellung der Verfolgbarkeit von Anforderungen explizite Aktivitäten und Verantwortlichkeiten vor.
[58] [Rational, 1998]
[59] Vgl. [Yu, 1994].
[60] Für einen Überblick, vgl. [Hood et al., 2005].

ermöglichen. Die Werkzeuge *microTool in-Step*, *TECHMOD*[61] und *PRO-ART*[62] verwenden dieses Grundprinzip.

Allen hier vorgestellten Ansätzen ist ein Grundprinzip gemeinsam: In der Regel wird zunächst eine formale Struktur mit bestimmten Elementen definiert bzw. vorausgesetzt. Dabei dominieren universelle Elemente, die stellvertretend für eher allgemeine Konzepte der realen Welt oder der Vorstellungswelt (z.B. Ziel, Leistungsmerkmal, Absicht) stehen. Diese Konzepte dienen dann der Formalisierung der für die Verfolgbarkeit relevanten Informationen. Mit einer derartigen Formalisierung geht allerdings das Risiko einher, die relevanten Informationen auf ihre formalen Aspekte zu reduzieren, was zwar ihre Handhabung erleichtert, aber nicht zwangsläufig die Verständlichkeit ihrer Verfolgbarkeit fördert.[63] Um diesen Widerspruch zu lösen, wird im Rahmen dieser Arbeit der Auffassung von Ravichandar gefolgt, dass zwischen dem informellen Ursprung einer Anforderung und der strikten Struktur der fixierten Anforderung ein Zwischenraum besteht, der durch einen Übergangsbereich gekennzeichnet ist (transition space).[64] Durch speziell eingeführte Abstraktionen gelingt die formale Ausgestaltung des Übergangs, ohne die Ursprünge aus ihrem Kontext zu lösen. Den Kontext bilden in dieser Arbeit die Texte der Ausgangsdokumente, hier speziell der Rechtsvorschriften mit Vollzugscharakter (siehe Abschnitt 2.5).

[61] Vgl. [Jarke&Marquardt, 1995].

[62] Vgl. [Dömges et al., 1996], [Pohl, 1996].

[63] Beispielsweise weist Pinheiro auf die aus einer Formalisierung resultierende Gefahr des Semantikverlustes hin („The reduction of this type of information to some formal structure may facilitate their manipulation but it is likely not to fulfill traceability needs since what relates a text, graph, or diagram to a requirement is not their formal features, if any." [Pinheiro, 1996], S. 10), während Cyriaks und Köhler auf die bremsende Wirkung der Formalisierung auf den kreativen Entwicklungsprozess in den frühen Phasen der Softwareentwicklung hinweisen ("Jede formale Einschränkung, die Methoden und Software-Tools mit sich bringen, wirkt hier als Beschränkung und kann den kreativen Prozess ins Stocken bringen."; [Cyriaks&Köhler, 2005], S. 185). Auch Ravichandar sieht ein ausgeprägtes Missverhältnis zwischen den informellen Benutzerbedürfnissen, die den Ursprung von Anforderungen bilden (vgl. [Gotel&Fineklstein, 1994]), und den strikten Strukturen fixierter Anforderungen („[...] capturing relationships between requirements and their sources, which are primarily user needs [...] is challenged by the vast disparity between the informality of a user need and the rigidity of a system requirement.", S. [Ravichandar, 2008], S. 94f.).

[64] Vgl. [Ravichandar, 2008], S. 94f.

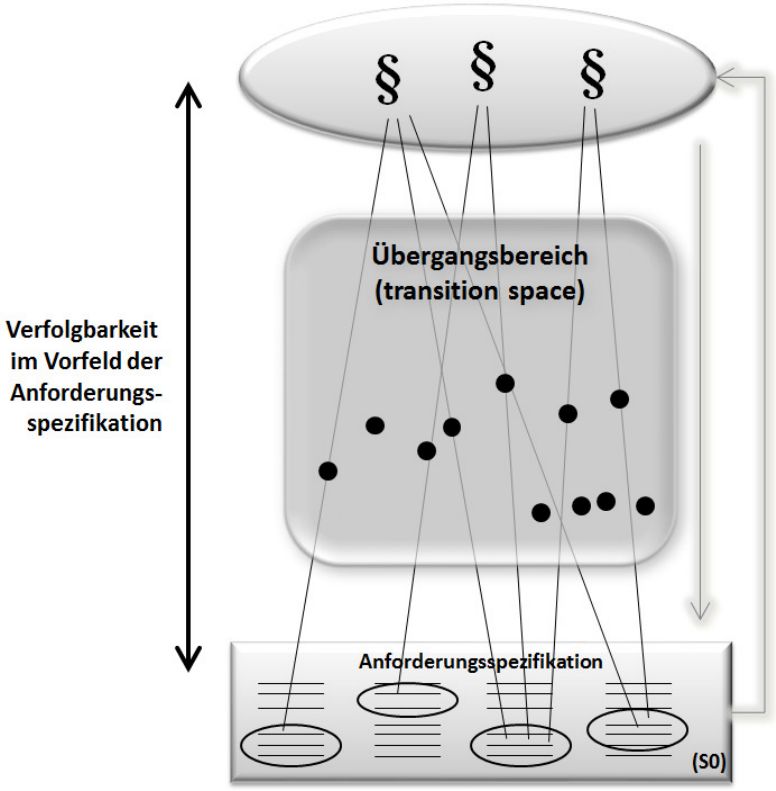


Abbildung 5 – Verfolgbarkeit im Vorfeld der Anforderungsspezifikation inkl. Übergangsbereich[65]

Die hier zugrunde liegende Arbeitshypothese geht davon aus, dass diese Abstraktionen durch eine geeignete Ontologie definiert und mit Techniken des Semantic Web auf Ursprünge von Anforderungen in Texten von Rechtsvorschriften angewendet werden können. Hierdurch bliebe der Kontext erhalten, und den Zwischenraum würden instanziierte Konzepte der Ontologie füllen. Darin unterscheidet sich der Ansatz dieser Arbeit insbesondere vom allgemein gültigen Capabilities-Ansatz, den Ravichandar zur Lösung vorschlägt. Weil die Effektivität der Nutzung von Verfolgbarkeitsinformationen aus einer geeigneten Definition ihrer Semantik resultiert, verschenken allgemein gültige Ansätze das Potenzial, das durch eine domänenspezifische Definition gegeben wäre.[66]

Insbesondere die Werkzeuge des Anforderungsmanagements erwecken den Eindruck, dass sie die Probleme der Verfolgbarkeit auch an dieser Stelle umfassend lösen können. Beispielsweise ermöglicht IBM Rational RequisitPro die Verfolgbarkeit zu Textpassagen innerhalb von Dokumenten. Ein wesentlicher Unterschied im Vergleich zum Ansatz dieser Arbeit ergibt sich eben durch den Einsatz einer speziellen Ontologie im Übergangsbereich. Aus ihr resultiert nicht nur die Semantik der Textelemente, sie wird auch als Grundlage für die anschließende Weiterverarbeitung verwendet. In der Definition einer geeigneten Semantik für die Verfolgbarkeitsbeziehungen sowie die Bestimmung einer geeigneten Granularität der beteiligten Elemente liegt eine wesentliche Herausforderung für die Verfolgbarkeit.[67] Weiterhin grenzt sich der Ansatz dieser Arbeit durch die Umsetzung basierend auf offenen Standards (z.B. des W3C, der OMG) und den Verzicht auf die

[65] Eigene Illustration der in [Ravichandar, 2008] beschriebenen Zusammenhänge.
[66] „In order to effectively utilize links and understand the underlying traceability relationships, it is necessary to define the semantics (e.g., type) of a link, however defining a formalism to represent the semantics is a non-trivial task and may be domain-specific. [...]Knowing and establishing the granularity of the elements being linked is important [...].“, vgl. [Huang et al., 2006], Abschnitt „D. Link Semantics“, o.S.
[67] Vgl. [Huang et al., 2006], Abschnitt „D. Link Semantics“, o.S.

Implementierung in proprietären Werkzeugfunktionen ab (vgl. Anforderung Nr. 5, Seite 7). Der Ansatz bietet auch die Möglichkeit, zusätzliche Informationen (z.B. den Grund, aus dem ein bestimmtes Element eines Gesetzestextes in dieser Form semantisch markiert wurde) zu dokumentieren und verwaltungsspezifische Aspekte (z.B. Hintergründe und Entscheidungen zu ihrer Überführung in Anforderungen) in den Übergang von Ursprüngen zu Anforderungen einfließen zu lassen.[68]

Zusammenfassend kann festgestellt werden, dass Verfolgbarkeit für das Vorfeld der Anforderungsspezifikation noch nicht als abschließend bearbeitet betrachtet werden kann und grundsätzlich Forschungsbedarf besteht. Es gibt bisher keinen Ansatz, der die domänenspezifische Verfolgbarkeit von Elementen in Rechtsvorschriften zu Anforderungen ermöglicht. Weiterhin sprechen Indizien in anderen Forschungsarbeiten dafür, dass die Techniken des Semantic Web basierend auf einer Ontologie geeignet sind, den Zwischenraum zwischen informellen Ursprüngen und fixierten Anforderungen geeignet zu überbrücken. Dadurch verbleiben die Ursprünge innerhalb ihres Kontext, instanziierte Konzepte der Ontologie füllen den Zwischenraum und dienen zur Herstellung von Beziehungen zu Anforderungen. In der damit verbundenen Konzeptionalisierung liegt aber generell eine Herausforderung, da hierdurch eine domänenspezifische Semantik definiert und die geeignete Granularität der verfolgbaren Elemente gefunden werden muss. Die domänenspezifische Semantik resultiert aus dem Anwendungsbereich im E-Government, der in Abschnitt 2.5 beschrieben wird. Die Granularität der Elemente muss geeignet sein, Anschluss an die klassischen Phasen der modellgetriebenen Softwareentwicklung zu finden. Dafür bietet die Model Driven Architecture mit ihren Konzepten zur Formalisierung von Anforderungen den Orientierungsrahmen.

2.2 Model Driven Architecture

Die Model Driven Architecture (MDA) ist ein Standard der OMG, der die Erstellung und Nutzung von bestimmten Modelltypen beschreibt, um die Entwicklung von Softwareanwendungen durch interoperable, wiederverwendbare, portable Softwarekomponenten zu ermöglichen, die auf standardisierten Modellen basieren.[69] Im Gegensatz zum verwandten Model Driven Software Development (MDSD), dessen Ziel die Generierung des Quellcodes in einer Programmiersprache als Ergebnis der Modellierung ist, bildet die MDA eine Architektur für Modelle. Im Rahmen dieser Architektur stellt sie die Modellierung mit ihren statischen Bestandteilen und deren dynamischem Zusammenwirken in den Vordergrund.[70] Die MDA definiert als wichtige Bestandteile ihrer Architektur drei Standpunkte (engl. viewpoints), aus deren Sicht Modelle im Rahmen des Softwareentwicklungsprozesses erstellt werden. Mit diesen Standpunkten lässt sich der Fokus einer modellhaften Systembeschreibung auf informationstechnikunabhängige,

[68] Damit folgt der Ansatz dieser Arbeit dem Vorschlag von Pinheirio zur Lösung des von ihm aufgezeigten Problems, indem er unstrukturierte Informationen berücksichtigt, um getroffene Entscheidungen auch nachträglich noch re-interpretierbar zu machen. Allerdings beschränkt sich der Ansatz auf textutelle Informationen und berücksichtigt die vorgeschlagenen Video- oder Ton-Aufzeichnungen von Interviews und Diskussionen nicht.

[69] Obwohl die MDA als OMG Standard im Sinne einer Spezifikation angesehen wird, ist die Dokumentation zur MDA noch nicht vollständig frei verfügbar. Diese Arbeit stützt sich auf die offiziellen OMG-Dokumente in [OMG MDA, 2000], [OMG MDA, 2001] und [OMG MDA, 2003a], sowie die minimal aktualisierte Version [OMG MDA, 2003b]. Die zuletzt im Jahr 2003 aktualisierte Dokumentation „MDA Guide, Version 1.0.1“ wird auf der OMG-Hompage (http://www.omg.org/mda/specs.htm, letzter Aufruf: 21.08.2011) als „Interims-Version“ bezeichnet, die Mitte 2005 durch das „MDA Foundation Model“ abgelöst werden sollte: „The MDA Guide, Version 1.0.1, just mentioned, defines the MDA. OMG members expect to replace this interim version with an update, based on the Foundation Model also just mentioned, around mid-2005.“ Der Arbeitsstand des „MDA Foundation Model“ ist zum Zeitpunkt der Erstellung dieser Arbeit nur für OMG-Mitglieder einsehbar.

[70] Vgl. [Nolte, 2009], S. 8.

plattformunabhängige und plattformspezifische Aspekte eines Softwaresystems und seiner Umgebung richten.[71] Jedes im Rahmen der MDA-Anwendung erstellte Modell besitzt daher immer einen der folgenden Modelltypen, der zu einem der Standpunkte korrespondiert. Der Modelltyp *Computation Independent Model (CIM)* stellt das System vom informationstechnikunabhängigen Standpunkt aus in einer Perspektive dar, die die Umgebung des Systems und die Anforderungen an das System zeigt. Die strukturellen Details des Softwaresystems und die (Informations-)Verarbeitung innerhalb des Systems werden von diesem Standpunkt aus nicht dargestellt. Ein *Platform Independent Model (PIM)* stellt das System vom plattformunabhängigen Standpunkt dar, ohne dass auf die Details einer speziellen Plattform eingegangen wird. Unter einer Plattform wird dabei eine Menge von Subsystemen oder Technologien verstanden, die eine zusammenhängende Funktionalität über Schnittstellen anbietet sowie deren Nutzung und zugrunde liegende Konzepte beschreibt.[72] Unter einem *Platform Specific Model (PSM)* wird ein Modelltyp verstanden, der auf die spezifische Nutzung einer konkreten Plattform (z.B. deren Schnittstellen und Konzepte) eingeht. Der plattformspezifische Standpunkt kombiniert die relevanten Elemente der Plattform mit den Informationen aus dem PIM. Zu diesem Zweck reicht es nicht, dass die Plattform in Form von Handbüchern oder nur im Kopf des Architekten existiert.[73] Die MDA sieht vor, dass die Plattform in einem *Plattformmodell (PM)* formalisiert wird, um verwendet werden zu können. Dieses Plattformmodell stellt technik-orientierte Konzepte bereit, die Bestandteile der Plattform und die durch die Plattform bereitgestellten Services beschreiben. Diese Konzepte können in plattformspezifischen Modellen verwendet werden, um auszudrücken, wie die modellierte Anwendung die Services der spezifischen Plattform benutzt. Außerdem stellt das Plattformmodell auch Anforderungen an die Verbindungen und an die Nutzung der Plattformbestandteile und an die Verbindungen zwischen Anwendung und Plattform.[74] Die MDA sieht vor, dass ein Plattformmodell mit UML und/oder OCL (siehe unten, Seite 23) erstellt wird. Einen Überblick über die Standpunkte und korrespondierenden Modelle der MDA gibt die nachfolgende Abbildung 6.

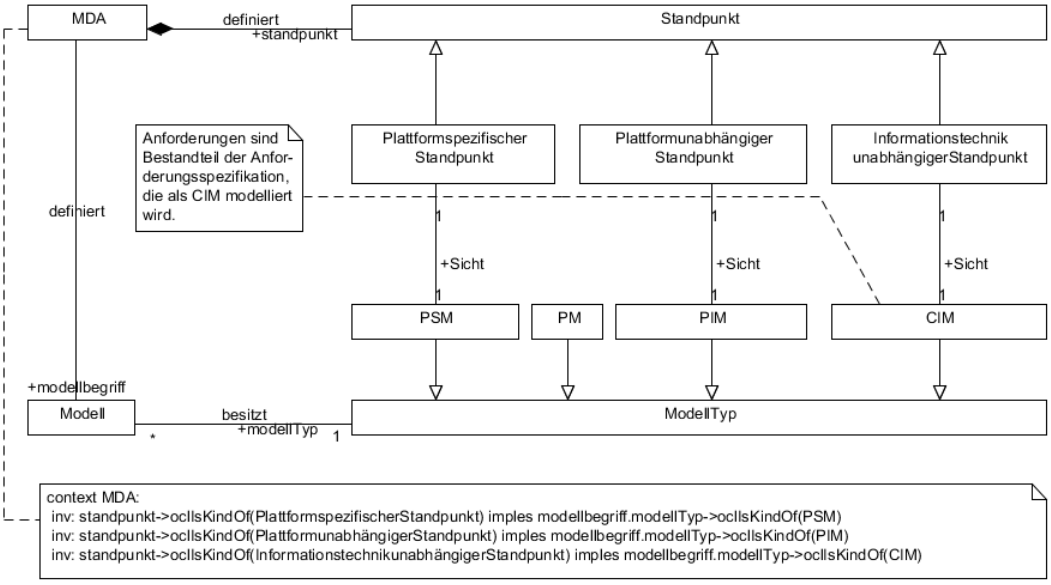


Abbildung 6 – Überblick über Modelle und Standpunkte der MDA

[71] Die MDA definiert neben den drei Standpunkten (Viewpoints) auch Sichten (Views). Unter einer Sicht versteht sie die Darstellung eines Systems von einem ausgewählten Standpunkt aus. [OMG MDA, 2003b], S. 2-3.
[72] Vgl. [OMG MDA, 2003b], S. 2-3.
[73] Vgl. [OMG MDA, 2003b], S. 3-2, [Petrasch&Meimberg, 2006], S. 103.
[74] Vgl. [OMG MDA, 2003b], S. 2-3.

Legt man einen generellen Entwicklungsprozess zugrunde, der mindestens die aufeinander aufbauenden Aktivitäten Geschäftsmodellierung, Anforderungsanalyse, Analyse, Design und Implementierung umfasst, gibt es in der Literatur unterschiedliche Auffassungen, welche dieser Aktivitäten den jeweiligen Standpunkten und Modelltypen zuzuordnen sind.[75] Im Rahmen dieser Arbeit erfolgt die Einordnung entsprechend des OMG MDA Guide[76]. Er sieht vor, dass eine Anforderungsspezifikation immer die Perspektive des informationstechnikunabhängigen Standpunktes einnimmt, da sie definitionsgemäß Anforderungen an das zu entwickelnde System darstellt, ohne auf die Details ihrer Umsetzung einzugehen. Im Rahmen der Anforderungsanalyse werden deshalb Modelle des Typs CIM erstellt. Weil E-Government als Durchführung von Prozessen der öffentlichen Verwaltung unter intensiver Nutzung der Informationstechnik verstanden wird, muss ein CIM für E-Government-Anwendungen einerseits Verwaltungsprozesse, andererseits Anforderungen an die unterstützende Informationstechnik darstellen. Das Modell der Verwaltungsprozesse ist unabhängig von der sie unterstützenden Informationstechnik, weil die Prozesse die aktuelle oder zukünftige fachliche Arbeitsweise darstellen. Mit der Entscheidung, welche Teile dieser Prozesse durch Informationstechnik unterstützt werden sollen, findet der Übergang von der Geschäftsprozessmodellierung zur Anforderungsanalyse statt. Sowohl die in der Geschäftsprozessmodellierung als auch die im Rahmen der Anforderungsanalyse erstellten Modelle sind vom Typ CIM. Sie spezifizieren somit verständlich, korrekt, eindeutig, vollständig, widerspruchsfrei, umsetzbar und verifizierbar die Anforderungen an die unterstützende Informationstechnik. Für das CIM definiert die MDA weiterhin, dass die Modellelemente dieses Typs verfolgbar bis zu den Elementen im PIM und PSM sein sollen und umgekehrt.[77] Für ein vorliegendes CIM kann somit per Definition die Verfolgbarkeit zu den weiteren Modellen der MDA als gegeben angesehen werden.

Die MDA erweitert den allgemeinen Modellbegriff, der generell ein Modell als eine vereinfachte und zweckbezogene Abbildung der Realität definiert. Ein Modell der MDA basiert immer auf einer Sprache, die über eine wohldefinierte Syntax und Semantik verfügt (und darüber hinaus weitere Elemente enthalten kann).[78] Die Syntax kann beispielsweise in Form von Text oder Grafik dargestellt werden, die Semantik hingegen mehr oder weniger formal in Form von „Dingen, die sich in der Welt beobachten lassen“ und deren Beziehungen definiert werden.[79] Um Modelle vom Typ CIM erstellen zu können, wird im Rahmen dieser Arbeit der OMG-Modellierungsstandard *Unified Modeling Language (UML)*[80] eingesetzt, der mit dem UML-Metamodell über eine wohldefinierte Syntax und Semantik verfügt. Mit der UML werden standardisierte Modelle erstellt, die statische Bestandteile (objektorientierter) Softwaresysteme und deren dynamisches Zusammenwirken aus verschiedenen Sichten in Form von Diagrammen zeigen. Beispielsweise können mit der UML Anwendungsfalldiagramme (Use Case diagram) und Klassendiagramme erstellt werden. Die Anwendungsfalldiagramme zeigen, wie Akteure das zu entwickelnde Systeme nutzen und können benutzt werden, um Anforderungen an das Systemverhalten zu

[75] Frankel ordnet die Geschäftsmodellierung nicht in die MDA-Modelltypen ein. Er sieht das "Business Model" als vorgelagertes Modell an, aus dem das CIM abgeleitet wird (vgl. Frankel, 2003], S. 192f.). Das CIM ist in seiner Einordnung das Ergebnis der Anforderungsanalyse, dass im PIM im Rahmen von Analyseaktivitäten weiter verfeinert wird (vgl. [Frankel, 2003]). Kleppe et al. ordnen die Geschäftsmodellierung dem CIM-Typen zu (vgl. [Kleppe et al., 2003], S. 19). Petrasch und Meimberg führen aus, dass mit dem CIM beispielsweise Anforderungen in Form von Klassenmodellen spezifiziert werden können ([Petrasch&Meimberg, 2006], S. 100). Das PIM hingegen stellt "ausschließlich die rein fachlichen Aspekte" dar, so dass auch Modelle ohne Bezug zu einer zu entwickelnden Software (z.B. als Ergebnis der Geschäftsprozessanalyse und -optimierung) von Petrasch und Meimberg als PIM angesehen werden.

[76] Vgl. [OMG MDA, 2003b], S. 2-5.

[77] Vgl. [OMG MDA, 2003a], S. 3-1; [OMG MDA, 2003b], S. 3-1 und weiterhin [Kleppe et al., 2003], S. 75f.; [Frankel, 2003], 209f.

[78] Vgl. [OMG MDA, 2001], S. 3.

[79] Vgl. [OMG MDA, 2001], S. 3.

[80] Vgl. [OMG UML, 2009a], [OMG UML, 2009b].

spezifizieren.[81] In verschiedenen Methoden zur Softwareentwicklung (z.B. OOSE[82], RUP[83], actiF[84]) ist eine Anforderungsanalyse vorgesehen, die u.a. auch die Erstellung eines Domänenmodells (Anwendungsbereichsmodells) in Form von UML-Klassendiagrammen vorsieht. Mit dem Domänenmodell werden zentrale Dinge der realen oder vorstellbaren Welt benannt, beschrieben, in Form von Klassen formalisiert und mit ihren Beziehungen untereinander dargestellt. Durch Domänenmodelle soll die Ermittlung von Anforderungen an das zu entwickelnde System unterstützt werden, in dem unter anderem eine gemeinsame Sprache zwischen allen Beteiligten gefunden wird. Gleichzeitig ermöglicht es durch seine graphische Darstellung auf Basis einer verbreiteten und akzeptierten Notation auch komplexe Zusammenhänge nahezu allgemeinverständlich darzustellen. Durch Einsatz der *Object Contraint Language (OCL)*[85] kann die Aussagekraft von UML-Modellen, z.B. durch Definition von Invarianten zwischen Modellelementen sowie Vor- und Nachbedingungen von Methoden weiter erhöht werden.

Modelle mit Anwendungsfall- und Klassendiagrammen werden im Rahmen der Anforderungsanalyse primär zur Formalisierung funktionaler Anforderungen verwendet. Um nicht-funktionale Anforderungen in der MDA nutzbar zu machen, müssen diese als Modelle bzw. Modellelemente formalisiert sein. In der Literatur gibt es Ansätze, die eine Integration nicht-funktionaler Anforderungen in Anwendungsfall- und Klassendiagramme erlauben, so dass sie mit den Standard-Modelltypen der MDA verwendet werden können.[86] Es werden in diesen Ansätzen beispielsweise Notizen oder Constraints als Modellelemente in Verbindung mit Stereotypen verwendet, um nicht-funktionale Anforderungen zu formalisieren.

Einen speziellen als *Non-Functional MDA (NFMDA)* bezeichneten Ansatz schlagen Cortellessa et al. vor.[87] Sie erweitern die bekannten Modelltypen der MDA um korrespondierende Modelltypen für die Darstellung der nicht-funktionalen Anforderungen (CI*NF*M, PI*NF*M, PS*NF*M) und entsprechende Transformationen zwischen den Modelltypen und den Erweiterungen. Weil im Rahmen dieser Arbeit der Schwerpunkt auf dem Vorfeld der Anforderungsanalyse liegt, verlässt die Erweiterung um zusätzliche Modelltypen für den plattformunabhängigen und plattformspezifischen Standpunkt den Rahmen dieser Arbeit. Stattdessen werden hier die nicht-funktionalen Anforderungen zusammen mit den funktionalen Anforderungen in den Modellen der Anforderungsanalyse dargestellt.

Zusammenfassend wird unter einer Anforderungsspezifikation im Rahmen dieser Arbeit eine Menge von Anwendungsfall- und Klassendiagrammen verstanden, die durch Fließtext oder strukturierten Text ergänzt sind und funktionale sowie nicht-funktionale Anforderungen an das zu erstellende System formalisieren.

Weil E-Government die Durchführung der Prozesse betrifft, sind Geschäftsprozessmodelle ein weiterer wichtiger Modelltyp. Mit ihnen wird der Ablauf von Geschäftsprozessen, bestehend aus den durchgeführten Aktivitäten und weiteren relevanten Informationen, dargestellt. Damit beschreiben Geschäftsprozesse das Umfeld eines zu entwickelnden

---

[81] Details einer solchen Nutzung können durch weitere UML-Diagramme verfeinert werden (z.B. durch Interaktions- oder Aktivitätendiagramme). Es ist darüber hinaus Stand der Technik, dass die Interaktion zwischen menschlichen Akteuren und dem System über eine Benutzeroberfläche erfolgt, deren Entwurf ebenfalls Teil der Anforderungsspezifikation ist. Für den Entwurf der Benutzeroberfläche stellt die UML keine standardisierten Diagramme oder Elemente zur Verfügung. In der Praxis werden daher alternative Darstellungsformen (z.B. Storyboards oder GUI-Prototypen) genutzt.

[82] OOSE = Object Oriented Software Engineering (vgl. [Jacobson et al., 1994]).

[83] RUP = Rational Unified Process (vgl [Kurchten, 1998]).

[84] Vgl. [microTool, 1998].

[85] Vgl. [OMG OCL, 2010].

[86] Vgl. u.a. [Tonu, 2006], [Cysneiros&Leite, 2001], [Berenbach&Gall, 2006], [Ciano et al., 2010], [Supakkul&Chung, 2005], [Zhu&Gorton, 2007], [Botella et al., 2001].

[87] Vgl. [Cortellessa et al., 2007].

Systems, ohne Aussagen über die zugrunde liegende Informationstechnik zu treffen. Dies entspricht der Sichtweise des CIM. Zum Zweck der Geschäftsprozesssmodellierung wurde von der OMG die spezielle *Business Process Model and Notation (BPMN)*[88] standardisiert. Sie löst international zunehmend die lange Zeit eingesetzten *Ereignisgesteuerten Prozessketten (EPK)*[89] ab.[90] Gleichzeitig wird seitens der OMG mit dem *UML Profile for BPMN Processes RFP*[91] an einer Möglichkeit gearbeitet, den Umfang der BPMN direkt mit den UML Aktivitätendiagrammen auszudrücken. Mit dieser Möglichkeit kann noch nicht gearbeitet werden, da noch keine Ergebnisse des Standardisierungsprozesses öffentlich zugänglich sind. Sie legt aber nahe, dass die UML Aktivitätendiagramme (zukünftig) als gleichwertige Alternative zu BPMN angesehen werden können.

Sowohl UML- als auch BPMN-Modelle werden jeweils mit einer Sprache ausgedrückt, die auf einem Metamodell basiert, dass die zulässigen Modellelemente und ihre möglichen Beziehungen beschreibt. Da ein Metamodell auch ein spezielles Modell ist, ist es ebenfalls mit einer Sprache, der so genannten Metasprache geschrieben. Eine Metasprache wird entsprechend durch ein Meta Metamodell definiert. Obwohl sich die in Abbildung 7 skizzierte Ebenenbildung beliebig fortsetzen ließe, haben sich insgesamt vier Ebenen (M0 bis M3) etabliert.[92] Für die UML sind Meta Object Facilities (MOF) das Meta-Metamodell. Auch die BPMN wird in Version 2.0 MOF als Meta-Metamodell verwenden, die Version 1.2 verwendet MOF noch nicht. In der nachfolgenden Abbildung 8 werden die zusätzlichen Aspekte der Syntax- und Semantik-Definition im Kontext der MDA-Modelle dargestellt.

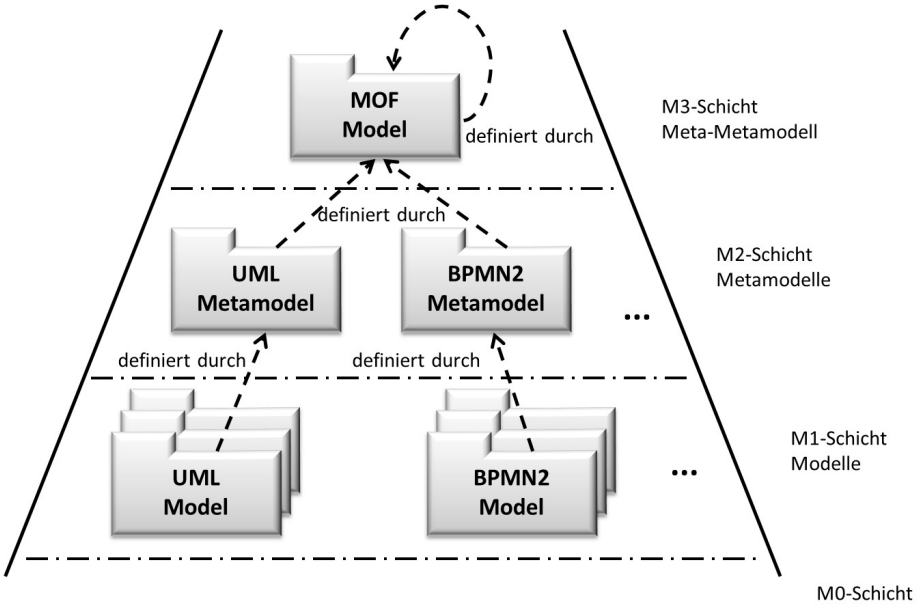


Abbildung 7 – Modell und Meta-Modellebenen[93]

[88] Die BPMN (vgl. [OMG BPMN, 2010a] für die aktuelle Version 2 und [OMG BPMN, 2009] für die letzte Fortschreibung der BPMN 1 mit Versionsnummer 1.2.) ermöglicht es, Erstellern und Nutzern von Geschäftprozessmodellen auf standardisierte Weise zusammenzuarbeiten. Darüber hinaus ist die Notation auf die Überführung in eine ausführbare Prozesssprache ausgerichtet. Sie verfügt über Notationselemente, mit denen sich beispielsweise Aktivitäten, Subprozesse Verzweigungen, Ereignisse und der Kontrollfluss zwischen ihnen ausdrücken lassen. Ihre Syntax und Semantik sind durch das BPMN-Metamodel definiert. (Die in Version 1.0 mit "Business Process Modeling" bezeichnete Abkürzung BPMN stand ab Version 1.1 für "Business Process Model and Notation".)

[89] Vgl. [Scheer, 2001].

[90] Vgl. [Decker et al., 2009], S. 92.

[91] Vgl. [OMG UML, 2010].

[92] Die oberste Schicht M3 wird durch Instanzen ihrer eigenen Modellelemente beschrieben, wodurch weitere Schichten nicht notwendig sind („selbst beschreibendes" Meta-Metamodell).

[93] Darstellung nach [OMG MOF, 2002], S. 2-3 und [OMG UML, 2009a], S. 14.

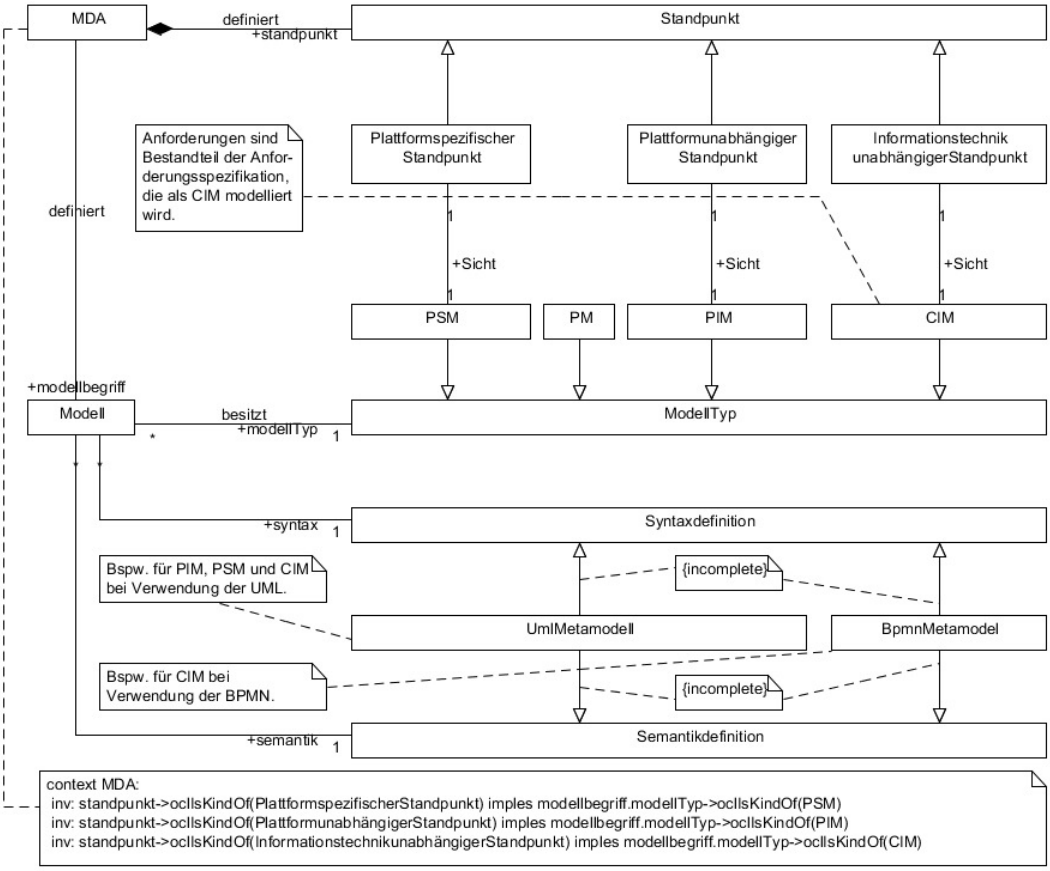


Abbildung 8 – Überblick über den Zusammenhang zwischen Modellen, Standpunkten, Syntax- und Semantikdefinition

Es gibt zwei Untermengen des MOF-Standards. Das Essential MOF (EMOF) dient dazu, Metamodelle auf einfache Weise mit einfachen Modellelementen zu konstruieren.[94] Das Complete MOF (CMOF) erweitert das EMOF um fortgeschrittene Modellierungsmöglichkeiten. Für diese Arbeit sind die Modellierungselemente des EMOF für die Konstruktion eigener Metamodelle in Kombination mit vorhandenen Metamodellen der UML und der BPMN ausreichend. Innerhalb der populären und zur prototypischen Implementierung des Ansatzes dieser Arbeit verwendeten Eclipse-Werkzeugunterstützung wird die hier relevante EMOF-Untermenge des Standards weitestgehend durch das ECORE-Metamodell implementiert, das im Rahmen der Projekte *Eclipse Modeling Framework (EMF)* [95]und *Graphical Modeling Framework (GMF)*[96] bereitgestellt wird. Alle im Rahmen dieser Arbeit verwendeten und auf Eclipse basierenden Werkzeuge verwenden das ECORE-Metamodell.

Neben den bisher beschriebenen statischen Bestandteilen definiert die MDA auch das dynamische Zusammenwirken dieser Bestandteile. Sie definiert Transformationen von Modellen der einzelnen Typen ineinander. Dabei werden durch die Transformation ein oder mehrere Ausgangsmodelle in ein oder mehrere Zielmodelle transformiert. Wenn Ausgangsmodelle und Zielmodelle unterschiedliche MDA-Standpunkte repräsentieren (z.B. CIM und PIM), repräsentieren sie unterschiedliche Aspekte des Systems. Diese Unterschiede zeigen sich in den zugrunde liegenden Konzepten, den verwendeten Modellelementen und deren Semantik. Die zuvor beschriebenen Anwendungsfalldiagramme mit Akteuren und Anwendungsfällen eines CIM sind ein Beispiel für Konzepte der Anforderungsanalyse und

[94] Vgl. [OMG MOF, 2006], S. 31.
[95] Die Homepage des EMF-Projektes ist abrufbar unter: http://www.eclipse.org/modeling/emf/ (letzter Aufruf: 21.08.2011).
[96] Die Homepage des GMF-Projektes ist abrufbar unter: http://www.eclipse.org/gef/ (letzter Aufruf: 21.08.2011).

der daraus resultierenden Modellelemente *Akteur* und *Anwendungsfall*. Im Rahmen der weiteren Modellierung erfolgt ihre Transformation in *Klassen* mit *Attributen* und *Methoden* in einem PIM oder nachfolgend in einem PSM, da Akteure und Anwendungsfälle im Gegensatz zu Klassen in objektorientierten Programmiersprachen keine direkte Entsprechung haben. Verbunden mit dieser Transformation ist auch die Überführung der Semantik der Ausgangselemente in die Semantik der Zielelemente. Die Tabelle 1 illustriert für die Konzepte Akteur, Anwendungsfall und Klasse die unterschiedliche Semantik, die sich aus dem UML-Metamodell ergibt.

<table>
<tr><th>Modellelemente des CIM</th><th>Modellelement des PIM</th></tr>
<tr><td>Semantik des Modellelements „Akteur“:

„Actors model entities external to the subject. When an external entity interacts with the subject, it plays the role of a specific actor. [...]“[97]</td><td rowspan="2">Semantik des Modellelements „Klasse“:

„The purpose of a class is to specify a classification of objects and to specify the features that characterize the structure and behavior of those objects. Objects of a class must contain values for each attribute that is a member of that class, in accordance with the characteristics of the attribute, for example its type and multiplicity. [...] Operations of a class can be invoked on an object, given a particular set of substitutions for the parameters of the operation. An operation invocation may cause changes to the values of the attributes of that object. It may also return a value as a result, where a result type for the operation has been defined. Operation invocations may also cause changes in value to the attributes of other objects that can be navigated to, directly or indirectly, from the object on which the operation is invoked, to its output parameters, to objects navigable from its parameters, or to other objects in the scope of the operation’s execution. Operation invocations may also cause the creation and deletion of objects. [...]“[98]</td></tr>
<tr><td>Semantik des Modellelements „Anwendungsfall“:

„An execution of a use case is an occurrence of emergent behavior. Every instance of a classifier realizing a use case must behave in the manner described by the use case.

Use cases may have associated actors, which describes how an instance of the classifier realizing the use case and a user playing one of the roles of the actor interact. [...]“[99]</td></tr>
</table>

Tabelle 1 – Semantik ausgewählter Modellelemente des CIM und des PIM bzw. PSM

Die zwischen Modellen unterschiedlicher Standpunkte existierenden Unterschiede in Konzepten, Modellelementen und ihrer Semantik müssen im Transformationsprozess berücksichtigt werden. Aus diesen Unterschieden kann sich auch eine *semantische Lücke*[100] ergeben, so dass zusätzliche Informationen notwendig sind, um im Transformationsprozess die Lücke schließen zu können.

Im Kontext der MDA wird häufig die Transformation plattformunabhängiger Modelle in plattformabhängige Modelle in den Vordergrund gestellt. Auch bei dieser Transformation gibt es Unterschiede zwischen den Modellen, obwohl im Wesentlichen die gleichen Elemente zur Modellierung genutzt werden können. Die Unterschiede ergeben sich, weil Modellelemente im PIM unabhängig von einer Plattform sind, sie im PSM aber hinsichtlich einer definierten Plattform konkretisiert sein müssen, um als plattformspezifisch gelten zu können. Hier wird die Lücke durch den Einsatz eines Plattformmodells geschlossen, das die Bestandteile und Services der Plattform so formalisiert, dass sie zur Spezifikation der Anwendungsfunktionen im PSM verwendet werden können. Der tatsächliche Einsatz eines Plattformmodells, das, wie von der MDA empfohlen, mittels UML und/oder OCL

[97] [OMG UML, 2009b], S. 588.
[98] [OMG UML, 2009b], S. 50
[99] [OMG UML, 2009b], S. 597
[100] Der Begriff der *Semantischen Lücke* bezeichnet u.a. eine „Inkompatibilität (das Nicht-zueinander-passen) [...] von Zeichenrealemen verschiedener Sprachen“ ([Jungclaussen, 2001], S. 591), wobei hier (vereinfacht) die Inkompatibilitäten von Elementen verschiedener Sprachen oder von verschiedenen Elementen der gleichen Sprache gemeint ist. Die Lücke ist regelmäßig zu überbrücken, wenn Modelle, die mit einem (bestimmten) Formalismus spezifiziert wurden, in Modelle transformiert werden sollen, die einen anderen Formalismus nutzen (vgl. [Amstel et al., 2008], S. 61).

formalisiert ist, ist in den in der Literatur dokumentierten Ansätzen vergleichsweise selten zu finden. In der Regel werden stattdessen plattformspezifische Abbildungsregeln verwendet, ohne dass ein PM explizit formalisiert wurde.[101] Die Transformation von informationstechnikunabhängigen Modellen in plattformunabhängige Modelle steht im Vergleich zur PIM-PSM-Tranformation eher im Hintergrund. Es sind allerdings Beispiele für CIM-PIM-Transformation in der Literatur dokumentiert.[102] Diese Ansätze transformieren die Modellelemente des CIM anhand spezifischer Abbildungsregeln in Elemente des PIM, ohne ein zusätzliches Modell (z.B. analog zum Plattformmodell) zu verwenden.

Die MDA sieht verschiedene Typen von Transformation und daraus resultierend verschiedene Ansatzpunkte für die Gestaltung von Abbildungsregeln vor.[103] Zum einen können Abbildungsregeln durch Abbildung von Modellelementtypen des Ausgangsmodells auf Elementtypen des Zielmodells definiert werden (Model Type Mapping, engl. für Modelltypabbildungen). In einer grundlegenden Variante dieser Modelltypabbildung werden Typen aus der Sprache des Ausgangsmodells auf Typen aus der Sprache des Zielmodells abgebildet. Für Sprachen, deren Typen auf einem Metamodell basieren, kann die Variante durch Abbildung von Metamodellelementen des Ausgangsmodells auf Metamodellelemente des Zielmodells umgesetzt werden. Die Abbildung kann auch Regeln oder Algorithmen umfassen, die sich beispielsweise auf Eigenschaftsausprägungen der Modellelemente (als Instanzen dieser Metamodellelemente) beziehen. Dadurch ergeben sich Steuerungsmöglichkeiten des Transformationsprozesses. Zum anderen kennt die MDA Transformationen, die die Modellelemente unabhängig von ihrem Typ oder zugehörigen Metamodellelement verarbeiten. Als Beispiel wird die Markierungstechnik angeführt, bei der von einer Menge so genannter Marken ausgegangen wird. Marken werden benutzt, um Modellelemente des Ausgangsmodells zu markieren. Jeder Marke (und nicht jedem Typ, wie bei der Modelltypabbildung) ist dabei eine spezielle Bedeutung zugeordnet. Diese Bedeutung steuert die Abbildung markierter Elemente des Ausgangsmodells auf das Zielmodell. Die Kombination dieser Varianten ist möglich und bietet eine effiziente und flexiblere Anwendung der Transformation.[104] Als dritte Variante können Vorlagen (Templates) in den Abbildungsregeln verwendet werden, um die Erzeugung von Elementen des Zielmodells zu steuern. Bei diesen Vorlagen kann es sich um parametrisierbare Modelle oder definierte Muster handeln. Parametrisierbare Modelle werden durch Werte oder Elemente des Ausgangsmodells gefüllt, dabei ggf. modifiziert und ergeben so das Zielmodell. Die Transformation anhand von Mustern überführt beispielsweise ein bestimmtes Muster von Modellelementen des Ausgangsmodells in ein Muster von Elementen des Zielmodells. Diese vorlagenbasierte Variante ist auch in Zusammenhang mit dem von der MDA eingeführten Mischen von Modellen im Rahmen einer Transformation relevant. In diesem Fall wird das Ausgangsmodell mit einem weiteren Modell zusammengeführt und bildet dadurch das Zielmodell. Am Beispiel einer PIM-PSM-Transformation ist dies in Abbildung 9 dargestellt.

[101] Freundliche Auskunft von Siegfried Nolte, Verfasser von [Nolte, 2009] (per E-Mail vom 13.02.2011).
[102] Vgl. u. a. [Zhang et al., 2005], [Rodríguez et al., 2007a], [Rodríguez et al., 2007b], [Rodríguez et al., 2008], [Suarez et al., 2008], [Kherraf et al., 2008], [Koch et al., 2008], [Kardoš&Drozdová, 2010].
[103] Vgl. hier und im Folgenden [OMG MDA, 2003b], S. 3-2 ff.
[104] Vgl. [OMG MDA, 2003b], S. 3-4.

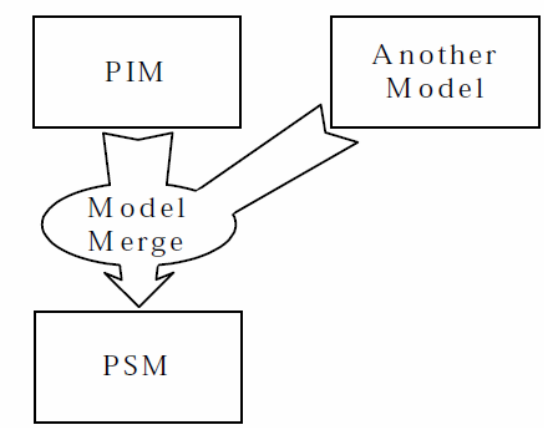


Abbildung 9 – Mischen von Modellen im Rahmen einer MDA-Transformation[105]

Durch die Aufzeichnung von Informationen, die während der Durchführung einer solchen Transformation von einem oder mehreren Ausgangsmodellen in ein oder mehrere Zielmodelle anfallen, können Verfolgbarkeitsbeziehungen zwischen den transformierten Elementen konstruiert werden. Für jedes Element eines Zielmodells ist dann entlang der Beziehung verfolgbar, aus welchem bzw. welchen Elementen des Ausgangsmodells es hervorgegangen ist.[106] Zusätzlich zur Aufzeichnung dieser Beziehungen kann die Aufzeichnung der jeweils eingesetzten Abbildungsregel Aufschluss darüber geben, warum eine Transformation des Ausgangselements in ein oder mehrere Zielelemente in dieser Weise stattgefunden hat.

Für die Definition von Transformationen und Abbildungsregeln zwischen Modellen wurde von der OMG der Standard *Meta Object Facility 2.0 Query/View/Transformation (QVT)*[107] entwickelt. Mit QVT können die für die Transformation relevanten Elemente oder Teile eines Ausgangsmodells ermittelt werden ("Query"). Es kann definiert werden, wie die Elemente des Zielmodells aussehen sollen ("View"). Die Beschreibung der Transformation, mit der die ermittelten Elemente in ein konkretes Zielmodell überführt werden, erfolgt ebenfalls mit QVT. Die Spezifikation definiert dazu einen QVT-Kern und die beiden Teilsprachen *QVT Relations Language (QVT-R)* und die *QVT Operational Mapping (QVT-OM)*. Durch QVT-R werden Transformationen deskriptiv mit Relationen zwischen den beteiligten Modellelementen und entsprechenden Regeln definiert. Mit QVT-OM erfolgt die „Formulierung von Relationen und Transformationen mit imperativen Sprachkonstrukten, die denen von bekannten Programmiersprachen ähnlich sind."[108] Durch den Einsatz von QVT-R wird die Konsistenz der Modelle durch die Abbildungsregeln automatisch in beide Richtungen sichergestellt, was bei QVT-OM nicht der Fall ist. Beide Teilsprachen setzen an den Metamodellen der beteiligten Ausgangs- und Zielmodelle an und ermöglichen, Modellelemente, ausgehend vom Metamodell, zu instanziieren.[109] Bei der Formulierung von Regeln kann innerhalb von QVT auf die Möglichkeiten von OCL zurückgegriffen werden. Für den Ansatz dieser Arbeit wird die QVT-R eingesetzt, weil sie die Konsistenz von Ausgangs- und Zielmodellen sicherstellt, was die Qualität der Verfolgbarkeitsbeziehung gegenüber QVT-OM verbessern hilft.

Mit QVT-R werden Abbildungsregeln zwischen Elementen der Ausgangs- und Zielmodelle definiert. Die QVT-Ausführungsumgebung trifft selbständig die Entscheidung, in welcher

[105] Abbildung aus [OMG MDA, 2003b], S. 3-12.
[106] Vgl. [OMG MDA, 2003b], S. 3-7 f.
[107] Vgl. Version 1.0 des Standards in [OMG QVT, 2008], Version 1.1 (Beta 2) in [OMG QVT, 2009] und Version 1.1 in [OMG QVT, 2011].
[108] [Nolte, 2009], S. 20.
[109] Von den im MDA-Guide ([OMG MDA, 2003b])prinzipiell dargestellten Alternativen zur Transformationen ist für QVT ist die Modelltypabbildung der dominierende Ansatz. Die Markierungstechnik kann ebenfalls eingesetzt werden.

Reihenfolge diese Regeln ausgeführt werden müssen und sorgt durch Änderung des Ausgangs- und/oder Zielmodells dafür, dass die Modellelemente (entsprechend einer vorgegebenen Transformationsrichtung) den in der Abbildungsregel definierten Bedingungen genügen. Eine Regel könnte beispielsweise definieren, dass sich der Name des Zielmodellelements aus Bezeichnung und Typ des Ausgangselements zusammensetzt. In diesem Fall würde die Ausführungsumgebung bei Anwendung der Regel prüfen, ob der Name des Zielmodellelements dieser Bedingung genügt. Wenn nicht, weil sich beispielsweise seit der letzten Transformation Typ oder Bezeichnung des Ausgangselementes geändert haben, wird der Name des Zielmodellelementes entsprechend aktualisiert.

Die MDA legt in ihren Konzepten fest, dass die sich während der Transformationen ergebenden Informationen aufgezeichnet werden sollen, damit die Verfolgbarkeit der Transformation des Ausgangsmodells in das Zielmodell gewährleistet ist.[110] Der QVT-Standard setzt diese Anforderung um, indem Verfolgbarkeitsinformationen während der Ausführung von Transformationen aufgezeichnet werden. Dazu unterscheidet die QVT-Spezifikation zwei Arten von Verfolgbarkeitsinformationen:[111]

- Verfolgbarkeitsklasse (Trace Class): Eine MOF-konforme Klasse, deren Attribute sich auf Objekte und Werte in einer Transformation beziehen. Für die Verfolgbarkeit von Transformationen mit der in dieser Arbeit verwendeten Relation Language entspricht eine Verfolgbarkeitsklasse jeweils einer Relation. Ihre Attribute repräsentieren die Domänen der Relation.

- Verfolgbarkeitsinstanz (Trace Instance): Eine Instanz einer Verfolgbarkeitsklasse, die während der Ausführung einer Transformation erzeugt wird. Sie belegt die Attribute mit konkreten Werteausprägungen, wodurch eine Beziehung zwischen den transformierten Modellelementen hergestellt wird.

Der QVT-Standard unterstützt mit der Aufzeichnung von Informationen in Form von Verfolgbarkeitsklassen und deren Instanzen das inkrementelle Update von Modellteilen des Zielmodells nach Änderung des Ausgangsmodells. Für die Visualisierung von Verfolgbarkeitsinformationen und deren Nutzung durch Bearbeiter macht der Standard keine Vorgaben.

Um QVT-R praktisch einsetzen zu können, ist das auf Eclipse basierende Werkzeug *medini™ QVT*[112] frei verfügbar und wird im Rahmen dieser Arbeit verwendet.[113] Für die praktische UML-Modellierung werden die UML-Werkzeuge des *Eclipse Modeling Project* und das frei verfügbare Werkzeug *TOPCASED* für Eclipse verwendet.[114] Die Modellierung mit

---

[110] Vgl. u.a. [OMG MDA, 2003], S. 3-1; [Kleppe et al., 2003], S. 75f.; [Frankel, 2003], 209f.

[111] Vgl. hier und im Folgenden [OMG QVT, 2008], S. 5.

[112] Die Homepage des Werkzeugs ist erreichbar unter: http://projects.ikv.de/qvt (letzter Aufruf: 19.08.2011).

[113] Es gibt auch Komponenten des M2M Eclipse-Projektes, die QVT-R als Eclipse-Erweiterung implementieren (Hompage des Projektes unter http://www.eclipse.org/m2m/, letzter Aufruf: 21.08.2011). Sie sind zum Zeitpunkt der Erstellung dieser Arbeit in einer Entwicklungsversion verfügbar. Eine weitere Alternative ist das kommerzielle Produkt *ModelMorf* des *Tata Research Development and Design Centre (TRDDC)* auf dessen Homepage eine Beta-Version des Produktes unter http://www.tcs-trddc.com/trddc_website/scripts/project_detail.php?lab=SWRD&project_id=44 (letzter Aufruf: 21.08.2011) heruntergeladen werden kann. Das Werkzeug ModelMorf verfügt über keine Eclipse-Integration und ist seit April 2009 nicht weiterentwickelt worden. Da mediniQVT ein freies Produkt mit aktuellem Releases-Plan und aktivem Support ist, wurde es für die prototypische Implementierung im Rahmen dieser Arbeit verwendet.

[114] TOPCASED wurde als Werkzeug gewählt, weil es direkt auf den Erweiterungen des Eclipse Modeling Project aufsetzt und unmittelbar in Eclipse integriert ist. Deshalb ist es für einen konsequent an den OMG-Standards ausgerichteten MDA-Ansatz innerhalb von Eclipse besser geeignet, als andere Werkzeuge, die proprietäre Implementierungen anstelle der OMG-Standards enthalten oder Inhalte in einem Repository oder Format speichern, das von außen nicht zugänglich ist (z.B. Omondo, ObjectiF für eclipse).

BPMN-Version 1.2 wird durch das frei verfügbare Werkzeug *BPMN Modeler*, basierend auf der Eclipse-Plattform (Teil des Eclipse-Projektes *SOA Tools*), unterstützt. Die entsprechende Werkzeugunterstützung für die BPMN-Version 2.0 befindet sich momentan in Entwicklung.

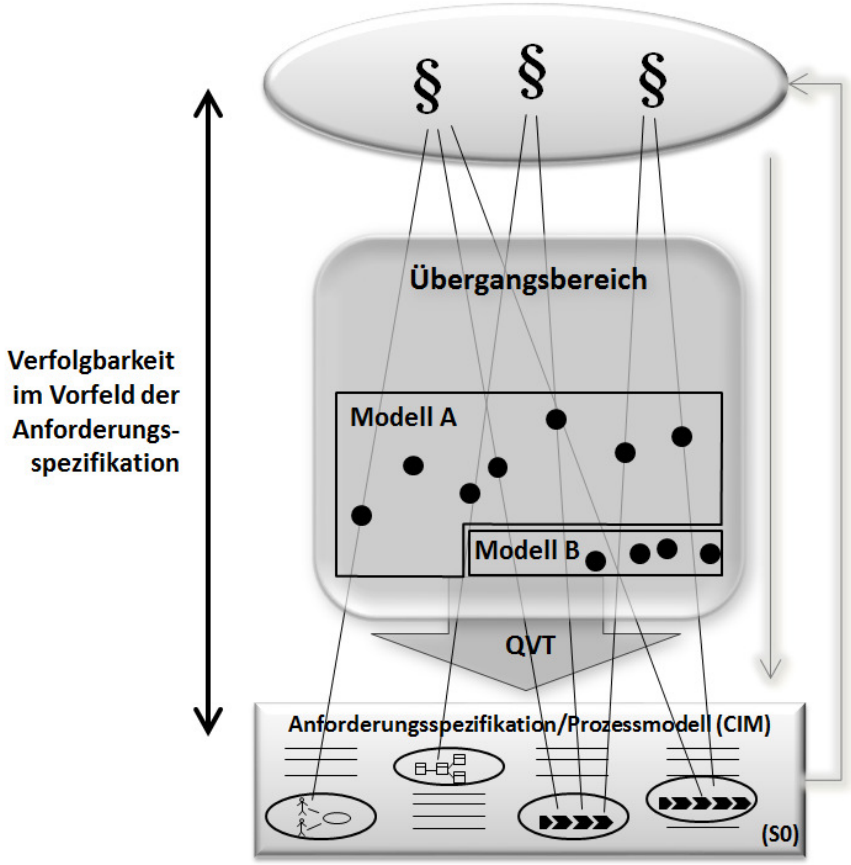


Abbildung 10 – Ausgestaltung des Übergangs im Vorfeld der Anforderungsspezifikation mit der MDA

Die dargestellten Konzepte der MDA machen deutlich, dass Elemente im Übergangsbereich zwischen den Ursprüngen von Anforderungen und fixierten Anforderungen benötigt werden, deren Granularität Anschluss an die Elemente von UML-Modellen der Anforderungsanalyse und an die Elemente von Geschäftsprozessmodellen (als die relevanten CIM-Vertreter) bieten muss. Wenn aus den Ursprüngen von Anforderungen Modelle im Sinne der MDA erstellt werden können, die das Vorfeld der Anforderungsspezifikation formalisieren (Modell A in Abbildung 10), ohne die Ursprünge aus ihrem Kontext in den Texten von Rechtsvorschriften zu lösen, können diese als Ausgangsmodell einer hier favorisierten QVT-R Transformation dienen.

Im Rahmen dieser Arbeit wird aber keine „modellierungsgerechte Gesetzgebung" angestrebt, die analog zur eingangs kritisierten automationsgerechten Gesetzgebung der 1970er Jahre vom Gesetzgeber die Formulierung eindeutiger Anforderungen fordert. Obwohl der Charakter der Gesetzgebung sich ändert[115], würde eine solche Forderung an den gleichen Hindernissen wie ihre Vorgängerforderung scheitern. Vielmehr wird hier davon ausgegangen, dass im Vorfeld der Anforderungsspezifikation lediglich Ursprünge von Anforderungen existieren, die nicht die formalen Eigenschaften und den vollständigen Informationsgehalt von Anforderungen haben. Wenn eine Transformation eines Modells mit QVT-R aus dem Übergangsbereich in die Anforderungsspezifikation gelingen soll, müssen zusätzliche Informationen einfließen. Analog zum Einsatz eines Plattformmodells in PIM-PSM-Transformationen wird hier ein vergleichbares Modell als zusätzliches Ausgangsmodell

[115] [Burghart, 1996], S. 24.

benötigt (Modell B in Abbildung 10), das den Ursprüngen von Anforderungen als Rahmen bei der Bildung einer Anforderungsspezifikation dient.

Mit dieser Transformation können die Ausgangsmodelle in Zielmodelle des Typs CIM transformiert werden. Ein Zielmodell würde dann Prozesse mittels UML oder BPMN darstellen, das zweite Modell spezifiziert die Anforderungen an die unterstützende Informationstechnik mit Anwendungsfall-, Aktivitäts- und Klassendiagrammen der UML. Zwischen dem Ausgangs- und den Zielmodellen kann durch Anwendung von QVT die Verfolgbarkeit als sichergestellt betrachtet werden. Die durchgeführten Transformationen werden aufgezeichnet, so dass sie im Sinne der Verfolgbarkeit ausgewertet werden können. Für die Transformation werden die Metamodelle der beteiligten Modelle benötigt. Die Metamodelle für die BPMN- und UML-Zielmodelle sind durch MOF bereits definiert. Für das Ausgangsmodell muss im Rahmen dieser Arbeit eine geeignete Formalisierung der Syntax und Semantik im Sinne eines Metamodells erfolgen, um es im Rahmen von QVT verarbeiten zu können. Hierfür können aufgrund konvergierender Standards u.a. Ontologien eingesetzt werden.

2.3 Ontologien und Konvergenz mit Modellierungsstandards

Zur Wissensrepräsentation im Forschungsgebiet der Künstlichen Intelligenz haben sich Ontologien zur Repräsentation von semantischen Zusammenhängen zwischen den Konzepten und Dingen eines Anwendungsbereichs etabliert. Besonders häufig wird der Ontologie-Begriff in der Literatur als eine „explizite formale Spezifikation einer gemeinsamen Konzeptionalisierung“ definiert.[116] Unter einer Konzeptionalisierung (engl. conceptualization) wird die abstrakte Zusammenfassung von Typen von Dingen, Begriffen und Konzepten der realen Welt bzw. eines Ausschnittes der realen Welt unabhängig von einer konkreten Sprache oder Notation verstanden.[117] Die Konzeptionalisierung basiert auf Ontological Commitment, das als die allgemeine Akzeptanz zur kohärenten und konsistenten Nutzung des in der Ontologie abgebildeten Gegenstandsbereichs durch ein gemeinsames Vokabular verstanden wird.[118] Die explizite Spezifikation gibt dieser Konzeptionalisierung dann ihre konkrete Form.

In der Literatur ist die Klassifikation von Ontologien nach ihrem Umfang gebräuchlich.[119] Allgemeine, bereichsübergreifende *top level ontologies* beschreiben generelle Konzepte wie Zeit, Raum, Materie, Objekt und Aktion und sind damit unabhängig von einem konkreten Anwendungsbereich. Anwendungsbereichsspezifische *domain ontologies* und *task ontologies* umfassen Konzepte aus einem bestimmten Anwendungsbereich (domain), wie beispielsweise der öffentlichen Verwaltung oder einer bestimmten, generischen Aufgabe (task) wie dem Verkaufen. Anwendungsspezifische Ontologien (*application ontologies*) stellen die semantische und konzeptionelle Basis von Daten- und Klassenmodellen einer Anwendung dar und sind Spezialisierungen einer domain ontology und/oder task ontology.

---

[116] Deutsche Übersetzung der Definition von T. R. Gruber (vgl. [Gruber, 1993]) in [Hesse, 2002], S. 477.

[117] Die Spezifikation der Ontologie führt eine abstrakte Konzeptionalisierung mit (Symbolen) einer Sprache (oder auch mehrerer Sprachen) zusammen, indem die Ontologie diese Sprache zur Benennung von Begriffen/Konzepten und zur Beschreibung ihrer Bedeutung(en) nutzt. Das Ziel der Ontologie ist es, die Zahl der möglichen Bedeutungen für ein sprachliches Symbol idealer Weise auf genau eine zu reduzieren. Eine explizite Spezifikation nutzt zur Beschreibung nur solche Elemente, die zuvor ausdrücklich definiert wurden (vgl. [Gómez-Pérez et. al, 2004], S. 6).

[118] Die Definition erfolgt in Anlehnung an [Gruber, 1993], S. 200 f., wo sie im Kontext der Nutzung portabler Ontologien durch Agenten gegeben wird, und in Übereinstimmung mit [Gómez-Pérez et al., 2004], der diese Art der Definition als intuitiv bezeichnet („[…] an intuitive way what an ontological commitment is.“, [Gómez-Pérez et al., 2004], S. 36).

[119] Vgl. [Guarino, 1998], S. 9 f.

Ontologien können weiterhin hinsichtlich der Form ihrer Beschreibung in formale und informelle Ontologien unterschieden werden. Eine informelle Ontologie (auch leichtgewichtige Ontologie) ist ein Katalog von benannten Konzepten, die in der Regel in natürlicher Sprache beschrieben sind. Die Verwendung von definierten Strukturen bei der Beschreibung und die Dokumentation in einer maschinenlesbaren Form bilden den Übergang zu einer formalen Ontologie.[120] Obwohl eine Ontologie bereits als formal gelten kann, wenn sie die Sammlung von benannten Konzepten durch Typen von Beziehungen in eine teilweise Ordnung (z.B. durch Untertypen-Beziehung) bringt[121], wird von einer formalen Ontologie in der Regel erst gesprochen, wenn sie mit Sprachen aus dem Forschungsbereich der Künstlichen Intelligenz (z.B. ONTOLINGUA, LOOM, OCML, FLogic) oder speziellen Ontologie Markup-Sprachen (z.B. RDF(S), DAML+OIL, OWL) beschrieben wird.[122] Diese formalen Sprachen zur Beschreibung einer Ontologie sind außerhalb ihrer speziellen Forschungs- bzw. Einsatzgebiete (z.B. Künstliche Intelligenz, Wissensrepräsentation) im Vergleich zu den in der Softwaretechnik akzeptierten Standards UML und OCL weniger verbreitet.[123] Ontologien lassen sich aber auch mit diesen Modellierungssprachen (und auch mit anderen, wie der ER-Modellierung[124]) beschreiben, obwohl sie hierfür bisher keine breite Akzeptanz gefunden haben. Für die Darstellung von Ontologien in MOF-konformen Modellierungsstandards wie UML (ergänzt durch OCL) wurden zahlreiche Ansätze entwickelt, die unterschiedliche Stärken und Schwächen hatten und in ihrer Gesamtheit vor allem den Bedarf an einem einheitlichen Modellierungsstandard deutlich werden ließen.[125] Die OMG hat deshalb[126] im Jahr 2003 eine Initiative begründet, die im Jahr 2009 das *Ontology Definition Metamodel* (ODM) als Standard in Version 1.0[127] verabschiedete. Dieses Metamodell ermöglicht die Abbildung von RDFS- und OWL-Ontologien auf andere MOF-konforme Modellierungssprachen, indem es MOF-konforme Meta-Modelle für beide Ontologie-Sprachen definiert. Weiterhin definiert es UML-Profile für beide Ontologie-Sprachen, so dass diese mit vorhandenen Modellierungswerkzeugen der Softwaretechnik bearbeitet werden können. Durch diese standardisierte Verbindung zwischen Ontologien und Modellierung ergibt sich Forschungsbedarf hinsichtlich der neuen Nutzungsmöglichkeiten.[128]

---

120 Vgl. [Gómez-Pérez et. al, 2004], S. 6.
121 Vgl. [Gómez-Pérez et. al, 2004], S. 9.
122 Vgl. [Gómez-Pérez et. al, 2004], S. 25.
123 Vgl. [Kogut et al., 2002], S. 61.
124 Vgl. [Gómez-Pérez et. al, 2004], S. 23.
125 Bekannte Ansätze sind u.a. dokumentiert in [Parreiras et. al, 2007], [Cranefield, 2001], [Baclawski et al., 2002a], [Baclawski et al., 2002b], [Falkovych et al., 2003], [Brockmans et al., 2004], [Djuric et al., 2004], [Djuric et al., 2005] und implementiert in den Werkzeugen Visual Ontology Modeler (http://www.sandsoft.com/products.html, letzter Aufruf: 21.08.2011), Protégé, (http://protege.stanford.edu, letzter Aufruf: 21.08.2011), DUET (DAML UML Enhanced Tool, http://codip.grci.com/wwwlibrary/wwwlibrary/DUET_Docs, letzter Aufruf: 18.02.2011, im August 2011 nicht mehr erreichbar) und Xpetal (http://www.langdale.com.au/styler/xpetal/, letzter Aufruf: 21.08.2011). Einen vergleichenden Überblick über die relevanten Ansätze gibt [Gaevic et al., 2006], S. 145-172.
126 Vgl. [Gaevic et al., 2006], S. 172.
127 [OMG ODM, 2009]
128 Über die Frage hinausgehend, wie Ontologien mit Techniken und Werkzeugen der Softwareentwicklung modellhaft dargestellt werden können, sind speziell im Kontext der modellgetriebenen Architektur und/oder modellgetriebenen Softwareentwicklung wenige aktuelle Ansätze in der Literatur dokumentiert. Einen Ansatz zur Transformation von Ontologie-Axiomen in OCL-Constraints für UML-Klassendiagramme, der als MDA-basiert bezeichnet wird, ist in [Kalibatiene et al., 2009] beschrieben. Schneider nutzt (vgl. [Schneider, 2010a], [Schneider, 2010b]) OWL-Ontologien, um ECORE-Modelle innerhalb von Eclipse zu annotieren, um dem Entwickler weitergehende Möglichkeiten (z.B. zur Definition von OWL Property Restrictions für Attribute oder SPARQL-Abfragen für Operationen) zu bieten. Um die Wiederverwendung MOF-basierter Modelle zu erleichtern, nutzen Wolter et al. (in [Wolter et al., 2010]) eine Ontologie, mit der diese Modelle annotiert werden können. Anschließend erläutern Sie anhand ihrer durchgeführten Experimente, wie mit Standardsoftware (z.B. Protegé) die Auswertung der Ontologien und Modelle möglich ist. Im

Beispielsweise können durch die Abbildung von Konzepten der Ontologie Markup-Sprachen RDF(S) und OWL auf MOF die formalisierten Konzepte und Individuen solcher Ontologien als Instanzen des ODM modelliert und mit QVT genutzt werden.

Es sind in der Literatur verschiedene Ontologien dokumentiert, die sehr allgemeine Konzepte formalisieren und deshalb als *Upper Ontologies* bezeichnet werden (z.B. *OpenCyc*[129], *DOLCE*[130], *Suggested Upper Merged Ontology*[131], *Universal Core*[132]). Sie beschreiben mit ihren Konzepten die physische und/oder mathematische Welt und eignen sich nicht für die Beschreibung sozialer Systeme.[133] Es existieren darüber hinaus auch Ontologien, die speziell Konzepte aus dem Bereich der allgemeinen Rechtsanwendung formalisieren, wie sie auch dem Verwaltungshandeln zugrunde liegen.[134] Beispiele derartiger Ontologien sind die von McCarty entwickelte *language for legal discourse*[135], die von Stamper entwickelte *Logic of Norms and Affordances (NORMA)*[136], die von Valente et al. entwickelte *Functional Ontology of Law*[137], die von van Kralingen et al. entwickelte *Conceptual Frame-based Ontology for the Law*[138] und die von Visser darauf aufbauend erarbeitete Formalisierung[139]. Sie formalisieren abstrakte Konzepte wie die Rechtsnorm oder das Subjekt, an das sich eine Rechtsnorm richtet, sind aber primär auf die Verwendung in wissensbasierten Rechtsinformationssystemen ausgerichtet und nicht für Annotationen im Rahmen des Semantic Web gedacht.[140] Die *LRI-Core ontology of law*[141] des *Leibniz Centers for Law*[142] und die im Rahmen des *ESTRELLA*-Projektes[143] (weiter-)entwickelte *Core ontology of basic legal concepts*[144], auf der auch das *Legal Knowledge Interchange Format (LKIF)*[145] basiert, sind Beispiele für Ontologien, die einerseits abstrakte Konzepte des Rechts formalisieren und andererseits deren Konkretisierung an so

---

ServCASE-Projekt (Homepage ist erreichbar unter: http://servcase.informatik.uni-leipzig.de/opencms/export/sites/servcase/servcase/, letzter Aufruf: 21.08.2011) wurde ein Ansatz entwickelt, mit dem aus gesprochener Konversation und geschriebenen Texten der frühen Phasen eines kreativen Entwicklungsprozesses mit automatischen Methoden semantische Repräsentationen (als RDF-Datei) erzeugt werden können (vgl. [Raether, 2004], S. 100f.). Diese Repräsentationen können bearbeitet werden. Um sie anschließend mit einem Konverter in das Geschäftsprozessmanagement-Werkzeug ARIS überführen zu können, müssen sie durch Typisierung formalisiert werden (vgl. [Cyriaks&Köhler, 2008], S. 193). Die Autoren schätzen ein, dass prinzipiell auch die Generierung von UML oder BPEL (Business Process Execution Language) möglich sei, was aber im Projekt nicht vertieft wurde (vgl. [Cyriaks&Köhler, 2008], S. 192).

129 Die Homepage des Projektes ist erreichbar unter http://sw.opencyc.org/ (letzter Aufruf: 21.08.2011).

130 Die Hompage des DOLCE-Projektes ist erreichbar unter http://www.loa-cnr.it/DOLCE.html (letzter Aufruf: 21.08.2011).

131 Die Hompage des Projektes ist erreichbar unter http://www.ontologyportal.org/ (letzter Aufruf: 21.08.2011).

132 [Smith et al., 2009] (Die Homepage des Projektes ist unter http://ucore.gov/ ist im Februar 2011 nicht mehr erreichbar.)

133 Vgl. [Breuker et al., 2002], S. 75.

134 Einen Überblick über die Ontologien aus dem Bereich des Rechts zur Anwendung in wissensbasierten Systemen gibt [Casellas, 2008], S. 113ff. Darüber hinausgehende

135 Vgl. [MacCarty, 1989], [MacCarty, 2002].

136 Vgl. [Stamper, 1991], [Stamper, 1996].

137 Vgl. [Valente&Breuker, 1994].

138 Vgl. [Kralingen et al., 1996].

139 Vgl.[Visser&Bench-Capon, 1996], [Visser&Bench-Capon, 1997], [Visser&Bench-Capon, 1998].

140 Vgl. [Breuker et al., 2002], S. 74.

141 Vgl. [Breuker&Hoekstra, 2004a], [Breuker&Hoekstra, 2004b].

142 Homepage des Leibniz Center of Law an der Universität von Amsterdam unter: http://www.leibnizcenter.org/ (letzter Aufruf: 21.08.2011).

143 Hompage des ESTRELLA-Projektes unter: http://www.estrellaproject.org/ (letzter Aufruf: 21.08.2011).

144 Vgl. u.a. [Hoekstra et al., 2007], [Breuker et al., 2007].

145 Das Legal Knowledge Interchange Format (LKIF) dient zur Repräsentation einer rechtlichen Wissensbasis (vgl. [Boer et al., 2006]) im OWL-Format (siehe unten, Abschnitt 2.4).

genannten „Ankerpunkten“ durch konkrete domänenspezifische Ontologien erlauben.[146] Die in ihrer Konkretisierung erstellten Ontologien lassen sich, wie es von den Autoren anhand einer Ontologie des niederländischen Strafrechts gezeigt wird, zur Annotation von Rechtstexten im Sinne des Semantic Web verwenden. Sie können für diese Arbeit einen Orientierungsrahmen im Sinne einer Checkliste bilden und die Wahrnehmung verschiedener Perspektiven auf die formalisierten Konzepte ermöglichen.[147] Casellas et al. sehen aktuell einen Trend, dass neuere Ontologien im Bereich des Rechts zunehmend domänen- oder anwendungsspezifisch sind, womit sie den Bezug zu einer konkreten Rechtsvorschrift oder einem abgegrenzten Bereich von Vorschriften haben. Es gibt jedoch keine domänen- oder anwendungsspezifischen Ontologien, die für den Ansatz dieser Arbeit verwendet werden können.[148] Allerdings ist innerhalb der domänenspezifischen Ansätze die von Steinmann und Nejdl formulierte These, dass Gesetzestexte bereits selbst eine spezielle Form einer Ontologie darstellen, von besonderer Bedeutung.[149] In Abhängigkeit vom Grad der Allgemeingültigkeit des zugrunde gelegten Gesetzes können anwendungsbereichsspezifische oder auch anwendungsbereichsübergreifende Konzepte im Text identifiziert werden. In der Literatur sind verschiedene Ansätze dokumentiert, die dieses Grundprinzip auf konkrete Gesetze (z.B. die Straßenverkehrsordnung[150] oder das österreichischen Fachhochschulstudiengesetz[151]), auf zusammengehörige Themengebiete (z.B. das Themengebiet Kompensationsausgleich bei Eingriffen in Natur und Landschaft gemäß Bundesnaturschutzgesetz, nordrheinwestfälischem Landschaftsgesetz und Verfahrensvorschriften[152]) oder auf ganze Rechtssysteme anwenden (z.B. auf das französische Rechtssystems[153] oder die Rechtsvorschriften der EU[154]).

Die direkte Wiederverwendung der vorliegenden Ontologien des Rechts im Rahmen dieser Arbeit wird erschwert, weil sie statische Zusammenhänge zwischen den Rechtskonzepten formalisieren und Aspekte der methodischen Anwendung von Recht (wie sie Kern des Verwaltungshandelns sind, vgl. Abschnitt 2.5) nicht weitergehend berücksichtigen. Da in dieser Arbeit die intensive Unterstützung des Verwaltungshandelns durch Informationstechnik im Rahmen des E-Governments den Gegenstand bildet, stehen hier die im Rahmen des Verwaltungshandelns zu erledigenden Aufgaben im Vordergrund. Deshalb muss die für den Ansatz benötigte Ontologie auch eine Formalisierung von Konzepten der Rechtsanwendung umfassen.

Neben der Formalisierung von Elementen des Rechts gibt es vergleichsweise wenige Ansätze, die Verwaltungskonzepte in Form von Ontologien formalisieren. Das Ontologie-

---

[146] In [Breuker et al., 2002] wird am Beispiel des niederländischen Strafrechts dargestellt, wie die Core Ontology of Law konkretisiert werden kann. Die im Rahmen dieser Arbeit formalisierten Konzepte lassen sich in vergleichbarer Weise als Konkretisierung der Core Ontology of Law auffassen.

[147] Vgl. [Breuker et al., 2002], S. 77.

[148] Als Beispiele für diesen von Casellas geschilderten Trend (vgl. [Casellas et al., 2010]) können Ontologien aus dem Bereich des digitalen Rechte-Managements (z.B. Copyright Ontology in [Carcia, 2005]), zur Unterstützung der professionellen Entscheidungsfindung (Ontology of Professional Judicial Knowledge in [Casellas, 2008]), zur Formalisierung von fallbasiertem Wissen (Legal Case OWL Ontology in [Wyner&Hoekstra, 2010]) oder zum EU-Recht (z.B. Transposition der EU-Dienstleistungsrichtlinie in [Liebwald, 2009], Ontologie des EU-Konsularrechts in [Schweighofer, 2010]) gelten.

[149] Steinmann und Nejdl führen in [Steinmann&Nejdl, 2004] als Beispiel das Bürgerliche Gesetzbuch (BGB) an, in dessen „Allgemeinem Teil einige Grundbegriffe und damit die Dinge, die für die nachfolgenden Gesetze als existent vorausgesetzt werden“ ([Steinmann&Nejdl, 2004], S. 5) beschrieben sind. Sätze in derartigen Gesetzen „tragen klar ontologische Züge; sie bilden zusammen mit den restlichen Paragraphen ein Modell […] im Anwendungsbereich“ ([Steinmann&Nejdl, 2004], S. 5).

[150] [Kuhn, 2000]

[151] [Hausenblas, 2004], S. 85ff.

[152] [Lutz, 2001]

[153] [Lame, 2000]

[154] [Despres&Szulman, 2004]

Framework für die integrierte öffentliche Leistungserbringung, das von Overbeek et al. entwickelt wurde, unterstützt die Zusammenarbeit zwischen Verwaltungen, indem es aus der Prozessmodellierung bekannte Konzepte formalisiert.[155] Das Projekt *OntoGov*, das von der europäischen Union im Rahmen des Forschungsprogramms *Information Society Technologies (IST)* von 2004 bis 2006 durchgeführt wurde, definierte u.a Ontologien für die Modellierung und Entwicklung von E-Government-Services.[156] Unterschieden wurden Meta-Ontologien, domänenorientierte Ontologien und Administrationsontologien.[157] Die Meta-Ontologien definieren die Sprache, mit der E-Government-Services modelliert werden. Die domänenorientierten Ontologien dienen zur Modellierung konkreter E-Government-Services und der von ihnen benötigten Daten. Die Administrationsontologie definiert die automatische Umsetzung von E-Government-Services und die Überwachung ihrer Ausführung. Von den im Projekt definierten Ontologien hat die Legal Ontology (eine der Meta-Ontologien) einen Bezug zu Rechtsvorschriften. Sie spiegelt die formale Struktur der Dokumente mit Paragraphen, Absätzen usw. wider. Dadurch wird es möglich, die Auswirkung von Änderungen an Gesetzestexten auf E-Government-Services zu verfolgen. Der Ansatz des OntoGov-Projektes ist eine Form von nachträglicher Dokumentation, wird nicht im Vorfeld der Anforderungsspezifikation angewendet und berücksichtigt primär die formale Struktur von Rechtsvorschriften, weniger die Semantik der in ihr enthaltenen fachlichen Elemente. Dennoch zeigt er, dass die Verwendung von Ontologien geeignet ist, um die Verfolgbarkeit von Anforderungen zu verbessern.[158] Das Projekt *SemanticGov*[159], das ebenfalls im Forschungsprogramm *Information Society Technologies (IST)* von 2006 bis 2009 durchgeführt wurde, entwickelte u.a. für die gesamte öffentliche Verwaltung eine Ontologie (*Overall Public Administration Domain Ontology*), mit deren Hilfe die semantische Interoperabilität bei der Definition von Diensten zur Erbringung von Verwaltungsleistungen innerhalb von E-Government-Systemen sichergestellt werden soll. Entsprechend formalisiert diese Ontologie einerseits dienstorientierte Konzepte wie Serviceinput, Serviceoutput, Serviceoutcome, Servicevorbedingung und Beteiligtenrollen. Andererseits formalisiert sie auch universelle Konzepte wie Gesetz und Ort.[160] In konkreten Anwendungsszenarien sollen diese Konzepte dann bezogen auf konkrete Verwaltungsleistungen weiter verfeinert werden, was in drei Beispielontologien illustriert wird.[161] Aber insbesondere die generischen Konzepte weisen aufgrund dieser Ausrichtung der Ontologie ein hohes Abstraktionsniveau auf, so das sie für die Annotation von Rechtsvorschriften nicht verwendet werden können. Sie können allerdings als Orientierungsrahmen für diese Arbeit herangezogen werden.

In Großbritannien wurden im Rahmen der Entwicklung des Zuständigkeitsfinders *Electronic Service Delivery* (ESD)[162] auch mehrere Ontologien entwickelt, die einerseits verschiedene Aspekte des Verwaltungshandelns und von Verwaltungsleistungen formalisieren (z.B. die Befugnisse und Pflichten der Verwaltungsträger, Lebenslagen und Bedürfnisse der Bürger) und andererseits das Vokabular der Verwaltung in ihrem Handeln definieren. Von besonderer Relevanz für die vorliegende Arbeit sind die ESD-Standards *Generic Process List* und die *Interaction List*. Die Generic Process List definiert generische Prozesse der

[155] Vgl. [Overbeek et al., 2009].
[156] Vgl. [Thönssen, 2004].
[157] Vgl. [Stojanovic et al., 2006], S. 79 f.
[158] Freundliche Auskunft von OntoGov-Projektmitarbeiterin Barbara Thönssen per E-Mail am 20.04.2011. (Die Homepage des Projektes im April 2011 nicht mehr erreichbar. Aus den verfügbaren Projektergebnissen ist nicht direkt ersichtlich, wie die Ontologie eingesetzt und die Verbindung zu Rechtsvorschriften genutzt wurde.)
[159] Die Homepage des Projektes ist verfügbar unter: http://www.semantic-gov.org (letzter Aufruf: 21.08.2011).
[160] Die Ontologie ist unter den folgenden Andressen verfügbar: http://www.semantic-gov.org/Ontologies/GEA/GEA_Ontology_17.wsml und http://www.semantic-gov.org/Ontologies/GeneralKnowledge/GeneralKnowledge_Ontology.wsml (letzter Aufruf: 21.08.2011).
[161] [SemanticGov, 2007]
[162] Die Homepage des ESD-Projekts ist erreichbar unter: http://www.esd.org.uk/ (letzter Aufruf: 21.08.2011).

Verwaltung als Konzepte einer Ontologie und formalisiert diese in RDF. Beispielsweise sind als generische Prozesse die *Interpretation der Rechtssprechung*[163], die *Unterstützung eines Bürgers durch Hinweise oder Informationen*[164], die *Bereitstellung von Informationen über Verwaltungsleistungen*[165] und *der Abschluss bzw. die Kommentierung einer Fallbearbeitung oder einer Anfrage*[166] identifiziert, verbal beschrieben und als RDF-Ressource abrufbar. In der Interaction List sind Typen möglicher Interaktionen (z.B. die *Beantragung einer Ausnahmeregelung*[167], sich *Ändernde Umstände*[168] oder die *Bezahlung von Waren und Dienstleistungen*[169] formalisiert, verbal beschrieben und zusätzlich als RDF-Ressource abrufbar. Es fehlt bei den Prozessen und den Interaktionen eine weitergehende Beschreibung, beispielsweise ihres Ablaufs, der benötigten Informationen, der produzierten Ergebnisse und der nachfolgenden Prozesse, so dass diese Aspekte des Verwaltungshandelns nicht enthalten sind. In den Vereinigten Staaten wurden im Rahmen der *U.S. Federal Enterprise Architecture (FEA)* vergleichbare Ontologien definiert, die Konzepte der Verwaltung und die ausgetauschten Daten formalisieren.[170]

Die vorgestellten Ontologien aus dem Bereich der öffentlichen Verwaltung zeigen Beispiele für die Formalisierung von Verwaltungskonzepten und Verwaltungshandeln. Sie machen deutlich, dass das Verwaltungshandeln über ausschließliche Anwendung von Rechtsnormen hinausgeht. Andererseits vernachlässigen die Ontologien Inhalte, wie sie in den Ontologien des Rechts formalisiert sind. Wenn im Rahmen dieser Arbeit davon ausgegangen wird, dass Verwaltungshandeln die Anwendung des in Vorschriften dokumentierten Rechts darstellt, muss eine Ontologie des Verwaltungshandeln auch Konzepte des Rechts und der Rechtsanwendung explizit berücksichtigen. Andernfalls könnte beispielsweise nicht sichergestellt werden, dass die aus Sicht der Rechtsanwendung in den Vorschriften relevanten Elemente innerhalb des Verwaltungshandelns zum Einsatz kommen. Es ist folglich für den Ansatz dieser Arbeit eine Ontologie zu verwenden, die Verwaltungshandeln mit der Rechtsanwendung verbindet, es aber nicht darauf reduziert. Eine solche Ontologie existiert zum Zeitpunkt der Erstellung dieser Arbeit nicht und muss deshalb hier entwickelt werden.

[163] Process "Interpret legislation" resource URI: http://id.esd.org.uk/process/47 (letzter Aufruf: 21.08.2011).
[164] Process "Respond to a customers request for information or advice" resource URI: http://id.esd.org.uk/process/38 (letzter Aufruf: 21.08.2011).
[165] Process "Provide information on services" resource URI: http://id.esd.org.uk/process/22
[166] Process "Complete or annotate a service request or case file record" resource URI http://id.esd.org.uk/process/57 (letzter Aufruf: 21.08.2011).
[167] Interaction "Application for exemption" resource URI: http://id.esd.org.uk/interaction/30 (letzter Aufruf: 21.08.2011).
[168] Interaction "Changing circumstances" resource URI: http://id.esd.org.uk/interaction/11 (letzter Aufruf: 21.08.2011).
[169] Interaction "Paying for goods and services" resource URI: http://id.esd.org.uk/interaction/4 (letzter Aufruf: 21.08.2011).
[170] Informationen zur FEA können abgerufen werden unter: http://www.whitehouse.gov/omb/e-gov/fea (letzter Aufruf: 21.08.2011).

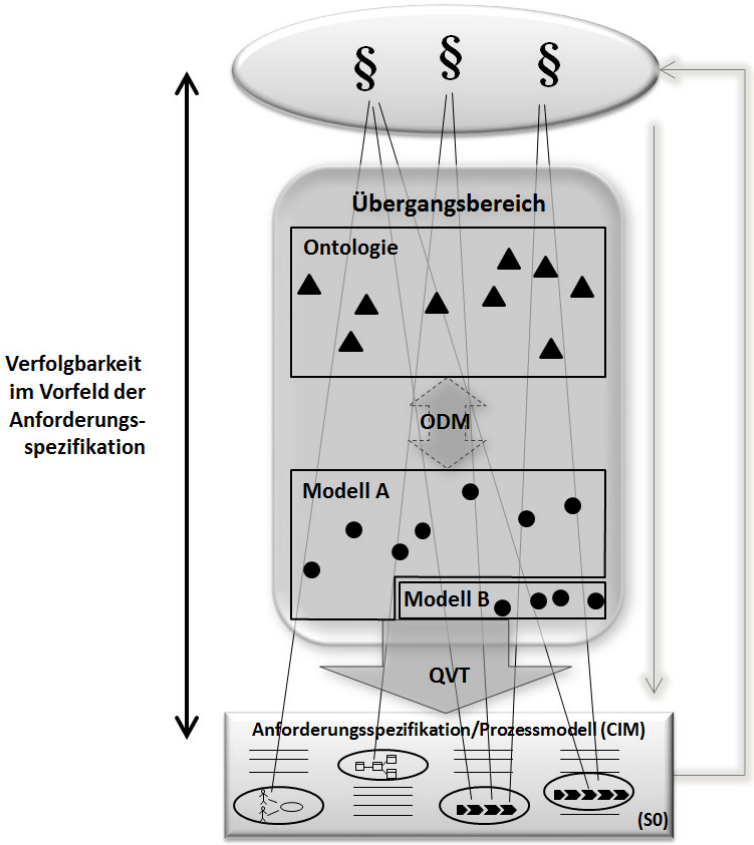


Abbildung 11 – Ausgestaltung des Übergangs im Vorfeld der Anforderungsspezifikation mit MDA/Ontologie

Diese Ontologie wird benutzt, um die Semantik von Konzepten innerhalb des Übergangsbereichs zwischen Ursprüngen von Anforderungen und fixierten Anforderungen zu definieren. Durch diese domänenspezifische Semantik und eine geeignete Granularität der Konzepte wird eine wesentliche Voraussetzung geschaffen, um die Verfolgbarkeitsbeziehungen effektiv einsetzen zu können. Mit Hilfe des ODM ist eine Abbildung dieser Elemente auf Modellelemente möglich, die innerhalb des Übergangsbereichs existieren.

2.4 Semantic Web

Als *Semantic Web* wird im Rahmen dieser Arbeit die Aufbereitung, Bereitstellung und Verknüpfung der Informationen innerhalb des World Wide Web (WWW) in einer Form verstanden, die ihre maschinelle Verarbeitung erlaubt.[171] Dazu werden die für menschliche Nutzer durch *HTML (HyperText Markup Language)*[172] und zunehmend *XHTML (eXtensible HyperText Markup Language)*[173] präsentierten Informationen auf Webseiten erweitert, so dass der Information eine wohldefinierte Bedeutung (Semantik) zugeordnet wird, die Computern die Verarbeitung dieser Semantik und der mit ihr verbundenen Inhalte ermöglicht. Zu den grundlegenden Technologien des Semantic Web gehören die Sprache *Extensible Markup Language* (*XML*) und die Sprache *Resource Description Framework* (*RDF*)[174].

[171] Das W3C gibt in den *W3C Semantic Web Frequently Asked Questions* (http://www.w3.org/2001/sw/SW-FAQ, letzter Aufruf: 21.08.2011) an, dass es zum Begriff des Semantic Web keine einheitliche, formale Begriffsdefinition gibt. Deshalb orientiert sich die Definition in dieser Arbeit an [Brenders-Lee et al., 2001], [Shadbolt et al., 2006] und [Hitzler et al., 2008].
[172] [W3C HTML, 1999], [W3C HTML, 2011]
[173] [W3C XHTML, 2002], [W3C XHTML, 2010]
[174] [W3C RDF, 2004a], [W3C RDF, 2004b], [W3C RDF, 2004c]

XML ist eine allgemeine Auszeichnungssprache (engl. Markup Language), durch die Informationen hierarchisch strukturiert als Textdokument in maschinell verarbeitbarer Form repräsentiert werden. Dazu dienen Markierungen (Auszeichnungen), denen abhängig von der konkreten XML-Sprache eine bestimmte Semantik zukommt. RDF definiert beispielsweise für die verwendeten Markierungen diese Semantik, so dass ein entsprechendes Vokabular entsteht. Auch XHTML ist ein Beispiel für ein solches Vokabular von Markierungen. Dabei handelt es sich um einen Standard des W3C, der im WWW zunehmend zur Präsentation von Informationen verwendet wird. Ein weiteres Beispiel einer XML-Sprache ist XMI (XML Metadata Interchange), ein OMG-Standard zum Austausch von MOF-basierten Modellen zwischen Softwareentwicklungswerkzeugen (vgl. Abschnitt 2.2). Die Familie der XML-Standards umfasst weitere Standards, von denen für den Ansatz dieser Arbeit die *Extensible Stylesheet Language Transformations (XSLT)*[175] besondere Relevanz hat. Mit XSLT kann die Transformation von XML-Dokumenten in andere XML-Dokumente in Form von XSLT-Stylesheets definiert und mit XSLT-Prozessoren ausgeführt werden.

Mit RDF können Informationen über identifizierbare Ressourcen in Form von Markierungen im WWW repräsentiert und verarbeitet werden. Dazu nutzt sie eindeutige Bezeichner im URI-Format[176] um Ressourcen identifizierbar zu machen. Die Informationen über diese Ressourcen werden in Form von Tripeln, bestehend aus einem Subjekt, einem Prädikat und einem Objekt, repräsentiert. Das Subjekt ist die Ressource, auf die sich die Information bezieht, d.h. über die eine Aussage durch Prädikat und Objekt getroffen wird. Dieser grundlegende Mechanismus wird durch *RDF-Schema* (RDFS)[177] und die *Web Ontology Language* (OWL)[178] erweitert. Mit RDFS können im Sinne einer Ontologie „Aussagen über die semantischen Beziehungen der Termini eines beliebigen nutzerdefinierten Vokabulars“[179] gemacht werden. Allerdings ist RDFS hinsichtlich seiner Modellierungsfähigkeit eingeschränkt.[180] Diese Einschränkungen werden durch OWL aufgehoben, die auf RDFS basiert und weitere Konstrukte einführt, um maschinelle Schlussfolgerungen aus den Informationen zu ermöglichen. So finden die Methoden und Techniken der Wissensrepräsentation aus dem Forschungsbereich der Künstlichen Intelligenz im Semantic Web einen Anwendungsbereich.[181] Dadurch ist es möglich, mit RDFS und OWL im Rahmen dieser Arbeit eine Ontologie des Verwaltungshandels zu formalisieren und, basierend auf den Konzepten der Ontologie, Aussagen über Ressourcen und Individuen zu treffen.

Damit Informationen im WWW maschinell verarbeitet werden können, müssen zum einen semantische Markierungen innerhalb der Webseiten und zum anderen ein definiertes Vokabular in Form einer Ontologie vorhanden sein, die die Interpretation und Verarbeitung dieser Markierungen ermöglichen. Weil das Semantic Web kein eigenständiges Netz ist, sondern das vorhandene WWW erweitert, müssen diese Markierungen zu vorhandenen Inhalten hinzugefügt werden.[182] Dies kann während der Erstellung einer Webseite erfolgen, indem die Markierungen in das HTML oder XHTML direkt eingefügt werden. Es ist auch möglich, die Markierungen zu einem späteren Zeitpunkt hinzuzufügen und zusammen mit der Seite oder separat zu speichern. In beiden Fällen ist eine Werkzeugunterstützung für die

[175] [W3C XSLT, 2007]
[176] Ein Uniform Resource Identifier (URI) ist eine Zeichenkette, "die alle zur Identifikation notwendigen Informationen [...] kodiert" ([Meinel&Sack, 2004], S. 21). Man unterscheidet zwei Arten von URIs. Universal Resource Locators (URL) bezeichnen eine Adresse, die über eine Abbildung auf Netzwerkprotokollalgorithmen Zugang zu einer Informationsressource zu einem Zeitpunkt bietet" (vgl. [IETF URI, 1994], S. 1), während Uniform Resource Names (URNs) einen Namensraum zur dauerhaften und eindeutigen Identifikation von Informationsressourcen bieten (vgl. [Meinel&Sacks, 2004], S.731, [IETF URI, 1994] S. 17, [IETF URN, 2002], S. 2).
[177] [W3C RDFS, 2004]
[178] [W3C OWL, 2004]
[179] [Hitzler et al., 2008], S. 67.
[180] Vgl. [Hitzler et al., 2008], S. 198 f.
[181] Vgl. [Brenders-Lee et al., 2001].
[182] Vgl. [Brenders-Lee et al., 2001].

Erstellung der Markierungen hilfreich. Es gibt eine Reihe von Werkzeugen, mit denen Annotationen zu HTML-Dokumenten hinzugefügt werden können.

Unter einer Annotation wird hier eine semantische Hinzufügung bzw. Ergänzung verstanden, die es nicht nur einem Menschen, sondern auch einer Maschine ermöglicht, Inhalte von HTML-Dokumenten zu verarbeiten. Für die Verfolgbarkeit ist es notwendig, dieses Verständnis noch zu präzisieren: „An annotation A is a tuple (a_s, a_p, a_o, a_c), where a_s is the subject of the annotation (the annotated data) a_o is the object of the annotation (the annotating data) a_p is the predicate (the annotation relation) that defines the type of relationship between a_s and a_o, and a_c is the context in which the annotation is made."[183] Die Verfolgbarkeit einer Annotation ergibt sich, basierend auf dieser Definition, durch den Annotationskontext (a_c), der Bestandteil des Tupels A ist. Eine Annotation ist entsprechend dieser Definition stets zu den annotierten Elementen verfolgbar, da diese Teil des Annotationskontextes sind.

Im Rahmen dieser Arbeit wird davon ausgegangen, dass die relevanten Rechtsvorschriften und sonstigen relevanten Texte als HTML-Dokumente im Internet oder Intranet verfügbar sind oder in dieses Format transformiert werden können.[184] Im Rahmen dieser Arbeit sind diese in den beabsichtigten Transformationsprozessen zu verarbeitenden Rechtsvorschriften die Dokumente, die mit Annotationen versehen werden.[185] Zu diesem Zweck werden Werkzeuge benötigt, mit denen Annotationen auch noch nach Erstellung des HTML-Dokuments (z.B. auf einer Webseite) ohne weitere Programmierkenntnisse in HTML oder XHTML hinzugefügt werden können. Anhand von zwei Werkzeugen kann exemplarisch gezeigt werden, wie Texte von Rechtsvorschriften mit Annotationen versehen werden. Das frei verfügbare Werkzeug *OntoMat-Annotizer*[186] vermittelt einen Eindruck davon, wie beispielsweise Fachexperten Annotationen von Rechtsvorschriften, basierend auf einer Ontologie, vornehmen könnten. Das Werkzeug ist am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB) der Universität Karlsruhe im Rahmen des Forschungsprojektes *OnTo-Agents*[187] Anfang der 2000er-Jahre entwickelt worden und steht kostenlos zur Verfügung. Seit 2004 wurde dieses Werkzeug nicht mehr weiterentwickelt und ist deshalb an aktuelle Entwicklungen nicht angepasst worden. Eine dieser aktuellen Entwicklungen ist der W3C-Standard *Resource Description Framework in attributes (RDFa)*[188], der Elemente definiert, mit denen weitergehende Möglichkeiten zur Annotation von und Extraktion aus Sprachen wie HTML und XHTML bestehen als mit den Metadaten-Elementen der HTML-Sprachen.[189] Dazu wird ein Satz von Attributen definiert, der eine

---

[183] [Oren et al., 2006] o.S.

[184] Online verfügbar sind Landesrecht (z.B. für das Land Brandenburg unter http://www.bravors.brandenburg.de, letzter Aufruf: 21.08.2011), Bundesrecht (z.B. unter http://www.gesetze-im-internet.de, http://www.verwaltungsvorschriften-im-internet.de, letzter Aufruf: 21.08.2011) und EU-Rechte (z.B. unter http://eur-lex.europa.eu/de/index.htm, letzter Aufruf: 21.08.2011) als HTML- oder PDF-Dokumente. Für die Transformation von PDF-Dokumenten in HTML-Dokumente sind Konverter verfügbar (z.B. das kostenlose Werkzeug "Some PDF to Html Converter" unter http://www.somepdf.com/some-pdf-to-html-converter.html, letzter Aufruf: 21.08.2011).

[185] Im Rahmen dieser Arbeit wird davon ausgegangen, dass sich immer ein menschlicher Anwender mit Hilfe seiner geistigen Fähigkeiten die Bedeutung eines Textes erschließt und die Annotationen erstellt. Er wendet zur Entwicklung des Textverständnisses neben den im allgemeinen Prozess der menschlichen Verständigung üblichen Methoden und Techniken weitere fachspezifische (hier rechtswissenschaftliche) Formen an, um sich die Bedeutung des Normtextes zu erschließen. Eine automatische Verarbeitung ist im Rahmen des Ansatzes dieser Arbeit nicht vorgesehen.

[186] Homepage des Werkzeugs OntoMat-Annotizer unter: http://annotation.semanticweb.org/ontomat/index.html (letzter Aufruf: 21.08.2011).

[187] Homepage des Projektes OntoAgents unter: http://infolab.stanford.edu/OntoAgents/ (letzter Aufruf: 21.08.2011).

[188] [W3C RDFa, 2008a], [W3C RDFa, 2008b]

[189] Zu RDFa gibt es vergleichbare Alternativen. Beispiele hierfür sind Embedded RDF (eRDF) (http://research.talis.com/2005/erdf/wiki/Main/RdfInHtml, letzter Aufruf: 18.03.2011) und die

Erweiterung von HTML und XHTML darstellt und Information in diesen Elementen in Verbindung zu Konzepten und Instanzen einer Ontologie setzen kann. Um die Annotationen aus HTML- und XHTML-Dokumenten zu extrahieren, hat das W3C die Technik *Gleaning Resource Descriptions from Dialects of Languages (GRDDL)* standardisiert.[190] Bei annotierten XHTML-Dokumenten realisiert GRDDL diese Transformation durch die Integration eines Verweises auf ein W3C-Profil und auf die URL eines XSL-Stylesheets.[191] Anwendung soll diese Technik nach Einschätzung der *eGovernment Interest Group des W3C* insbesondere für Webseiten der öffentlichen Verwaltung finden. Unter dem Slogan „Your Website is your API"[192] sieht sie die Notwendigkeit, Informations- und Transaktionsdienste mit Hilfe von RDFa maschinell verarbeitbar zu machen. Texte von Rechtsvorschriften werden dabei nicht explizit erwähnt, sind aber als Bestandteil der Informationsdienste vorstellbar.

Bis zur Fertigstellung dieser Arbeit war es mit den verfügbaren RDFa Annotations-Werkzeugen nicht möglich, in einer mit dem Werkzeug OntoMat vergleichbaren Weise RDFa-Annotationen basierend auf einer benutzerspezifischen Ontologie zu existierenden Dokumenten hinzuzufügen. In der Regel können Annotationen basierend auf populäre Ontologien des Semantic Web (z.B. der Dublin Core Metadata Initiative[193] oder des Friend-of-a-Friend-Projektes[194]) problemlos erstellt werden. Bei der Verwendung der im Rahmen dieser Arbeit erstellten Ontologie zeigten sich bei frei und kostenlos verfügbaren Werzeugen (z. B. bei RADiFy, FUZZ) Kompatibilitäts- und Stabilitätsprobleme. Im Rahmen dieser Arbeit wird deshalb RDFa exemplarisch mit Hilfe eines XML-Editors zu den XHTML-Dokumenten hinzugefügt, um die Nutzung dieses Standards zu verdeutlichen.

Mikroformate (http://microformats.org/, letzter Aufruf: 21.08.2011). Diese Alternativen sind prinzipiell für den Ansatz dieser Arbeit ebenfalls geeignet. Sie haben allerdings nicht den Status eines W3C-Standards, weshalb in dieser Arbeit RDFa gewählt wurde.

[190] Vgl.[W3C GRDDL, 2007a].

[191] Vgl.[W3C GRDDL, 2007b].

[192] Vgl. Use Case 5 der W3C eGovernment Interest Group unter http://www.w3.org/egov/wiki/Use_Case_5_-_Your_Website_is_your_API (letzter Aufruf: 21.08.2011).

[193] Die Homepage der Dublin Core Metadata Initiative ist erreichbar unter: http://dublincore.org. (letzter Aufruf: 08.08.2011.)

[194] Die Homepage des Friend-of-a-Friend-Projektes ist erreichbar unter: http://www.foaf-project.org. (letzter Aufruf: 08.08.2011).

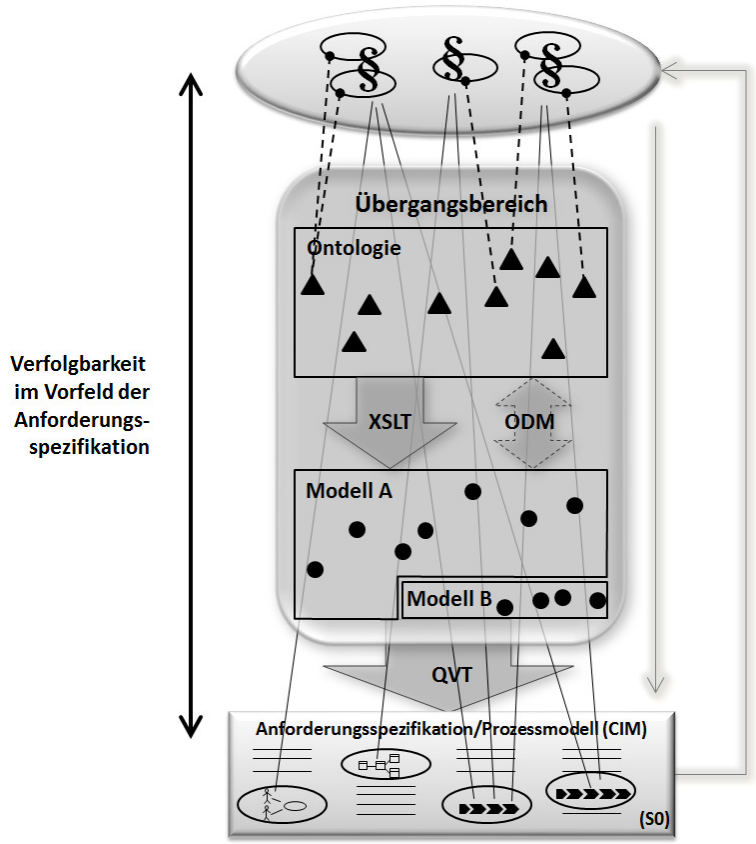


Abbildung 12 – Ausgestaltung des Übergangs im Vorfeld der Anforderungsspezifikation mit MDA/Ontologie/Semantic Web

Durch den Einsatz der Techniken, Methoden und Werkzeuge des Semantik Web können die Rechtsvorschriften in Form von (X)HTML-Dokumenten im Rahmen dieses Ansatzes, basierend auf einer geeigneten Ontologie, mit Annotationen versehen werden. Weil die Annotationen immer mit ihrem Kontext verbunden sind, wird der mit der Formalisierung einhergehende Informationsverlust verringert. Die Textelemente erhalten durch die Formalisierung mit Ontologie Markup-Sprachen eine Semantik, die ihre maschinelle Verarbeitung mit XSLT ermöglicht. Dadurch ist beispielsweise anhand der durch das ODM implizierten Abbildungsregeln die einfache Transformation von RDF(S)- oder OWL-Dokumenten in die XMI-Darstellung von Modellen, basierend auf dem MOF-Metamodell, möglich.

2.5 E-Government und Bürgerdienste

In der Literatur gibt es eine Reihe von Definitionen des E-Governments, die jeweils unterschiedliche Aspekte des Begriffs betonen.[195] Winkel schätzt ein, dass mittlerweile, ausgehend von der Definition des Fachausschusses Verwaltungsinformatik der Gesellschaft für Informatik und des Fachbereichs 1 der Informationstechnischen Gesellschaft im Verband Deutscher Elektrotechniker (VDE), Konsens zur Abgrenzung des Begriffs E-Government besteht.[196] Deshalb werden im Rahmen dieser Arbeit unter E-Government die Prozessdurchführung der öffentlichen Willensbildung, der Entscheidung und der Leistungserstellung in Politik, Staat und Verwaltung unter sehr intensiver Nutzung der Informationstechnik verstanden.[197] Im Rahmen dieser Arbeit wird von dem breiten Spektrum, das diese Definition eröffnet, nur der Ausschnitt betrachtet, der für die

[195] Vgl. [Hagen, 2001], S. 23ff.
[196] Vgl. [Winkel, 2004], S. 126f.
[197] Vgl. [GI&VDE, 2000].

Entscheidung und Leistungserstellung der Verwaltung, basierend auf dem Vollzug von Rechtsvorschriften, im Rahmen des E-Government relevant ist.

Den bisherigen Ausführungen lag der Begriff „öffentliche Verwaltung“ in einem intuitiven Verständnis zugrunde, das für die weitere Eingrenzung des E-Government-Begriffs konkretisiert werden muss: Grundsätzlich ist die öffentliche Verwaltung als Exekutive eine der drei auf die Idee von Montesquieu zurückzuführenden Gewalten im Staat. Ihr obliegt als primäre Aufgabe der Vollzug politischer Entscheide (z.B. Gesetze, Rechtsvorschriften), die ihr von dazu legitimierten Organen zugewiesen wurden sowie einer zuvor eventuell notwendigen Konkretisierung der Entscheidungsprogramme.[198] Im Rahmen dieser Vollzugsaufgabe wird regelmäßig über die Erstellung und Abgabe von Verwaltungsleistungen entschieden. Entsprechend der getroffenen Entscheidung werden diese Leistungen dann erstellt und im Wesentlichen in Form von Dienstleistungen, Informationsgütern, Sachgütern und sonstigen Gütern (z.B. geldlichen Zuwendungen) an Dritte abgegeben. Solches Tätigwerden wird im Folgenden als *Verwaltungshandeln* bezeichnet und weitergehend betrachtet. Entsprechend wird die *öffentliche Verwaltung* hier als eine Organisation verstanden, die Träger der zuvor beschriebenen Aufgabe ist und im Sinne dieses Verwaltungshandelns tätig wird.

Wenn im Rahmen des E-Governments definitionsgemäß die Prozesse und die sie unterstützende Informationstechnik eine integrale Einheit bilden, sind für den im Rahmen dieser Arbeit betrachteten Ausschnitt des E-Governments die Prozesse des Verwaltungshandelns (Verwaltungsprozesse) relevant. Ein Verwaltungsprozess ist (in Anlehnung an den etablierten Begriff des Geschäftsprozesses) eine Folge von Aktivitäten, die mehrfach durchgeführt werden und einen Beitrag zum Ziel und Zweck der Verwaltung liefern, indem ein von außen wahrnehmbares Ergebnis erstellt wird.

Aus Sicht der Softwaretechnik werden Prozesse in Form von Geschäftsprozessmodellen definiert. Sie können dann in eine ausführbare Form gebracht und mit Werkzeugunterstützung (z.B. in Form von Workflow-Management-Systemen) technisch unterstützt oder mit Hilfe von Groupware-Systemen als Ad-hoc-Prozesse durchgeführt werden. Die Informationstechnik besteht aus Sicht der Softwaretechnik vor allem aus Software und hier aus der Anwendungssoftware, mit der die fachlichen Aufgaben innerhalb eines Prozesses bearbeitet werden.[199] Damit Prozesse und Anwendungssoftware eine integrale Einheit bilden können, müssen sowohl die sich aus den Prozessen ergebenden Anforderungen an die Anwendungssoftware als auch die sich durch die Informationstechnik ergebenden neuen Möglichkeiten bei der Prozessgestaltung berücksichtigt werden.[200] Für den Ansatz dieser Arbeit bedeutet dies, dass sowohl für Anforderungen an Prozesse als auch für Anforderungen an Anwendungssoftware die Verfolgbarkeit zu ihren Quellen in den Texten von Rechtsvorschriften sichergestellt werden muss.

Im Rahmen dieser Arbeit können nicht alle Formen und Arten von Rechtsvorschriften betrachtet werden. Die Arbeit konzentriert sich auf solche Rechtsvorschriften, die auf administrativen Vollzug ausgerichtet sind und Verwaltungsleistungen mit einem unmittelbaren Bezug zum Einzelfall eines Bürgers regeln.[201] Diese Verwaltungsleistungen

---

[198] Neben der primären Aufgabe sieht Becker die sekundäre Aufgabe der öffentlichen Verwaltung in der Unterstützung der Politik bei der Herstellung von politischen Entscheidungen (vgl. [Becker, 1989], S. 119).

[199] Neben der Anwendungssoftware sind Systemsoftware sowie Hardware und Kommunikationsinfrastruktur Bestandteil moderner informationstechnischer Systeme. Diese Komponenten sind im Vergleich zur Anwendungssoftware in der Regel universell einsetzbar, so dass sie im Rahmen dieser Arbeit nicht weiter betrachtet werden.

[200] Vgl. [Schuppan, 2005], S. 29.

[201] Gesetze, die nicht direkt vollzogen werden, wie beispielsweise das Signaturgesetz (SiG) oder das Bundesdatenschutzgesetzt (Bundesdatenschutzgesetz in der Fassung der Bekanntmachung vom 14. Januar 2003 - BGBl. I S. 66 -, das zuletzt durch Artikel 1 des Gesetzes vom 14. August 2009 - BGBl. I S. 2814 - geändert worden ist, im Folgenden kurz: BDSG), enthalten Regelungen, die im Rahmen des

wurden im Rahmen der E-Government-Diskussion besonders eingehend als so genannte *Bürgerdienste* betrachtet.[202] In der Literatur fehlt eine anerkannte, einheitliche Definition des Begriffs Bürgerdienste. Im Rahmen dieser Arbeit wird er in Anlehnung an Lenk definiert und dabei die Verbindung zum E-Government weiter betont: Bürgerdienste sind Leistungen der öffentlichen Verwaltung (Verwaltungsleistungen), die mit sehr intensiver Nutzung der Informationstechnik, insbesondere (aber nicht ausschließlich) über das Internet angeboten werden und sich an Endbenutzer richten, die ihre Rolle als Bürger einnehmen.[203]

Für die Gestaltung des Leistungsangebotes für Bürgerdienste ist eine „Orientierung und Ausrichtung der Verfahren nach dem Verständnis und den Bedürfnissen der (externen) Zielgruppen"[204] notwendig. Im Konzept der Lebenslagenorientierung wird dem Rechnung getragen, indem eine *Lebenslage* die Situationen bzw. Lebensabschnitte eines Bürgers bezeichnet, in denen Verwaltungsleistungen möglich oder notwendig sind.[205] Die Lebenslage dient somit der Strukturierung von Verwaltungsleistungen und bietet für den Bürger einen optimierten Einstiegspunkt um die eigentliche Leistungserbringung anzustoßen. Die Erbringung von Verwaltungsleistungen im Sinne von Bürgerdiensten erfolgt nach gängigem Organisationsmuster arbeitsteilig, medienbruchfrei und unter Ausschöpfung der Effizienz- und Effektivitätspotenziale, indem Bürgern der Zugang zu Leistungen über das Front-Office angeboten, die Leistungserbringung aber im Back-Office gebündelt wird. Dieser organisatorische Rahmen für Bürgerdienste hat sich nach Einschätzung von Schuppan durchgesetzt.[206] In der Literatur gibt es dennoch keine einheitliche Definition des Begriffspaares für das E-Government. Im Rahmen dieser Arbeit werden Front- und Backoffice wie folgt definiert: Das *Front-Office* wird als die Menge aller Prozessschritte verstanden, die zur Erbringung einer Leistung abgewickelt werden und einen direkten Nachfragerkontakt erfordern. Der Kontakt im Front-Office kann dabei über verschiedene Zugangswege erfolgen, die in Abhängigkeit von der Intensität der Interaktion von den an diesen Prozessschritten beteiligten Verwaltungsträgern angeboten werden. Das *Back-Office* ist hingegen die Menge alle Prozessschritte, die zur Erbringung einer Leistung abgewickelt werden, unabhängig vom direkten Nachfragerkontakt sind und keinen Zugangsweg erfordern.[207] Für die Abwicklung von Bürgerdiensten wurden in der Literatur verschiedene Phasenmodelle entwickelt, von denen das Modell der niederländischen E-Govnernment-Initiative *Behördenschalter 2000*[208] als repräsentativ gilt.[209] Dieses Modell führt insgesamt 6 Phasen ein, die zum Zweck der Leistungserbringung durchlaufen werden. In Phase 1 wird dem Antragsteller bewusst, dass ein Verwaltungskontakt aufgrund seiner Lebenslage sinnvoll oder notwendig ist. In der anschließenden Orientierungsphase (Phase 2) beschafft sich der Antragsteller die notwendigen Informationen zur Vorbereitung des Verwaltungskontaktes. In Phase 3 wird der Kontakt zur Verwaltung hergestellt und das Anliegen des Bürgers mit

---

Vollzugs anderer Gesetze zu beachten sind. Ist dies für ein mit dem Ansatz dieser Arbeit bearbeitetes Gesetz der Fall, so kann der Ansatz auf die Gesamtheit der sich ergebenden Regelungen angewendet werden. Die Übertragung der Ergebnisse dieser Arbeit auch auf darüber hinausgehende Bereiche des Verwaltungshandelns und die ihnen zugrunde liegenden Gesetze erscheint prinzipiell möglich, kann in dieser Arbeit aber nicht vollständig dargelegt werden.

[202] Die intensive Betrachtung von Bürgerdiensten im Rahmen des E-Government kritisiert Lenk, da mit ihr eine „ideologische Verkürzung von Staatstätigkeit auf Dienstleistungen" ([Lenk, 2005], S. 96) und eine Fokussierung auf kommunale Aufgaben einherging. Andererseits sind Bürgerdienste im Kontext des E-Government nach Auffassung von Lenk „von großem Interesse, weil man hier gut verdeutlichen kann, zu welchen grundlegenden Umgestaltungen eine weitblickende Nutzung der Informationstechnik führen kann." ([Lenk, 2005], S. 97).

[203] Vgl. [Lenk, 2011b], S. 23.

[204] [Wimmer, 2002], S. 60.

[205] Vgl. [Wimmer, 2002], S. 60; übereinstimmend [Müller, 2011], S. 80.

[206] Vgl.[Schuppan, 2005], S. 44.

[207] Vgl. [Fink et al., 2001], S. 196.

[208] Hompage des Behördenschalters 2000 unter: http://www.ol2000.nl, eine deutschsprachige Zusammenfassung ist in [KGSt, 2002] enthalten.

[209] Für eine Übersicht vgl. [Wimmer&Tambouris, 2002], S. 8 ff.

dem Leistungsangebot der Verwaltung in Übereinstimmung gebracht. Die Verwaltung beginnt darauf hin mit der Leistungserstellung (in Phase 4), um die fertig gestellte Leistung in Phase 5 an den Bürger abzugeben. Eine abschließende Phase der Nachbereitung bietet Möglichkeiten, Rückmeldungen des Bürgers (z.B. in Form von formlosem Feedback oder Widerspruch) zu bearbeiten oder interne Abschlussaktivitäten (z.B. Aktivitäten im Rahmen des Wissensmanagements) durchzuführen. Dieses allgemeine Phasenmodell bietet nicht nur einen Orientierungsrahmen für die Prozessgestaltung von Bürgerdiensten. Es wirkt sich indirekt auch auf die Leistungsmerkmale aus, die von Anwendungssystemen zur Unterstützung der Erbringung von Bürgerdiensten im Rahmen des E-Government erwartet werden. Damit kommt derartigen Phasenmodellen ein Referenzcharakter zu, weshalb es in der Literatur auch als „Referenzmodell" geführt wird.

Dem Begriff des Referenzmodells wird im Rahmen dieser Arbeit ein spezielleres, nutzungsorientiertes Verständnis[210] zugrunde gelegt, das ein Referenzmodell als spezielles Modell versteht, „dessen Zweck es ist, für den Entwurf anderer Modelle nützlich zu sein und daher hierfür auch herangezogen wird"[211]. Anhand dieser Definition wird deutlich, dass ein solches Referenzmodell auch die Rolle des Modells einnehmen kann, dass in einer MDA-Transformation die semantische Lücke (vgl. Abschnitt 2.2, Seite 27) zwischen Ausgangs- und Zielmodell schließt. Für diesen Einsatzzweck müssen die Referenzmodelle einerseits auf einem Abstraktionsniveau bereitgestellt werden, das aus dem Vorfeld der Anforderungsspezifikation heraus die Abbildung auf Elemente in informationstechnikunabhängigen Modellen der Anforderungsspezifikation ermöglicht. Andererseits müssen sie das Verwaltungshandeln mittels wieder verwendbarer Modelle formalisieren, so dass sie bei der Entwicklung von Modellen für konkrete Einsatzzwecke und Aufgabenstellungen innerhalb der Verwaltung nützlich sein können. Im E-Government sind relativ wenige derartige Ansätze zur Referenzmodellierung bekannt.[212] In der Regel haben diese Ansätze für einen relativ begrenzten Anwendungsbereich Referenzcharakter, stellen den Fluss von Dokumenten innerhalb der Verwaltung in den Vordergrund, beschränken sich auf die Abläufe innerhalb der Verwaltung oder fokussieren nicht primär auf Prozesse.[213] Nach Einschätzung von Hinkelmann et al. wird aber durch die Arbeiten zur Referenzmodellierung im Projekt *Local Electronic Government (eLoGo)* am Kommunalwissenschaftlichen Institut der Universität Potsdam ein „Vorstoß"[214] unternommen, der insbesondere durch die Konkretisierung der Referenzprozesssicht durch das Referenzanforderungsmodell bemerkenswert ist. Darüber hinaus integrieren die Modelle die Lebenslagenorientierung und den Organisationsansatz des Front- und Back-Office. Im Rahmen der Projektgruppe BundOnline2005 wurden für einzelne, querschnittliche Aspekte (z.B. Antragsstellung), für Beschaffung (eVergabe), für Förderung und für den Zahlungsverkehr (ePayment) Prozessmodelle, Anforderungsbeschreibungen und

[210] Vgl. [Thomas, 2006], S. 17.
[211] [Horn&Off, 2004b], S. 24.
[212] Vgl. [Hinkelmann et al., 2005], S. 360.
[213] u.a. [Engel, 1999], [Menne-Haritz, 1996], [KBSt, 1997], sowie Ergebnisse des Projektes *RAFEG* (Hompage des Projektes unter: http://www.tu-chemnitz.de/informatik/PI/rafeg/, letzter Aufruf: 21.08.2011), das zur Erstellung einer Referenzarchitektur für E-Government auch Prozesse des Planfeststellungsverfahrens modelliert, die einen Referenzcharakter für dieses Anwendungsgebiet haben und Schlussfolgerungen für die Konstruktion von Referenzmodellen zulassen, des Projektes *OntoGov* (Hompage des Projektes unter http://www.ontogov.com ist im April 2011 nicht mehr erreichbar), in dem konkrete Prozesse (z.B. Umzug von Personen) auf Referenzniveau modelliert und für die praktische Anwendung auf unterschiedlichem Abstraktionsniveau bereitgestellt wurden oder des Projektes HERA (Homepage des Projektes unter: http://www.ipmsg.ch/~ktigs2/de/home.html, letzter Aufruf: 21.08.2011), das ebenfalls die Erstellung einer Referenzarchitektur für E-Government als Ziel hatte und dazu auch Prozessmodelle zugrunde legte.
[214] [Hinkelmann et al., 2005], S. 360.

Architekturmodelle entwickelt, die vergleichbar mit den eLoGo-Referenzmodellen sind.[215] Sie zeigen die Relevanz des eLoGo-Ansatzes, der mit Bezug zur Kommunalverwaltung des ländlich geprägten Raumes entwickelt wurde, ebenso für die allgemeine Verwaltung bis zur Bundesebene. Allerdings besitzen die BundOnline2005 bezogen auf Bürgerdienste nicht die gleiche Durchgängigkeit wie die eLoGo-Referenzmodelle, weshalb im Folgenden von den eLoGo-Referenzmodellen ausgegangen wird.

Die eLoGo-Referenzmodelle für E-Government bestehen aus insgesamt drei Modellen, einem Referenzprozessmodell, einem Referenzanforderungsmodell und einer Referenzarchitektur.[216] Das Referenzprozessmodell ist aus den Phasenmodellen für Bürgerdienste abgeleitet und beschreibt die jeweiligen Phasen als Prozesse in Anlehnung an die EPK-Notation. Es unterscheidet dabei einen Anliegen- und einen Leistungsprozess. Der Anliegenprozess umfasst die Aktivitäten, die von der Bewusstwerdung des Anliegens eines Bürgers über die Identifikation von möglichen Verwaltungsleistungen bis zu dessen Befriedigung führen. Aus dem Anliegenprozess wird (ggf. mehrfach) der Leistungsprozess initiiert, in dessen Rahmen die eigentlichen Verwaltungsleistungen erstellt, an den Bürger abgegeben und ggf. nachbereitet werden. Um den Zusammenhang zu den Phasenmodellen zu erhalten, sind zu den Phasen korrespondierende Subprozesse modelliert. Einen Überblick über beide Prozesse geben nachfolgend die Abbildung 13 und die Abbildung 14. Die jeweiligen Subprozesse sind weitergehend detailliert. Am Beispiel des Subprozesses „Bewusstwerdung“ ist dies in Abbildung 15 dargestellt und in Tabelle 2 weitergehend beschrieben. Für alle Subprozesse und die enthaltenen Elemente (z.B. Aktivitäten, Ereignisse, Geschäftsentitäten und Rollen) existiert eine detaillierte Beschreibung.

[215] Die Ergebnisse des Projektes BundOnline2005 können abgerufen werden unter: http://www.cio.bund.de/DE/Standards/Musterprozesse/musterprozesse_node.html (letzter Aufruf: 21.08.2011).
[216] Vgl.[Horn&Off, 2004b].

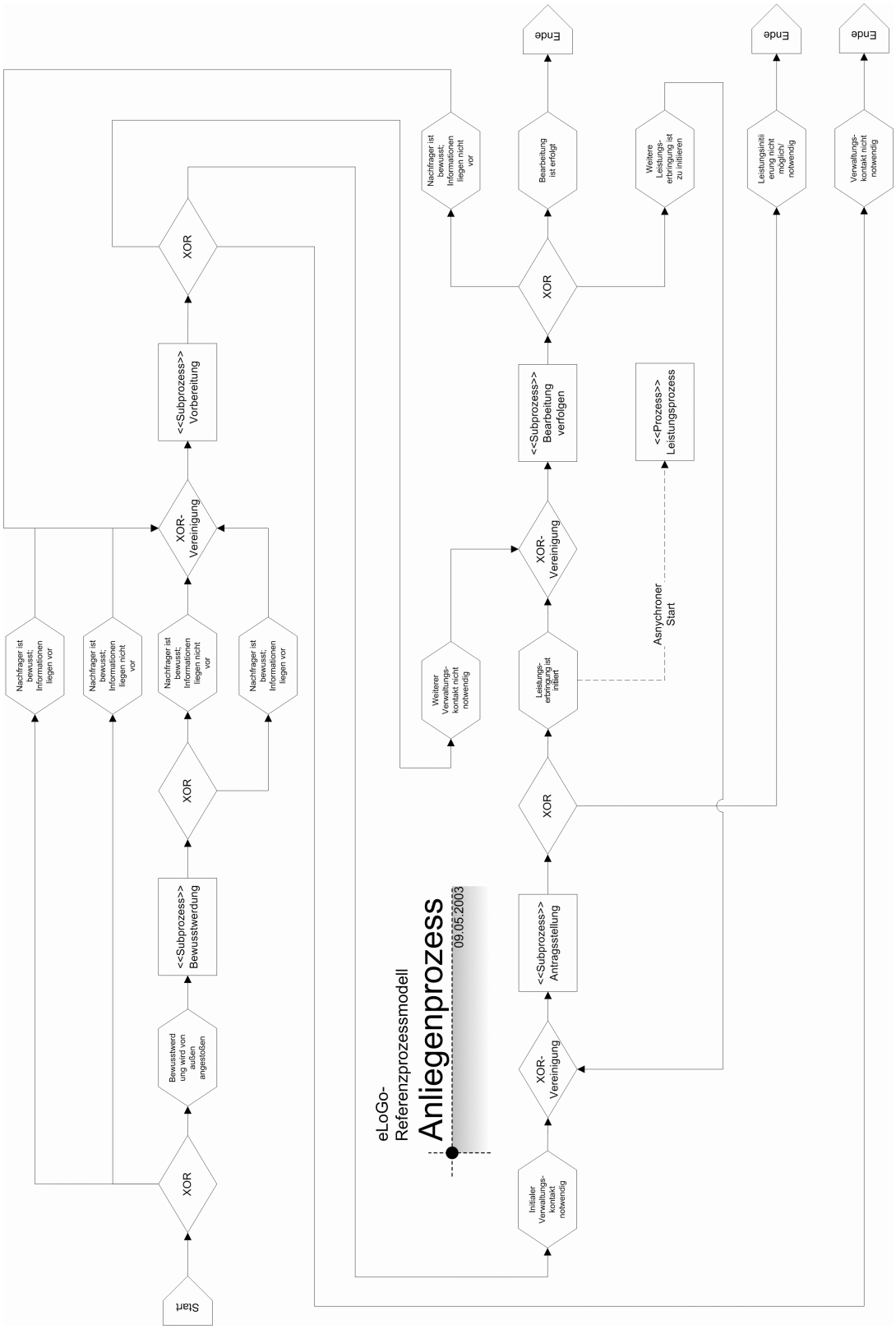


Abbildung 13 – Anliegenprozess aus den eLoGo-Referenzmodellen[217]

[217] Abbildung aus [Horn&Off, 2004b], S. 32.

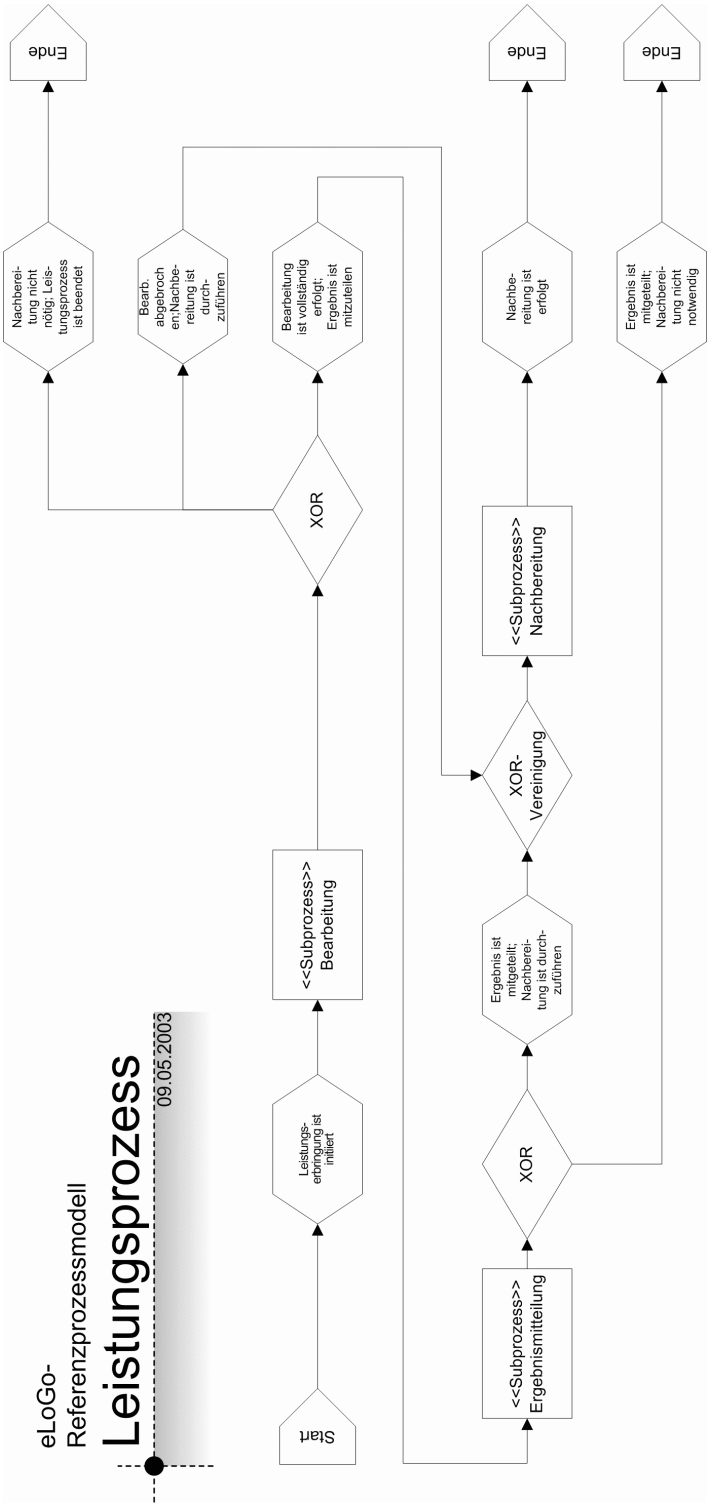


Abbildung 14 – Leistungsprozess aus den eLoGo-Referenzmodellen[218]

[218] Abbildung aus [Horn&Off, 2004b], S. 38.

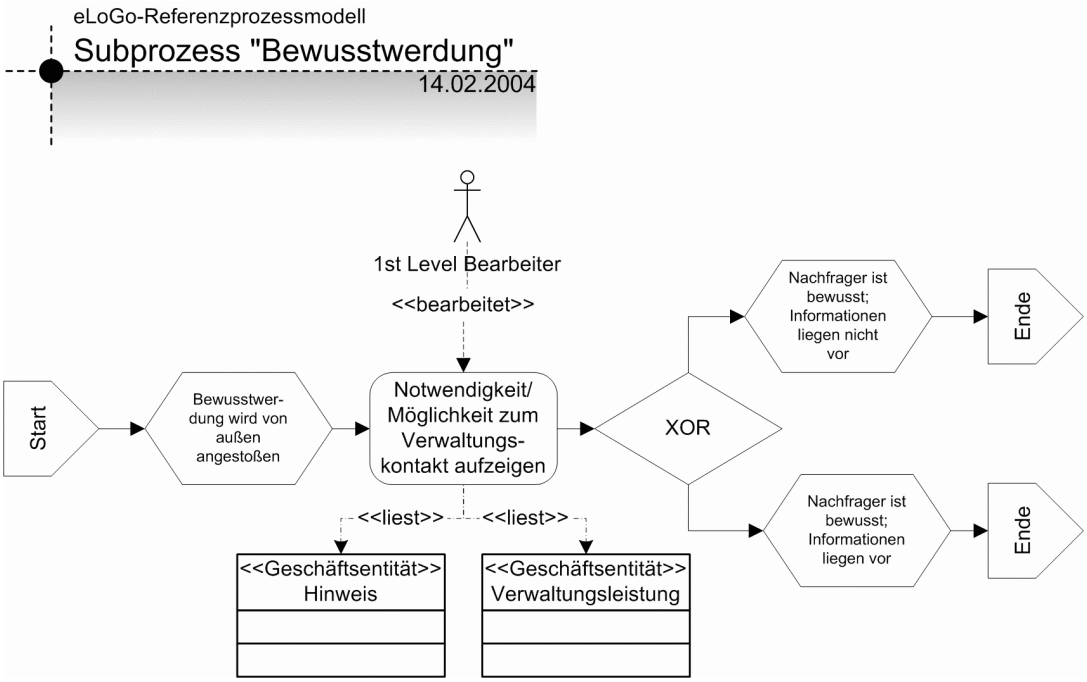


Abbildung 15 – Subprozess "Bewusstwerdung" aus dem Anliegenprozess[219]

Aktivität	Beschreibung
Notwendigkeit/Möglichkeit zum Verwaltungskontakt aufzeigen	Es werden Hinweise zusammengestellt, die die Notwendigkeit und/oder Möglichkeit zu einem Verwaltungskontakt aufzeigen. Im Rahmen dieser Aktivität wird der Nachfrager über die Hinweise und ggf. über weitere Informationen, die in Zusammenhang mit den Hinweisen stehen (z.B.: über mögliche Verwaltungsleistungen zur aktuellen Lebenslage), informiert. Der Nachfrager nimmt die ihm gegebenen Hinweise und Informationen zu Kenntnis.
	Eingangsereignis
	Bewusstwerdung wird von außen angestoßen
	Ausgangsereignis
	Entweder: Nachfrager ist bewusst; Informationen liegen nicht vor Oder: Nachfrager ist bewusst; Informationen liegen vor
	Geschäftsentitäten
	Liest: Hinweis Liest: Verwaltungsleistung Liest: Lebenslage
	Rollen
	1st Level Bearbeiter

Tabelle 2 – Beschreibung der Aktivität aus dem Subprozess "Bewusstwerdung"[220]

Das Referenzanforderungsmodell konkretisiert die im Referenzprozessmodell beschriebenen Elemente weitergehend. Aus diesen Elementen werden die Anforderungen an eine E-

[219] Abbildung aus [Horn&Off, 2004b], S. 39.
[220] Tabelle aus [Horn&Off, 2004a], S. 126.

Government-Anwendung abgeleitet, die die modellierten Aktivitäten der Referenzprozessmodelle intensiv unterstützen kann. Zu diesem Zweck werden Anwendungsfallmodelle der UML verwendet. Das Referenzanforderungsmodell gliedert die Anwendungsfälle in Subsysteme, eines, das die Anwendungsfälle umfasst, die aus dem Anliegenprozess abgeleitet wurden (Abbildung 16) und ein anderes mit den aus dem Leistungsprozess abgeleiteten Anwendungsfällen (Abbildung 17). Darüber hinaus gibt es ein Subsystem mit querschnittlichen Anwendungsfällen, die von den anderen beiden Subsystemen verwendet werden können.

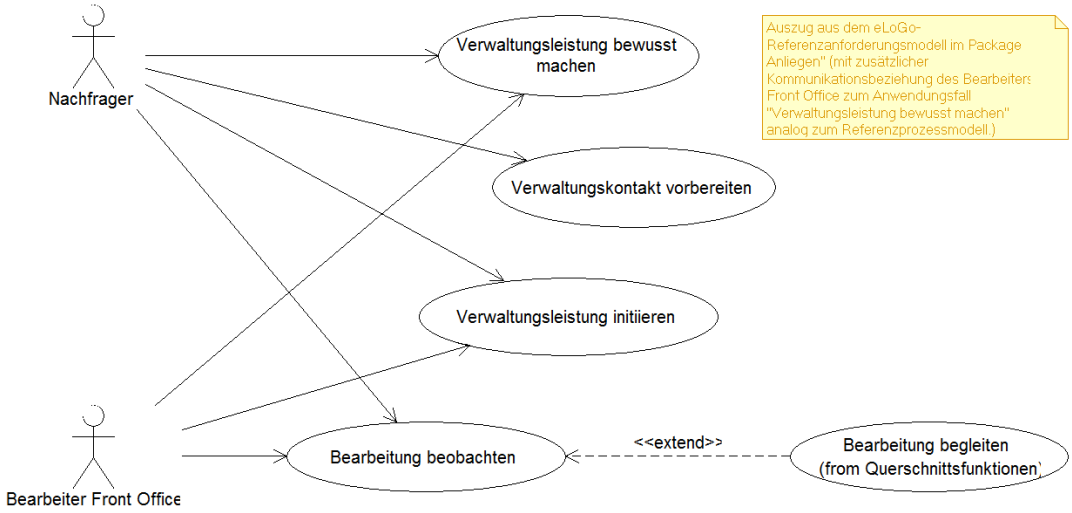


Abbildung 16 – Anwendungsfälle des Subsystems "Anliegen"[221]

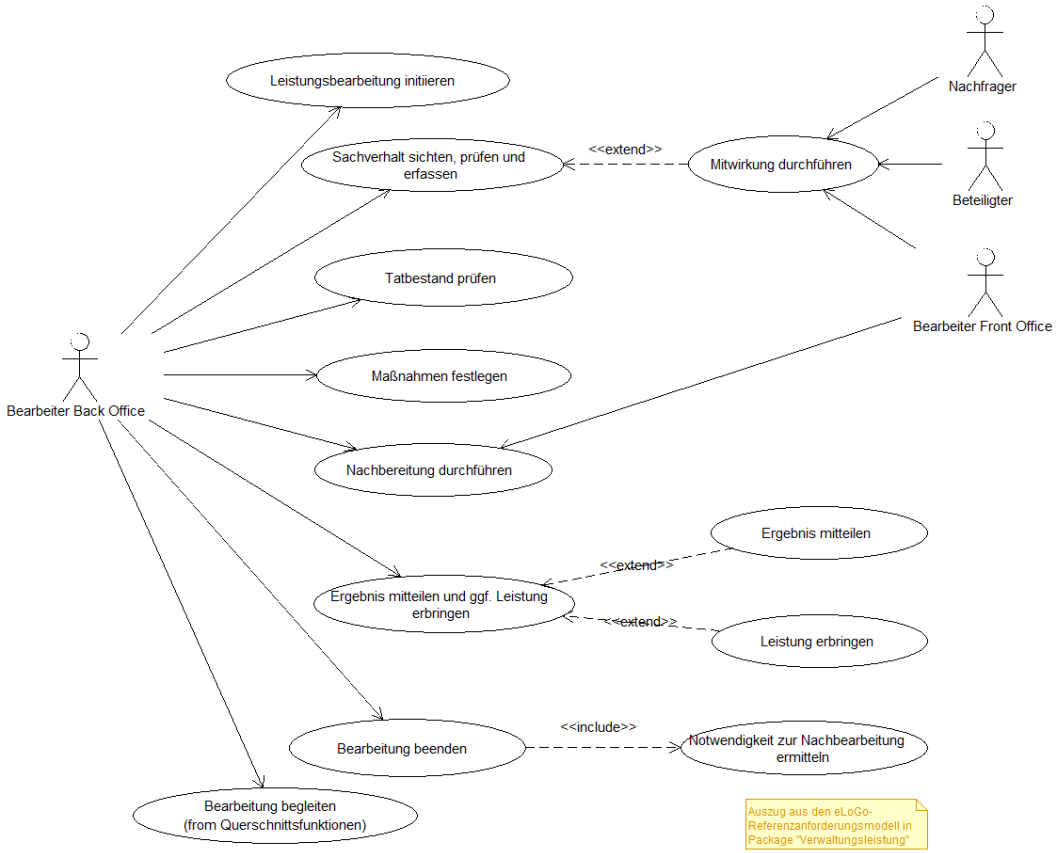


Abbildung 17 – Anwendungsfälle des Subsystems "Verwaltungsleistungen"[222]

[221] Eigene Abbildung nach [Horn&Off, 2004b], S. 45.

Zu jedem Anwendungsfall wurde jeweils ein Anwendungsfalldiagramm erstellt, das durch eine allgemeine Anwendungsfallbeschreibung, die Beschreibung der Interaktion des Akteurs mit der Anwendungssoftware (als Konkretisierung der Kommunikationsbeziehung mit dem Anwendungsfall) und Domänenmodelle ergänzt wird. Am Beispiel des Anwendungsfalls „Verwaltungsleistung bewusst machen“ sind das Anwendungsfalldiagramm in Abbildung 18 und die Domänenklassen in Abbildung 19 dargestellt.

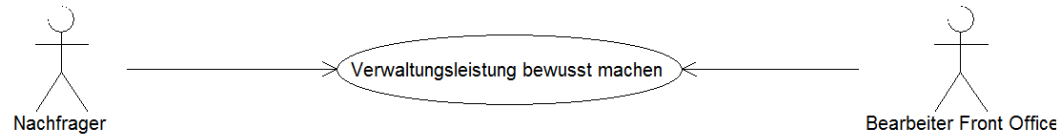


Abbildung 18 – Anwendungsfalldiagramm zum Anwendungsfall "Verwaltungsleistung bewusst machen"[223]

Ziel	Das Ziel des Anwendungsfalls ist es, den Benutzer (als potenziellen Nachfrager) einer Verwaltungsleistung auf die Notwendigkeit und/oder Möglichkeit eines Verwaltungskontaktes aufmerksam zu machen.
Ursprung	Der Anwendungsfall „Verwaltungsleistung bewusst machen“ beschreibt die aus dem Subprozess „Bewusstwerdung“ des Referenzprozesses resultierenden Anforderungen.
Abgrenzung	Die im Referenzprozess vorgesehene Aktivität zum Aufzeigen der Notwendigkeit/Möglichkeit eines Verwaltungskontaktes erfolgt vom Akteur „Nachfrager“ in Interaktion mit dem System. Die Rolle des Bewusstmachers wird im Rahmen des Anwendungsfalls durch das System übernommen, da (entsprechend der Definition von eGovernment) der Prozessablauf intensiv unterstützt werden soll.
Beteiligte Akteure	Am Ablauf des Anwendungsfalls ist der Akteur „Nachfrager“ beteiligt. Die Kommunikation mit dem Nachfrager erfolgt über eine Benutzeroberfläche.
Vor- und Nachbedingungen	Vorbedingung für den Anwendungsfall gibt es keine.Nachbedingung ist, dass Verwaltungsleistungen initiiert wurden und/oder Hinweise verworfen oder keine Änderung des Systems stattgefunden hat.

Tabelle 3 – Anwendungsfallbeschreibung zum Anwendungsfall „Verwaltungsleistung bewusst machen“[224]

[222] Eigene Abbildung in Anlehnung an [Horn&Off, 2004b], S. 47; unter Berücksichtigung zusätzlicher interner eLoGo-Projektergebnisse.
[223] Eigene Abbildung nach [Horn&Off, 2004b], S. 50.
[224] Tabelle aus [Horn&Off, 2004b], S. 50.

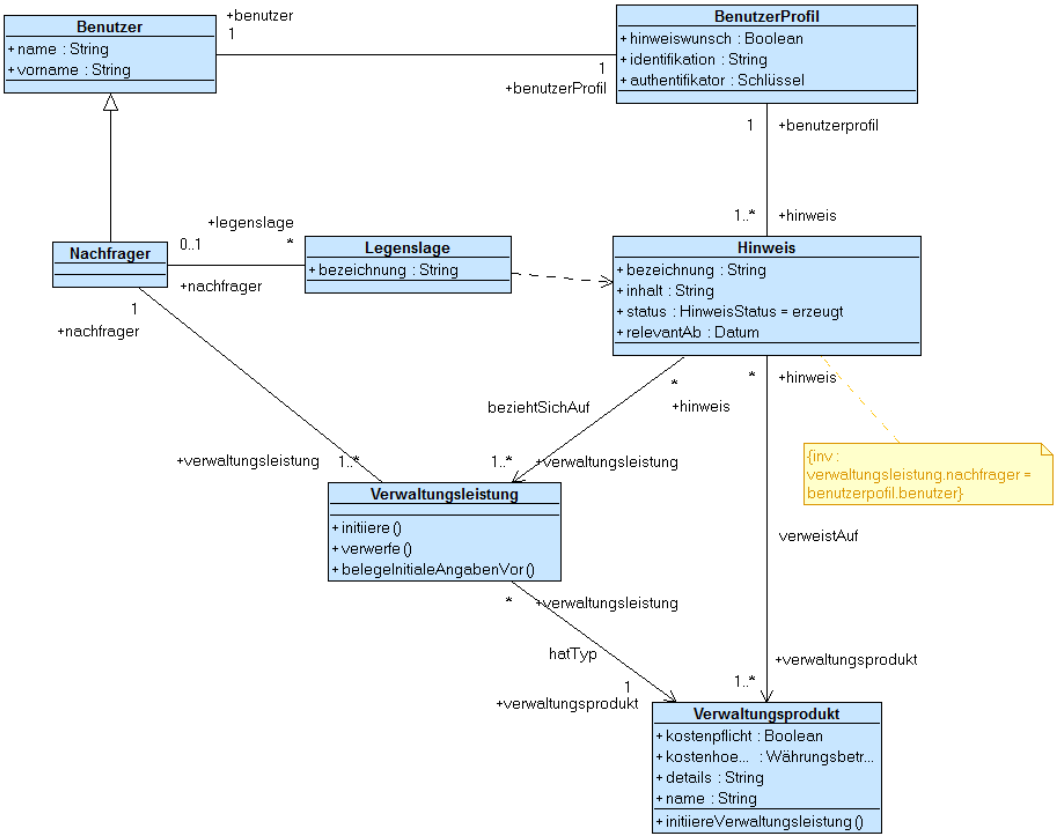


Abbildung 19 – Domänenklassen zum Anwendungsfall "Verwaltungsleistung bewusst machen"[225]

Darüber hinausgehend enthält das Referenzanforderungsmodell prototypische Oberflächenentwürfe eines Bürgerinformationssystems, deren Berücksichtigung über den Rahmen dieser Arbeit hinausgehen würde, da sie den Anwendungsbereich der Modellierung verlassen.

Aus dem Referenzanforderungsmodell wird eine Referenzarchitektur abgeleitet, die mit ihren Komponenten und Diensten die geforderten Leistungsmerkmale der Anwendung erbringt, um den Referenzprozess intensiv im Sinne des E-Government zu unterstützen. Während das Referenzprozessmodell und das Referenzanforderungsmodell eine informationstechnikunabhängige Sicht auf das System darstellen, ist die Referenzarchitektur eine plattformunabhängige Sicht und liegt deshalb außerhalb des Betrachtungsrahmens dieser Arbeit.

Anhand der eLoGo-Referenzmodelle wird einerseits deutlich, dass eine integrierte Betrachtung des Prozesses und der daraus resultierenden Anforderungen notwendig ist, um die intensive Unterstützung im Sinne des E-Governement erreichen zu können. Andererseits ist anhand des Detailgrades deutlich geworden, dass diese Modelle (im Gegensatz zu anderen in der Literatur dokumentierten Referenzmodellen) geeignet sind, um prinzipiell innerhalb einer MDA-Transformation eingesetzt zu werden.

2.6 Bestandsaufnahme

Aus den zuvor betrachteten Entwicklungen lässt sich auf das Potenzial der vorgestellten Technologien und Konzepte schließen, die Verfolgbarkeit von Textelementen in Rechtsvorschriften zu Anforderungen an E-Government-Anwendungen zu ermöglichen. Im folgenden wird der Lösungsansatz dieser Arbeit dargestellt und gegenüber existierenden und verwandten Ansätzen abgegrenzt.

[225] Eigene Abbildung in Anlehnung an [Horn&Off, 2004b], S. 53.

2.6.1 Lösungsansatz

Die Entwicklungen im Bereich der modellgetriebenen Architektur, die Konvergenz von Modellierungsstandards mit Ontologiesprachen, die Techniken und Standards des Semantic Web sowie die Entwicklungen im Rahmen des E-Government bieten Ansatzpunkte für die Verbesserung der Verfolgbarkeit im Vorfeld der Anforderungsspezifikation. Mit den Techniken des Semantic Web ist es möglich, ein Vokabular in Form einer Ontologie zu definieren und darauf basierend Inhalte im World Wide Web des Internets (WWW) mit einer definierten Semantik zu versehen. Für Ontologien ist durch das Ontology Definition Metamodel (ODM) eine prinzipielle Möglichkeit geschaffen worden, ihre Konzepte auf Konzepte aus dem Bereich der Modellierung abzubilden. Da die Bedeutung der Modellierung in der Softwaretechnik in den vergangenen Jahren stark zugenommen hat, ist heute die modellgetriebene Softwareentwicklung mit der MDA (Model Driven Architecture) als ihrem wichtigsten Vertreter ein zentrales Paradigma der Softwaretechnik. Auf diesen Entwicklungen gründete die Arbeitshypothese, die von der Existenz eines neuartigen Ansatzes ausging, der durch geeignete Kombination ausgewählter Elemente dieser Entwicklungen entstehen würde. Die Herausforderung liegt darin, sie in einem Gesamtansatz zu kombinieren und zu integrierten.

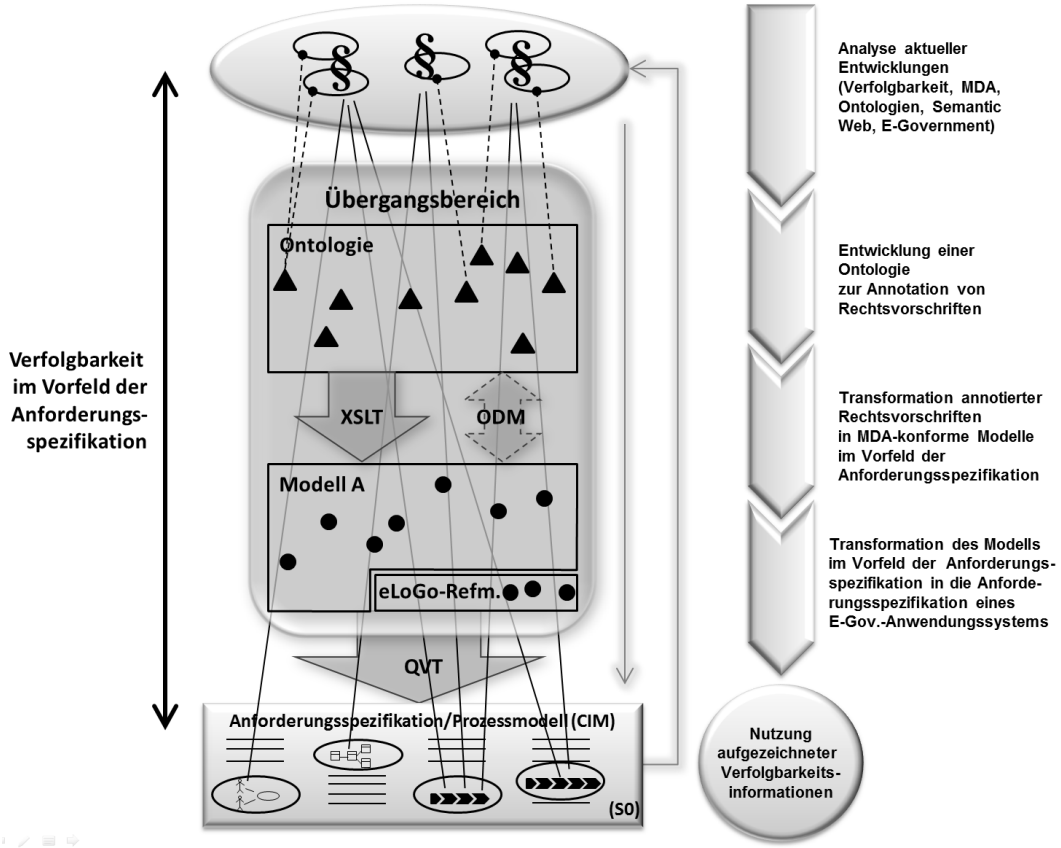


Abbildung 20 – Gesamtansatz mit Bezug zur Vorgehensweise

Für die Kombination in einem Gesamtansatz ist der Ausgangspunkt eine im Rahmen dieser Arbeit entwickelte Ontologie des Verwaltungshandelns. Sie ermöglicht es, in Texten von Rechtsvorschriften relevante Passagen durch Nutzung der Techniken des Semantic Web und mit Hilfe eines geeigneten Werkzeugs zu annotieren. Die Annotationen formalisieren Wissen über die Inhalte und Zusammenhänge im Text der Rechtsvorschrift und machen es für die weitere maschinelle Verarbeitung zugänglich. Die anschließende Abbildung der Konzepte der Ontologie auf den Bereich der Modellierungsstandards anhand des ODM lässt dieses Wissen in ein initiales Modell im Sinne modellgetriebener Ansätze einfließen. Dies wird durch die Konvergenz von Standards beider Bereiche ermöglicht. Gleichzeitig kann die

durchgeführte Abbildung verfolgbar dokumentiert werden. Innerhalb dieses initialen Modells können verwaltungsspezifische Aspekte berücksichtigt werden, die über den bloßen Vollzug der Rechtsvorschrift hinausgehen und deshalb keine Entsprechung in ihrem Text haben. Um den Anschluss an die etablierten Modelle der modellgetriebenen Architektur zu finden, werden Transformationen definiert, die das aus den Annotationen gewonnene und verwaltungsspezifisch weiterentwickelte Modell in das Modell einer Anforderungsspezifikation im Sinn der modellgetriebenen Architektur (vom Typ CIM) überführen. Dazu werden die eLoGo-Referenzmodelle als unterstützende Modelle geeignet in die Transformation eingebunden. Im Ergebnis liegen die aus dem Text der Rechtsvorschrift gewonnenen und um verwaltungsspezifische Aspekte ergänzten Anforderungen in einer Anforderungsspezifikation einer E-Government-Anwendung vor und sind verfolgbar zu ihren Ursprüngen dokumentiert. Diese Anforderungsspezifikation kann mit den etablierten modellgetriebenen Techniken weiterentwickelt werden. Durch diesen Ansatz wird es möglich, Ursprünge von Anforderungen an modellbasierte E-Government-Anwendungen für den Vollzug konkreter Rechtsvorschriften per Transformation verfolgbar in die modellgetriebene Softwareentwicklung einfließen zu lassen.

Ist in diesem Sinn der Anschluss an die modellgetriebene Softwareentwicklung auf Basis der MDA gefunden, kann von der vollständigen Verfolgbarkeit bis zum ausführbaren Anwendungssystem ausgegangen werden, da die weitergehende Verfolgbarkeit von dokumentierten Anforderungen einer Anforderungsspezifikation durch die folgenden Aktivitäten des Softwareentwicklungsprozesses in der Praxis bereits gut beherrscht wird.

Bei der Konstruktion des Ansatzes ist insbesondere auf die Verwendung offener Standards zu achten, so dass nicht nur mit der MDA selbst, sondern insbesondere auch im Umfeld der Ontologien und des Semantic Web auf Standards gesetzt wurde. Für die E-Government-Anwendungen gibt es mit den *Standards und Architekturen für E-Government-Anwendungen (SAGA)* einen verbindlichen Rahmen, welche Technologien für die Umsetzung verwendet werden sollten.[226] Die für die vorgestellten Entwicklungen relevanten Standards sind in der SAGA-Dokumentation empfohlen (z.B. RDF, XMI, XSLT, BPMN), als verpflichtend erklärt (z.B. HTML, XML, UML) oder als relevant eingeschätzt (z.B. XHTML). Insofern ist ihre Verwendung im Rahmen des Ansatzes dieser Arbeit prinzipiell möglich (vgl. Anforderung Nr. 5 auf S. 7).

2.6.2 Alternative und verwandte Ansätze

Der Erkenntnisfortschritt, der durch diese Arbeit erzielt wird, liegt in der Konstruktion der Ontologie des Verwaltungshandelns, deren Nutzung zur Annotation von Rechtsvorschriften mit den Techniken und Standards des Semantic Web und ihrer Abbildung auf Modelle im Sinne der MDA sowie der hierdurch ermöglichten Verbesserung der Verfolgbarkeit im Vorfeld der Softwareentwicklung. Dies erfordert die Verknüpfung und Formalisierung bisher unabhängig von einander existierender Ansätze und Forschungsergebnisse aus unterschiedlichen Fachgebieten. In diesen Fachgebieten existieren bereits vergleichbare oder verwandte Lösungsansätze.

Ansätze der zielbasierten bzw. zielorientierten Anforderungsanalyse stellen den Aktivitäten der Anforderungsanalyse eine Phase voran oder zur Seite, die der Identifikation, Analyse und Formalisierung von organisatorischen Zielen und/oder Zielen der Beteiligten dient.[227] Die Ansätze ermöglichen darüber hinaus beispielsweise die Verifikation und die Validierung von Zielen, das Konfliktmanagement, die Konfliktauflösung und die Alternativenwahl. Dazu formalisieren sie verschiedene Ziele (z.B. funktionale und nicht-funktionale), deren Eigenschaften und Beziehungen. Ziele können solche Beziehungen insbesondere zu Anforderungen in einer Anforderungsspezifikation haben, die beispielsweise der Umsetzung dieser Ziele dienen. Die KAOS- und die GORE-Methode sind Beispiele für Methoden der

[226] Vgl. [BMI SAGA, 2008], [BMI SAGA, 2010].

[227] Vgl. u. a. [Antón, 1996], [Yu&Mylopoulos, 1998], [Berkem, 1999] [Lamsweerde, 2000], [Lamsweerde, 2001], [Donzelli&Bresciani, 2003].

zielorientierten Anforderungsanalyse, wobei Teil der GORE Werkzeugunterstützung auch eine Annotationskomponente für Texte ist, mit der Hyperlinks zu Modellelementen hergestellt werden können.[228]

Innerhalb der zielbasierten und zielorientierten Ansätze hat das *Business Motivation Model (BMM)* als Standard der OMG eine besondere Rolle.[229] Es definiert ein MOF-konformes[230] Metamodell um Elemente eines Business-Plans ausdrücken zu können, zu denen auch Ziele und Absichten sowie die an ihrer Erreichung bzw. Umsetzung beteiligten Geschäftsprozesse, Geschäftsregeln und Organisationseinheiten.[231] Mit dem BMM lässt sich beispielsweise auch die von einer Organisationseinheit definierte Vorgabe als „Gesetz" für eine andere Organisationeinheit (z.B. auf niedrigerer Hierarchieebene) abbilden.[232] Das BMM ist keine vollständige Methode,[233] bietet aber den Anschluss an Elemente aus dem BPMN-Metamodell[234]. Aus der informellen Beschreibung im BMM lassen sich Abbildungsregeln für eine MDA-konforme Transformation ableiten.[235] Von Amsden wird ein Ansatz präsentiert, der zeigt, wie mit der IBM Rational Produktfamilie das BMM für die Anforderungsanalyse praktisch benutzt werden kann.[236]

Im Rahmen der Anforderungsanalyse mit der User Requirements Notation (URN), die von der International Telekommunication Union (ITU) standardisiert wurde, können Ziele, weiche Ziele, Aufgaben und Glauben (Belief) spezifiziert und modelliert werden. Das Konzept des Glaubens soll die Verfolgbarkeit der Ziele zu ihren Quellen ermöglichen.[237] Um die Verbindung zu diesen Quellen herzustellen, verweist die URN auf die eindeutige Identifizierbarkeit der Modellelemente und die Möglichkeit, diese durch Metadaten zu ergänzen (Tupel aus Name und Wert).[238]

Die zielbasierten und zielorientierten Ansätze unterstützen somit die Verfolgbarkeit im Vorfeld der Anforderungsspezifikation, da sie zielorientierte Aspekte formalisieren, bevor diese in Form von Anforderungen in eine Spezifikation eingeflossen sein müssen. Aufgrund der Fokussierung auf Rechtsvorschriften, die auf den Vollzug in der öffentlichen Verwaltung ausgerichtet sind, umfassen sie eine Beschreibung der Ziele und Zwecke der getroffenen Regelungen. Da Rechtsvorschriften weitere wesentliche Aspekte umfassen, stellen Ansätze zur Zielanalyse keine vollständige Lösung für die Verfolgbarkeit der Ursprünge von Anforderungen in den Texten von Rechtsvorschriften dar.

Breaux[239] schlägt einen Ansatz vor, bei dem der Anwender abstrakte Konzepte wie Rechte, Pflichten, Bedingungen, Vollmachten und Definitionen nach einer vorgegebenen Methode aus dem Text einer Rechtsvorschrift extrahiert und Referenzen auf den Abschnitt der

---

228 Vgl. [Lamsweerde, 2004], S. 6 f.
229 [OMG BMM, 2010]
230 Im BMM-Standard fehlt der Hinweis auf die MOF-Konformität des zugrunde liegenden Metamodells. Eine Anfrage bei der OMG (Issue 16397) resultierte in der freundlichen Auskunft von Pete Rivett, Mitglied der OMG BMM Revision Task Force, per E-Mail vom 02.08.2011, dass es sich um ein Problem mit der Veröffentlichung des Standards handelt und für das BMM das MOF-konforme Metamodell unter den folgenden Adressen verfügbar ist: http://www.omg.org/cgi-bin/doc?dtc/08-11-13 und als aktuellere Version http://www.omg.org/cgi-bin/doc?dtc/09-03-10 (letzter Aufruf: 21.08.2011).
231 Vgl. [OMG BMM, 2010], S. 1 f.
232 Vgl. [OMG BMM, 2010], S. 33 f.
233 Vgl. [OMG BMM, 2010], S. 3.
234 Vgl. [OMG BMM, 2010], S. 89.
235 Die seit Mai 2010 verfügbare Version 1.1 des BMM beschreibt informell u.a. die Zusammenhänge mit Konzepten der BPMN. In Version 1.0 aus dem Jahr 2008 wird stattdessen ursprünglich das Business Process Definition Metamodel ([OMG BPDM, 2008a], [OMG BPDM, 2008a]) berücksichtigt (vgl. [OMG BMM, 2008], S. 91 ff.).
236 [Amsden, 2008]
237 [ITU-T, 2008], S. 23.
238 [ITU-T, 2008], S. 137 f.
239 [Breaux, 2006]

Rechtsvorschrift anlegt, aus dem sie hervorgegangen sind. Sie werden mit Mitteln der formalen Logik weitergehend formalisiert, um sie als Anforderungen in die Softwareentwicklung einfließen lassen zu können.[240] Der Ansatz lässt offen, wie die eigentliche Verbindung zu Anforderungen einer Anforderungsspezifikation herzustellen wäre. Die Weiterentwicklung[241] dieses Ansatz verwendet als Ausgangspunkt Rechtsvorschriften in Form von XML-Dokumenten, basierend auf einem speziellen XML-Schema. Abstrakte Konzepte (z.B. Erlaubnis, Pflicht, Tatsache, Absicht, Objekt, Ort) werden in einer Upper Ontology in Form von UML-Modellen formalisiert und dienen entsprechend einer vorgegebenen Heurisitik zur Annotation der Rechtsvorschrift in ihrem speziellen XML-Format. Für die Annotation wird eine spezielle Frame-based Markup Language verwendet. Im Ergebnis des Ansatzes kann aus den XML-Dokumenten per XSLT eine tabellarische Zusammenfassung der aus den annotierten Textpassagen resultierenden Anforderungen als XHTML-Dokument generiert werden. Wie der Anschluss dieser Anforderungen an die Anforderungsanalyse stattfindet, bleibt weiterhin offen. In der Weiterentwicklung des Ansatzes von Breaux wird von Kiyavitskays et al. ein System vorgeschlagen, das den Extraktionsprozess automatisiert, d. h. die Texte von Rechtsvorschriften automatisch mit Annotationen der Konzepte versieht. Die Annotationen werden von diesem Werkzeug mit einfachen XML-Tags vorgenommenen.[242]

Das *Nòmos Framework* ist ein Ansatz zur Entwicklung gesetzeskonformer Anforderungen, der dem Ansatz von Breaux ähnlich ist.[243] Er umfasst unter anderem eine Sprache zur Modellierung von Anforderungen und zur Modellierung des Einflusses rechtlicher Vorschriften auf Anforderungen.[244] Weiterhin umfasst er einen Prozess zur systematischen Erzeugung gesetzeskonformer Anforderungen.[245] Die Sprache wird als Metamodell mit der UML ausgedrückt. Sie umfasst abstrakte Konzepte wie das Recht, den Akteur, das Ziel und die Umsetzung.[246] Eine spezielle Notation unterstützt den Prozess der Modellierung.[247] Dieser Prozess soll iterativ begleitend zur eigentlichen Anforderungsanalyse durchgeführt werden, in der die gesetzeskonformen Anforderungen weitergehend verfeinert werden.[248] Er umfasst Aktivitäten, wie die Identifikation der relevanten Rechtsvorschriften, die Modellierung der gesetzeskonformen Anforderungen mit den formalisierten Konzepten, die Verifikation der Konformität mit der Rechtsvorschrift und eine Schwachstellenanalyse, um Verbesserungsmöglichkeiten zu identifizierten und zu lösen, die aus der Notwendigkeit zur Konformität resultieren.[249]

Im Gegensatz zum Ansatz von Breaux und zum Nòmos Framework verwendet der Ansatz dieser Arbeit anstelle einer Upper Ontologie oder eines Metamodells mit abstrakten universellen Konzepten eine spezielle Ontologie, für die Annotation von Rechtsvorschriften. Weiterhin berücksichtigt er verwaltungsspezifische Besonderheiten im Vorfeld der Anforderungsspezifikation und nutzt anstelle einer speziellen Annotationssprache die Techniken des Semantic Web bzw. anstelle einer speziellen Modellierungsnotation die UML. Er bietet außerdem Anschluss an die MDA-basierte, modellgetriebene Softwareentwicklung, indem eine initiale Anforderungsspezifikation als CIM erzeugt wird.

Der Ansatz dieser Arbeit grenzt sich auch von Projekten aus dem Bereich Corporate Governance und Compliance ab. Corporate Governance legt die „allgemeinen Grundsätze zu Entscheidungskompetenzen und Verantwortung, Regeln für die Sicherstellung von

[240] Vgl. [Breaux, 2006].
[241] [Breaux, 2009]
[242] [Kiyavitskaya et al., 2008]
[243] [Siena, 2010]
[244] Vgl. [Siena, 2010], S. 3.
[245] Vgl. [Siena, 2010], S. 3.
[246] Vgl. [Siena, 2010], S. 39 ff.
[247] Vgl. [Siena, 2010], S. 44.
[248] Vgl. [Siena, 2010], S. 73.
[249] [Siena, 2010], S. 76.

Transparenz und Einhaltung moralischer Grundsätze [...]“[250] fest. Die Corporate Governance dient somit der „verantwortungsvollen Steuerung des Unternehmens“[251] und wird durch Corporate Compliance ergänzt, die auf „die Umsetzung der notwendigen Kontrollmaßnahmen“[252] zielt. Corporate Compliance umfasst „alle organisatorischen Aufsichts-, Schulungs- und Kontrollmaßnahmen der Geschäftsleitung (einschließlich der Einrichtung eines Berichts- und Dokumentationswesens), welche einen Verstoß des Unternehmens gegen gesetzliche Pflichten verhindern sollen“[253]. Für das Thema dieser Arbeit ist der Teilbereich IT-Compliance relevant, deren Gegenstand die Aspekte der Compliance sind, die sich aus der eingesetzten Informationstechnik ergeben. [254] Zusammenfassend wird hier unter dem Betriff IT-Compliance „die Kenntnis und Einhaltung sämtlicher regulatorischer Vorgaben und Anforderungen an das Unternehmen, die Aufgabe und die Einrichtung entsprechender Prozesse […], sowie die Kontrolle und Dokumentation der Einhaltung der relevanten Bestimmungen gegenüber internen und externen Adressaten“[255] verstanden.

Für die Gestaltung und Modellierung von Geschäftsprozessen unter Beachtung von Compliance-Anforderungen gibt es eine Reihe von Ansätzen, die auf der Formalisierung der Anforderungen mittels abstrakter, generischer Elemente eines Metamodells oder mit formalen Regeln basieren.[256] Diesen Ansätzen ist gemein, dass sie erst nach der Identifikation der einzuhaltenden Compliance-Regelungen ansetzen und deshalb nicht die Auseinandersetzung mit dem Text einer Rechtsvorschrift umfassen bzw. den Zusammenhang zwischen identifizierten Compliance Regelungen und ihrer Umsetzung vernachlässigen. Hierdurch unterscheiden sie sich grundsätzlich vom Ansatz dieser Arbeit.

Im Rahmen des Projektes REALM wurde hingegen ein Metamodell entwickelt, mit dem sich rechtliche Anforderungen für das Compliance Management im Unternehmen formalisieren und Beziehungen zu Ursprüngen in Rechtsvorschriften verfolgen lassen.[257] Das Metamodell umfasst abstrakte, generische Konzepte wie Person, Organisation, Prozess und Aktion, Artefakt und Ressource, Prinzip und Absicht sowie Ort und Kosten. Im Metamodell sind mögliche Beziehungen von Instanzen dieser Konzepte definiert. Es wird von Giblin et al. vorgeschlagen, (unternehmens-)spezifische Elemente durch Spezialisierung der Elemente des Metamodells abzuleiten.[258] Durch die Instanziierung der Konzepte entsteht ein Compliance Regelsatz, der mit temporaler Logic beschrieben wird und mittels eines speziellen UML-Profils in Form von Klassendiagrammen ausgedrückt werden kann. Im Klassendiagramm können die Elemente mit zusätzlichen Metadaten versehen werden. Diese Metadaten ermöglichen es, u. a. auch eine Beziehung zu Ursprüngen in Rechtsvorschriften zu dokumentieren[259], womit das Projekt einen Beitrag zur Verfolgbarkeit zu den Ursprüngen von Rechtsvorschriften leistet. Die mit REALM modellierten Zusammenhänge werden dann jedoch nicht auf Anforderungen einer Anforderungsspezifikation abgebildet. Vielmehr sehen die Autoren die REALM-Modelle als plattformunabhängige Modelle, die in plattformspezifische Modelle überführt werden können: „Depending on the type of requirements captured, target systems can be process definitions, access control lists, privacy policies, storage policies or correlation rules for event monitoring. The models are mapped onto these artifacts by means of model transformations. […] In terms of the Model Driven Architecture (MDA) [...], one can regard the REALM models as platform-independent

[250] [Fröhlich&Glasner, 2007], S. 28.
[251] [Rath, 2009], S. 119.
[252] [Rath, 2009], S. 119.
[253] [Rath, 2009], S. 119.
[254] Im Folgenden werden deshalb die Begriffe „Compliance“ und „IT-Compliance“ synonym verwendet.
[255] [Rath&Sponholz, 2009], S. 24.
[256] Vgl. u.a. [Sadiq et al., 2007], [Liu et al., 2007], [Padmanabhan et al., 2006], [Schleicher et al., 2010]
[257] Vgl. hier und im Folgenden [Giblin et al., 2005].
[258] Vgl. [Giblin et al., 2005], S. 6.
[259] Vgl. [Giblin et al., 2005], S. 8.

models (PIM) and the target models as platform-specific models (PSM)."[260] Durch die Generierung plattformspezifischer Modelle aus den REALM-Modellen wird die in der Softwareentwicklung etablierte Anforderungsanalyse vernachlässigt, die in einer Anforderungsspezifikation als zentralem Element der Verfolgbarkeit resultiert. Die Ausrichtung des REALM-Ansatzes auf ein breites Spektrum von Compliance Anforderungen resultiert darüber hinaus in einem vergleichsweise generischen Metamodell, das die Potenziale der domänenspezifischen Gestaltung, wie sie im Ansatz dieser Arbeit für E-Governement-Anwendungen genutzt werden, nicht auszuschöpfen vermag. Der REALM-Ansatz geht weiterhin davon aus, dass die Elemente mit Verweisen auf ihre Ursprünge in Rechtsvorschriften versehen werden, nachdem diese im REALM-Modell modelliert wurden. Er nutzt im Gegensatz zum Ansatz dieser Arbeit nicht die Techniken des Semantic Web, um die Elemente aus den Rechtsvorschriften zu extrahieren.

Das Projekt *COMPAS (Compliance-driven Models, Languages, and Architectures for Services)*, das im Rahmen des 7. Framework-Programms der Europäischen Kommission von 2008 bis 2011 durchgeführt wurde, entwickelte einen IT-Compliance Ansatz für Geschäftsprozesse.[261] Er wurde auf Basis modellgetriebener Softwareentwicklungstechniken (MDSD), domänenspezifischer Modellierungsprachen (DSL) und service-orientierter Architektur-komponenten (SOA-Komponenten) implementiert. Der COMPAS-Ansatz ermöglicht die Modellierung und Spezifikation von Compliance-Anwendungen mittels einer erweiterten Spracherweiterung der Business Process Execution Language (BEPL). Über einen Generator unterstützt er die Code-Erzeugung zur Ausführung der Geschäftsprozesse und ermöglicht ihre Überwachung (z. .B. mittels Dashboard und durch Abfragesprachen). COMPAS konzentriert sich damit auf die Prozesssicht, die auch Bestandteil von E-Government-Anwendungen ist, lässt aber die Spezifikation von Anforderungen an die weiteren Bestandteile des Anwendungssystems unberücksichtigt. Es werden Techniken aus dem MDSD eingesetzt, die ihren Einsatzschwerpunkt auf Code-Erzeugung legen und sich darin von einem MDA-basierten Ansatz unterscheiden. Die Verfolgbarkeit zu den Quellen von Compliance Anforderungen wurde innerhalb des Projektes als wichtig herausgearbeitet: „[...] how do compliance requirements relate to compliance sources such as laws or regulations?"[262] Deshalb werden innerhalb der konzeptionellen Modellierung im Projekt die Ursprünge von Anforderungen als *Compliance Sources* formalisiert und eine Verbindung zu *Compliance Requirements* definiert. Das im Projekt entwickelte Werkzeug verfügt u.a. über die Komponente *Compliance Requirements Manager (CompRM).* Sie ermöglicht die Erfassung der Ursprünge von Compliance Requirements, indem sie die Erfassung des Namens, einer Beschreibung, des Typs und weiterer einfacher Angaben zur Compliance Source erlaubt. Weiterhin können Compliance Source untereinander durch eine Eltern-Kind-Beziehung verbunden werden. Im Gegensatz zum REALM-Ansatz können Compliance Sources erfasst werden, ohne dass das zugehörige Compliance Requirements bereits definiert wurde. Dies erlaubt prinzipiell die Betrachtung des Vorfelds der Anforderungen, da diese noch nicht formalisiert sein müssen. Der COMPAS-Ansatz gibt keine weitergehende Hilfestellung, um die Elemente aus den als Compliance Source identifizierten Rechtsvorschriften zu extrahieren. Darin unterscheidet sich der COMPAS-Ansatz vom Ansatz dieser Arbeit, der hierfür die Techniken des Semantic Web in Verbindung mit einer speziellen Ontologie verwendet.

Richten sich die Compliance-Regelungen nicht (ausschließlich) an die Elemente eines Geschäftsprozesses, sondern auch an die sie unterstützende Anwendungssoftware, werden diese im Rahmen der Anforderungsanalyse der Anwendungssoftware betrachtet. Rath und Sponholz sehen deshalb das so genannte Compliance Engineering als Bestandteil der

[260] Vgl. [Giblin et al., 2005], S. 5.
[261] Die Hompage des Projektes COMPAS ist erreichbar unter: http://www.compas-ict.eu/index.php (letzter Aufruf: 21.08.2011).
[262] [COMPAS, 2011], S. 10.

Anforderungsanalyse und schlagen unter Verweis auf den Ansatz von Abel[263] die Verwendung der etablierten Verfolgbarkeitstechniken der Anforderungsanalyse vor[264], ohne auf die Problematik der Verfolgbarkeit im Vorfeld der Anforderungsspezifikation einzugehen.

Die dargestellten Projekte aus dem Bereich Compliance zeigen, dass ein Ansatz für die Verfolgbarkeit der Ursprünge von Anforderungen in Rechtsvorschriften zu (geschäfts-)prozessbasierten Anwendungen auch außerhalb des E-Government Relevanz besitzt. Sie unterscheiden sich vom Ansatz dieser Arbeit in der Auseinandersetzung mit dem Text der Rechtsvorschrift, in der Formalisierung der Ursprünge von Anforderungen, basierend auf einer Ontologie und dem Anschluss an die MDA-basierte, modellgetriebene Softwareentwicklung. Im Rahmen dieser Arbeit wird die Ursache für diesen Unterschied darin gesehen, dass Compliance-Ansätze sich eben auf *Unternehmens*handeln beziehen. Der Kern dieses Unternehmenshandelns ist die Produktion von Sach- oder Dienstleistungen für den Bedarf Dritter, um sie am Markt anzubieten.[265] Das Unternehmenshandeln wird nicht im gleichen Maße durch Rechtsvorschriften geregelt, wie dies für die öffentliche Verwaltung der Fall ist, deren Handeln streng an Recht und Gesetz gebunden ist. Compliance greift zwar sehr viele einzelne Aspekte auf, zu denen das informationstechnikunterstützte Unternehmenshandeln konform sein muss[266], strebt aber definitionsgemäß danach Verstöße des Unternehmens gegen gesetzliche Pflichten zu verhindern. Die unmittelbar wertschöpfenden Handlungen zur Produktion der Sach- oder Dienstleistungen werden dabei nicht im gleichen Maße durch Rechtsvorschriften geregelt, wie es für die Produktion von Entscheidungen in der öffentlichen Verwaltung der Fall ist. Für die intensive Unterstützung des Handelns besitzt die Analyse der Ursprünge von Anforderungen in Rechtsvorschriften für Unternehmen daher nicht die gleiche Bedeutung, wie für die öffentliche Verwaltung.

In diesem Zusammenhang zeigt sich auch, dass die Mehrzahl der Compliance Ansätze eine reaktive Natur hat.[267] Sie erlauben, von einem existierenden Geschäfts(prozess)modell ausgehend, die Beurteilung, ob dieses zu Regelungen konform ist: „[...] they only allow to determine after design whether the resulting business process model is compliant."[268] Hierin liegt ein weiterer wesentlicher Unterschied zum Ansatz dieser Arbeit, der von den Ursprüngen der Anforderungen in Rechtsvorschriften ausgeht, um das Gestaltungspotenzial durch Berücksichtigung verwaltungsspezifischer Aspekte für eine intensive Unterstützung des Verwaltungshandelns im Sinne des E-Government sicherzustellen.

Der Ansatz dieser Arbeit soll nicht die Basis für ein Rechtsinformationssystem, ein regelbasiertes oder wissensbasiertes System sein, obwohl Elemente in Texten von Rechtsvorschriften identifiziert, semantisch aufbereitet und auch für die werkzeugunterstützte Weiterverarbeitung zugänglich gemacht werden. Insofern unterscheidet er sich beispielsweise auch von Ansätzen, die von Giblin et al. als Beispiele für Modelle von konkreten Regulierungsanforderungen bezeichnet werden („Exemplary models of concrete regulatory requirements"[269]), aus Sicht dieser Arbeit jedoch als Formalisierungen von (berechenbaren) Regeln einzuordnen sind. Hierzu zählen die Formalisierung des *British Nationality Act* durch Regeln in der Programmiersprache PROLOG[270], des Niederländischen Steuergesetzes, des *Ontario Freedom of Information and Protection of Privacy Act*[271] durch Regeln in der Enterprise Privacy Authorization Language (EPAL)[272] sowie des *US Code of Federal*

[263] [Abel, 2007], S. 49; nach [Rath&Sponholz, 2009], S. 138.
[264] Vgl. [Rath&Sponholz, 2009], S. 138.
[265] Vgl. [Peters et al., 2005], S. 7 f.
[266] Vgl. [Wecker&vLaak, 2009], S. 151.
[267] Vgl. [COMPAS, 2008], S. 23.
[268] [COMPAS, 2008], S. 7.
[269] [Giblin et al., 2005], S. 3.
[270] [Sergot et al., 1986]
[271] [Powers et al., 2004]
[272] Enterprise Privacy Authorization Language ist eine von IBM entwickelte XML-basierte Sprache zur Beschreibung von Regeln des Datenschutzes, die von IBM entwickelt und im Jahr 2003 zur

Regulations Title 40 (40 CFR): Protection of the Environment in einer speziell entwicklten XML-basierten Sprache für Regelungen zum Umweltschutz, die auch Verarbeitungsregeln, basierend auf Prädikatenlogik, umfasst.[273] Die Formalisierung und Nutzung von (berechenbaren) Regeln, die sich aus den Rechtsvorschriften ableiten lassen, ist nicht Ziel dieser Arbeit. Die Arbeit sucht vielmehr nach einem Ansatz zur Konstruktion des Zusammenhangs, der für die Verfolgbarkeit von Anforderungen im Vorfeld der modellgetriebenen Softwareentwicklung verwendet werden kann.

Der Ansatz dieser Arbeit geht vom Vorliegen der Rechtsvorschriften als HTML- oder XHTML-Dokumente aus, wie es zum Zeitpunkt der Arbeit Stand der Darstellung im WWW ist. Er unterscheidet sich darin von Ansätzen, die spezielle Ausgangsformate für die Rechtsvorschriften erfordern (z.B. das im Rahmen des MetaLex Projektes[274] entwickelte *MetaLex XML interchange format*[275]).

Ebenfalls abzugrenzen ist diese Arbeit von *Domänenspezifischen Sprachen* (DSL, Domain Specific Languages). Einerseits nutzt der Ansatz dieser Arbeit das domänenspezifische UML-Profil für RDF und OWL, das zur Modellierung von Ontologien benutzt wird, so dass das Modell im Vorfeld der Anforderungsspezifikation als domänenspezifisch gelten könnte. Andererseits bietet das Profil selbst vergleichsweise universell einsetzbare Elemente, mit denen sich Ontologien aus verschiedenen Anwendungsbereichen formalisieren lassen. Deshalb sollte der Ansatz nicht zu den domänenspezifischen Sprachen gezählt werden. Die Nutzung eines Teils der eLoGo-Referenzmodelle erfolgt ebenfalls durch ein UML-Profil, das im Rahmen dieser Arbeit entwickelt wurde. Mit dem Profil wird die Semantik der Modellelemente eines CIM weitergehend domänenspezifisch konkretisiert. Die Motivation hierfür ist ausschließlich, die semantische Lücke zwischen dem Vorfeld der Anforderungsspezifikation und dem CIM zu überbrücken.[276] Eine sich anschließende domänenspezifische Modellierung im Rahmen der Anforderungsanalyse, basierend auf dem eLoGo UML-Profil, geht über das Thema dieser Arbeit hinaus, da sie sich auf die Verfolgbarkeit im Vorfeld der Anforderungsspezifikation konzentriert.

Vergleichbar ist diese Arbeit mit dem Ansatz des *Continuous Model-Driven Engineering (CMDE)*[277]. Beiden Ansätzen liegt die Erkenntnis zugrunde, dass modellgetriebene Softwareentwicklung zwar die Modellierung in den Vordergrund stellt, diese aber weiterhin im Aufgabenbereich der IT verbleibt: „Model-driven design shifts the attention […] to the modeling level, but still remains in the IT realm."[278] CMDE ändert dies, indem durch das *eXtreme Model-Driven Design (XMDD)*, ausgehend von fachlichen Modellen, die Lücke zwischen Anforderungsanalyse und Implementierung iterativ verringert wird, Anforderungen ermittelt und stabilisiert werden.[279] Dazu werden die Fachmitarbeiter (bzw. Kunden/Endbenutzer) entlang des gesamten System-Lebenszyklus eingebunden.[280] An dieser Stelle geht die vorliegende Arbeit einen anderen Weg zur Optimierung der Anforderungsanalyse, indem aus den durch Fachmitarbeiter annotierten Rechtsvorschriften eine initiale

---

Standardisierung durch das W3C eingerichtet wurde (vgl. http://www.w3.org/Submission/2003/SUBM-EPAL-20031110/).

[273] [Kerrigan&Law, 2003]

[274] Die Homepage des Projektes ist erreichbar unter: http://www.metalex.eu (letzter Aufruf: 21.08.2011).

[275] Metalex ist ein offenes, XML-basiertes Austauschformat für Rechtsquellen. Es wird durch ein XML-Schema definiert. Die Definition der Bestandteile dieses Schemas ist auch als OWL-Ontologie verfügbar.

[276] Hierin liegt ein Unterschied zu Ansätzen, die mit domänenspezifischen Sprachen versuchen, die Aufgabenstellungen innerhalb des Anwendungsbereichs der öffentlichen Verwaltung besser zu formalisieren (vgl. u. a. [Klischewski, 2003], [Palkovits&Karagiannis, 2003], [Kühne&Wenzel, 2006], [Becker et al., 2007], [Heitkoetter, 2011]).

[277] Vgl. [Margaria&Steffen, 2009b].

[278] [Margaria&Steffen, 2009b], S. 94.

[279] Vgl. [Margaria&Steffen, 2009b], S. 95.

[280] Vgl. [Margaria&Steffen, 2009c], S. 490.

Anforderungsspezifikation erzeugt wird. Er erweitert die MDA auf das Vorfeld der Anforderungsanalyse und verbindet es mit den Annotationen in Ausgangsdokumenten. Auch hier können mit geeigneter Werkzeugunterstützung Fachmitarbeiter tätig werden und Ergebnisse produzieren, die in die weitere Entwicklung einfließen. Die Betrachtung weitergehender System-Lebenszyklusphasen liegt jedoch außerhalb des Themas dieser Arbeit, worin ein Unterschied zu XMDD liegt.

Ähnlich dem *One Thing Approach (OTA)*, der verwandt mit XMDD ist, ermöglicht der Ansatz dieser Arbeit die Endbenutzer-orientierte Modellierung, die schrittweise verfeinert werden kann, bis ein ausreichender Detaillierungsgrad für die Implementierung erreicht ist.[281] In OTA wird dazu genau ein zentrales Arbeitergebnis in verschiedenen Dimensionen verfeinert: „[...]there is only one artifact during the whole systems' life cycle. This artifact is successively refined in various dimensions [...]"[282] Weil in dieser Arbeit die MDA eingesetzt wird, kommen hier mehrere Modelle zum Einsatz, die auf einander aufbauen und durch Transformation in einander überführt werden. Der Unterschied resultiert auch aus der Ausrichtung dieser Arbeit auf die Nutzung der Modellierungsstandards der OMG und des W3C, die (mit wenigen Ausnahmen) als Standards und Architekturen für E-Government-Anwendungen (SAGA) vom Beauftragten der Bundesregierung für Informationstechnik anerkannt sind.

[281] [Margaria&Steffen, 2009a], S. 1.
[282] [Margaria&Steffen, 2009a], S. 1.

3 Ontologie des Verwaltungshandelns und ihre Anwendung

In diesem Kapitel wird die im Rahmen dieser Arbeit entwickelte Ontologie des Verwaltungshandelns vorgestellt. Die Ontologie soll verwendet werden, um in den Texten von Rechtsvorschriften Elemente mit einer definierten Semantik zu versehen, die sie als möglichen Ursprung einer Anforderungen kennzeichnet.

Der Ontologie liegt zugrunde, dass die Verwaltung in ihrem Handeln und ihren Entscheidungen an Recht und Gesetz gebunden ist und somit Verwaltungshandeln im Kern insbesondere die Anwendung von Recht darstellt. Für die Anwendung von Recht gibt es etablierte Methoden und Techniken der Rechtswissenschaft zur Arbeit mit Rechtstexten. Bei der Entwicklung der Ontologie wurde von diesen Methoden und Techniken ausgegangen, um relevante Konzepte in Texten von Rechtsvorschriften zu identifizieren. Darauf aufbauend wurde die Anwendung dieser Konzepte innerhalb der Rechtsanwendung formalisiert. Hieraus leiten sich Konzepte des Verwaltungshandeln ab, die auf Konzepte der Rechtsanwendung zurückgehen, diese erweitern und spezialisieren. Da die Ontologie Besonderheiten des Verwaltungshandelns berücksichtigen muss, die über die reine Rechtsanwendung hinausgehen, werden exemplarische Konzepte eingeführt, die keine Entsprechung in der Rechtsanwendung haben und Spezifika des Verwaltungshandelns sind. Analog zu diesen Konzepten ist die Erweiterung der Ontologie um weitere Spezifika möglich.

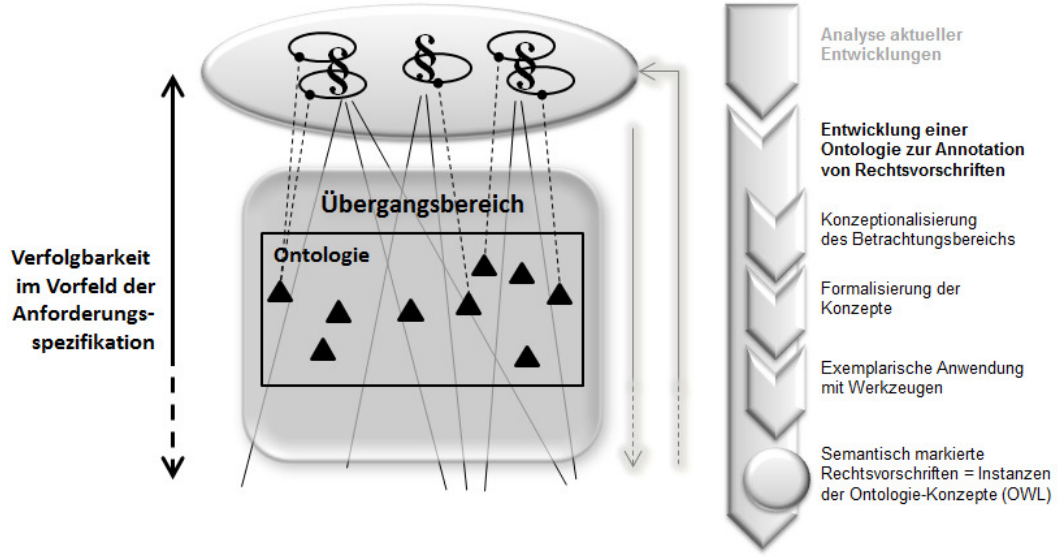


Abbildung 21 – Ontologie und ihre Anwendung auf Rechtsvorschriften als Inhalte dieses Abschnittes

Die Gliederung dieses Abschnittes folgt dem allgemeinen Entwicklungsprozess einer Ontologie. Die Mehrzahl der Methoden des *Ontology Design* bzw. *Ontology Engineering* beschreibt die Entwicklung einer Ontologie, indem diese zunächst als informelle Ontologie entwickelt wird. In einem weiteren Schritt wird dann eine formale Beschreibungssprache ausgewählt und die Ontologie anschließend mit dieser Sprache formal beschrieben.[283] Diese

[283] Beispielsweise unterscheidet die von Uschold und King (u.a. [Uschold&King, 1995]) entwickelte Methode die Erstellung einer Ontologie in Schritte zum Identifizieren von Konzepten und Beschreibung (Ontology capture) von den Schritten der Kodierung (Coding) der Ontologie in einer formalen Sprache. Auch die von Grünniger und Fox (u.a. [Grüninger&Fox, 1995])entwickelte Methode unterscheidet in die Erstellung einer Ontologie und deren anschließende Spezifikation in einer formalen Sprache. Die *Methontology* (vgl. [Fernández-López et al. 1997]) unterscheidet ebenfalls zwischen Schritten zur Entwicklung einer informellen Konzeptionalisierung und deren anschließender Formalisierung.

Arbeit folgt ebenfalls diesem Prinzip. Zunächst wird der Gegenstands- und Betrachtungsbereich der zu entwickelnden Ontologie natürlichsprachig eingegrenzt, indem Begriffe und Zusammenhänge auf allgemein akzeptierte Techniken, Methoden und Definitionen zurückgeführt werden (Abschnitt 3.1). Dies ist notwendig, um das Ontological Commitment (siehe Seite 32 in Abschnitt 2.3) herzustellen. Anschließend wird diese natürlichsprachige Darstellung in eine formale Struktur überführt (Abschnitt 3.2). Dadurch ergeben sich in den nachfolgenden Abschnitten Redundanzen, die aus methodischen Gründen jedoch nicht vollständig vermieden werden können. Die Nutzung der Ontologie in einem Werkzeug und das dadurch erzielte Ergebnis werden in dem anschließenden Abschnitt 3.3 beschrieben. Das Kapitel schließt mit einer Zusammenfassung der Zwischenergebnisse (Abschnitt 3.4), die zum nächsten Kapitel überleitet.

3.1 Konzeptionalisierung

Die Beschreibung der Konzepte des Gegenstandsbereichs der Ontologie (Konzeptionalisierung) gliedert sich in einen Teil, in dem die grundlegenden Konzepte in der Arbeit mit Rechtstexten, basierend auf den rechtstheoretischen Grundlagen, identifiziert werden. Dann werden die darauf aufbauenden allgemeinen Konzepte der Rechtsanwendung identifiziert und beschrieben. Das Ontological Commitment geht auf die Methodenlehre der Rechtswissenschaft und auf allgemein anerkannte Begriffsdefinitionen zurück.[284] Es verwendet als Orientierungsrahmen die vorhandenen Ontologien des Rechts (vgl. Abschnitt 2.3). Anschließend werden die Konzepte der Rechtsanwendung auf den Bereich des Verwaltungshandelns übertragen und um die verwaltungsspezifischen Aspekte ergänzt. Im Sinne des Ontological Commitment wird einerseits die Konsistenz zu den Konzepten der Rechtsanwendung hergestellt. Andererseits werden die Besonderheiten des Verwaltungshandelns berücksichtigt, indem Konzepte der Verwaltungswissenschaften zugrunde gelegt und Konzepte aus den vorhandenen Ontologien der öffentlichen Verwaltung als Ansatzpunkt verwendet werden. Im Ergebnis ist der Gegenstandsbereich abgegrenzt und aufbereitet, so dass sich der Bezug, ausgehend von Rechtssätzen eines Rechtstextes bis zu Handlungen der Verwaltung, herstellen lässt.

3.1.1 Normtext und Rechtssatz

Im Rahmen dieser Arbeit wird von Rechtsvorschriften ausgegangen, die dadurch gekennzeichnet sind, dass Sie Gebote oder Verbote für Verhalten von *Adressaten* innerhalb und außerhalb der Verwaltung beschreiben. In der Regel werden *Rechtsvorschriften* vor Ihrer Anwendung im Verwaltungshandeln weitergehend konkretisiert und detailliert (z.B. in Verwaltungsvorschriften oder Satzungen), um gängigen Maximen des Verwaltungshandelns zu genügen (z.B. der Gleichheit vor dem Gesetz). Für die juristische Rechtsanwendung sind nur die Rechtsvorschriften im engeren Sinne verbindlich. Vorschriften, die sich beispielsweise die öffentliche Verwaltung selbst zur Konkretisierung und Festlegung ihrer Arbeitsweise gegeben hat, sind für den juristischen Rechtsanwender grundsätzlich nicht relevant, wenn er beispielsweise die Zulässigkeit von Verwaltungshandeln im Einzelfall zu prüfen hat.[285] In diesem Abschnitt wird deshalb das Konzept *Rechtsnorm* im Sinne der juristischen Rechtsanwendung eingeführt. Eine Rechtsnorm (syn. Normtext) besteht in ihrer reinen sprachlichen und grammatischen Form aus allgemeinen sprachlichen Elementen (Wörtern, Wortgruppen, Sätzen mit deren Satzteilen) und hat eine inhaltliche Gliederung

[284] Vgl. u.a. [Larenz, 1983], [Larenz, 1991], [Looschelders&Roth, 1996], [Weber, 2000].

[285] Aus Sicht der operativen Verwaltungstätigkeit spielt dieser Unterschied im Vollzug eine untergeordnete Rolle. Es ist für einen Sachbearbeiter weitgehend unerheblich, ob er aufgrund einer Vorgabe des Gesetzgebers oder einer Vorgabe seiner Behörde verpflichtet ist, eine Entscheidung entsprechend zu treffen. Hier werden daher die Aussagen, die zunächst für die Rechtsanwendung von Rechtsnormen getroffen werden, auch auf die Rechtsanwendung in der Verwaltung übertragen. Dies ist zulässig, da es sich bei ihnen lediglich um Konkretisierungen handeln kann, sie also immer an die entsprechenden Rechtsvorschriften gebunden sind.

(z. B. durch Absätze, Abschnitte, Paragraphen und Kapitel).[286] Sie befasst sich immer mit einem bestimmten Thema, hat einen inneren Zusammenhang der Textelemente und ist immer abgeschlossen.[287] Über diese sprachlichen Eigenschaften einer Rechtsnorm hinausgehend kennt die rechtswissenschaftliche Methodenlehre weitere Elemente der Rechtsnorm.[288] Rechtsnormen sind besondere Texte, die ein zusammengehöriges Rechtsgebiet regeln und immer aus Rechtssätzen bestehen. Ein *Rechtssatz* ist die sprachliche Form zum Ausdruck von Bestandteilen und Zusammenhängen einer Rechtsnorm.[289] Er verfügt immer über einen normativen Sinn, indem eine Geltungsanordnung getroffen wird, durch die eine bestimmte Regelung erfolgt bzw. erfolgen soll. Der normative Sinn unterscheidet den Rechtssatz vom Aussagesatz, der lediglich Tatsachenbehauptungen oder Feststellungen enthält. Ein vollständiger Rechtssatz verknüpft im Allgemeinen einen generell und abstrakt beschriebenen *Tatbestand*, der auf eine Vielzahl konkreter *Sachverhalte* zutreffen kann, mit einer oder mehreren ebenso abstrakt beschriebenen *Rechtsfolgen*. Neben diesen *vollständigen Rechtssätzen* gibt es auch *unvollständige Rechtssätze*, die zwar sprachlich vollständige Sätze sind, aber nicht alle Elementtypen eines vollständigen Rechtssatzes aufweisen. Sie dienen dazu, den Tatbestand, die Rechtsfolge oder deren Merkmale näher zu beschreiben (erläuternde Rechtssätze), bestimmte Fallgruppen aus der Anwendung des Rechtssatzes herauszunehmen (einschränkende Rechtssätze) oder auf Elemente einer anderen Rechtsnorm zu verweisen (verweisende Rechtssätze), um Rechtsfolge oder Tatbestand zu bestimmen. Unvollständige Rechtssätze entfalten ihren normativen Sinn somit nur im Zusammenspiel mit anderen Rechtssätzen. Die Unterscheidung der Rechtssätze in vollständige und unvollständige Rechtssätze macht deutlich, dass im Textfluss eines Normtextes die vollständigen Rechtssätze nicht immer unmittelbar aufeinander folgen. Rechtssätze können aber durch Anwendung juristischer Techniken und Methoden aus dem Normtext gewonnen werden (vgl. Abschnitt 3.1.2). Eine Zusammenfassung der identifizierten Konzepte im Gegenstandsbereich ist in der nachfolgenden Tabelle 4 dargestellt.

Name des Konzepts	Beschreibung
Adressat	Adressat der Rechtsvorschrift, die ihm Tun oder Unterlassen vorschreibt oder ihm ein bestimmtes Recht einräumt oder entzieht.
Rechtsnorm	Eine Rechtsnorm ist ein spezieller Text, der sich in der Regel in mehrere inhaltlich zusammengehörige Abschnitte gliedert, die ihrerseits aus Wörtern, Wortgruppen, Sätzen und Satzteilen bestehen und sich in weitere Unterabschnitte gliedern können. Die Rechtsnorm beschreibt die Regelung eines zusammengehörigen Themas bzw. Rechtsgebietes. Per Definition besteht jede Rechtsnorm aus mindestens einem Rechtssatz, in der Regel aber aus mehreren Rechtssätzen.
Rechtsvorschrift	Bezeichnet eine schriftlich fixierte Rechtsnorm, die sich an Adressaten außerhalb und innerhalb der Verwaltung richtet und ihnen Handlungsvorgaben macht.
Rechtssatz	Ein Rechtssatz ist die sprachliche Form zum Ausdruck von Bestandteilen und Zusammenhängen einer Rechtsnorm, die in einem Normtext dokumentiert ist.

[286] Vgl. [DudenBd4, 1998], S. 833ff.
[287] Vgl. [DudenBd4, 1998], S. 833ff.
[288] Eine Alternative wäre es, die Elemente der formalen Gliederung von Rechtsnormen zugrunde zu legen. Sie würden die Verfolgbarkeit zu Abschnitten, Paragraphen, Absätzen usw. der Rechtsnorm ermöglichen. Beispielsweise wäre verfolgbar, dass die §42, §44 und §43 Abs. 1 im Gesetz XYZ mit dem Anwendungsfall „ABC“ in Beziehung stehen. (Der Begriff Anwendungsfall bezeichnet hier ein Konzept der Anforderungsanalyse, das der Spezifikation von funktionalen Anforderungen an das Softwaresystem dient und in Abschnitt 2.2 eingeführt wurde.) Die tatsächliche Bedeutung der durch die Gliederungselemente repräsentierten Informationen bliebe aber ungenutzt. Es ist dann nicht ersichtlich, warum Beziehungen existieren und welcher Art sie sind.
[289] Vgl. hier und im Folgenden [Larenz, 1991], S. 250.

Name des Konzepts	Beschreibung
	Der Rechtssatz verfügt immer über einen normativen Sinn, indem eine Geltungsanordnung getroffen wird, durch die eine bestimmte Regelung erfolgt bzw. erfolgen soll. Es können vollständige und unvollständige Rechtssätze anhand ihrer Bestandteile unterschieden werden. Da eine Rechtsnorm keine bloße Abfolge von Rechtssätze ist, setzt sich der Rechtssatz aus den sprachlichen Bestandteilen des konkret vorliegenden Textes zusammen. Rechtssätze beschreiben als Rechtskonzept dadurch den Tatbestand mit dessen abstrakten Merkmalen und Bedingungen, sowie die vorgesehene rechtliche Konsequenz mit deren abstrakten Merkmalen in sprachlicher Form.
Rechtsfolge	Die Rechtsfolge ist die in einem Rechtssatz für einen Tatbestand vorgesehene abstrakte Beschreibung der rechtlichen Konsequenz.
Sachverhalt	Der Sachverhalt ist die Abstraktion tatsächlich vorliegender tatbestandsrelevanter Eigenschaften. Ein Sachverhalt ist somit ein Ausschnitt eines tatsächlich vorliegenden Umstandes oder einer vorliegenden Situation und deshalb nicht unmittelbarer Bestandteil eines Rechtssatzes.
Tatbestand	Beschreibung bestimmter abstrakter Merkmale und Voraussetzungen, die ein Rechtssatz für das Eintreten einer Rechtsfolge vorsieht.
Text	Eine abgegrenzte, kommunikative Einheit, die aus einer zusammenhängenden Folge von sprachlichen Elementen (Worten, Sätzen, Absätzen, Abschnitten) besteht und ein Thema behandelt.[290]
Unvollständiger Rechtssatz	Ein unvollständiger Rechtssatz ist ein sprachlich vollständiger Satz, der über einen normativen Sinn verfügt. Er weist im Unterschied zu vollständigen Rechtssätzen nicht alle Elemente eines Rechtssatzes auf und wirkt stattdessen auf die Bestandteile eines anderen Rechtssatzes erläuternd, einschränkend oder verweisend.
Vollständiger Rechtssatz	Ein vollständiger Rechtssatz weist alle Elemente eines Rechtssatzes auf. Er verknüpft einen generell und abstrakt beschriebenen Tatbestand, der auf eine Vielzahl konkreter Sachverhalte zutreffen kann, mit einer oder mehreren ebenso abstrakt beschriebenen Rechtsfolgen.

Tabelle 4 – Identifizierte Normtext- und Rechtssatzkonzepte

3.1.2 Rechtsanwendung

Etablierte Methoden[291] geben vor, wie Recht und insbesondere Rechtssätze (mit ihren nachfolgend detaillierter beschriebenen Elementen) auf einen Einzelfall anzuwenden sind.[292] Dies beginnt grundsätzlich mit der *Rechtsfindung*, in deren Rahmen die auf den vorliegenden Einzelfall anzuwendenden Rechtsnormen identifiziert werden. Sie sind nicht auf methodisch vorgezeichnetem Wege ermittelbar, denn es kommt vielmehr auf die Intuition, Urteilskraft, Erfahrung und Kenntnisse des Rechtsanwenders an.[293] Zunächst wird daher der im konkreten Einzelfall vorliegende *Lebenssachverhalt* aufbereitet.[294] Der Lebenssachverhalt ist

[290] Vgl. [DudenBd4, 1998], S. 833ff.

[291] Vgl. hier und im Folgenden [Larenz, 1983], [Larenz, 1991], [Looschelders&Roth, 1996], [Weber, 2000].

[292] Die hier beschriebenen Rechtshandlungen erfolgenden durch einen juristischen Rechtsanwender (z.B. einen Richter).

[293] Vgl. [Bierling, 1911],S. 45ff. zitiert nach [Looschelders&Roth, 1996], S. 86.

[294] Auf den Unterschied zwischen dem tatsächlich vorliegenden oder erdachten Lebenssachverhalt und dem Sachverhalt, der die tatbestandsrelevanten Eigenschaften umfasst, wird in der Literatur verwiesen, ohne jedoch einen expliziten Schritt in der Rechtsanwendung vorzusehen. Es wird stattdessen implizit vorausgesetzt, dass im Schritt der Subsumtion die Abstraktion des rechtlich relevanten Sachverhaltes bereits erfolgt ist. Im Schritt der Rechtsfindung kann dies aber typischerweise nicht erfolgt sein, da „nur" nach den Rechtsnormen gesucht wird, die die Grundlage

durch eine beliebige Menge von Eigenschaften gekennzeichnet, von denen für die Rechtsanwendung jedoch nur eine definierte Teilmenge relevant ist. Diese Teilmenge umfasst alle Eigenschaften, denen „im Angesicht einer bestimmten Norm rechtliche Bedeutung zugemessen werden kann."[295] Einer Eigenschaft kommt beispielsweise dann eine rechtliche Bedeutung zu, wenn der in einem Rechtssatz der Rechtsnorm beschriebene Tatbestand eine Bedingung an diese Eigenschaft stellt. Die tatbestandsrelevanten Eigenschaften des Lebenssachverhalts müssen herausgearbeitet, abstrahiert und in die subsumtionsfähige Form eines rechtlich relevanten *Sachverhaltes* transformiert werden. In der juristischen Rechtsanwendung kann es damit einhergehend notwendig sein, die abstrakt formulierten Tatbestandsbedingungen hinsichtlich ihrer Bedeutung zu konkretisieren. (In der Rechtsanwendung der öffentlichen Verwaltung ist dies in der Regel bereits im Vorfeld der Rechtsanwendung erfolgt und ihre Bedeutung in verwaltungsinternen Regelungen und sonstigen Konkretisierungen definiert.) Daran schließt sich die *Subsumtion* an, die die Unterordnung des Sachverhaltes unter den Tatbestand der Rechtsnorm bezeichnet. Der *Tatbestand* wird im Allgemeinen als eine Beschreibung bestimmter abstrakter Merkmale und Voraussetzungen verstanden, die ein Rechtssatz für das Eintreten einer Rechtsfolge vorsieht. Der Tatbestand kann somit auch als eine Menge von Sachverhalten aufgefasst werden, zu der alle Sachverhalte gehören, deren Eigenschaften den Bedingungen der Tatbestandsmerkmale genügen.[296] Wird festgestellt, das der Sachverhalt unter den Tatbestand subsumiert werden kann, so bedeutet dies, das der Sachverhalt ein Element der Menge ist, die der Tatbestand bezeichnet. Tatbestand und Sachverhalt werden im Folgenden insbesondere dadurch unterschieden, dass unter abstrakten Tatbestandmerkmalen immer Bedingungen bzw. Voraussetzungen verstanden werden, denen konkrete Eigenschaften eines Sachverhaltes genügen müssen. Die Prüfung, ob der im Einzelfall vorliegende Sachverhalt den Bedingungen des Tatbestandes genügt und deshalb unter den Tatbestand subsumiert werden kann, vollzieht sich in einzelnen Schritten. In jedem Schritt wird jeweils einem Merkmal des Tatbestandes die korrespondierende Eigenschaft des Sachverhaltes gegenübergestellt und geprüft, ob die Sachverhaltseigenschaft den Bedingungen genügt, die durch das Tatbestandsmerkmal formuliert sind. Dieser Schritt wird für jedes weitere Tatbestandsmerkmal wiederholt. Sind alle Merkmale des Tatbestandes durch den Sachverhalt erfüllt, so ist die Subsumtion gelungen und der vorliegende Sachverhalt ist ein Element der durch den Tatbestand bezeichneten Menge. Ist dies nicht der Fall, hat ein Tatbestandsmerkmal keine Entsprechung bzw. ergibt die Prüfung der Bedingungen, dass ein Merkmal durch den Sachverhalt nicht erfüllt ist, so scheitert die Subsumtion. Bei der Subsumtion kann dem Rechtsanwender ein *Beurteilungsspielraum* bei der Wertung von Tatbestandsmerkmalen vor dem Hintergrund eines konkreten Sachverhaltes zugestanden werden. Ist die Subsumtion gelungen, so wird in einem nächsten Schritt der Rechtsanwendung die Rechtsfolgenseite angewendet (*Rechtsfolgenbestimmung*). In der Rechtsfolgenbestimmung wird die Rechtsfolge bzw. werden die Rechtsfolgen ermittelt, die in der Rechtsnorm bei erfülltem Tatbestand vorgesehen ist bzw. sind. Die *Rechtsfolge* ist die in einem Rechtssatz für einen Tatbestand vorgesehene abstrakte Beschreibung der rechtlichen Konsequenz. Sie kann daher als eine Menge von konkreten *Maßnahmen* aufgefasst werden, zu der alle Maßnahmen gehören, die der abstrakten Beschreibung der Rechtsfolge genügen. Der Rechtsanwender bestimmt die rechtlich zulässigen Rechtsfolgen, die von einer Rechtsnorm für den vorliegenden Tatbestand vorgesehen sind. Nachdem die abstrakte Rechtsfolge bestimmt wurde, die in der Rechtsnorm für den vorliegenden Tatbestand vorgesehen ist, muss sie in einem nächsten Schritt der Rechtsanwendung für den vorliegenden Einzelfall konkretisiert werden. In diesem Schritt der *Einzelfallfestsetzung* sind die Merkmale der

---

der weiteren Rechtsanwendung sind. Im Rahmen dieser Arbeit wird die Sachverhaltsermittlung als expliziter Schritt der Rechtsanwendung ausgewiesen, um Klarheit der Darstellung zu fördern. Dies schließt nicht aus, dass dieser Schritt auch als Teilschritt der Subsumtion oder der Rechtsfindung gesehen werden kann.

[295] [Looschelders&Roth, 1996], S. 93.

[296] Vgl. [Looschelders&Roth, 1996], S. 88.

abstrakten Rechtsfolge zu identifizieren und für jedes dieser Merkmale eine konkrete Ausprägung für den Einzelfall vorzusehen. Häufig wird in der Literatur nicht zwischen Rechtsfolgenbestimmung und Einzelfallfestsetzung unterschieden. Diese Verkürzung auf einen Schritt ist unzulässig; beide Schritte sind klar voneinander zu trennen.[297] Denn im Rechtssatz ist eine abstrakte Rechtsfolge beschrieben, die mindestens durch entsprechende Angaben über Personen, Orte und Zeitpunkte konkretisiert werden muss. In komplexen Fällen ist es auch möglich, dass zunächst eine vorzusehende konkrete Maßnahme als eine „Form" der Rechtsfolge abgeleitet werden muss. Abschließend wird ein Schritt durchgeführt, der in der Darstellung der Rechtsanwendung in der Regel vernachlässigt wird. Die im Rahmen der Rechtsanwendung vorgenommene Einzelfallfestsetzung muss in der Regel dem bzw. den Betroffenen mitgeteilt werden, damit sie ihre Rechtswirkung entfalten kann (*Mitteilung*). Darüber hinaus können Aktivitäten für die tatsächliche Umsetzung der getroffenen Entscheidung notwendig sein, die jedoch über die methodischen Schritten der klassischen Rechtsanwendung hinausgehen und deshalb innerhalb der hier formalisierten Konzepte der Rechtsanwendung keine Entsprechung haben.

Die rechtswissenschaftliche Methodenlehre richtet sich primär an Personen, die Recht anwenden. Ihre Aufgabe ist es, aus Normtexten zunächst die Rechtssätze mit ihren Bestandteilen abzuleiten, die daraus resultierenden Rechtshandlungen durchzuführen und dabei die rechtlich relevanten Handlungsgegenstände zu bearbeiten. Diese Personen nehmen die Rolle eines *Rechtsanwenders* ein. Ein Normtext richtet sich aber nicht (nur) an den Rechtsanwender, sondern an weitere Adressaten, deren Tun oder Unterlassen vorgeschrieben oder denen ein bestimmtes Recht eingeräumt oder entzogen wird. Adressaten sind somit Beteiligte oder Betroffene (hier kurz *Adressat*) oder werden dies im Laufe der Rechtsanwendung. Damit die Maßnahme zur Umsetzung gelangt, müssen sich Aktivitäten zur *Mitteilung* der getroffenen Entscheidung und ggf. der tatsächlichen Ausführung der Entscheidung anschließen. Da die tatsächliche Ausführung über die klassische Rechtsanwendung im methodisch-rechtlichen Sinne hinausgeht, ist diese Aufgabe nicht zwangsläufig beim Rechtsanwender und nicht beim Adressaten selbst zu sehen. In Bezug auf die tatsächliche Ausführung bleibt hier zunächst eine konzeptionelle Lücke, die durch Konzepte des Verwaltungshandelns und der Verwaltungsakteure geschlossen werden muss.

Die identifizierten Konzepte können in Anlehnung an die LRI-Core ontology of law (vgl. Abschnitt 2.3) in Konzepte für Handlungen, für mentale oder physische Handlungsgegenstände und Akteure unterschieden werden.[298] Die nachfolgenden Tabellen fassen die identifizierten Konzepte der Rechtsanwendung entsprechend dieser Einteilung und die sich für das Konzept des Rechtssatzes ergebenden neuen Aspekte zusammen.

Name des Konzepts	Beschreibung
Einzelfallfestsetzung	Ein Schritt der Rechtsanwendung, in dessen Rahmen aus einer abstrakten Rechtsfolge durch Konkretisierung vor dem Hintergrund des vorliegenden Einzelfalls eine konkrete Maßnahme abgeleitet und festgesetzt wird.
Mitteilung	Ein Schritt der Rechtsanwendung, in dessen Rahmen die vorgenommene Einzelfallfestsetzung gegenüber dem/den Adressaten verbindlich kommuniziert wird, u.a. damit die Einzelfallfestsetzung ihre Rechtswirkung entfalten kann.
Rechtsfindung	Ein Schritt der Rechtsanwendung, in dessen Rahmen die auf den vorliegenden

[297] Vgl. hier und im Folgenden [Larenz, 1983], S. 265.

[298] Die LRI-Core ontology of law unterscheidet Occurrences, *Roles*, *Abstract Concepts*, *Mental Concepts*, *Physical Concepts*, *Processes*, sowie *Quality* und *Quantity*. Im Rahmen dieser Arbeit werden Mental Concepts und Physical Concepts gemeinsam als Handlungsgegenstände betrachtet, da eine weitergehende Unterscheidung für das Ziel dieser Arbeit nicht relevant ist. Processes werden als Handlungen und Roles als Akteure verwendet. Die Occurrences, Abstract Entities, Quality und Quantity sind für die Anwendung im Ansatz dieser Arbeit nicht relevant.

Name des Konzepts	Beschreibung
	Einzelfall anzuwendenden Rechtsnormen identifiziert werden.
Rechtsfolgenbestimmung	Ein Schritt der Rechtsanwendung, in dessen Rahmen eine oder mehrere abstrakte Rechtsfolgen, die von einer Rechtsnorm für einen Tatbestand vorgesehen und rechtlich zulässig sind, bestimmt werden.
Sachverhaltsermittlung	Ein Schritt der Rechtsanwendung, in dessen Rahmen aus dem tatsächlich vorliegenden Lebenssachverhalt alle tatbestandsrelevanten Eigenschaften abstrahiert und in die subsumtionsfähige Form des rechtlich relevanten Sachverhalts gebracht werden.
Subsumtion	Ein Schritt der Rechtsanwendung, in dessen Rahmen versucht wird, einen vorliegenden Sachverhalt unter den Tatbestand zu subsumieren. Vollzieht sich für jedes Tatbestandsmerkmal, indem jeweils die zu einem Merkmal/einer Bedingung des Tatbestandes korrespondierende Eigenschaft des Sachverhaltes gegenübergestellt und geprüft wird, ob die Sachverhaltseigenschaft den Bedingungen genügt, die durch das Tatbestandsmerkmal formuliert sind. Der Schritt endet mit der Entscheidung, ob alle Merkmale des Tatbestandes durch den Sachverhalt erfüllt sind und damit die Subsumtion gelungen ist, oder ob dies nicht der Fall ist, weil ein Tatbestandsmerkmal keine Entsprechung hat bzw. die Prüfung ergibt, dass ein Merkmal durch den Sachverhalt nicht erfüllt ist.

Tabelle 5 – Konzepte für Rechtshandlungen

Name des Konzepts	Beschreibung
Lebenssachverhalt	Der Lebenssachverhalt ist ein tatsächliches Ereignis oder eine tatsächlich vorliegende Situation, die durch eine beliebige Menge von Eigenschaften gekennzeichnet ist, von denen für die Rechtsanwendung jedoch nur eine durch den Sachverhalt definierte Teilmenge relevant ist. (syn. Einzelfall)
Maßnahme	Konkretisierung einer abstrakten Rechtsfolge, in dem alle abstrakten Merkmale der Rechtsfolge für den vorliegenden Einzelfall konkretisiert werden. Die Maßnahme ist Element der Menge von möglichen Maßnahmen, die durch die Rechtsfolge repräsentiert werden.
Maßnahmeneigenschaft	Eine konkretisierte Eigenschaft einer Maßnahme wird durch eine Instanz des Konzepts Maßnahmeneigenschaft ausgedrückt. Es ist immer eine Konkretisierung eines abstrakten Merkmales einer Rechtsfolge.
Rechtsfolge	Die Rechtsfolge ist die in einem Rechtssatz für einen Tatbestand vorgesehene abstrakte Beschreibung der rechtlichen Konsequenz und kann als eine Menge von konkreten Maßnahmen aufgefasst werden, zu der alle Maßnahmen gehören, die der abstrakten Beschreibung der Rechtsfolge genügen. Das Konzept baut auf dem gleichnamigen Konzept in Tabelle 4 auf, begründet seine Existenz in der Rechtsanwendung (auch) durch sein Vorkommen in den Texten von Rechtsvorschriften.
Rechtsfolgenmerkmal	Ein Rechtsfolgenmerkmal ist die abstrakte Beschreibung eines Merkmals der Rechtsfolge, das in einem Rechtssatz für einen Tatbestand vorgesehen ist. Dieses Merkmal muss für die Anwendung auf einen vorliegenden Einzelfall konkretisiert werden. Das Konzept baut auf dem Konzept „Rechtsfolge" in Tabelle 4 auf, begründet seine Existenz in der Rechtsanwendung (auch) durch sein Vorkommen in den Texten von Rechtsvorschriften.

Name des Konzepts	Beschreibung
Rechtssatz	Rechtssätze beschreiben den Tatbestand mit dessen abstrakten Merkmalen und Bedingungen sowie die vorgesehene rechtliche Konsequenz mit deren abstrakten Merkmalen in sprachlicher Form. Rechtssätze können aus Normtexten gewonnen werden, in dem Wörter, Wortgruppen, Sätze/Teilsätze durch einen Rechtsanwender interpretiert und in ihnen Rechtssätze mit ihren Bestandteilen identifiziert und zueinander in Beziehung gesetzt werden. Das Konzept baut auf dem gleichnamigen Konzept in Tabelle 4 auf, begründet seine Existenz in der Rechtsanwendung (auch) durch sein Vorkommen in den Texten von Rechtsvorschriften.
Sachverhalt	Der Sachverhalt ist das Ergebnis der Extraktion und Abstraktion tatbestandsrelevanter Eigenschaften eines vorliegenden Lebenssachverhalts in einer subsumtionsfähigen Form. Die tatbestandsrelevanten Eigenschaften ergeben sich aus den Eigenschaften, die der Tatbestand aufweist, unter den der Sachverhalt subsumiert werden soll. Das Konzept baut auf dem gleichnamigen Konzept in Tabelle 4 auf.
Sachverhaltseigenschaft	Eine Sachverhaltseigenschaft ist genau eine tatbestandsrelevante Eigenschaft, die durch den vorliegenden Lebenssachverhalt belegt wird. Das Konzept baut auf dem Konzept „Sachverhalt" in Tabelle 4 auf.
Tatbestand	Der Tatbestand ist die Beschreibung bestimmter abstrakter Merkmale und Voraussetzungen, die ein Rechtssatz für das Eintreten einer Rechtsfolge vorsieht. Der Tatbestand kann als eine Menge von Sachverhalten aufgefasst werden, zu der alle Sachverhalte gehören, deren Eigenschaften den Bedingungen der Tatbestandsmerkmale genügen. Das Konzept baut auf dem gleichnamigen Konzept in Tabelle 4 auf, begründet seine Existenz in der Rechtsanwendung (auch) durch sein Vorkommen in den Texten von Rechtsvorschriften.
Tatbestandsmerkmal	Ein Tatbestandsmerkmal ist genau ein abstraktes Merkmal oder eine Voraussetzung, die für das Vorliegen eines Tatbestandes gegeben oder erfüllt sein muss. Das Konzept baut auf dem Konzept „Tatbestand" in Tabelle 4 auf, begründet seine Existenz in der Rechtsanwendung (auch) durch sein Vorkommen in den Texten von Rechtsvorschriften.

Tabelle 6 – Konzepte für Gegenstände von Rechtshandlungen

Name des Konzepts	Beschreibung
Adressat	Adressat der Rechtsvorschrift, die ihm Tun oder Unterlassen vorschreibt oder ihm ein bestimmtes Recht einräumt oder entzieht.
Rechtsanwender	Person oder Organisation, die Recht anwendet, d.h. aus Rechtsnormen zunächst die Rechtssätze mit ihren Bestandteilen ableitet, die daraus resultierenden Rechtshandlungen durchführt und dabei die rechtlich relevanten Handlungsgegenstände bearbeitet.

Tabelle 7 – Konzepte für Akteure in der Rechtsanwendung

3.1.3 Verwaltungshandeln

In diesem Abschnitt werden die zur Rechtsanwendung korrespondierenden Konzepte des Verwaltungshandelns identifiziert. Damit werden jetzt nicht mehr nur Rechtsvorschriften im engeren juristischen Sinne, sondern auch ihre Konkretisierungen (z.B. Verwaltungsvorschriften, Satzungen) betrachtet. Die öffentliche Verwaltung verwendet die Konzepte der Rechtsanwendung in spezifischer Weise, ergänzt sie durch weitere Aspekte und neue

ausschließlich verwaltungsspezifische Konzepte. Dies kann nicht nur durch Unschärfen und Mehrdeutigkeiten in der Terminologie der beiden Anwendungsbereiche und durch die Fokussierung der Verwaltung auf Rechtsnormen mit administrativem Vollzugscharakter erklärt werden. Es gibt vielmehr verwaltungsspezifische Besonderheiten, aufgrund derer eine direkte Übernahme der Konzepte der Rechtsanwendung in den Anwendungsbereich des Verwaltungshandelns zu kurz greift. Beispielsweise beschränkt sich Verwaltungshandeln nicht auf die Prüfung der Rechtmäßigkeit einer Rechtsfolge, sondern wählt von mehreren möglichen und rechtlich zulässigen Rechtsfolgen diejenige, die beispielsweise den größten Beitrag zur Zielerreichung liefert. Dies unterscheidet den Rechtsanwender in der Verwaltung vom Juristen. Der juristische Rechtsanwender kann mit „dem ihm zur Verfügung stehenden wissenschaftlichen Instrumentarium nicht feststellen, welche von mehreren zulässigen Alternativen einer Ermessensentscheidung die beste ist. Dieses ist nur durch die Anwendung verwaltungswissenschaftlicher Methoden möglich."[299]

Obgleich Verwaltungshandeln sehr vielgestaltig ist, wird es häufig danach typisiert, wie stark die Handlungen und Aufgaben durch Gesetze[300] determiniert sind.[301] Becker bezeichnet dies als Grad der Aufgabendeterminiertheit und stellt verschiedene Typen von Gesetzen modellhaft dar.[302] Sein Modell benennt auch Merkmale, die auf durch Gesetze determinierte Handlungsgegenstände des Verwaltungshandelns schließen lassen. Das Modell macht grundsätzlich deutlich, dass Verwaltungshandeln aus Sicht des Rechtsanwenders in der Verwaltung entweder determiniert ist oder ihm bewusst ein Beurteilungs- bzw. Ermessensspielraum eingeräumt wurde, der zu nutzen ist.[303] Auf der Ebene der Sachbearbeitung liegen beispielsweise nur in seltenen Fällen keine Angaben über Auslöser, über die zu prüfenden tatsächlichen Umstände, über Maßnahmen und Auswahlkriterien vor. Sollte dies doch der Fall sein, so ist davon auszugehen, dass die Elemente des Verwaltungshandelns, die auf der operativen Ebene nicht geregelt sind, nachträglich geregelt werden müssen oder dem Beurteilungsspielraum bzw. dem Ermessen des Bearbeiters unterliegen. Unterliegen sie nicht dem Beurteilungsspielraum oder dem Ermessen, so werden sie als Regelungslücke geschlossen oder als Ausnahme- bzw. Sonderregelung getroffen.

Betrachtet man die im Modell von Becker definierten einzelnen Typen hinsichtlich des Merkmals der Aufgabendeterminiertheit weitergehend, werden Parallelen zu den Konzepten der Rechtsanwendung deutlich. Nach diesem Modell sind immer die *Ziele und Zwecke* des Verwaltungshandelns und bei mehreren Zielen und Zwecken deren Prioritäten festgelegt. In der Rechtsanwendung spielen Ziele und Zwecke von Rechtsnormen hingegen dann eine Rolle, wenn konkurrierende Rechtssätze aufeinander stoßen und der Jurist nach dem Ziel und Zweck der zugrunde liegenden Rechtsnormen fragt, um im vorliegenden Fall den richtigen Rechtssatz zur Anwendung zu bringen. Da die Verwaltung aber nicht, wie in der Rechtsanwendung üblich, lediglich die Entscheidung trifft, ob und welche Maßnahmen rechtlich zulässig sind oder welcher Rechtssatz zur Anwendung gebracht werden kann, sondern von den möglichen Maßnahmen diejenige wählt, die am besten zur Zielerreichung geeignet ist, benötigt die Verwaltung immer diese Zielvorgaben, aus denen ein Zielsystem konstruiert wird, das während der Entscheidungsfindung benutzt wird. „Um zum Zweck der eigenen Tätigkeit zu gelangen […], kann auf die Absicht des Gesetzgebers abgestützt werden, die mit der Aufgabe verfolgt wird."[304] Hierin liegt ein Unterschied zum

[299] [Thieme, 1981], S. 32 f.

[300] Die Typisierung erfolgt bezogen auf Gesetze, wie sie von der Legislative verabschiedet wurden. Deshalb wird hier der Begriff Gesetz verwendet, um ihn von der eingangs definierten Rechtsvorschrift in erweiterter Bedeutung abzugrenzen. Luhmann und Becker verwenden stattdessen z.B. den Begriff des legislativen bzw. politischen Handlungsprogramms.

[301] Vgl. [Luhmann, 1983], S. 207.

[302] Vgl. hier und im Folgenden [Becker, 1989], S. 451 ff.

[303] Beispielsweise muss die Bewertung von Persönlichkeitsmerkmalen (z.B. der Zuverlässigkeit, des Verhaltens oder der beruflichen Leistungsfähigkeit) immer von einem Menschen verantwortet werden und darf nicht automatisiert getroffen werden (§6a BDSG).

[304] [Schedler&Proeller, 2003], S. 123.

vergleichbaren Konzept der Rechtsanwendung, so dass für die Verwaltung ein spezielles Konzept für *Ziele und Zwecke* eingeführt werden muss. Eine auf den Vollzug ausgerichtete Rechtsvorschrift kann abstrakt beschreiben, wann Handeln der Verwaltung möglich oder erforderlich ist (z.B. Handeln auf Antrag oder von Amts wegen). Hierfür wird für das Verwaltungshandeln das Konzept *Auslöser* eingeführt, der ein rechtlich relevantes Ereignis darstellt, dessen Auftreten von der Verwaltung wahrgenommen wird und Verwaltungshandeln ermöglicht oder erfordert.[305] Die abstrakte Beschreibung, wann Handeln möglich und erforderlich ist, lässt auch auf Merkmale und Eigenschaften von Situationen schließen, in denen Parallelen zum Rechtskonzept des Tatbestandes deutlich werden. Der Begriff „Tatbestand" wird aber häufig auf eine strafrechtliche Bedeutung reduziert. Er benennt jedoch im Allgemeinen Voraussetzungen und Bedingungen für das Eintreten einer Rechtsfolge und findet nicht nur im Strafrecht Anwendung. Um Fehlinterpretationen der Begriffsbedeutung zu vermeiden und andererseits den Voraussetzungscharakter zu betonen, wird anstelle des Tatbestand-Begriffs der Begriff *Voraussetzung* für das korrespondierende Verwaltungskonzept verwendet. Eine Voraussetzung wird dabei als Zusammenfassung einzelner Voraussetzungsbedingungen verstanden, die bei der Entscheidung zugrunde gelegt werden. Der *Sachverhalt* ist im Verwaltungshandeln reduziert auf seine für das Verwaltungshandeln relevanten Eigenschaften, die als *Sachverhaltseigenschaften* zu den *Voraussetzungsbedingungen* korrespondieren. Die Ausprägungen dieser Eigenschaften werden aus dem tatsächlichen *Lebenssachverhalt* des vorliegenden Einzelfalls ermittelt und der Verwaltungstätigkeit zugrunde gelegt.[306] Die beiden Konzepte Sachverhalt und Lebenssachverhalt entsprechen damit den gleichnamigen Konzepten der Rechtsanwendung. Bei der Beurteilung des Sachverhaltes kann der Bearbeiter einen *Beurteilungsspielraum* haben, der sich auf nicht hinreichend oder nicht vollständig definierte Tatbestandsmerkmale bezieht. Auf Vollzug ausgerichtete Rechtvorschriften umfassen auch abstrakte Beschreibungen der möglichen Alternativen des Verwaltungshandelns und deren Eigenschaften auf der Rechtsfolgenseite. Sie entsprechen dem Konzept der *Rechtsfolge* und dem Konzept des Rechtsfolgenmerkmals. Im Zuge des Verwaltungshandelns werden für einen konkreten Einzelfall die Rechtsfolge und die Ausprägungen der Rechtsfolgenmerkmale festgelegt. Weil diese Arbeit E-Government mit dem Fokus auf Bürgerdienste betrachtet, kann davon ausgegangen werden, dass Rechtsfolgen in Form eines aus dem Blickwinkel eines Dritten wahrnehmbaren Ergebnissen (Leistungen der Verwaltung) erbracht werden.[307] Entsprechend wird hier der Begriff *Verwaltungsleistung* anstelle des Begriffs Maßnahme der Rechtsanwendung verwendet. Es können auch die *Auswahlkriterien* für die Auswahl der Rechtsfolge definiert sein. Sowohl bei dieser Wahl als auch bei der konkreten Ausgestaltung von Eigenschaften der Verwaltungsleistung (z.B. Höhe des Geldleistung oder des Bußgeldes) kann dem Rechtsanwender in der Verwaltung ein Entscheidungsspielraum zukommen, der als *Ermessen* bezeichnet wird. Die *Ausführung* der ausgewählten Maßnahme und deren Erfolgskontrolle sind der Abschluss des Modells von Becker.

Alle identifizierten Konzepte bilden zunächst ein Zwischenergebnis für das hier zusammenfassend argumentiert wird, dass die Handlungsgegenstände Lebenssachverhalt, Sachverhalt, Tatbestand, Rechtsfolge und Maßnahme (jeweils mit Ihren Eigenschaften bzw. Merkmalen und Beziehungen) als Konzepte der Rechtsanwendung in der Vollzugsfunktion der öffentlichen Verwaltung in den Konzepten Lebenssachverhalt, Sachverhalt, Voraussetzung, Rechtsfolge und Leistung eine Entsprechung haben.[308] Neu hinzu kommen spezielle Konzepte für den Auslöser, die Ziele und Zwecke und die Auswahlkriterien für die Maßnahmenwahl. Sie bilden zusammen den Gegenstand des Verwaltungshandelns bzw. die Handlungsgegenstände.

[305] Vgl. [Menne-Haritz, 1999], S. 335 f.
[306] Vgl. VwVfG §§ 24 (1), 26.
[307] Vgl. [Schedler&Proeller, 2003], S. 63.
[308] Vgl. u.a. [Reichard, 1987], S. 29; [Thieme, 1981], S. 31.

Name des Konzepts	Beschreibung
Antrag	Der Antrag repräsentiert eine spezielle Form des Handlungsauslösers von Verwaltungshandeln.
Auswahlkriterien	Bei Vorliegen der Voraussetzungen werden die in Rechtsvorschriften dokumentierten Auswahlkriterien für die Auswahl einer von mehreren möglichen Rechtsfolgen herangezogen. Dieses Konzept hat keine Entsprechung in den hier identifizierten Konzepten der Rechtsanwendung.
Beurteilungsspielraum	Entscheidungsspielraum, der sich während der Beurteilung von Sachverhaltseigenschaften vor dem Hintergrund nicht vollständig oder hinreichend definierter Voraussetzungsbedingungen ergibt.
Ermessen	Entscheidungsspielraum, der sich bei der Wahl einer Rechtsfolge und/oder der Ausgestaltung der resultierenden Verwaltungsleistungseigenschaft ergibt.
Handlungsauslöser	Repräsentiert den Auslöser des Verwaltungshandelns, d.h. das Auftreten eines rechtlich relevanten Ereignisses, das Verwaltungshandeln auslöst.
Lebenssachverhalt	Der Lebenssachverhalt ist ein tatsächliches Ereignis oder eine tatsächlich vorliegende Situation, die durch eine beliebige Menge von Eigenschaften gekennzeichnet ist, von denen für die Rechtsanwendung in der Verwaltung jedoch nur eine definierte Teilmenge relevant ist.
Rechtsfolge	Die Rechtsfolge ist die in einem Rechtssatz bei Vorliegen der Voraussetzungen vorgesehene abstrakte Beschreibung der rechtlichen Konsequenz. Ihre rechtliche Zulässigkeit wird durch die Methoden der Rechtsanwendung ermittelt, ihre Auswahl aber wird auch durch ihren Beitrag zur Zielerreichung des Verwaltungshandelns bewertet. Eine Rechtsfolge kann als eine Menge von konkreten Leistungen aufgefasst werden, zu der alle Leistungen gehören, die der abstrakten Beschreibung der Rechtsfolge genügen.
Sachverhalt	Ergebnis der Extraktion und Abstraktion tatbestandsrelevanter Eigenschaften eines real vorliegenden Lebenssachverhalts in einer entscheidungsfähigen und damit auch subsumtionsfähigen Form. Ein Sachverhalt weist neben der Subsumtionsfähigkeit auch die Entscheidbarkeit hinsichtlich der Ziele des Verwaltungshandelns auf.
Sachverhaltseigenschaft	Repräsentiert genau ein aus Sicht des Verwaltungshandelns relevantes Merkmal eines Sachverhalts.
Voraussetzung	Zusammenfassung bestimmter abstrakter Merkmale und Voraussetzungen, die ein Rechtssatz für das Eintreten einer Rechtsfolge vorsieht. Sie kann als eine Menge von Sachverhalten aufgefasst werden, zu der alle Sachverhalte gehören, deren Eigenschaften den Bedingungen der Voraussetzung genügen. Hinweis: Das Konzept übernimmt nicht die Bezeichnung Tatbestand in das Verwaltungshandeln, um deutlich zu machen, dass hiermit die Voraussetzung für die Erbringung einer Verwaltungsleistung in der Leistungsverwaltung und nicht primär ein Straftatbestand in der Ordnungsverwaltung oder richterlichen Rechtsanwendung gemeint ist.
Voraussetzungs-bedingung	Repräsentiert einen Teil der Voraussetzung, der für sich genommen eine vollständige Bedingung an einen vorliegenden Sachverhalt darstellt.
Verwaltungsleistung	Konkretisierung einer abstrakten Rechtsfolge für den vorliegenden Einzelfall. Sie kann als Element einer Menge von ggf. mehreren möglichen Leistungen aufgefasst werden, die durch die Rechtsfolge repräsentiert werden. Das Streben nach optimaler Zielerreichung führt dazu, dass die Konkretisierung anhand der Ziele des Verwaltungshandelns durchgeführt wird. Hinweis: Das Konzept übernimmt nicht die allgemeine Bezeichnung Maßnahme in das Verwaltungshandeln, um deutlich zu machen, dass in der

Name des Konzepts	Beschreibung
	Leistungsverwaltung Maßnahmen in Form von Leistungen erbracht werden. Semantisch sonst identisch.
Verwaltungsleistungs-eigenschaft	Repräsentiert genau eine Eigenschaft der Verwaltungsleistung, die bezogen auf den Einzelfall unterschiedlich ausgeprägt sein kann.
Ziele & Zwecke	Die Zwecke und Ziele repräsentieren Bestandteile von Rechtsvorschriften, die auf den Vollzug in der Verwaltung ausgerichtet sind und geben die Motivation für die Festlegungen und die beabsichtigte Wirkung an. Sie werden von der Verwaltung als Bestandteil eines Zielsystems verwendet, an dem sich die öffentliche Verwaltung bei Maßnahmenwahl und Leistungsfestlegung orientiert.

Tabelle 8 – Identifizierte Konzepte für Handlungsgegenstände im Verwaltungshandeln

Durch die Identifikation dieser Konzepte als Gegenstände des Verwaltungshandelns ist noch keine Aussage darüber getroffen, in welchen Handlungen sie verwendet werden. Zwar liegen dem Modell von Becker die aggregierten Phasen eines generellen administrativen Handlungsvorgangs zugrunde, tatsächlich erfolgt die eigentliche Fertigung von Verwaltungsleistungen in der Regel in Form eines Verwaltungsverfahrens. Unter einem Verwaltungsverfahren wird „die nach außen wirkende Tätigkeit der Behörden, die auf die Prüfung der Voraussetzungen, die Vorbereitung und den Erlass eines Verwaltungsaktes oder auf den Abschluss eines öffentlich-rechtlichen Vertrags gerichtet ist“[309] und den Erlass des Verwaltungsaktes einschließt, verstanden.[310] Aus verwaltungstheoretischer Sicht ist ein Verfahren eine konkrete Instanz eines sozialen Handlungssystems, das verbindliche Entscheidungen in Anwendung von Rechtsvorschriften (bei Luhmann: legislativer Handlungsprogramme) in einem komplexitätsreduzierten Handlungszusammenhang erstellt.[311] Ein solches soziales System weist eine bestimmte Struktur auf, indem es von der Umwelt abgegrenzt ist, ganz bestimmte Informationstypen aus seiner Umwelt aufnimmt und an diese abgibt, verschiedene (Beteiligten-)Rollen kennt, eine begrenzte zeitliche Dauer hat und dadurch das Treffen einer Entscheidung garantiert. Aufgrund dieser Merkmale des Verfahrens reduziert sich die Komplexität, da die unendliche Vielzahl möglicher Verhaltensweisen begrenzt wird.[312] Damit bietet das Verwaltungsverfahren die Möglichkeit, die Konzepte für Rechtshandlungen hinsichtlich Ihrer Übertragbarkeit auf das Verwaltungshandeln zu prüfen. Die Durchführung von Verwaltungsverfahren im Allgemeinen ist im Verwaltungsverfahrensgesetz (VwVfG)[313] geregelt.[314] Entsprechend des von Steinmann und Nejdl beschriebenen Prinzips (vgl. Abschnitt 2.3) und unter

[309] VwVfG § 9.

[310] Neben dem Erlass eines Verwaltungsaktes kann das Verwaltungsverfahren auch den Abschluss des öffentlich-rechtlichen Vertrags mit einschließen. Der Abschluss eines öffentlich-rechtlichen Vertrags ist für die Erbringung von Verwaltungsleistungen in Form von Bürgerdiensten unüblich. Deshalb wird im Folgenden der Begriff Verwaltungsverfahren auf den Erlass des Verwaltungsaktes beschränkt.

[311] Vgl. [Luhmann, 1983], S. 41 f.

[312] Vgl. [Luhmann, 1983], S. 42.

[313] Verwaltungsverfahrensgesetz in der Fassung der Bekanntmachung vom 23. Januar 2003 (BGBl. I S. 102), das zuletzt durch Artikel 2 Absatz 1 des Gesetzes vom 14. August 2009 (BGBl. I S. 2827) geändert worden ist (im Folgenden kurz VwVfG).

[314] Neben dem Verwaltungsverfahrensgesetz auf Bundesebene gibt es eine Reihe von Verwaltungsverfahrensgesetzen auf Landesebene, die dem Bundesgesetz weitgehend entsprechen. Neben der Regelung allgemeiner Verfahrensfragen gibt es fachspezifische Verfahrensfestlegungen, beispielsweise im Sozialgesetzbuch I und X und in entsprechenden Fachgesetzen (z.B. §§36, 105 BBergG). Weitere Festlegungen in anderen Gesetzen betreffen bestimmte Aspekte des Verwaltungshandelns (z.B. die Vollstreckung wegen Geldforderungen oder die Erzwingung von Handlungen im VwVG). Um im Rahmen dieser Arbeit mit der Ontologie zur Annotation von Rechtsvorschriften einen Beitrag zur Verbesserung der Verfolgbarkeit im Vorfeld der Anforderungsspezifikation zu leisten, ist die Betrachtung des VwVfG zunächst ausreichend.

Berücksichtigung der im Rahmen des RAFEG-Projektes von Thomas et al. dokumentierten Erfahrungen erfolgt die Identifikation von Konzepten auf Basis des VwVfG.[315]

Jedes Verwaltungsverfahren beginnt hiernach immer mit dem Schritt der *Verfahrenseinleitung*, der sich aus § 22 VwVfG ableitet. Grundsätzlich entscheidet die Behörde „nach pflichtgemäßem Ermessen, ob und wann sie ein Verwaltungsverfahren durchführt".[316] In den Fällen, in denen die Behörde von Amts wegen oder auf Antrag tätig werden muss, obliegt ihr diese Entscheidung nicht. Sie obliegt ihr auch nicht, wenn sie nur auf Antrag tätig werden darf und ein Antrag nicht vorliegt.[317] Das Konzept der Verfahrenseinleitung hat keine direkte Entsprechung in den zuvor identifizierten Konzepten der Rechtsanwendung. An die Einleitung des Verfahrens schließt sich die *Sachverhaltsermittlung* an. Sie umfasst alle Handlungen, die eine Behörde im Rahmen eines Verwaltungsverfahrens durchführt, um die tatbestandsrelevanten Eigenschaften eines vorliegenden Lebenssachverhalts zu untersuchen, herauszuarbeiten, zu abstrahieren und in die subsumtionsfähige Form eines rechtlich relevanten Sachverhaltes zu transformieren, um dem Untersuchungsgrundsatz nach § 24 VwVfG zu genügen. Damit entspricht die Sachverhaltsermittlung semantisch teilweise den im Rahmen der Rechtsanwendung durchgeführten Handlungen zur Rechtsfindung. Im Wesentlichen ist die öffentliche Verwaltung bei der Erbringung von Verwaltungsleistungen von der Rechtsfindung entlastet, da diese bereits im Vorfeld des Vollzugs erfolgte (z.B. im Zuge der Definition der Verwaltungsleistung und der für sie relevanten Rechtsvorschriften). Im Folgenden wird deshalb davon ausgegangen, dass mit der Erbringung einer Verwaltungsleistung immer auch geklärt ist, welche Rechtsnormen auf den vorliegenden Einzelfall anzuwenden sind, eine Rechtsfindung im Einzelfall des Verwaltungshandelns daher nicht zu erfolgen hat. Die Anwendung der Subsumtion im Rahmen eines Verwaltungsverfahrens erfolgt durch *Prüfung definierter Voraussetzungen,* so dass festgestellt wird, ob die Ausprägungen des im Einzelfall vorliegenden Sachverhalts die Voraussetzungen für die Erbringung einer Verwaltungsleistung erfüllen. Im Verwaltungsverfahren kann die Voraussetzungsprüfung auch die *Nutzung eines Beurteilungsspielraumes* umfassen. Beurteilungsspielraum wird durch die Wertung von Voraussetzungsbedingungen, die auf unbestimmten Rechtsbegriffen basieren, bei der Anwendung auf einen konkreten Sachverhalt genutzt.[318] Die Aktivität der *Mittelwahl* entspricht der Rechtsfolgenbestimmung im Rahmen eines Verwaltungsverfahrens. Sie führt zur Festlegung einer abstrakten Rechtsfolge. Grundsätzlich kann bei der Mittelwahl zwischen gebundenem Verwaltungshandeln und Verwaltungshandeln nach Ermessen unterschieden werden. Im Falle des gebundenen Verwaltungshandelns ist eine Rechtsfolge zwingend vorgeschrieben, sobald die Prüfung der Voraussetzungen erfolgreich war (also die Subsumtion gelungen ist). Im Fall von ungebundenem Verwaltungshandeln wird ein Ermessen eingeräumt, ob zulässige Maßnahmen zu treffen sind, welche von mehreren zulässigen Maßnahmen zu treffen ist oder wie die getroffene Maßnahme auszugestalten ist (*Ausübung von Ermessen).* Dann wird von Verwaltungshandeln nach Ermessen gesprochen. Die *Maßnahmenfestlegung* steht als Handlung im Verwaltungsverfahren für die Anwendung der Einzelfallfestsetzung. Aus praktischen Überlegungen heraus muss sich für die Leistungserbringung in der öffentlichen Verwaltung in der Regel noch ein weiterer Schritt an die klassischen Schritte der Rechtsanwendung anschließen. Zum einen muss die Entscheidung in der Regel einem Betroffenen (z.B. einem Antragsteller) mitgeteilt werden. Dies schließt nicht nur den Fall ein, dass eine Verwaltungsleistung erbracht wird. Es kann auch notwendig oder sinnvoll sein, über das Versagen einer Leistung zu informieren. Für Verwaltungsleistungen, die den Erlass eines Verwaltungsaktes einschließen, ist es in jedem Fall zwingend erforderlich, den *Verwaltungsakt* (z.B. in Form eines Bescheides) zu kommunizieren, da er andernfalls die ihm definitionsgemäß zukommende „nach außen gerichtete Rechtswirkung"[319] nicht entfalten

[315] Vgl. [Thomas et al., 2004].
[316] § 22 Satz 1 VwVfG.
[317] Vgl. § 22 Satz 2 VwVfG.
[318] Vgl. [Sukow&Weidemann, 2007], S. 152 f.
[319] Vgl. § 35 Satz 1 VwVfG.

kann (*Ergebnismitteilung*). Solche Verwaltungsleistungen, die sich in einer zu treffenden Entscheidung im Sinne einer Feststellung (z.B. Erteilung einer Erlaubnis) erschöpfen, ist die Rechtsanwendung im Einzelfall abgeschlossen. Für andere Verwaltungsleistungen „bleibt nach der Feststellung oder Begründung von Leistungsansprüchen des Bürgers für die Verwaltung noch etwas zu tun, das über den Entscheidungsakt hinausgeht“[320]. Nur dadurch, dass über eine Rechtsfolge entschieden und sie kommuniziert wurde, ist die Verwaltungsleistung nicht in allen Fällen erbracht. Es müssen sich in einigen Fällen noch Tätigkeiten anschließen, um beispielsweise Geldzahlungen anzuweisen, einen physischen Bewohnerparkausweis zu erstellen, Kfz-Kennzeichen mit dem Siegel der Zulassungsstelle zu versehen oder weitere Aktivitäten anzustoßen (z.B. muss die Leitung einer Kindertagesstätte darüber informiert werden, welche verfügbaren Plätze welchen Eltern bzw. Kindern zugeordnet wurden). Die *Leistungserbringung* repräsentiert somit eine Verwaltungshandlung, die zur Kommunikation der getroffenen Entscheidung und zur tatsächlichen Erbringung der Verwaltungsleistung dient. Erfolgt die Handlung im Rahmen eines Verwaltungsverfahrens und wurde ein Verwaltungsakt erlassen oder der Erlass abgelehnt, so muss die getroffene Entscheidung kommuniziert werden.[321] Handlungen zur tatsächlichen Erbringung der Verwaltungsleistung können aufgrund ihres Inhaltes sehr unterschiedlich sein, sind aber immer auf die Gestaltung der tatsächlichen Umstände ausgerichtet. Die Leistungserbringung findet in den klassischen Schritten zur Rechtsanwendung daher keine direkte Entsprechung. Sind alle Schritte durchgeführt, so kann die Verwaltungsleistung als vollständig erbracht und das Verwaltungshandeln als beendet betrachtet werden. In der Realität können sich Aktivitäten und Prozesse zur Erfolgskontrolle oder zur Korrektur fehlerhaft erbrachter Leistungen anschließen. Derartige Aktivitäten und Prozesse gehen inhaltlich über das Thema dieser Arbeit hinaus, da sie sich an die Leistungserstellung anschließen. Einen Überblick über die bisher identifizierten Verwaltungshandlungen gibt im Sinne eines Zwischenergebnisses die nachfolgende Tabelle 9.

Name des Konzepts	Beschreibung
Verwaltungshandeln	Zusammenfassung aller Handlungen der öffentlichen Verwaltung zur Erbringung von Verwaltungsleistungen.
Verwaltungsverfahren	Nach außen wirkende Tätigkeit der öffentlichen Verwaltung, die auf die Prüfung der Voraussetzungen, die Vorbereitung und den Erlass eines Verwaltungsaktes gerichtet ist und den Erlass des Verwaltungsaktes einschließt. Spezialisierung des generellen Konzepts des Verwaltungshandelns. Dieses Konzept hat keine Entsprechung in den bisher identifizierten Konzepten der Rechtsanwendung.
Ermessensausübung	Eine Handlung, die von der Verwaltung im Falle von ungebundenem Verwaltungshandeln durchgeführt wird, um (entsprechend dem Zweck der Ermächtigung und in den gesetzlichen Grenzen) zu entscheiden, ob und/oder wie gehandelt werden soll. Das Konzept ist eine Spezialisierung des Konzepts der Rechtsfolgenbestimmung.
Sachverhaltsermittlung	Sie steht für alle Handlungen, die eine Behörde im Rahmen eines Verwaltungsverfahrens durchführt, um die tatbestandsrelevanten Eigenschaften eines vorliegenden Lebenssachverhalts zu untersuchen, herauszuarbeiten, zu abstrahieren und in die subsumtionsfähige Form eines rechtlich relevanten Sachverhaltes zu transformieren. Die Sachverhaltsermittlung repräsentiert die Aspekte der Rechtsfindung, die der Ableitung des rechtlich relevanten Sachverhalts aus dem vorliegenden Lebenssachverhalt dienen.

[320] [Krause, 1974], S. 52.

[321] Ein Verwaltungsverfahren kann auch mit dem Abschluss bzw. Nicht-Abschluss eines öffentlich-rechtlichen Vertrages enden. Dieser Fall ist jedoch in der Vollzugsfunktion der Leistungsverwaltung von untergeordneter Bedeutung und wird daher im Rahmen dieser Arbeit nicht betrachtet.

	Die Frage der anzuwendenden Rechtsnorm, die in der Rechtsanwendung ebenfalls Bestandteil der Rechtsfindung sind, kann während der Ausführung durch die Verwaltung als geklärt vorausgesetzt werden.
Ergebnismitteilung	Repräsentiert alle Handlungen, die zur Kommunikation der getroffenen Entscheidung dienen (Mitteilung).
Maßnahmenfestlegung	Die Maßnahmenfestlegung repräsentiert die Verwaltungshandlung zur Festlegung einer konkreten Maßnahme als Verwaltungsleistung im Einzelfall. Im Rahmen der Maßnahmenfestlegung werden Handlungen durchgeführt, die für jedes Merkmal der festgelegten abstrakten Rechtsfolge eine konkrete Ausprägung entsprechend des vorliegenden Einzelfalls vorsehen und dokumentieren. Die Maßnahmenfestlegung ist die Anwendung der Einzelfallfestsetzung in Form einer Verwaltungshandlung.
Mittelwahl	Ist durch die Rechtsvorschrift oder durch eine andere Vorschrift bei erfülltem Tatbestand genau eine abstrakte Rechtsfolge vorgesehen, so ist die durchzuführende Handlung trivial: Die vorgesehene Rechtsfolge wird als Mittel gewählt. Stehen mehrere Rechtsfolgen zur Auswahl, so wird von diesen genau eine Rechtsfolge ausgewählt. Es ist auch möglich, dass keine Rechtsfolge gewählt wird, d.h., dass sich die Verwaltung entscheidet, nicht zu handeln. Die Mittelwahl führt zur Auswahl einer abstrakten Rechtsfolge oder zur Auswahl keiner Rechtsfolge. Im Falle von ungebundenem Handeln kann die Mittelwahl die Ausübung von Ermessen umfassen. Die Mittelwahl repräsentiert die Anwendung der Rechtsfolgenbestimmung im Rahmen eines Verwaltungsverfahrens.
Nutzung von Beurteilungsspielraum	Repräsentiert Handlungen, die im Rahmen einer Prüfung von Leistungsvoraussetzungen (Voraussetzungsprüfung) erfolgen, bei denen die zugrunde liegende Rechtsvorschrift und sonstige Vorschriften einen Spielraum in der Beurteilung zulassen. Im Rahmen dieser Handlungen wird der gelassene Spielraum durch Wertung und Beurteilung vor dem Hintergrund eines konkreten Sachverhalts genutzt. Die Nutzung von Beurteilungsspielraum ist in der Rechtsanwendung Teil der Subsumtion.
Verfahrenseinleitung	Die Verfahrenseinleitung repräsentiert Handlungen der Verwaltung, die die Leistungserbringung im Rahmen eines Verwaltungsverfahrens einleiten und wird durch das Auftreten eines von der Verwaltung wahrgenommenen Ereignisses ausgelöst. Dieses Konzept hat keine Entsprechung in den bisher identifizierten Konzepten der Rechtsanwendung.
Voraussetzungsprüfung	Im Rahmen der Voraussetzungsprüfung wird geprüft, ob die Ausprägungen eines im Einzelfall vorliegenden Sachverhalts die Voraussetzungen für die Erbringung einer Verwaltungsleistung erfüllen. Im Rahmen der Voraussetzungsprüfung wird für jede Tatbestandsbedingung geprüft, ob die korrespondierende Eigenschaft des Sachverhaltes dieser Bedingung genügt. Die Voraussetzungsprüfung endet mit der Entscheidung, ob alle Bedingungen des Tatbestandes durch den Sachverhalt erfüllt sind und damit die Subsumtion gelungen ist oder ob dies nicht der Fall ist. Die Prüfung von Voraussetzungen kann auch die Nutzung von Beurteilungsspielraum umfassen. Das Konzept repräsentiert die Anwendung der Subsumtion im Rahmen eines Verwaltungsverfahrens, d.h. als Verwaltungshandlung.
Leistungserbringung	Solche Handlungen, die sich nicht ausschließlich in der Mitteilung der getroffenen Entscheidung erschöpfen, erfordern weitere Handlungen zur

	tatsächlichen Leistungserbringung. Diese Handlungen werden durch das Konzept Leistungserbringung repräsentiert. Eine Instanz des Konzepts Leistungserbringung steht in Beziehung zu einer Instanz des Konzepts Verwaltungsleistung, die die zu erbringende Leistung repräsentiert.

Tabelle 9 – Identifizierte Konzepte für Handlungen der Verwaltung

Neben den Handlungsgegenständen und den Handlungen, die von der Verwaltung durchgeführt werden, sind auch die agierenden Akteure ein relevanter Aspekt zur Formalisierung. Das VwVfG regelt die möglichen Akteure am Verwaltungsverfahren und unterscheidet diese Personen auch von denjenigen, die für die Behörde tätig werden.[322] Die für die Behörde tätigen Personen entsprechen dem bereits formalisierten Konzept des *Rechtsanwenders*. Weil der Verwaltung auch immer die Ausführung der getroffenen Entscheidung obliegt, wird der *Ausführende* als neues Konzept eingeführt. Das Verwaltungsverfahren kennt grundsätzlich auch den *Adressaten*, an den sich die Verwaltungsentscheidung richtet bzw. der Empfänger der Verwaltungsleistung ist und am Verfahren beteiligte bzw. von der Entscheidung betroffene Akteure (*Beteiligter/Betroffener*). Außerdem wird der *Antragsteller* als weiterer Akteur eingeführt, der keine Entsprechung in den zuvor identifizierten Konzepten der Rechtsanwendung hat. Die Tabelle 10 fasst die relevanten Akteure als Zwischenergebnis zusammen.

Name des Konzepts	Beschreibung
Adressat	Der Adressat ist der am Verwaltungshandeln beteiligte, an den sich die getroffene Entscheidung und die erbrachte Leistung richtet.
Antragsteller	Der Antragsteller ist ein Akteur, dessen Antrag das Verwaltungshandeln ausgelöst hat.
Anwender	Akteur in einer Verwaltungsorganisation oder Verwaltungsorganisation, die Recht anwendet, die dazu notwendigen Verwaltungshandlungen durchführt und dabei die relevanten Handlungsgegenstände bearbeitet.
Ausführender	Umfassen die Verwaltungshandlungen auch die tatsächliche Ausführung der getroffenen Entscheidung, so wird diese von einem Akteur ausgeführt, für den im Rahmen dieser Arbeit das Konzept Ausführender eingeführt wird.
Beteiligter/Betroffener	Repräsentiert einen Akteur, der am Verwaltungshandeln beteiligt oder von ihm betroffen ist.

Tabelle 10 – Identifizierte Konzepte für Akteure aus Sicht der Verwaltung

Würdigt man die bisher als Zwischenergebnisse identifizierten Konzepte vor dem Hintergrund der vorhandenen Ontologien des Verwaltungshandelns (Abschnitt 2.3), wird insbesondere durch Vergleich mit der ESD-Ontologie (siehe oben, Seite 36) deutlich, dass bestimmte Konzepte unterrepräsentiert sind. Es fehlen beispielsweise die Ansatzpunkte für ESD-Prozesse wie die Unterstützung eines Bürgers durch Hinweise oder Informationen, die Bereitstellung von Informationen über Verwaltungsleistungen, den Abschluss bzw. die Kommentierung einer Fallbearbeitung oder einer Anfrage und die Bezahlung von Waren und Dienstleistungen. Die für das E-Government im Kontext von Bürgerdiensten relevante Lebenslagenorientierung findet sich ebenso wenig wie Ansatzpunkte für einen organisatorischen Rahmen aus Front- und Backoffice. Die Ausprägung der Phasen typischer Phasenmodelle für Bürgerdienste ist in den Handlungen nur in Grundzügen zu erkennen. Hier wird dies als Hinweis gewertet, dass die ausschließliche Ableitung relevanter Konzepte aus der Rechtsanwendung und aus Bestandteilen von Rechtsvorschriften für die öffentliche Verwaltung nicht ausreichend ist. Deshalb muss der Betrachtungsbereich, um die

[322] VwVfG §13, §20.

Gestaltungspotenziale des E-Government nutzen zu können, auch die Spezifika des Verwaltungshandelns bei intensiver Unterstützung durch Informationstechnik berücksichtigen. Dazu wird auf die in Abschnitt 2.5 dargestellten Zusammenhänge zurückgegriffen, und es werden im Rahmen dieser Arbeit exemplarisch weitere Konzepte identifiziert:

Die *Lebenslagen* von Bürgern sind zentrales Element für die Gestaltung von Bürgerdiensten. Wird sich ein Bürger dessen bewusst, dass in einer individuellen Situation oder einem Lebensabschnitt (Lebenssachverhalt) ein Verwaltungskontakt sinnvoll oder notwendig ist, entwickelt er daraus ein *Anliegen* bezogen auf eine oder mehrere Verwaltungsleistungen.[323] Dieser Schritt der *Bewusstwerdung* kann von der Verwaltung durch bereitgestellte Hinweise und erste Informationen (*HinweisInformation*) unterstützt oder durch Handeln der Verwaltung (z.B. Erinnerung an einen ablaufenden Personalausweis) ausgelöst werden. Solches Handeln erfolgt durch einen Akteur im Sinne eines *Bewusstmacher*s innerhalb der Verwaltung (Person oder technische Komponente). Bereitgestellte Informationen nutzt der Bürger auch, um den Verwaltungskontakt vorzubereiten (z.B. um einen Kontaktweg zu wählen oder notwendige Unterlagen zusammenzustellen). Dies resultiert in den Konzepten *Informationsbeschaffung* und *Vorbereitung*. Anschließend nimmt der Bürger über ein Front Office Kontakt mit der Verwaltung auf, was zu einem Abgleich seines Anliegens mit den möglichen und sinnvollen Verwaltungsleistungen führt (*AnliegenLeistungenAbgleichen*). Im Ergebnis dieses Abgleichs ergeben sich Anträge, die die notwendigen initialen Bearbeitungsinformationen enthalten und die Handlungsauslöser für das Verwaltungshandeln sind. Die Durchführung der Leistungserbringung erfolgt dann in der Regel im Back-Office (durch Akteure in der Rolle *AnwenderBackOffice*), wobei notwendige Kontakte mit dem Bürger (z.B. Nachreichen von Unterlagen oder das Bezahlen kostenpflichtiger Verwaltungsleistungen) über ein Front-Office abgewickelt werden, in dem Akteure in der Rolle *AnwenderFrontOffice* tätig werden. Das Ergebnis der Bearbeitung im Back-Office wird dem Bürger anschließend mitgeteilt und parallel werden die eventuell noch notwendigen Schritte zur tatsächlichen Leistungserbringung eingeleitet. Die eigentliche Leistungserstellung kann sowohl im Back-Office (z.B. Anlegen eines wiederkehrenden Zahlauftrags für Wohngeld) oder im Front-Office (z.B. Siegeln von KFZ-Kennzeichen) durch Akteure in den Rollen *AusfuehrenderBackOffice* oder *AusfuehrenderFrontOffice* erfolgen. Die abschließende *Nachbereitung* umfasst interne Aktivitäten (z.B. den Abschluss bzw. die Kommentierung einer Fallbearbeitung) zur Dokumentation relevanter Informationen (*Nachbearbeitungsinformation*) oder ausgehend von bekannten Zusammenhängen und Rahmenbedingungen die Abgabe von Hinweisen auf weitere Verwaltungsleistungen, die im Anschluss sinnvoll oder notwendig sein könnten (*Leistungszusammenhang*). Daraus ergeben sich die nachfolgend zusammengefassten exemplarischen Konzepte für Handlungsgegenstände (Tabelle 11), Handlungen (Tabelle 12) und Akteure (Tabelle 13) im Kontext von Bürgerdiensten bei intensiver Unterstützung durch Informationstechnik. Die Identifikation weiterer Konzepte, die als Erweiterung für E-Government sinnvoll erscheinen, ist analog zu den hier aufgeführten Konzepten möglich. Für das Ziel dieser Arbeit, einen Ansatz für die Verfolgbarkeit im Vorfeld der Anforderungsspezifikation zu entwickeln, sind weitere Konzepte jedoch nicht notwendig.

Name des Konzepts	Beschreibung
HinweisInformation	Eine Information, die vom Erkennen der Notwendigkeit oder Eignung einer Verwaltungsleistung bis zur Beantragung der Leistung in einer bestimmten Lebenslage hilfreich sein kann.

[323] Müller unterscheidet die Begriffe Lebenssituation und Anliegen nicht weitergehend, sondern fasst beides unter dem Begriff „Bedarfslage“ zusammen (vgl. [Müller, 2011], S. 79).

Lebenslage	Abstrakte Situation bzw. abstrakter Lebensabschnitt eines Bürgers, in der/dem aufgrund „eines Bedürfnisses, eines Wunsches, eines Rechts, eines Ereignisses oder einer Pflicht“[324] Verwaltungsleistungen möglich oder notwendig sind.
Anliegen	Aus einer individuellen Situation ggf. durch Konkretisierung einer abstrakten Lebenslage und unter Verwendung von Hinweisen/Informationen abgeleiteter Bedarf eines Bürgers an einer oder mehreren Verwaltungsleistungen.
Verwaltungsleistung	In Ergänzung zur Definition in Tabelle 8 hat eine Verwaltungsleistung das Merkmal der Kostenpflicht. Die Kostenpflicht gibt an, ob die Leistung prinzipiell zu bezahlen ist.
Nachbearbeitungs-information	Informationen, die für den internen Abschluss der Verwaltungstätigkeit notwendig sind.
Leistungszusammenhang	Informationen, die bezogen auf die aktuell erbrachte Verwaltungsleistung und eine bekannte Lebenslage Hinweise auf sinnvolle oder notwendige Folgeleistungen geben können.

Tabelle 11 – Ergänzende Konzepte für Handlungsgegenstände von Bürgerdiensten bei intensiver Unterstützung durch Informationstechnik

Name des Konzepts	Beschreibung
Bewusstwerdung	Handlungen, die dazu führen, dass von außen angeregt oder aufgrund eigener Erkenntnis der Nachfrager ein Anliegen entwickelt.
Informationsbeschaffung	Handlungen, die dazu dienen, die notwendigen Informationen zur Vorbereitung eines Verwaltungskontaktes zu beschaffen.
Vorbereitung	Handlungen eines Nachfragers, um einen Verwaltungskontakt zu planen und vorzubereiten.
Nachbearbeitung	Handlungen, die dazu dienen, die Leistungserstellung intern zu dokumentieren oder abzuschließen und ggf. Hinweise auf nachfolgend relevante Verwaltungsleistungen an den Nachfrager zu geben.

Tabelle 12 – Ergänzende Konzepte für Handlungen im Kontext von Bürgerdiensten bei intensiver Unterstützung durch Informationstechnik

Name des Konzepts	Beschreibung
Bewusstmacher	Ein Akteur, der personalisierte Hinweise und Informationen abgibt, die einen potenziellen Nachfrager dazu anregen, ein konkretes Anliegen zu entwickeln.
Nachfrager	Ein Akteur, der ein zu befriedigendes Anliegen entwickelt hat. Er unterscheidet sich vom Antragsteller dadurch, dass ein Nachfrager nicht notwendigerweise bereits einen Antrag gestellt haben muss.
AnwenderFrontOffice	Ein spezieller Anwender, der Handlungen durchführt, die zur Erstellung einer Leistung durchgeführt werden und einen direkten Kontakt mit dem Nachfrager erfordern.
AnwenderBackOffice	Ein spezieller Anwender, der Handlungen durchführt, die zur Erstellung einer Leistung durchgeführt werden und unabhängig vom direkten Nachfragerkontakt sind.
AusfuehrenderFrontOffice	Ein spezieller Anwender, der Handlungen zur tatsächlichen Umsetzung einer Leistung in direktem Kontakt mit dem Nachfrager/Antragsteller ausführt.

[324] [Müller, 2011], S. 80.

AusfuehrenderBackOffice	Ein spezieller Anwender, der Handlungen zur tatsächlichen Umsetzung einer Leistung ohne direkten Nachfrager-/Antragstellerkontakt ausführt.

Tabelle 13 – Ergänzende Konzepte für Akteure im Kontext von Bürgerdiensten bei intensiver Unterstützung durch Informationstechnik

3.1.4 Zusammenwirken

Die bisherige Konzeptionalisierung hat zu Konzepten geführt, die relevant für die Auseinandersetzung mit Normtexten und Rechtssätzen, mit der Rechtsanwendung und dem Verwaltungshandeln sind. Bei der Herleitung und Identifikation dieser Konzepte ist deutlich geworden, dass Abhängigkeiten zwischen den Konzepten existieren. Diese Abhängigkeiten können verwendet werden, um auf einander aufbauende, idealtypische Schichten zu definieren. Der Schicht „Normtext/Rechtssatz" sind die Konzepte zugeordnet, die einen unmittelbaren Bezug zum Normtext und zu dessen Elementen haben. Darauf aufbauend sind der Schicht „Rechtsanwendung" solche Konzepte zugeordnet, die idealtypisch auf der Existenz der Textkonzepte in der darunter liegenden Schicht basieren. Der obersten Schicht sind die Konzepte des Verwaltungshandelns zugeordnet. Diese Konzepte begründen ihre Existenz idealtypischer Weise in Konzepten der Rechtsanwendung.

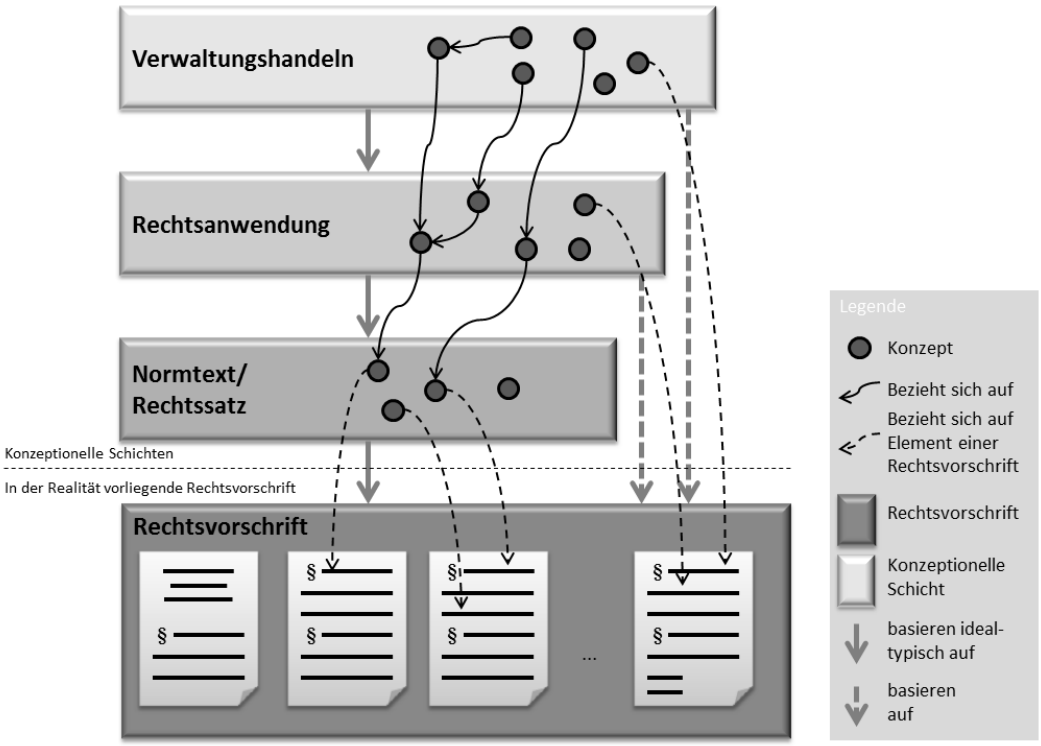


Abbildung 22 – Illustration konzeptioneller Schichten und ihrer Zusammenhänge

Die Konzeptionalisierung im Rahmen dieser Arbeit kommt zu einem Ergebnis, das von der idealtypischen Schichtenbildung abweicht. Es gibt sowohl in der Schicht der Rechtsanwendung, als auch in der Schicht des Verwaltungshandelns Konzepte, deren Existenz nicht unmittelbar auf hier formalisierte Konzepte der darunter liegenden Schicht zurückgeführt werden kann. In der Rechtsanwendung gehen beispielsweise die hier formalisierten durchzuführenden Handlungen nicht direkt auf Konzepte des Normtextes und des Rechtssatzes zurück. Ihre Existenz in der Ontologie begründet sich vielmehr durch die Tatsache, dass im Rahmen dieser Handlungen Bestandteile des Rechtssatzes gemäß anerkannter rechtswissenschaftlicher Methoden bearbeitet werden. Es existiert somit ein Bezug zu Konzepten der gleichen Schicht, die ihrerseits auf die zugrunde liegende Schicht der Textkonzepte zurückgeführt werden können. Es gibt aber auch in den hier formalisierten

Konzepten der Rechtsanwendung solche, die nicht auf Konzepte in Texten zurückgeführt werden können. Ein Beispiel hierfür ist der *Lebenssachverhalt*, der immer individuell ist.[325] Im Verwaltungshandeln gehen beispielsweise die *Ziele und Zwecke* des Handelns und solche Konzepte, die Besonderheiten dieses Handelns im Vergleich zur Rechtsanwendung darstellen, nicht auf die Konzepte der darunter liegenden Schicht zurück. Vom idealtypischen Standpunkt betrachtet, kann für solche Konzepte deshalb aus rechtswissenschaftlicher Sicht nicht zugesichert werden, dass sie innerhalb der Texte von Rechtsvorschriften tatsächlich vorkommen. Ohne dem Ergebnis des Abschnittes 3.2.4 vorwegzugreifen, gibt es praktische Beispiele, die zeigen, dass auch solche Konzepte im Text von Rechtsvorschriften identifiziert werden können. Weil das Ziel der Ontologie ist, Elemente im Text von Rechtsvorschriften mit einer definierten Semantik zu versehen, kann von der Idee einer idealtypischen Schichtenbildung abgewichen werden. Im Folgenden wird deshalb anstelle des Begriffs der Schicht der Begriff „Themenkomplex" verwendet.

3.2 Formalisierung in RDFS und OWL

In der vorangegangenen Beschreibung des Gegenstandbereichs wurden Konzepte isoliert dargestellt und ihre Beziehungen wurden informell angedeutet. In diesem Abschnitt werden sie innerhalb einer gemeinsamen Ontologie formalisiert, indem nicht nur ihre Semantik beschrieben wird, sondern sie auch zueinander in Beziehung gesetzt werden. Mittels einer Taxonomie werden die Konzepte des Verwaltungshandelns (soweit möglich) auf Konzepte der Rechtsanwendung und diese auf Konzepte von Normtexten zurückgeführt. Dazu ist es notwendig, weitere Konzepte (z. B. als abstrakte Oberklassen der Ontologie) einzuführen. Für die Ontologie-Entwicklung wurde zur Unterstützung das populäre Werkzeug *protégé* (Abbildung 23) verwendet. Es ermöglicht neben dem Ontologieentwurf auch den Export des Ergebnisses im OWL-Format, so dass die exportierte Ontologie mit Werkzeugen des Semantic Web eingesetzt werden kann.

Die Abbildung 23 zeigt die Ontologie auf oberster Ebene im Werkzeug *protége*. Vom allgemeinen Wurzel-Konzept `owl:Thing` wird die gemeinsame Oberklasse `DomainConcept` abgeleitet, die für die im Rahmen dieses Ansatzes formalisierten Konzepte die gemeinsame Ausgangsbasis ist. Von ihr sind weitergehend drei Subklassen abgeleitet, die allgemeine Verwaltungskonzepte, Textkonzepte und Rechtskonzepte repräsentieren und in jeweils eigenen Unterabschnitten (3.2.1 bis 3.2.3) beschrieben werden. Die grafische Darstellung erfolgt in der vom Werkzeug protegé im *OWLViz-Plug in* und im *OWLPropViz-Plug in* verwendeten Notation, die neben der für protegé mit OWLViz typischen „is-a"-Beziehung auch Beziehungen zeigt, die durch OWL Object Properties ausgedrückt werden. Zentrale Konzepte, die für das Verständnis der nachfolgenden Ausführungen (z.B. Beispiele, Implementierung) notwendig sind, werden zusätzlich in der als gut lesbar anerkannten Manchester-Notation[326] dargestellt. Die vollständige Ontologie, bestehend aus Klassen, Object Properties und Data Properties, ist als OWL-Listing im Anhang A enthalten.

[325] Müller identifiziert im Kontext von Bürgerdiensten das vergleichbare Element „Bedarfslage", bei dem sie zur Schlussfolgerung gelangt: „Eine umfassende Beschreibung [...] einer Bedarfslage kann aufgrund ihrer Komplexität und Individualität nicht gelingen" ([Müller, 2011], S. 79).
[326] Für eine Einführung in die Manchester-Notation vgl. [Horridge et al., 2006].

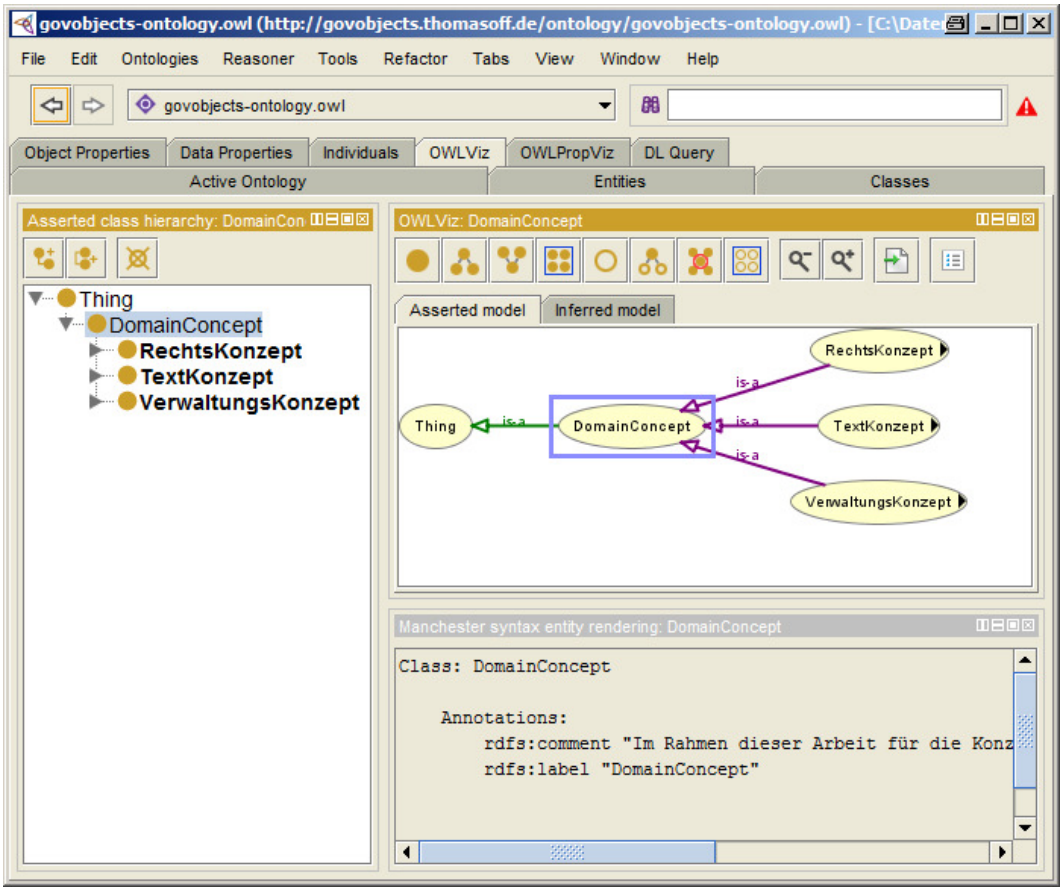


Abbildung 23 – Darstellung der Ontologie auf oberster Ebene im Werkzeug „protégé“

3.2.1 Normtext- und Rechtssatzkonzepte

Die Konzepte für `Text` und `Textbestandteil` (als Spezialisierung von `TextKonzept`) repräsentieren Texte als zusammenhängende, abgegrenzte kommunikative Einheiten in geschriebener Sprache mit ihren Bestandteilen. Die nachfolgende Abbildung 24 zeigt die weitergehenden Spezialisierungen dieser Konzepte.

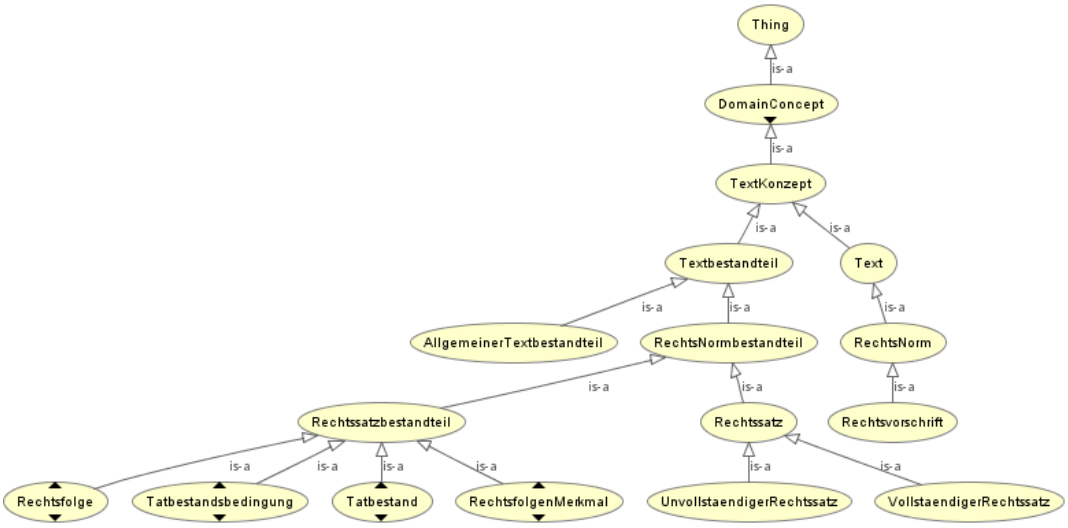


Abbildung 24 – OWL- Klassenhierarchie der Normtext- und Rechtssatzkonzepte

Die `RechtsNorm` wird durch eine Spezialisierung als besonderer Text aufgrund ihrer normativen Absicht von anderen Texten unterschieden. Als weitergehende Spezialisierung

wird die Rechtsvorschrift eingeführt, die im Rahmen dieser Arbeit synonym für Gesetze, Rechtsvorschriften und vor dem Hintergrund des hier betrachteten Verwaltungshandelns auch für deren Konkretisierungen steht. Die Ontologie formalisiert das Konzept AllgemeinerTextbestandteil, das solche Elemente repräsentiert, die Bestandteil von Texten aus sprachlicher Sicht sind (z.B. Wörter, Wortgruppen, Sätze, Absätze, Abschnitte, Paragraphen, Kapitel). Die Regeln ihres Aufbaus müssen für die beabsichtigte Verwendung der Ontologie zur Annotation vorhandener Rechtsvorschriften nicht weitergehend formalisiert werden, da vorhandene Texte mit Annotationen versehen werden und ihre sprachliche Struktur als Bezugspunkt nicht erforderlich ist. Das Konzept des AllgemeinenTextbestandteils dient in der Ontologie vor allem der Abgrenzung der weiteren Spezialisierung des Textbestandteils, den speziellen Bestandteilen von Rechtnormen (RechtsNormBestandteil). Diese speziellen Bestandteile einer Rechtsnorm sind Rechtssatz und Rechtssatzbestandteil. Im Abschnitt 3.1.1 wurde beschrieben, dass Rechtssätze als unvollständige Rechtssätze und vollständige Rechtssätze auftreten können, so dass sie hier als getrennte Klassen formalisiert werden. Als Rechtssatzbestandteile wurden Rechtsfolge, Rechtsfolgenmerkmal, Tatbestandbedingung und Tatbestand formalisiert. Die sich unter Berücksichtigung dieser Konzepte ergebende Klassenhierarchie ist in Abbildung 24 dargestellt.

3.2.2 Rechtskonzepte

Die Rechtskonzepte der Ontologie basieren auf den im Abschnitt 3.1.2 identifizierten Konzepten der Rechtsanwendung. Das RechtsKonzept führt als Subklassen RechtsHandlungsgegenstand, RechtsAkteur und RechtsHandlung in die Ontologie ein (Abbildung 25).

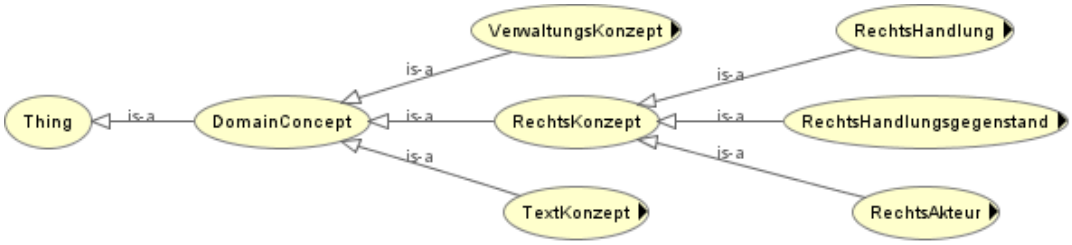


Abbildung 25 – Rechtskonzepte der Ontologie

Ein RechtsAkteur ist die Abstraktion für Personen oder Organisationen, die an der Rechtsanwendung beteiligt sind. Dies sind in der Regel der Rechtsanwender (Anwender) selbst und der Adressat einer Rechtsnorm. Dieser Teil der Ontologie ist in der nachfolgenden Abbildung 26 dargestellt, wobei die Oberklassen owl:Thing und DomainConcept von RechtsKonzept ausgeblendet wurden.

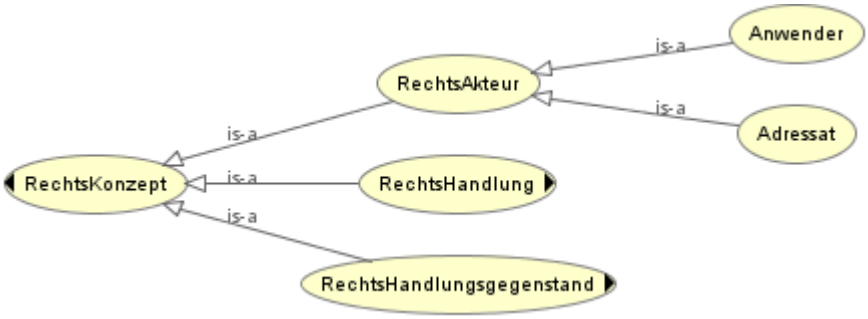


Abbildung 26 – Rechtsbeteiligte in der Ontologie

Das Konzept RechtsHandlung ist die Abstraktion für die in Abschnitt 3.1.2 eingeführten handlungsorientierten Konzepte. Sie repräsentiert Schritte der Rechtsanwendung, die durch etablierte Methoden der Rechtswissenschaft vorgegeben sind und dazu dienen, Recht und

insbesondere Rechtssätze auf einen Einzelfall anzuwenden. Die `Rechtsfindung` repräsentiert einen Schritt der Rechtsanwendung, in dessen Rahmen die auf den vorliegenden Einzelfall anzuwendenden Rechtsnormen identifiziert werden. Weiterhin werden als Konzepte die Ermittlung des Sachverhalts (`Sachverhaltsermittlung`), die `Subsumtion`, die Bestimmung der Rechtsfolge (`Rechtsfolgenbestimmung`), die Festsetzung einer Einzelfallentscheidung (`Einzelfallfestsetzung`) und die `Mitteilung` der getroffenen Entscheidung in der Ontologie formalisiert.

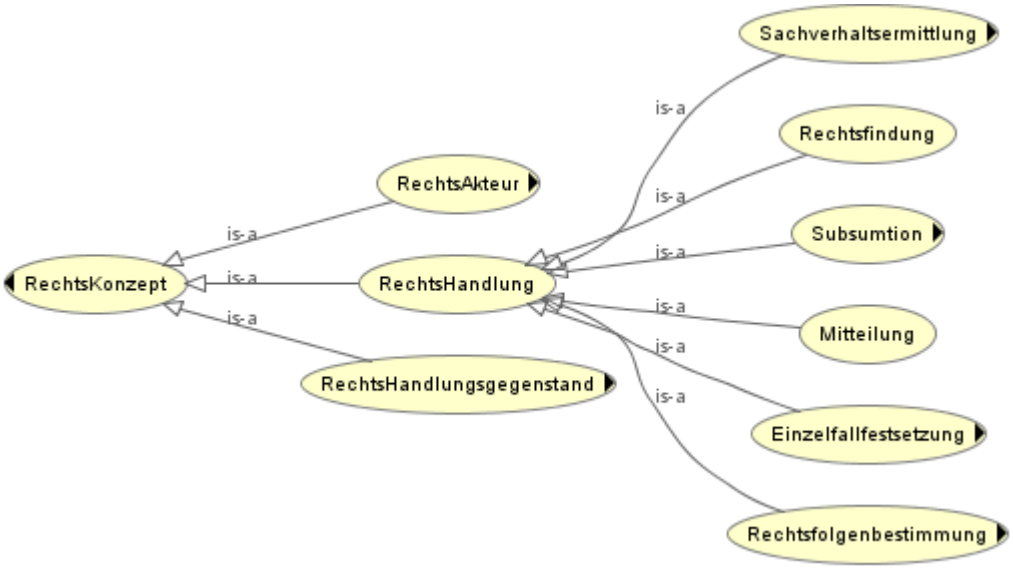


Abbildung 27 – Rechtshandlungen in der Ontologie

Das Konzept `RechtsHandlungsgegenstand` ist in der Ontologie die Abstraktion für abstrakte Konzepte und konkrete Dinge, die im Rahmen von Rechtshandlungen erzeugt, verwendet und/oder manipuliert werden. Seine Subklassen sind analog zu den Konzepten aus Abschnitt 3.1.2 definiert. Entsprechend ergibt sich eine Spezialisierung in `Sachverhalt`, `Sachverhaltseigenschaft`, `Tatbestand`, `Tatbestandsbedingung`, `Rechtsfolge`, `Rechtsfolgenmerkmal`, `Massnahme` und `MassnahmenEigenschaft` (Abbildung 28). In der Abbildung 29 ist die Klassenhierarchie der Rechtskonzepte in der Ontologie vollständig grafisch dargestellt.

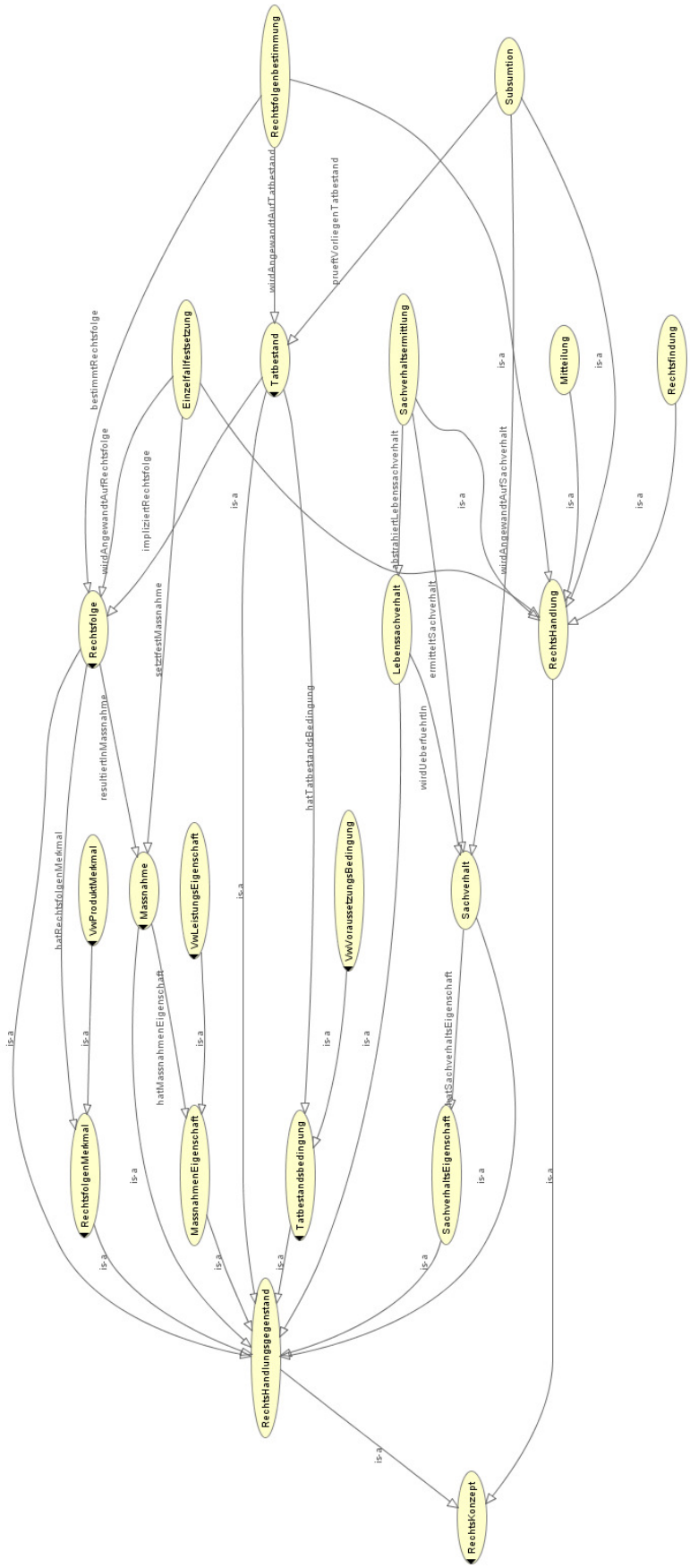


Abbildung 28 – Rechtshandlungsgegenstände und ihr Zusammenhang mit Rechtshandlungen in der Ontologie

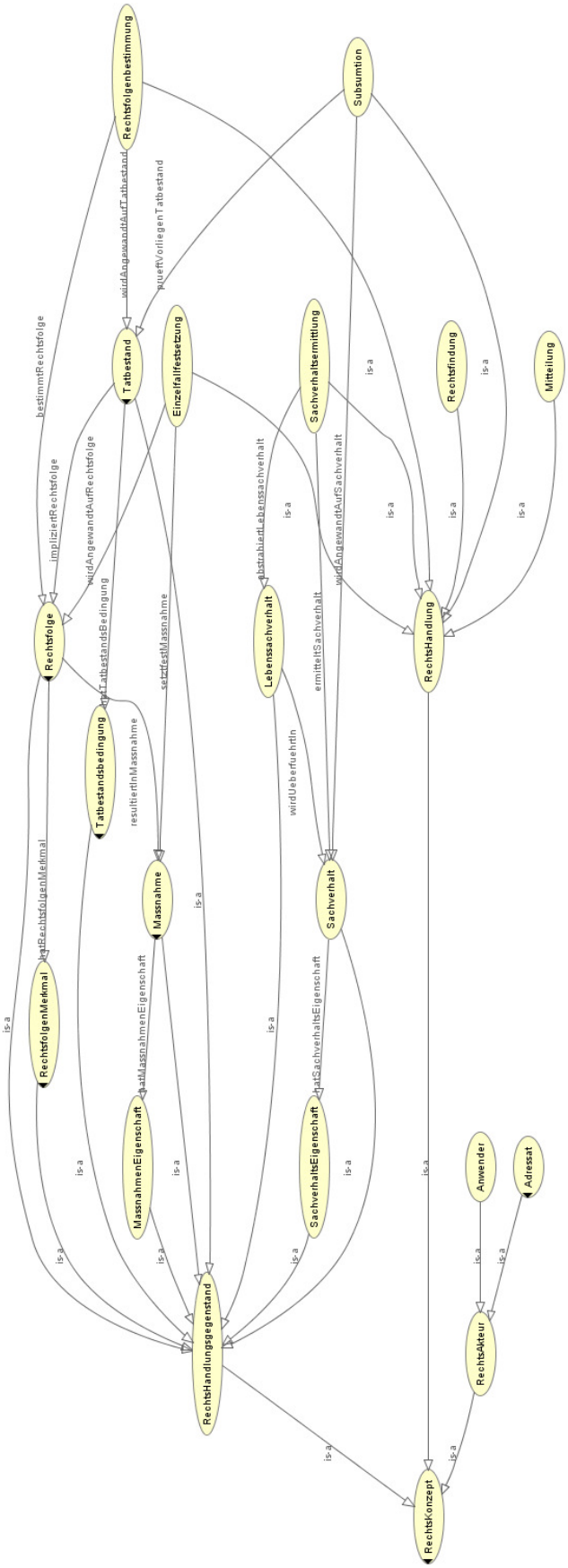


Abbildung 29 – OWL- Klassenhierarchie der Rechtskonzepte

3.2.3 Verwaltungskonzepte

Die Verwaltungskonzepte bilden den dritten Themenkomplex im Gegenstandsbereich.[327] In der Ontologie wird er durch Subklassen der Klasse `VerwaltungsKonzept` und deren Beziehungen ausgedrückt. Für `VerwaltungsKonzept` sind in der Ontologie Data Properties `Name`, `Beschreibung`, `Arbeitsnotiz` und `Rahmenbedingung` definiert (siehe Listing 6 auf Seite 95). Mit dem Data Property `Name` kann einer Instanz bei Bedarf ein beliebiger fachlicher Name zugewiesen werden, der in der nachfolgenden Verarbeitung (anstelle der aus der annotierten Textpasssage gewonnenen Bezeichnung der Instanz) verwendet werden kann. Die Data Properties `Beschreibung` und `Arbeitsnotiz` dienen zur weitergehenden fachlichen Beschreibung bzw. zur Dokumentation offener Punkte und relevanter Hinweise für die weitere Verarbeitung. Das Data Property `Rahmenbedingung` ermöglicht es, zu den Instanzen Rahmenbedingungen ihrer Umsetzung oder ihrer Anwendung zu dokumentieren. Diese Data Properties sind somit auch in den Spezialisierungen von `VerwaltungsKonzept` verfügbar. Einen Überblick über die Formalisierung der Klasse `VerwaltungsKonzept` gibt das nachfolgende Listing 1.

```
(1)   Class: VerwaltungsKonzept
(2)
(3)       Annotations:
(4)           rdfs:label "Verwaltungskonzept"^^xsd:string,
(5)           rdfs:comment "Verwaltungskonzept ist die Abstraktion und
      Formalisierung der Konzepte des Verwaltungshandelns"^^xsd:string
(6)
(7)       SubClassOf:
(8)           DomainConcept,
(9)           konkretisiertRechtsKonzept only RechtsKonzept,
(10)          Arbeitsnotiz some string,
(11)          Beschreibung some string,
(12)          Quelle some string
```

Listing 1 – Klassen „VerwaltungsKonzept“ (Auszug)

Analog zu den Rechtskonzepten werden auch bei den Verwaltungskonzepten die handlungsorientierten Konzepte (`VwHandlung`), die erzeugten, verwendeten oder manipulierten Handlungsgegenstände (`VwHandlungsgegenstand`) und die daran Beteiligten (`VwAkteur`) unterschieden. Darüber hinaus ergeben sich weitere Konzepte für Verwaltungshandeln im Kontext von Bürgerdiensten unter intensiver Unterstützung durch Informationstechnik. Diese Konzepte sind als jeweilige Erweiterung der Handlungsgegenstände, Handlungen und Akteure in der Ontologie formalisiert (Abbildung 30).

[327] Um Mehrdeutigkeiten in der Bezeichnung der Konzepte aufzulösen, wird für die Verwaltungskonzepte das Präfix "Vw" verwendet, sofern das Wort "Verwaltung" nicht im Namen des Konzepts enthalten ist.

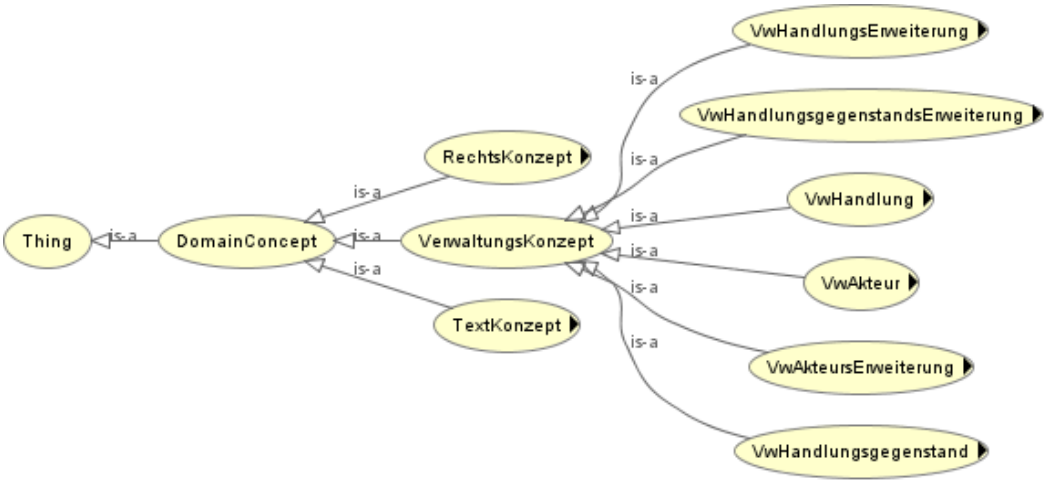


Abbildung 30 – Verwaltungskonzepte auf oberster Ebene

Die Handlungsgegenstände der Verwaltung, die während der Untersuchung des Gegenstandbereichs identifiziert wurden, sind in der Ontologie als Subklassen von Verwaltungshandlungsgegenstand (`VwHandlungsgegenstand`) repräsentiert. Für diese Verwaltungshandlungsgegenstände ergibt sich ein Zusammenhang zu den Handlungsgegenständen der Rechtsanwendung (siehe Abschnitt 3.2.4). Da die Semantik nicht identisch ist, werden die Rechtshandlungsgegenstände nicht direkt in das Verwaltungshandeln übernommen. Außerdem leiten sich die Verwaltungshandlungs-gegenstände `Handlungsauslöser`, `Antrag` und `ZielZweck` nicht direkt aus den in der Ontologie enthaltenen Rechtshandlungsgegenständen ab, weil sie spezifisch für das Verwaltungshandeln sind. Dennoch ist zu erwarten, dass sie aus Rechtsvorschriften, die auf den Vollzug ausgerichtet sind, abgeleitet werden können. Deshalb sind sie als Subklassen von `Verwaltungshandlungsgegenstand`, aber nicht als Subklassen der Rechtskonzepte formalisiert. Diese Zusammenhänge zwischen den Handlungsgegenständen der Rechtsanwendung und denen des Verwaltungshandelns sind in der Abbildung 31 dargestellt.

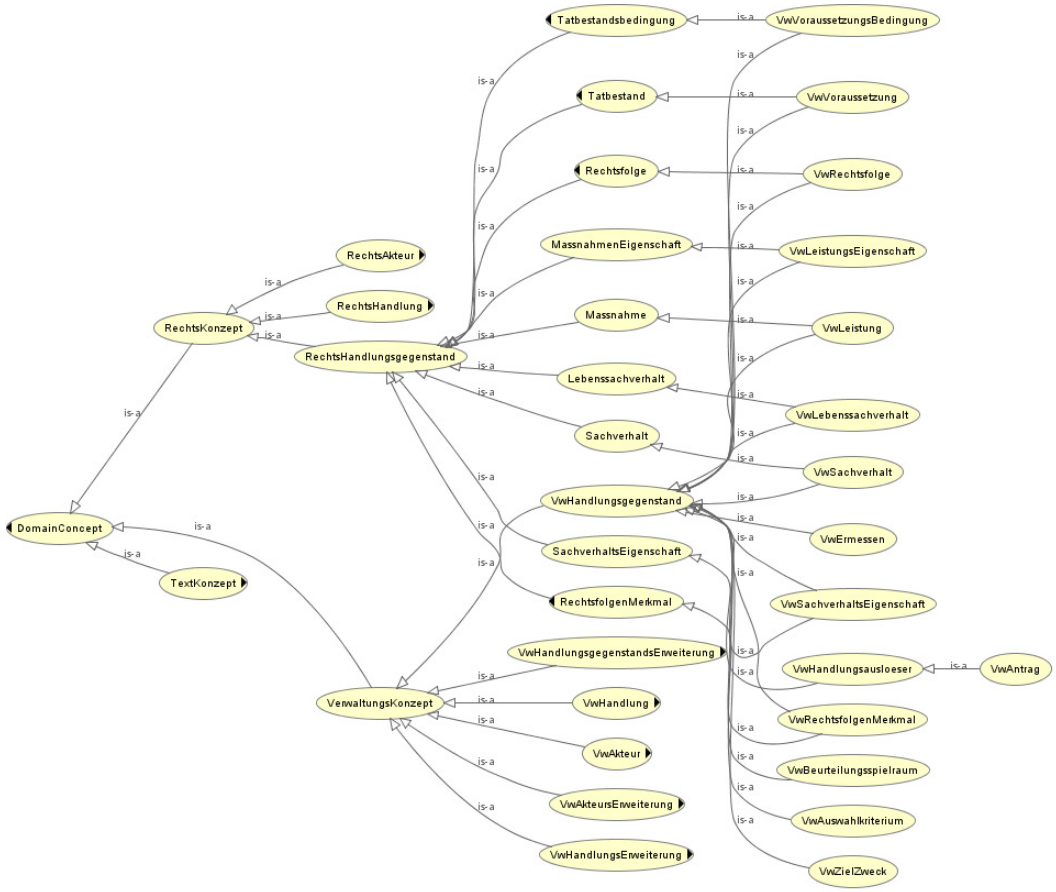


Abbildung 31 – Zusammenhänge zwischen den Handlungsgegenständen der Rechtsanwendung und des Verwaltungshandelns

```
(1)   Class: VwLeistung
(2)
(3)       Annotations:
(4)           rdfs:comment "Die Verwaltungsleistung repräsentiert die
      Konkretisierung einer abstrakten Rechtsfolge für den im Verwaltungshandeln
      vorliegenden Einzelfall. Sie kann als Element einer Menge von ggf. mehreren
      möglichen Leistungen aufgefasst werden, die durch die Rechtsfolge
      repräsentiert werden. Das Streben nach optimaler Zielerreichung führt dazu,
      dass die Konkretisierung anhand der Ziele des Verwaltungshandelns
      durchgeführt wird.Maßnahme: Das Konzept übernimmt nicht die allgemeine
      Bezeichnung Maßnahme in das Verwaltungshandeln, um deutlich zu machen, dass
      in der Leistungsverwaltung Maßnahmen in Form von Leistungen erbracht werden.
      Semantisch sonst identisch."^^xsd:string,
(5)           rdfs:label "VwLeistung"^^xsd:string
(6)
(7)       SubClassOf:
(8)           Massnahme,
(9)           VwHandlungsgegenstand,
(10)          istVomTyp only VwRechtsfolge
(11)
(12)  Class: VwRechtsfolge
(13)
(14)      Annotations:
(15)          rdfs:label "VwRechtsfolge"^^xsd:string,
```

```
(16)          rdfs:comment "Die Rechtsfolge im Verwaltungshandeln repräsentiert
      die in einem Rechtssatz bei Vorliegen der Voraussetzungen vorgesehene
      abstrakte Beschreibung der rechtlichen Konsequenz. Ihre rechtliche
      Zulässigkeit wird durch die Methoden der Rechtsanwendung ermittelt, ihre
      Auswahl aber wird auch durch ihren Beitrag zur Zielerreichung des
      Verwaltungshandelns bewertet. Eine Rechtsfolge kann als eine Menge von
      konkreten Leistungen aufgefasst werden, zu der alle Leistungen gehören, die
      der abstrakten Beschreibung der Rechtsfolge genügen. Die Semantik der
      Rechtsfolge wird an die veränderte Bezeichnung des Konzepts Maßnahme und des
      korrespondierenden Konzepts Leistung angepasst. Insbesondere erweitert sich
      der Charakter dadurch, dass sie nicht nur eine rechtliche Zulässigkeit hat
      (oder nicht hat), sondern auch an ihrem Beitrag zur Zielerreichung gemessen
      wird."^^xsd:string
(17)
(18)      SubClassOf:
(19)          Rechtsfolge,
(20)          VwHandlungsgegenstand,
(21)          liefertBeitragZuZielZweck only VwZielZweck,
(22)          istKostenpflichtig exactly 1 boolean
(23)
```

Listing 2 – Klassen „VwLeistung“ und „VwRechtsfolge“ (Auszug)

Analog zu den Verwaltungshandlungsgegenständen sind auch die Handlungen der Verwaltung als Subklassen der Verwaltungshandlung(VwHandlung) in der Ontologie formalisiert. Das Konzept VwHandlung ist die Oberklasse für spezielle Handlungen der Verwaltung. In der Ontologie sind die Data Properties HandlungsZiel, HandlungVorher und HandlungNachher definiert, mit denen eine VwHandlung weiter formalisiert werden kann (siehe Listing 7 auf Seite 95). Ihre Instanzen können hierdurch das Ziel der Handlung, die Voraussetzungen (Vorbedingungen) und die Ergebnisse (Nachbedingungen) der Handlung beschreiben. Die VwHandlung verfügt aufgrund ihrer Spezialisierung aus dem Verwaltungskonzept auch über das Data Property Beschreibung, das von ihren Instanzen benutzt wird, um die durchzuführenden Schritte der Handlung zu beschreiben. Die VwHandlung ist Oberklasse weiterer spezieller Konzepte, die ebenfalls über diese Data Properties verfügen. Einige dieser Spezialisierungen der VwHandlung haben einen Zusammenhang mit den Handlungen der Rechtsanwendung, der in Abbildung 32 dargestellt ist. Spezifisch für das Verwaltungshandeln ist die spezielle Form des Verwaltungsverfahrens (VwVerfahren) und dessen Einleitung (VwVerfahrenseinleitung), die jeweils keine Subklasse von Rechtshandlung sind. Ein Beispiel einer Formalisierung zeigt das für die Klasse VwSachverhaltsermittlung.

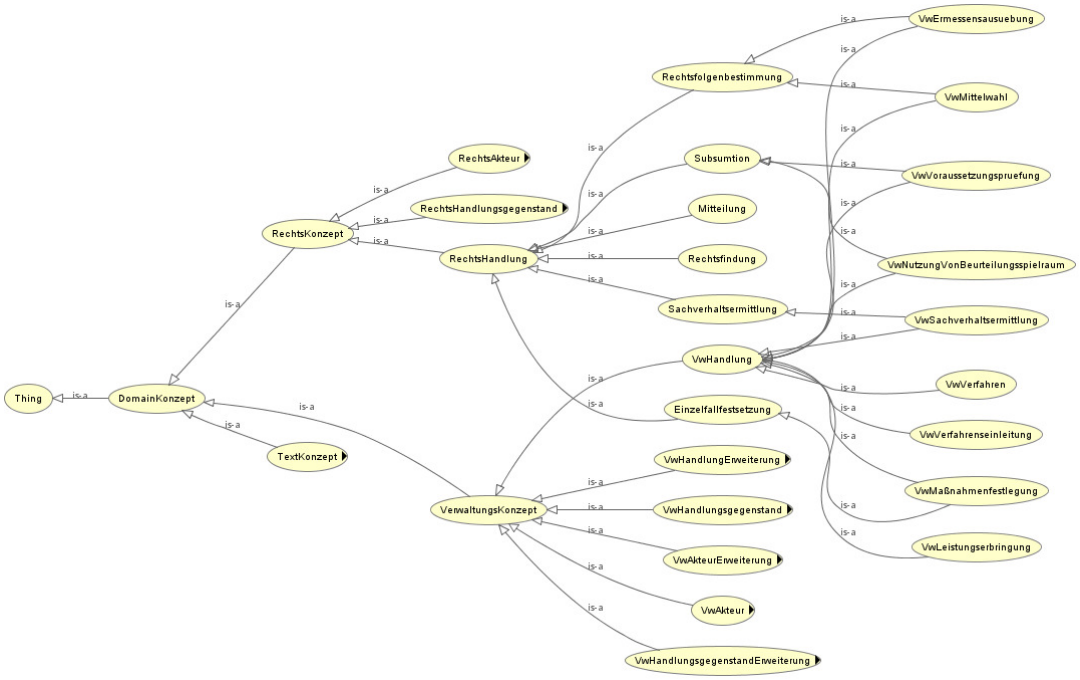


Abbildung 32 – Zusammenhang zwischen den Subklassen der Rechtshandlung und der Verwaltungshandlung

```
(1)   Class: VwSachverhaltsermittlung
(2)
(3)       Annotations:
(4)           rdfs:label "VwSachverhaltsermittlung"^^xsd:string,
(5)           rdfs:comment "Repräsentiert alle Handlungen, die ein Träger
      öffentlicher Aufgaben im Rahmen seiner Tätigkeit (in der Regel eines
      Verwaltungsverfahrens) durchführt, um die tatbestandsrelevanten
      Eigenschaften eines vorliegenden Lebenssachverhalts zu untersuchen,
      herauszuarbeiten, zu abstrahieren und in die subsumtionsfähige Form eines
      rechtlich relevanten Sacherhaltes zu transformieren, um z.B. dem
      Untersuchungsgrundsatz nach § 24 VwVfG zu genügen. Die
      Sachverhaltsermittlung repräsentiert die Aspekte der Rechtsfindung, die der
      Ableitung des rechtlich relevanten Sachverhalts aus dem vorliegenden
      Lebenssachverhalt dienen. Die Frage der anzuwendenden Rechtsnorm, die in der
      Rechtsanwendung ebenfalls Bestandteil der Rechtsfindung ist, kann während
      der Ausführung durch die Verwaltung als geklärt vorausgesetzt
      werden."^^xsd:string
(6)
(7)       SubClassOf:
(8)           Sachverhaltsermittlung,
(9)           VwHandlung,
(10)          ermitteltSachverhalt only VwSachverhalt,
(11)          hatNachfolger only VwVoraussetzungspruefung
```

Listing 3 – Klasse „VwSachverhaltsermittlung“ (Auszug)

Die handelnden Personen und Organisationen werden durch die Klasse `VerwaltungsAkteur` und deren Subklassen in der Ontologie repräsentiert. Dabei wird zwischen Beteiligten, Ausführenden, Antragsteller, Adressat und dem eigentlichen Anwender unterschieden. Der Beteiligte wird durch Subklassenbildung weiter verfeinert (z.B. Anzuhörende, betroffene Dritte). Diese Zusammenhänge sind in Abbildung 33 dargestellt. Das Listing 4 zeigt die Formalisierung der Klasse `VwAntragsterller`.

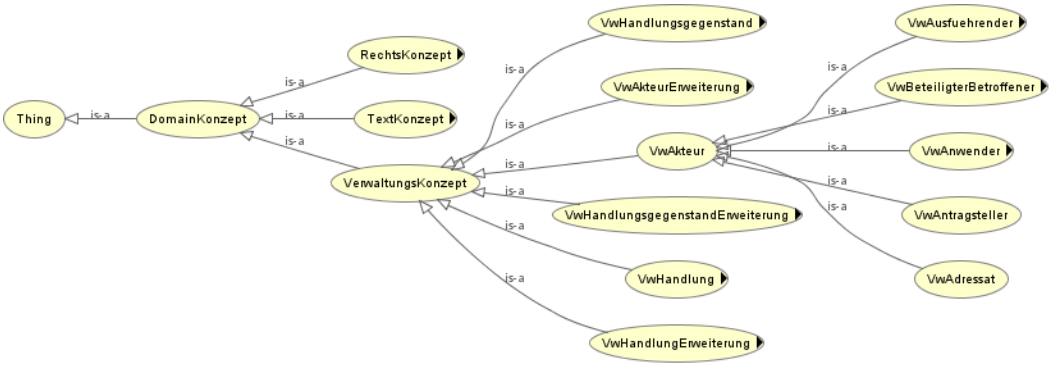


Abbildung 33 – Verwaltungsakteure

```
(1)   Class: VwAntragsteller
(2)
(3)       Annotations:
(4)           rdfs:label "VwAntragsteller"^^xsd:string,
(5)           rdfs:comment "Der Antragsteller ist ein Akteur, dessen Antrag das
      Verwaltungshandeln ausgelöst hat."^^xsd:string
(6)
(7)       SubClassOf:
(8)           VwAkteur,
(9)           hatLebenslage only VwLebenslage,
(10)          stelltAntrag only VwAntrag
```

Listing 4 – Klasse „VwAntragsteller" (Auszug)

Die bisher eingeführten Verwaltungshandlungsgegenstände, Verwaltungshandlungen und Verwaltungsakteure entsprechen im Wesentlichen vergleichbaren Rechtskonzepten. Zur Erbringung von Bürgerdiensten unter intensiver Unterstützung durch Informationstechnik wurden weitere notwendige Konzepte identifiziert. Sie sind als Spezialisierung von Konzepten formalisiert, die jeweils die Handlungsgegenstände, Handlungen und Akteure der Verwaltung erweitern (Abbildung 34) und haben keine Entsprechung in den Konzepten der Rechtsanwendung, des Normtextes und des Rechtssatzes. Es handelt sich bei ihnen um Besonderheiten des Verwaltungshandelns, von denen nicht erwartet wird, dass sie sich in Texten von Rechtsvorschriften identifizieren lassen. Unterhalb von `VwHandlungsgegenstandsErweiterung` werden deshalb exemplarisch die Konzepte `Legenslage` und `Anliegen`, unterhalb von `VwHandlungsErweiterung`[328] die Konzepte für `Bewusstwerdung` und `Vorbereitung` und unterhalb von `VwAkteursErweiterung` die im Front- und Backoffice tätigen Anwender oder Ausführenden (`VwAusfuehrenderBackOffice`, `VwAnwenderFrontOffice`) vorgesehen. Durch diese Konstruktion ist es möglich, bedarfsgerecht weitere Konzepte in der Ontologie einzuführen, wenn dies für die Berücksichtigung von Besonderheiten des Verwaltungshandelns sinnvoll ist. Ein Beispiel für die Formalisierung des Konzepts VwHinweisInformation ist in Listing 5 dargestellt.

[328] Für „VwHandlungsErweiterung" wurden die gleichen DataProperties, wie für „VwHandlung" definiert.

Abbildung 34 – Erweiterung um Konzepte für Bürgerdienste unter intensiver Unterstützung durch Informationstechnik

```
(1)   Class: VwHinweisInformation
(2)
(3)       Annotations:
(4)           rdfs:comment "Eine Information, die vom Erkennen der Notwendigkeit
      oder Eignung einer Verwaltungsleistung bis zur Beantragung der Leistung in
      einer bestimmten Lebenslage hilfreich sein kann."^^xsd:string,
(5)           rdfs:label "VwHinweisInformation"^^xsd:string
(6)
(7)       SubClassOf:
(8)           VwHandlungsgegenstandsErweiterung,
(9)           moeglicheRechtsfolge some VwRechtsfolge,
(10)          notwendigeRechtsfolge some VwRechtsfolge,
(11)          zugehoerigeLebenslage some VwLebenslage,
(12)          beziehtSichAufLeistung only VwLeistung
```

Listing 5 – Klasse „VwHinweisInformation" (Auszug)

Neben den Klassen sind in der Ontologie noch zwei weitere Typen formalisiert: Data Properties und Object Properties. Mit Hilfe der Data Properties werden Eigenschaften für Klassen der Ontologie definiert. Für die Instanzen dieser Klassen können dann Ausprägungen dieser Eigenschaften festgelegt werden. Die nachfolgenden Listings zeigen dies am Beispiel der Klassen `VerwaltungsKonzept` (Listing 6), `VwHandlung` und `VwHandlungsErweiterung` (Listing 7).

Um die möglichen Beziehungen der Instanzen von Klassen ausdrücken zu können, wurden in der Ontologie Object Properties formalisiert. Beispielsweise kann mit dem Object Property `hatAntragsteller` die Beziehung von einer Instanz der Klasse `VwAntrag` und zu einer Instanz der Klasse `VwAntragsteller` und mit dem inversen Object Property `stelltAntrag` die gleiche Beziehung aus umgekehrter Richtung hergestellt werden (Listing 8). Für das Verständnis der nächsten Kapitel weiterhin wichtige Object Properties sind die Beziehungen, die jeweils eine Instanz von `VwHinweisInformation` bzw. von `VwLeistung` zu einer Instanz von `VwRechtsfolge` haben können (Listing 9 bzw. Listing 10).

```
(1)   DataProperty: Arbeitsnotiz
(2)
(3)       Annotations:
(4)           rdfs:label "Arbeitsnotiz",
(5)           rdfs:comment "Bemerkungen, die während der weiteren Umsetzung
      hilfreich sein können."
(6)
(7)       Domain:
```

```
(8)           VerwaltungsKonzept
(9)
(10)      Range:
(11)          string
(12)
(13)  DataProperty: Beschreibung
(14)
(15)      Annotations:
(16)          rdfs:comment "Inhaltliche Beschreibung des durch die Annotation
      repräsentierten Individuums.",
(17)          rdfs:label "Beschreibung"
(18)
(19)      Domain:
(20)          VerwaltungsKonzept
(21)
(22)      Range:
(23)          string
(24)
(25)  DataProperty: Name
(26)
(27)      Annotations:
(28)          rdfs:label "Name",
(29)          rdfs:comment "Bietet die Möglichkeit zur Angabe eines alternativen
      Namens, der sich nicht direkt aus dem Text einer annotierten Textpassage
      ergibt."
(30)
(31)      Domain:
(32)          VerwaltungsKonzept
(33)
(34)      Range:
(35)          string
(36)
(37)  DataProperty: Rahmenbedingung
(38)
(39)      Annotations:
(40)          rdfs:comment "Beschreibung der besonderen Bedingungen für die
      Umsetzung oder den Einsatz des Konzepts.",
(41)          rdfs:label "Rahmenbedingung"
(42)
(43)      Domain:
(44)          VerwaltungsKonzept
(45)
(46)      Range:
(47)          string
(48)
```

Listing 6 – Data Properties der Klasse „Verwaltungskonzept“

```
(1)   DataProperty: HandlungNachher
(2)
(3)       Annotations:
(4)           rdfs:label "HandlungNachher",
(5)           rdfs:comment "Beschreibung, was durch die Handlung erreicht wurde
      (z.B. der erreichte Zustand)."
(6)
(7)       Domain:
(8)           VwHandlung,
(9)           VwHandlungsErweiterung
(10)
(11)      Range:
(12)          string
(13)
(14)  DataProperty: HandlungVorher
(15)
(16)      Annotations:
(17)          rdfs:comment "Beschreibung, was vor Durchführung der Handlungen
      bereits erfolgt sein muss.",
(18)          rdfs:label "HandlungVorher"
(19)
(20)      Domain:
(21)          VwHandlung,
(22)          VwHandlungsErweiterung
(23)
(24)      Range:
(25)          string
```

```
(26)
(27)  DataProperty: HandlungsZiel
(28)
(29)      Annotations:
(30)          rdfs:comment "Beschreibung des mit der Handlung verfolgten Ziels.",
(31)          rdfs:label "HandlungsZiel"
(32)
(33)      Domain:
(34)          VwHandlung,
(35)          VwHandlungsErweiterung
(36)
(37)      Range:
(38)          string
```

Listing 7 – Data Properties der Klassen „VwHandlung“ und „VwHandlungsErweiterung“

```
(1)   ObjectProperty: stelltAntrag
(2)
(3)       Annotations:
(4)           rdfs:label "stelltAntrag"^^xsd:string,
(5)           rdfs:comment "Repräsentiert die Beziehung von einem Antragssteller
      zu dem von ihm gestellten Antrag."^^xsd:string
(6)
(7)       Domain:
(8)           VwAntragsteller
(9)
(10)      Range:
(11)          VwAntrag
(12)
(13)      InverseOf:
(14)          hatAntragsteller
(15)
(16)  ObjectProperty: hatAntragsteller
(17)
(18)      Annotations:
(19)          rdfs:comment "Repräsentiert die Beziehung eines Antrags zu seinem
      Antragsteller."^^xsd:string
(20)
(21)      Domain:
(22)          VwAntrag
(23)
(24)      Range:
(25)          VwAntragsteller
(26)
(27)      InverseOf:
(28)          stelltAntrag
```

Listing 8 – Object Properties von “VwAntragsteller”

```
(1)   ObjectProperty: notwendigeRechtsfolge
(2)
(3)       Annotations:
(4)           rdfs:comment "Repräsentiert eine Beziehung zwischen einem
      Hinweis/einer Information und einer Rechtsfolge. Sie drückt für einen
      Hinweis/eine Information aus, dass die Rechtsfolge als Verwaltungsleistung
      erbracht werden muss."^^xsd:string,
(5)           rdfs:label "notwendigesVerwaltungsprodukt"
(6)
(7)       Domain:
(8)           VwHinweisInformation
(9)
(10)      Range:
(11)          VwRechtsfolge
(12)
(13)  ObjectProperty: moeglicheRechtsfolge
(14)
(15)      Annotations:
(16)          rdfs:label "moeglichesVerwaltungsprodukt",
(17)          rdfs:comment "Repräsentiert eine Beziehung zwischen einem
      Hinweis/einer Information und einer Rechtsfolge. Sie drückt für einen
      Hinweis/eine Information aus, dass die Rechtsfolge möglicherweise als
      Verwaltungsleistung erbracht werden kann."^^xsd:string
(18)
(19)      Domain:
(20)          VwHinweisInformation
(21)
```

```
(22)      Range:
(23)          VwRechtsfolge
(24)
(25)  ObjectProperty: beziehtSichAufLeistung
(26)
(27)      Annotations:
(28)          rdfs:comment "Repräsentiert eine Beziehung zwischen einem
      Hinweis/einer Information und einer Verwaltungsleistung. Sie drückt für
      einen Hinweis/eine Information aus, dass die Verwaltungsleistung hilfreich,
      nützlich oder erforderlich ist."^^xsd:string,
(29)          rdfs:label "beziehtSichAufLeistung"
(30)
(31)      Domain:
(32)          VwHinweisInformation
(33)
(34)      Range:
(35)          VwLeistung
```

Listing 9 – Object Properties von "VwHinweisInformation"

```
(1)   ObjectProperty: istVomTyp
(2)
(3)       Annotations:
(4)           rdfs:comment "Repräsentiert die Beziehung, die eine im konkreten
      Einzelfall erbrachte Verwaltungsleistung zur Rechtsfolge hat."^^xsd:string,
(5)           rdfs:label "istVomTyp"
(6)
(7)       Domain:
(8)           VwLeistung
(9)
(10)      Range:
(11)          VwRechtsfolge
```

Listing 10 – Object Properties von "VwLeistung"

3.2.4 Zusammenwirken

In Abschnitt 3.1.4 wurde im Rahmen der Konzeptionalisierung das Zusammenwirken der Normtext-/Rechtssatz-, Rechts- und Verwaltungskonzepte anhand eines idealtypischen Schichtenmodells beschrieben. Bei der Formalisierung der Konzepte können die Konzepte der verschiedenen Themenkomplexe folglich nicht für sich allein stehen. Ihr Zusammenhang ist in der Ontologie durch „is-a"-Beziehungen und Object Properties formalisiert. Die „is-a"-Beziehung verbindet ein Konzept mit dem Konzept aus einem anderen Themenbereich, auf das es in seiner Existenz direkt zurückgeht. Object Properties drücken aus, dass ein Konzept in seiner Existenz auf ein Konzept des gleichen Themengebietes zurückgeht.

Beispielweise gehen im Themenkomplex Rechtskonzepte die Rechtshandlungsgegenstände `Rechtsfolge`, `RechtsfolgenMerkmal`, `Tatbestand` und `Tatbestandsbedingung` auf Bestandteile eines Rechtssatzes (aus dem Themengebiet Normtext- und Rechtssatz) zurück. Sie sind deshalb sowohl als Subklassen von `RechtsHandlungsegenstand`, als auch als Subklassen von `Rechtssatzbestandteil` formalisiert werden. Dieser Zusammenhang ist in Abbildung 35 dargestellt.

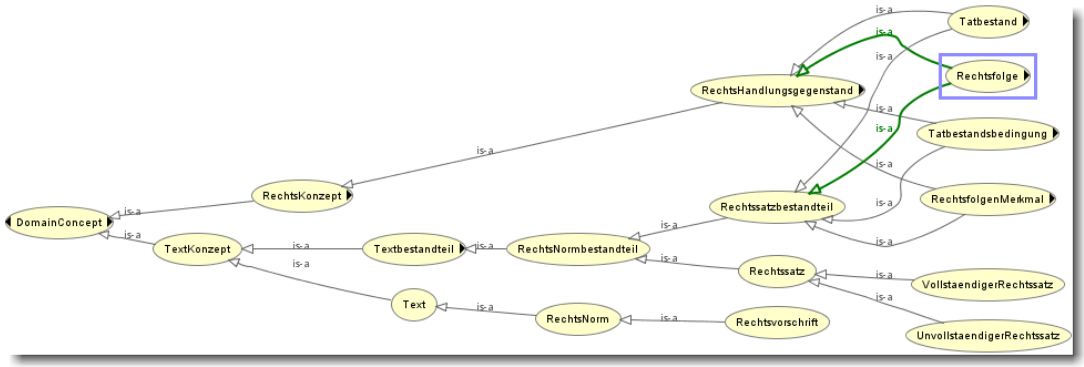


Abbildung 35 – Zusammenhang zwischen „Normtext-/Rechtssatzkonzepten" und „Rechtskonzepten"

Für solche Rechtskonzepte, die in in ihrer Existenz nicht direkt auf Normtext- und Rechtssatzbestandteile zurückgehen, werden Object Properties benutzt, um sie mit anderen Rechtskonzepten in Beziehung zu setzen.

Beispielsweise gehen die Rechtshandlungen in ihrer Existenz nicht direkt auf Normtext- und Rechtssatzbestandteile zurück. Weil in ihrem Rahmen aber mit den Rechtshandlungsgegeständen entsprechend rechtswissenschaftlicher Methoden gearbeitet wird, existiert eine Beziehung, die in der Ontologie durch Object Properties formalisiert wird. Ein Beispiel hierfür ist die Klasse `Subsumtion`, deren Instanzen über das Object Property `prueftVorliegenTatbestand` mit Instanzen der Klasse `Tatbestand` verbunden werden können (Listing 11). Weil der `Tatbestand` selbst direkt auf einen Rechtssatzbestandteil zurück geht, existiert eine indirekte Beziehung zwischen der Subsumtion und einem Rechtssatzbestandteil. Einen Überblick über die Object Properties im Themengebiet „Rechtsanwendung“ gibt Abbildung 36.

```
(12)  ObjectProperty: prueftVorliegenTatbestand
(13)
(14)      Annotations:
(15)          rdfs:comment "Repräsentiert die Verbindung, die im Rahmen der
      Durchführung einer Subsumtion zum anzuwendenden Tatbestand
      existiert."^^xsd:string,
(16)          rdfs:label "prueftVorliegenTatbestand"^^xsd:string
(17)
(18)      Domain:
(19)          Subsumtion
(20)
(21)      Range:
(22)          Tatbestand
(23)
```

Listing 11 – Object Property „prueftVorliegenTatbestand“

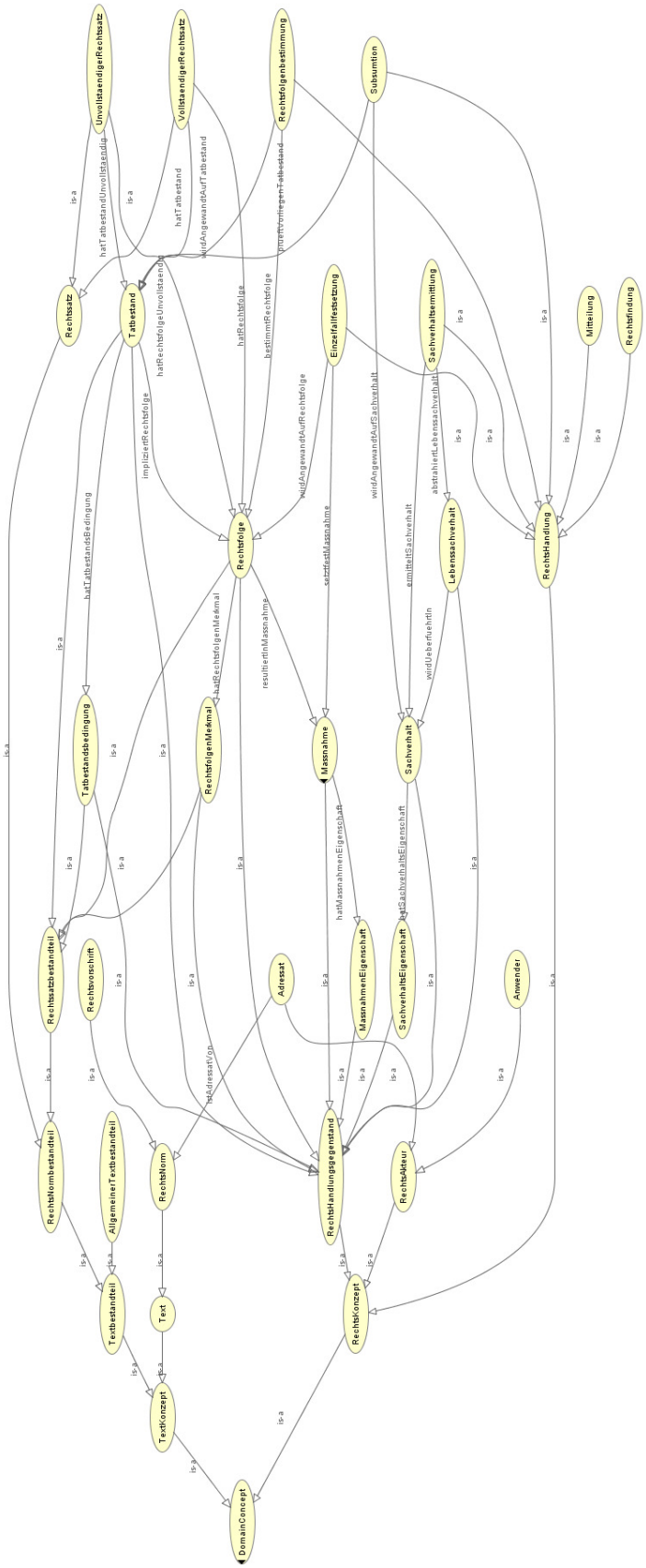


Abbildung 36 – Zusammenhang zwischen Textkonzepten und Rechtskonzepten

Dieses Muster, das benutzt wird, um die Beziehung zwischen den Themenkomplexen Normtext-/Rechtssatzkonzepte und Rechtskonzepte herzustellen, wird auch für den Zusammenhang zwischen den Themengebieten Verwaltungskonzepte und Rechtskonzepte benutzt..

Einige der Verwaltungskonzepte gehen in ihrer Existenz auf Rechtskonzepte und diese direkt auf Konzepte des Normtextes und des Rechtssatzes zurück. Ein Beispiel eines solchen Konzepts ist die `VwRechtsfolge`, die einerseits als Spezialisierung von `VwHandlungsgegenstand` und andererseits von `Rechtsfolge` ist (Abbildung 37).

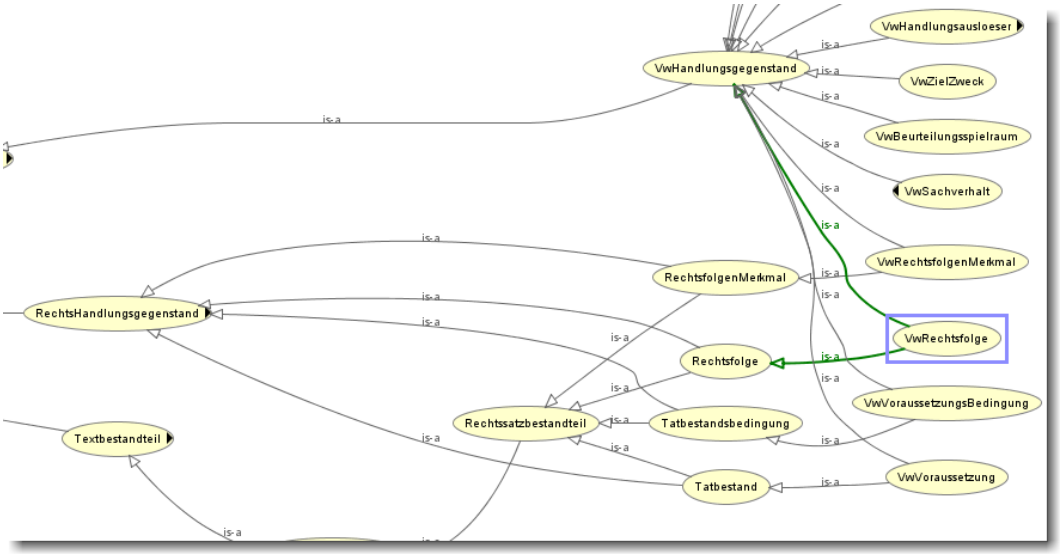


Abbildung 37 – Klasse „VwRechtsfolge" als Beispiel für die Formalisierung des Zusammenhangs zwischen Rechts- und Verwaltungskonzepten

Die Klasse `Rechtsfolge` ist eine Subklasse von `RechtsHandlungsgegenstand`. Sie repräsentiert damit abstrakte Konzepte und konkrete Dinge, die im Rahmen von Rechtshandlungen erzeugt, verwendet und/oder manipuliert werden. Die `Rechtsfolge` steht speziell für die in einem Rechtssatz für einen Tatbestand vorgesehene abstrakte Beschreibung der rechtlichen Konsequenz. Im Gegensatz dazu ist die `VwRechtsfolge` eine Subklasse von `VerwaltungsHandlungsgegenstand`, der analog zum Rechtshandlungsgegenstand jedoch mit einem konkreten Bezug zur Verwaltungstätigkeit definiert ist. Der `VerwaltungsHandlungsgegenstand` repräsentiert somit abstrakte Konzepte und konkrete Dinge, die im Rahmen von Rechtshandlungen eines Trägers von öffentlichen Aufgaben (d.h. einer öffentlichen Verwaltung) erzeugt, verwendet und/oder manipuliert werden. Entsprechend hat auch die `VwRechtsfolge` einen Bezug zur Verwaltungstätigkeit, wodurch neben der rechtlichen Konsequenz und deren Zulässigkeit auch der Beitrag zur Zielerreichung relevant ist. Deshalb handelt es sich bei der `VwRechtsfolge` um eine Spezialisierung von `Rechtsfolge` und nicht um diese Rechtsfolge selbst.

Mit der auf diese Weise formalisierten Ontologie sollen Elemente im Text einer Rechtsvorschrift mit einer definierten Semantik versehen werden können, um sie als Ursprünge von Anforderungen handhabbar zu machen. Ein Textelement (z.B. Wort, Wortgruppe, Satz, Absatz) wird zu diesem Zweck mit einer Annotation versehen und repräsentiert dadurch eine Instanz einer Klasse der Ontologie. Der Ansatz dieser Arbeit beschränkt sich auf Instanzen der Konzepte des Verwaltungshandelns, um einen Beitrag zur Verfolgbarkeit im Vorfeld der Anforderungsspezifikation zu liefern. Die darüber hinausgehende Annotation von Textelementen in Form von Instanzen der Normtext- und

Rechtssatzkonzepte oder der Rechtskonzepte ist möglich, aber aus Sicht der Verfolgbarkeit nicht erforderlich.[329]

3.3 Semantische Annotation von Rechtsvorschriften

Die praktische Anwendung der in den vorangegangenen Abschnitten vorgestellten Ontologie erfolgt, indem Texte von Rechtsvorschriften durch Instanziierung der Ontologie-Konzepte des Verwaltungshandelns mit Annotationen versehen werden. Diese Annotation erfolgt durch einen Bearbeiter, der dafür spezielle Werkzeuge einsetzt. Der Bearbeiter dokumentiert durch die Annotation das im konkreten Projektkontext erzielte Verständnis von den zugrunde liegenden Rechtsvorschriften.[330] Im einfachsten Fall erfolgt dies aufgrund seiner Erfahrung und seines Wissens. In komplexeren Projekten können Abstimmungsprozesse und Verhandlungen mehrerer Beteiligter erforderlich sein, um ein gemeinsames Verständnis durch Annotation der relevanten Rechtsvorschriften zu erarbeiten. Die Organisation dieser Abstimmungsprozesse und Verhandlungen wird hier als Aufgabe des Projektmanagements gesehen und deshalb im Rahmen dieser Arbeit nicht betrachtet. Das gemeinsame Verständnis kann mit verschiedenen Werkzeugen aus dem Kontext des SemanticWeb dokumentiert werden.[331]

3.3.1 Nutzung des Werkzeugs „OntoMat-Annotizer“

Im Rahmen dieser Arbeit wird exemplarisch das Werkzeug „OntoMat-Annotizer“[332] verwendet, um den Text von Rechtsvorschriften mit Annotationen zu versehen. Der nachfolgende Screenshot zeigt das Werkzeug (Abbildung 38). Zunächst wird es mit den wesentlichen Werkzeugfunktionen kurz vorgestellt. Dann folgen Beispiele aus den mit dem Werkzeug auf Basis der entwickelten Ontologie annotierten Rechtsvorschriften.

[329] Die Möglichkeit ist in der Ontologie durch die Object Properties „konkretisiertRechtskonzept“ und „konkretisiertTextkonzept“ gegeben (vgl. Anhang A).

[330] Es ist möglich, dass durch das erzielte Verständnis auch generische oder wiederverwendbare Annotationen erarbeitet wurden, die von anderen Projekten ebenfalls verwendet werden können. Die sich hieran schließenden Fragen der Wiederverwendung und des Wissensmanagements solcher Annotationen gehen über den Rahmen dieser Arbeit hinaus.

[331] Für eine Liste bekannter Annotationswerkzeuge vgl. http://annotation.semanticweb.org/tools/ und http://semanticweb.org/wiki/Category:Semantic_annotation_tool, letzter Aufruf: 21.08.2011.

[332] Weitere Informationen zum Werkzeug OntoMat-Annotizer und eine Möglichkeit zum Herunterladen sind unter der folgenden URL zu finden: http://annotation.semanticweb.org/ontomat/index.html (letzter Aufruf: 21.08.2011).

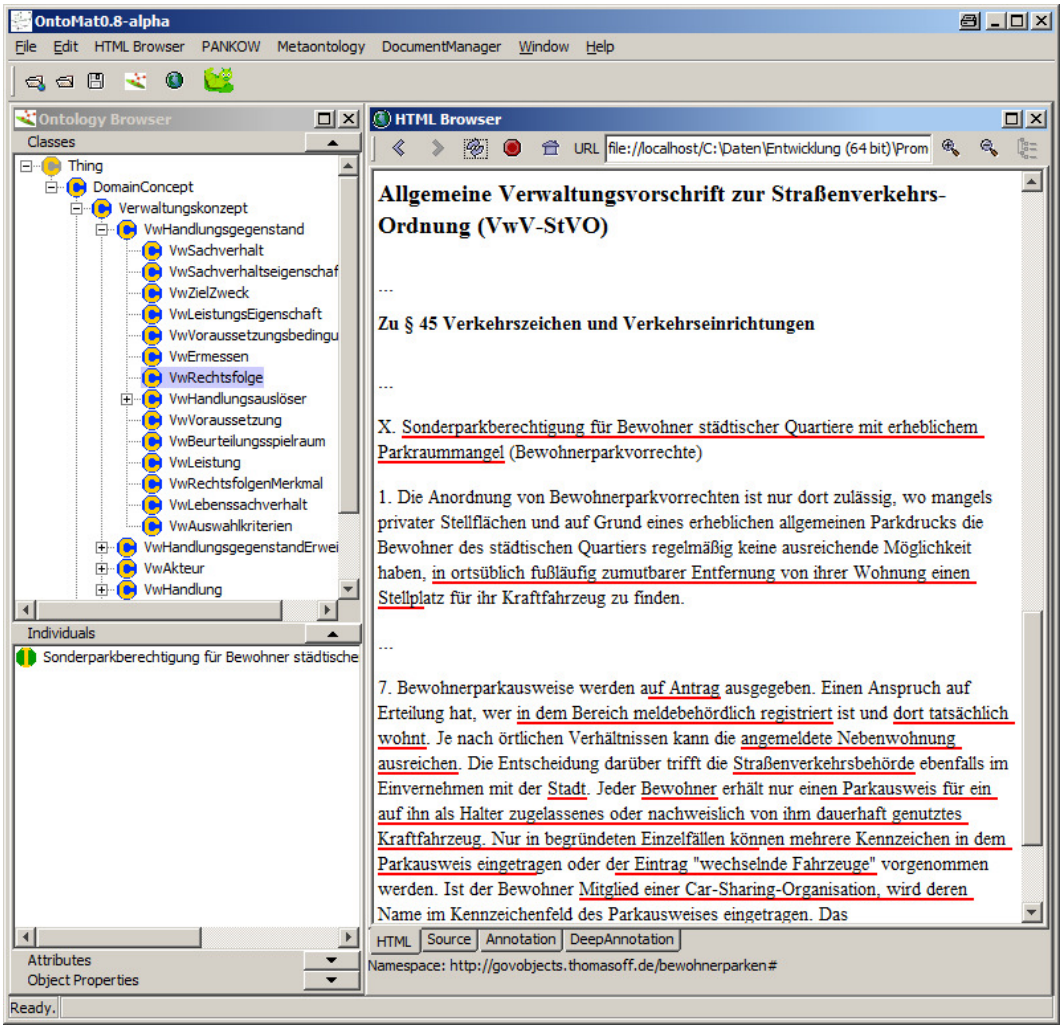


Abbildung 38 – Screeenshot des Werkzeugs OntoMat

Um eine Rechtsvorschrift mit Annotationen zu versehen, wird zunächst die Ontologie im OWL-Format in das Werkzeug geladen. Sie kann dort mit den enthaltenen Konzepten (im Unterfenster „Ontology Browser“ im linken Teil des Screenshots) eingesehen werden. Im nächsten Schritt werden die relevanten Rechtsvorschriften in einem Browserfenster geöffnet (im Unterfenster „HTML Browser“ im rechten Teil des Fensters). Relevante Textpassagen, die Instanzen der Ontologie-Konzepte repräsentieren sollen, werden mit der Maus direkt im Text markiert. Anschließend können sie per Drag und Drop mit den Ontologie-Klassen in Verbindung gebracht werden. Dadurch wird automatisch eine Instanz der Klasse angelegt, die in der Liste „Individuals“ im Ontology Browser angezeigt wird. Gleichzeitig wird der Text der Rechtsvorschrift mit einer entsprechenden Annotation versehen. Annotationen können im Text (durch Unterstreichung) sichtbar hervorgehoben werden. Außerdem werden Annotationen als Kommentar im Quelltext eines lokal gespeicherten HTML-Dokuments eingebettet und können auch in ihrer XML-Struktur (über Registerkarten im Unterfenster „HTML Browser“) angezeigt werden. Nachdem Annotationen in die Rechtsvorschrift eingefügt wurden, steht damit ein HTML-Dokument zur Verfügung, das als Input für weitere Transformationsschritte genutzt werden kann. Für die Arbeit mit dem OntoMat ist es wichtig, in einem vorbereitenden Schritt die relevanten Rechtsvorschriften innerhalb eines

HTML-Dokuments zusammenzufassen, weil das Werkzeug jeweils nur ein Dokument öffnen kann.[333]

Ausgangspunkt für die Instanziierung ist im Beispiel des Bewohnerparkens das Straßenverkehrsgesetz (StVG)[334]. In §6 (1) Nr. 14 werden in diesem Gesetzt zunächst die Ziele und Zwecke für das Handeln der Verwaltung vorgeben. Dort heißt es:

> „Das Bundesministerium für Verkehr, Bau und Stadtentwicklung wird ermächtigt, Rechtsverordnungen mit Zustimmung des Bundesrates zu erlassen über [...] die Beschränkung des Haltens und Parkens zugunsten der Bewohner städtischer Quartiere mit erheblichem Parkraummangel sowie die Schaffung von Parkmöglichkeiten für Schwerbehinderte mit außergewöhnlicher Gehbehinderung und Blinde, insbesondere in unmittelbarer Nähe ihrer Wohnung oder ihrer Arbeitsstätte; [...]."

Um Ziele und Zwecke des Verwaltungshandelns in Rechtsvorschriften ausdrücken zu können, enthält die Ontologie das Konzept des `ZielZweck`. Mit dem Werkzeug wird per Drag&Drop eine Instanz dieses Konzepts als Individuum erzeugt (Abbildung 39). Gleichzeitig wird die markierte Textpassage durch das Werkzeug mit einer Annotation versehen, die aus zwei Teilen besteht. Im folgenden Listing 12 sind die Zeilen 21 bis 23 die Repräsentation der Instanz des `ZielZweck`-Konzepts aus der Ontologie. Die Zeilen 25 bis 29 repräsentieren die korrespondierende Textpassage im Text der Rechtsvorschrift.

[333] Darüber hinaus erwies es sich als zwingend erforderlich, die Zeichenkodierung ISO-8859-1 für dieses Dokument festzulegen, da andernfalls die Darstellung der Umlaute (z. B: ö, Ä, ß) die weitere Verarbeitung der Annotationen erschweren.

[334] Straßenverkehrsgesetz in der Fassung der Bekanntmachung vom 5. März 2003 (BGBl. I S. 310, 919), das zuletzt durch Artikel 2 des Gesetzes vom 12. Juli 2011 (BGBl. I S. 1378) geändert worden ist (im Folgenden kurz: StVG).

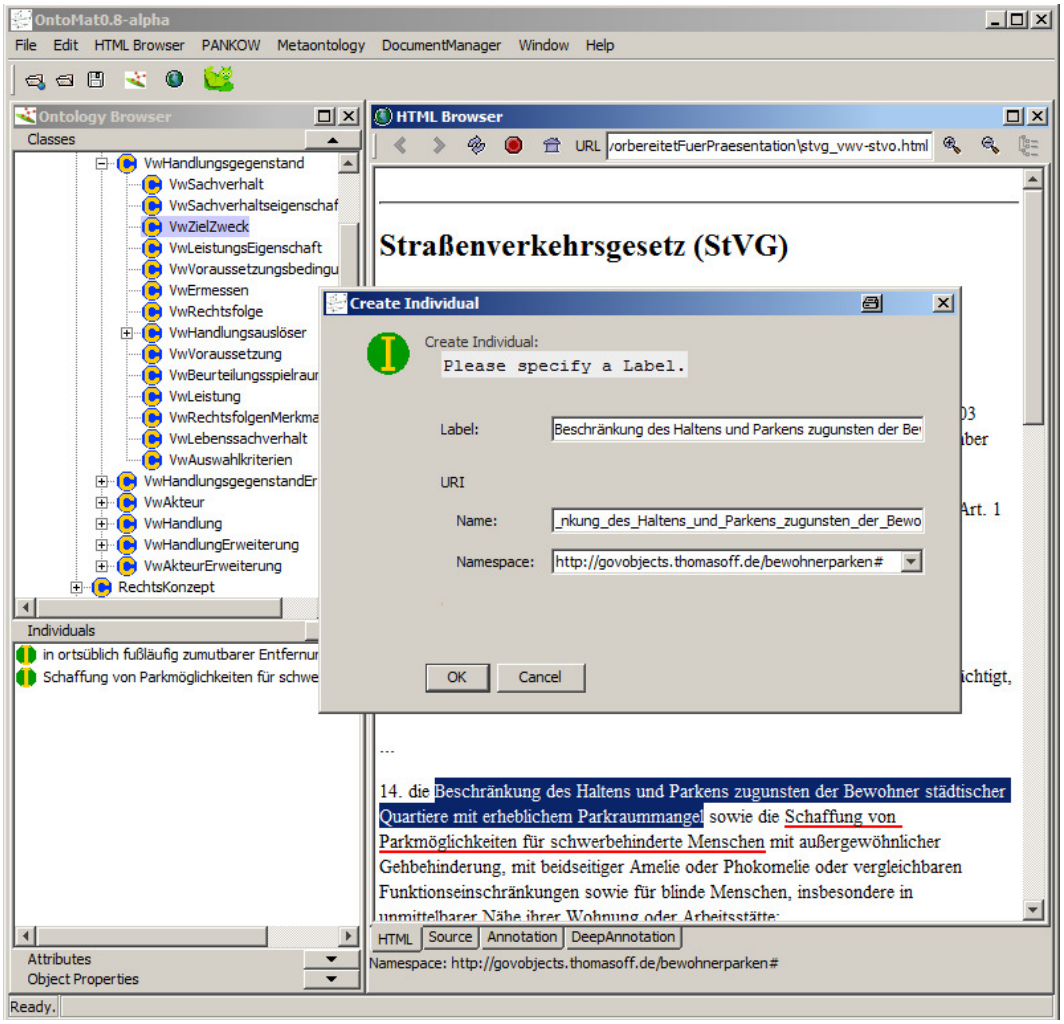


Abbildung 39 – Annotation der Ziele und Zwecke des Verwaltungshandelns im Straßenverkehrsgesetz

```
(1)    <rdf:RDF
(2)      xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
(3)      xmlns:dc="http://purl.org/dc/elements/1.1/"
(4)      xmlns:ontomat="http://annotation.semanticweb.org/ontologies/cream/ontomat#
       "
(5)      xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(6)      xmlns:rss="http://purl.org/rss/1.0/"
(7)      xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
(8)      xmlns="http://govobjects.thomasoff.de/bewohnerparken#"
(9)      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(10)     xmlns:govobjects-
       ontology="http://govobjects.thomasoff.de/ontology/govobjects-
         ontology.owl#"
(11)     xmlns:owl="http://www.w3.org/2002/07/owl#"
(12)     xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
(13)     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(14)     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
(15)
(16)     <owl:Ontology rdf:about="http://govobjects.thomasoff.de/bewohnerparken#">
(17)       <owl:imports rdf:resource="http://annotation.semanticweb.org/ontologies/
           cream/ontomat#"/>
(18)       <owl:imports
       rdf:resource="http://govobjects.thomasoff.de/ontology/govobjects-
           ontology.owl#"/>
(19)     </owl:Ontology>
(20)
(21)     <govobjects-ontology:VwZielZweck
       rdf:about="http://govobjects.thomasoff.de/bewohnerparken#Beschr_nkung_des_Ha
       ltens_und_Parkens_zugunsten_der_Bewohner_st_dtischer_Quartiere_mit_erheblich
       em_Parkraummangel">
```

```
(22)      <rdfs:label>Beschränkung des Haltens und Parkens zugunsten der Bewohner
      städtischer Quartiere mit erheblichem Parkraummangel</rdfs:label>
(23)    </govobjects-ontology:VwZielZweck>
(24)
(25)    <ontomat:ReificationDataIndividual>
(26)      <rdfs:label>about: http://govobjects.thomasoff.de/bewohnerparken#Beschr_
      nkung_des_Haltens_und
          _Parkens_zugunsten_der_Bewohner_st_dtischer_Quartiere_mit_erheblichem_Pa
      rkraummangel</rdfs:label>
(27)      <ontomat:CreationSource>http://govobjects.thomasoff.de/bewohnerparken#xp
      ointer(
          //point()[598]/range-to(//point()[710])</ontomat:CreationSource>
(28)      <ontomat:AboutIndividual>http://govobjects.thomasoff.de/bewohnerparken#B
      eschr_nkung_des_Haltens_
          und_Parkens_zugunsten_der_Bewohner_st_dtischer_Quartiere_mit_erheblichem
      _Parkraummangel</ontomat:AboutIndividual>
(29)    </ontomat:ReificationDataIndividual>
(30)
(31)  </rdf:RDF>
```

Listing 12 – Annotation der Ziele und Zwecke des Verwaltungshandelns im Straßenverkehrsgesetz

Schrittweise werden die im StVG enthaltenen Ziele und Zwecke durch weitere Rechts- und Verwaltungsvorschriften bis auf eine operative Ebene konkretisiert. Dazu zählen dann beispielsweise Vorschriften, die Regelungen zur Kennzeichnung von Parkflächen (z.B. in StVO §45 (1b) Nr. 2a.[335]) und zur Vergabe von Parkausweisen umfassen (z.B. in VwV-StVO zu §45). Auf dieser operativen Ebene werden sie durch den Bearbeiter in der Verwaltung in Einzelfallentscheidungen umgesetzt. Für das Bewohnerparken ist in der Allgemeinen Verwaltungsvorschrift zur Straßenverkehrsordnung (VwV-StVO) die Vergabe von Parkausweisen wie folgt geregelt:

„[...]

X. Sonderparkberechtigung für Bewohner städtischer Quartiere mit erheblichem Parkraummangel (Bewohnerparkvorrechte)

1. Die Anordnung von Bewohnerparkvorrechten ist nur dort zulässig, wo mangels privater Stellflächen und auf Grund eines erheblichen allgemeinen Parkdrucks die Bewohner des städtischen Quartiers regelmäßig keine ausreichende Möglichkeit haben, in ortsüblich fußläufig zumutbarer Entfernung von ihrer Wohnung einen Stellplatz für ihr Kraftfahrzeug zu finden. [...]

7. Bewohnerparkausweise werden auf Antrag ausgegeben. Einen Anspruch auf Erteilung hat, wer in dem Bereich meldebehördlich registriert ist und dort tatsächlich wohnt. Je nach örtlichen Verhältnissen kann die angemeldete Nebenwohnung ausreichen. Die Entscheidung darüber trifft die Straßenverkehrsbehörde ebenfalls im Einvernehmen mit der Stadt. Jeder Bewohner erhält nur einen Parkausweis für ein auf ihn als Halter zugelassenes oder nachweislich von ihm dauerhaft genutztes Kraftfahrzeug. Nur in begründeten Einzelfällen können mehrere Kennzeichen in dem Parkausweis eingetragen oder der Eintragung "wechselnde Kraftfahrzeuge" vorgenommen werden. Ist der Bewohner Mitglied einer Car-Sharing-Organisation, wird deren Name im Kennzeichenfeld des Parkausweises eingetragen. Das Bewohnerparkvorrecht gilt dann nur für das Parken eines von außen deutlich erkennbaren Fahrzeugs dieser Organisation (Aufschrift, Aufkleber am Fahrzeug); darauf ist der Antragsteller schriftlich hinzuweisen.

8. Der Bewohnerparkausweis wird von der zuständigen Straßenverkehrsbehörde erteilt. Dabei ist das Muster zu verwenden, das das Bundesministerium für Verkehr, Bau- und Wohnungswesen im Verkehrsblatt bekannt gibt. [...]“

Aus diesem Text lassen sich nach dem zuvor beschriebenen Mechanismus weitere Instanzen von Konzepten der Ontologie ableiten. Beispielsweise lässt sich außer den Voraussetzungen für die Erteilung eines Bewohnerparkausweises (z.B. dass der Bewohner Halter des zu parkenden KFZ ist oder es dauerhaft nutzt) und den Rechtsfolgenmerkmalen (z.B. einem

[335] Straßenverkehrs-Ordnung vom 16. November 1970 (BGBl. I S. 1565), die zuletzt durch Artikel 1 der Verordnung vom 1. Dezember 2010 (BGBl. I S. 1737) geändert worden ist (im Folgenden kurz: StVO).

Kennzeicheneintrag im Bewohnerparkausweis) anhand der Formulierung "in begründeten Einzelfällen können mehrere Kennzeichen in dem Parkausweis eingetragen werden" auch Ermessensspielraum identifizieren. Das entsprechende Ergebnis in der Benutzeroberfläche des OntoMat zeigt Abbildung 40 und die Annotationen als OWL-Individuen werden in Listing 13 dargestellt. Bei der Instanziierung der Konzepte wurden zur Illustration teilweise die markierten Textpassagen direkt als Label bzw. URI der Instanz übernommen (z.B. bei der Instanz von VwVoraussetzungsbedingung mit dem Label "nachweislich von ihm dauerhaft genutztes Kraftfahrzeug", Zeilen 29 bis 31). Teilweise wurde auch eine freie Bezeichnung gewählt, wie bei der Instanz des Konzepts `VwAntrag` mit dem Label "AntragBewohnerparken" (Zeilen 53 bis 55), um die unterschiedlichen Möglichkeiten zu demonstrieren und in der nachfolgenden Verarbeitung auf die damit verbundenen Unterschiede eingehen zu können.

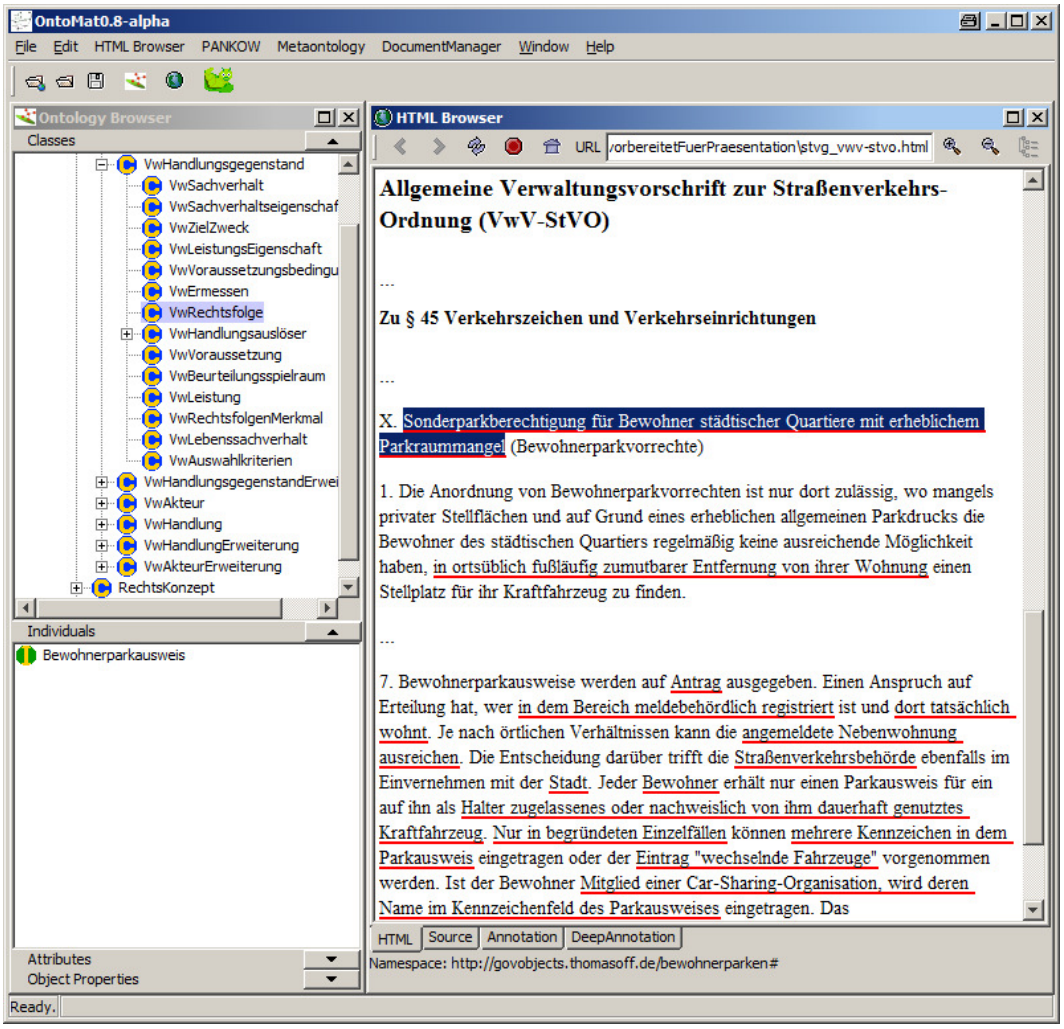


Abbildung 40 – Antrag auf Bewohnerparkausweis als Individuum im OntoMat

```
(1)   <rdf:RDF
(2)       xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
(3)       xmlns:dc="http://purl.org/dc/elements/1.1/"
(4)
      xmlns:ontomat="http://annotation.semanticweb.org/ontologies/cream/ontomat#"
(5)       xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(6)       xmlns:rss="http://purl.org/rss/1.0/"
(7)       xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
(8)       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(9)       xmlns="http://govobjects.thomasoff.de/ontology/bewohnerparken#"
(10)      xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-
          ontology.owl#"
```

```
(11)      xmlns:owl="http://www.w3.org/2002/07/owl#"
(12)      xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
(13)      xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(14)      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
(15)
(16)    <owl:Ontology
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#">
(17)      <owl:imports rdf:resource="http://govobjects.thomasoff.de/ontology/"/>
(18)      <owl:imports

      rdf:resource="http://annotation.semanticweb.org/ontologies/cream/ontomat#"/>
(19)    </owl:Ontology>
(20)
(21)  <!-- ... -->
(22)
(23)    <govobjects-ontology:VwRechtsfolgenMerkmal
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#Car_Sharin
      g_Organisation__wird_deren_Name_im_Kennzeichenfeld_des_Parkausweises_eingetr
      agen">
(24)      <rdfs:label>Car-Sharing-Organisation, wird deren Name im Kennzeichenfeld
      des Parkausweises eingetragen</rdfs:label>
(25)    </govobjects-ontology:VwRechtsfolgenMerkmal>
(26)
(27)  <!-- ... -->
(28)
(29)    <govobjects-ontology:VwVoraussetzungsbedingung
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#nachweisli
      ch_von_ihm_dauerhaft_genutztes_Kraftfahrzeug">
(30)      <rdfs:label>nachweislich von ihm dauerhaft genutztes
      Kraftfahrzeug</rdfs:label>
(31)    </govobjects-ontology:VwVoraussetzungsbedingung>
(32)
(33)  <!-- ... -->
(34)
(35)    <govobjects-ontology:VwAntragsteller
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#Bewohner">
(36)      <rdfs:label>Bewohner</rdfs:label>
(37)    </govobjects-ontology:VwAntragsteller>
(38)
(39)  <!-- ... -->
(40)
(41)    <govobjects-ontology:VwRechtsfolge
      rdf:about=""http://govobjects.thomasoff.de/Bewohnerparken#Sonderparkberechti
      gung_f_r_Bewohner_st_dtischer_Quartiere_mit_erheblichem_Parkraummangel"">
(42)      <rdfs:label>Sonderparkberechtigung für Bewohner städtischer Quartiere
      mit erheblichem Parkraummangel</rdfs:label>
(43)    </govobjects-ontology:VwRechtsfolge>
(44)
(45)  <!-- ... -->
(46)
(47)    <govobjects-ontology:VwZielZweck
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#orts_blich
      _fu_l_ufig_zumutbarer_Entfernung_von_ihrer_Wohnung_einen_Stellplatz_f_r_ihr_
      Kraftfahrzeug">
(48)      <rdfs:label>ortsüblich fußläufig zumutbarer Entfernung von ihrer Wohnung
      einen Stellplatz für ihr Kraftfahrzeug</rdfs:label>
(49)    </govobjects-ontology:VwZielZweck>
(50)
(51)  <!-- ... -->
(52)
(53)    <govobjects-ontology:VwAntrag
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#AntragBewo
      hnerparken">
(54)      <rdfs:label>AntragBewohnerparken</rdfs:label>
(55)    </govobjects-ontology:VwAntrag>
(56)
(57)  <!-- ... -->
(58)
(59)    <govobjects-ontology:VwRechtsfolgenMerkmal
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#Eintrag_we
      chselnde_Fahrzeuge">
(60)      <rdfs:label>Eintrag wechselnde Fahrzeuge</rdfs:label>
(61)    </govobjects-ontology:VwRechtsfolgenMerkmal>
(62)
(63)  <!-- ... -->
```

```
(64)
(65)    <govobjects-ontology:VwErmessen
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#in_begr_nd
      eten_Einzelf_llen_k_nnen">
(66)      <rdfs:label>in begründeten Einzelfällen können</rdfs:label>
(67)    </govobjects-ontology:VwErmessen>
(68)
(69)    <!-- ... -->
(70)
(71)  </rdf:RDF>
```

Listing 13 – Auszug aus der RDF-Definition für die Annotation des Antrags auf Bewohnerparken

In der Ontologie des Verwaltungshandelns wurden für die OWL-Klasse `VerwaltungsKonzept` die Data Properties `Name`, `Arbeitsnotiz`, `Beschreibung` und `Rahmenbedingung` formalisiert. Alle Subklassen des Verwaltungskonzepts verfügen über diese Data Properties. Sie können im Werkzeug zur Beschreibung der Instanzen verwendet werden. Im Unterfenster "Ontology Browser" sind im Bereich "Attributes" die Properties einer ausgewählten Instanz dargestellt und können bearbeitet werden (Abbildung 41). Das Werkzeug ergänzt die OWL-Annotation dann um die entsprechenden Properties (Listing 14, Zeilen 20 bis 25). Mit diesem Mechanismus ist es möglich, die getroffenen Entwurfsentscheidungen bei der Instanziierung der Konzepte oder das Ergebnis fachlicher Abstimmungen nachvollziehbar zu dokumentieren.

Die dokumentierten Rahmenbedingungen der Umsetzung bzw. des Einsatzes des Konzepts können ebenfalls in die nachfolgende Modellierung übernommen werden. Beispielsweise resultiert aus der Textpassage „Muster zu verwenden“ die Rahmenbedingung, das dieses Muster farbig wiederzugeben ist (Listing 14, Zeilen 32 bis 34).

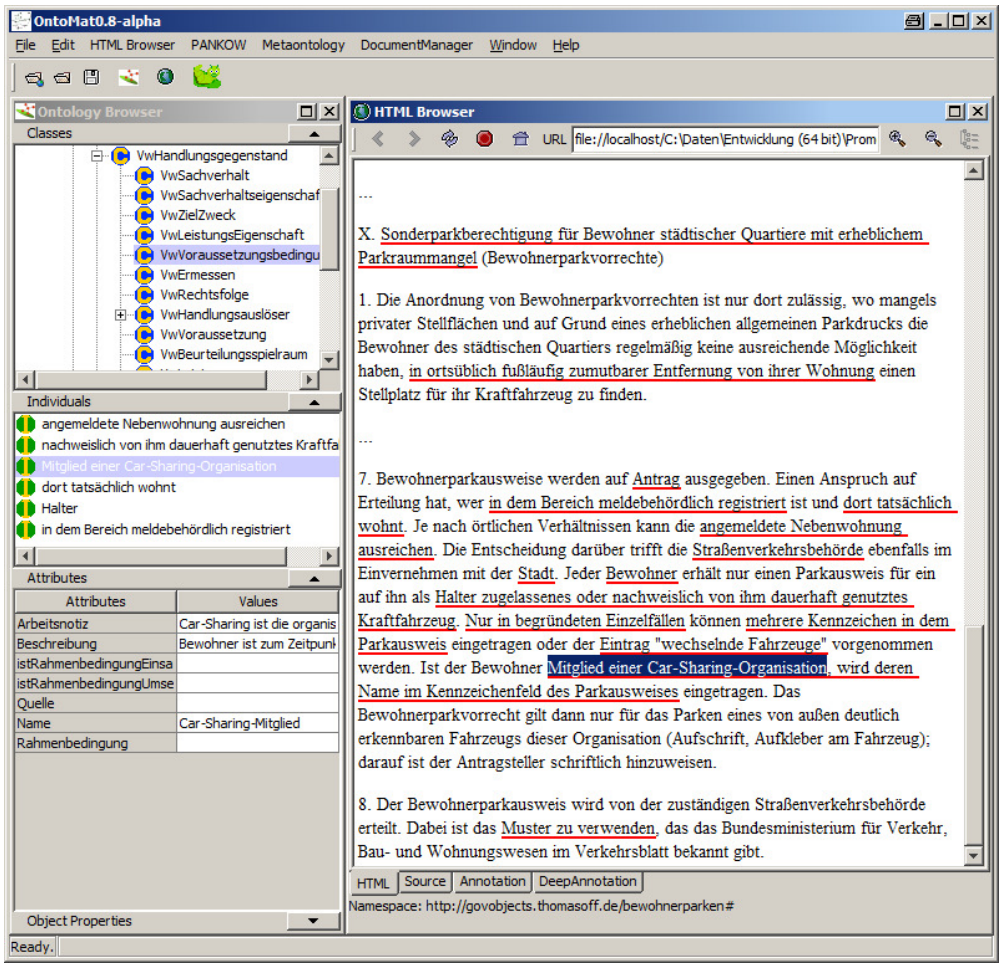


Abbildung 41 – Verwendung von Data Properties während der Instanziierung

```
(1)   <rdf:RDF
(2)       xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
(3)       xmlns:dc="http://purl.org/dc/elements/1.1/"
(4)       xmlns:ontomat="http://annotation.semanticweb.org/ontologies/cream/ontoma
      t#"
(5)       xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(6)       xmlns:rss="http://purl.org/rss/1.0/"
(7)       xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
(8)       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(9)       xmlns="http://govobjects.thomasoff.de/ontology/bewohnerparken#"
(10)      xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(11)      xmlns:owl="http://www.w3.org/2002/07/owl#"
(12)      xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
(13)      xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(14)      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
(15)
(16)    <!-- ... -->
(17)
(18)    <govobjects-ontology:VwVoraussetzungsbedingung
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#Bewohner_M
      itglied_einer_Car_Sharing_Organisation">
(19)      <rdfs:label>Bewohner Mitglied einer Car-Sharing-
      Organisation</rdfs:label>
(20)      <govobjects-ontology:Name
      rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
(21)      >Car-Sharing-Mitglied</govobjects-ontology:Name>
(22)      <govobjects-ontology:Beschreibung
      rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
(23)         Bewohner ist zum Zeitpunkt der Antragstelung Mitglied einer
      Carsharing-Organisation.</govobjects-ontology:Beschreibung>
(24)      <govobjects-ontology:Arbeitsnotiz
      rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
(25)      >CarSharing ist die organisierte gemeinschaftliche Nutzung eines oder
      mehrerer Autos. Das Autoteilen unter Nachbarn und Bekannten fällt im engeren
      Sinn heute nicht mehr unter den Begriff des Carsharings
      (Wikipedia).</govobjects-ontology:Arbeitsnotiz>
(26)    </govobjects-ontology:VwVoraussetzungsbedingung>
(27)
(28)    <!-- ... -->
(29)    <govobjects-ontology:VwRechtsfolgenMerkmal
      rdf:about="http://govobjects.thomasoff.de/Bewohnerparken#Muster_zu_verwenden
      ">
(30)    <govobjects-ontology:istMerkmalVonRechtsfolge
      rdf:resource="http://govobjects.thomasoff.de/Bewohnerparken#Sonderparkberech
      tigung_f_r_Bewohner_st_dtischer_Quartiere_mit_erheblichem_Parkraummangel"/>
(31)    <rdfs:label>Muster zu verwenden</rdfs:label>
(32)    <govobjects-ontology:istRahmenbedingungEinsatz
      rdf:datatype="http://www.w3.org/2001/ XMLSchema#boolean">true</govobjects-
      ontology:istRahmenbedingungEinsatz>
(33)    <govobjects-ontology:istRahmenbedingungUmsetzung
      rdf:datatype="http://www.w3.org/2001/ XMLSchema#boolean">true</govobjects-
      ontology:istRahmenbedingungUmsetzung>
(34)    <govobjects-ontology:Rahmenbedingung
      rdf:datatype="http://www.w3.org/2001/X MLSchema#string">Das Muster ist
      farbig. Obwohl die Farben nicht verbindlich festgelegt sind, ist die farbige
      Produktion notwendig.</govobjects-ontology:Rahmenbedingung>
(35)    </govobjects-ontology:VwRechtsfolgenMerkmal>
(36)
(37)  </rdf:RDF>
```

Listing 14 – Instanzen mit Ausprägungen von Data Properties

Zwischen den Instanzen können Beziehungen existieren, die sowohl für das Gesamtverständnis hilfreich als auch für die weitere Verarbeitung notwendig sind. In der Ontologie sind die möglichen Beziehungen der Instanzen durch die Object Properties formalisiert. Beispielsweise kann mit dem Object Property `hatAntragsteller` eine Instanz von `VwAntrag` mit einer Instanz von `VwAntragsteller` in Beziehung gesetzt werden. Im Werkzeug OntoMat werden für eine ausgewählte Instanz deren Object Properties angezeigt. Per Drag&Drop können Instanzen aus dem Wertebereich (range) zum Object Property zugeordnet werden (Abbildung 42). Die entsprechende Zuordnung wird automatisch zur OWL-Annotation hinzugefügt.

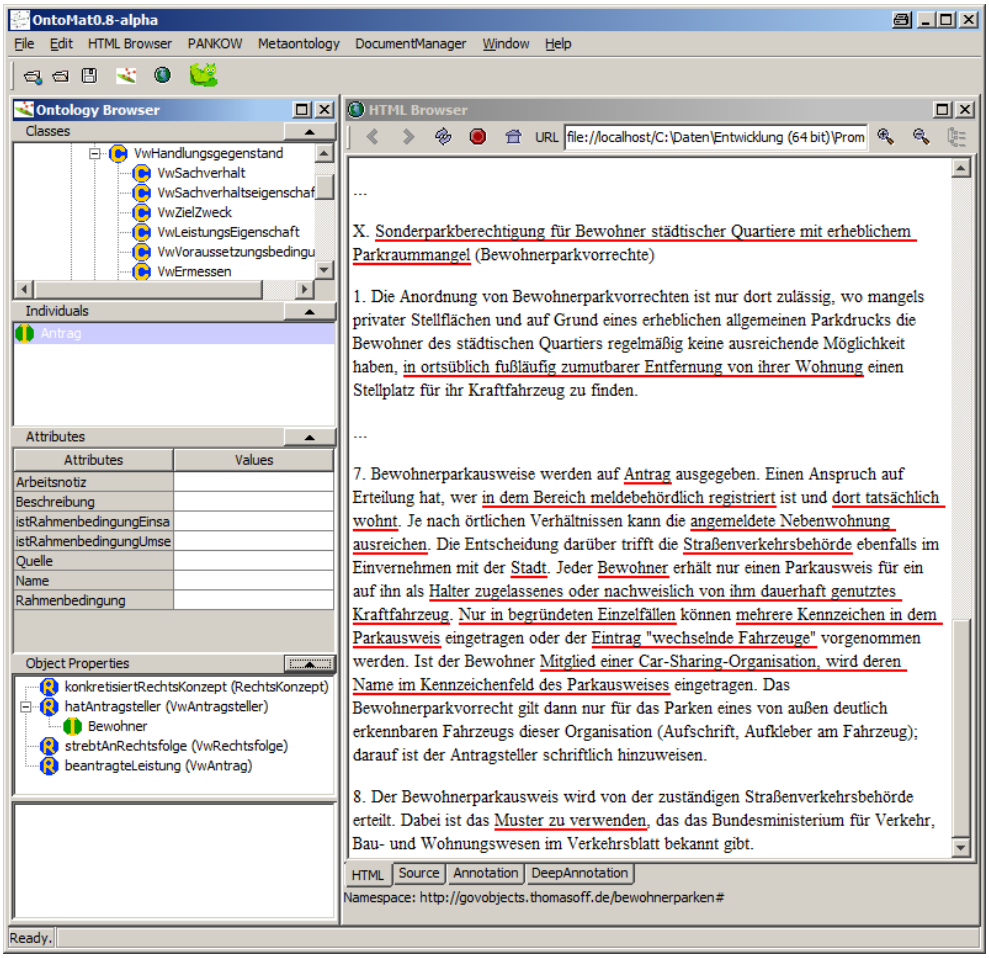


Abbildung 42 – Verwendung von Object Properties während der Instanziierung

```
(1)   <rdf:RDF
(2)       xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
(3)       xmlns:dc="http://purl.org/dc/elements/1.1/"
(4)       xmlns:ontomat="http://annotation.semanticweb.org/ontologies/cream/ontoma
      t#"
(5)       xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(6)       xmlns:rss="http://purl.org/rss/1.0/"
(7)       xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
(8)       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(9)       xmlns="http://govobjects.thomasoff.de/ontology/bewohnerparken#"
(10)      xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(11)      xmlns:owl="http://www.w3.org/2002/07/owl#"
(12)      xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
(13)      xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(14)      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
(15)
(16)    <!-- ... -->
(17)
(18)    <govobjects-ontology:VwAntrag
      rdf:about="http://govobjects.thomasoff.de/ontology/bewohnerparken#AntragBewo
      hnerparken">
(19)      <rdfs:label>AntragBewohnerparken</rdfs:label>
(20)      <govobjects-ontology:hatAntragsteller
      rdf:resource="http://govobjects.thomasoff.de/ontology/bewohnerparken#Bewohner"/>
(21)    </govobjects-ontology:VwAntrag>
(22)
(23)    <!-- ... -->
(24)
(25)  </rdf:RDF>
```

Listing 15 - Object Property hatAntragsteller

Das Werkzeug OntoMat bettet die Annotationen in den Quelltext eines HTML-Dokuments als Kommentar am Ende des Dokuments ein. Innerhalb dieses Kommentars sind neben den Instanzen auch Informationen über die repräsentierte Textpassage enthalten (vgl. Listing 12, Zeilen 25-29). Diese Informationen sind nicht unmittelbar in die annotierte Textpassage eingebettet. Vielmehr wird der seit 2002 als Entwurf vorliegende XPointer-Standard des W3C genutzt. Dadurch kann von außen auf die Inhalte des Textes verwiesen werden, ohne den Text selbst (z. B. durch Hinzufügungen) verändern zu müssen. Im OntoMat wird anhand von `xpointer(//point()[n]/range-to(//point()[n+l])` der Textbereich von Zeichenposition `n` bis Zeichenposition `n + l` als Annotationskontext eingegrenzt, wobei `l` die Zeichenlänge des Kontextes ist. Dieser XPointer-Bereich bezieht sich in der Implementierung im OntoMat auf den sichtbaren Text des Gesamtdokumentes, d.h. HTML-Sprachelemente bleiben bei der Positionsbestimmung unberücksichtigt. Dieses Prinzip wird von Oren et al. und Handschuh vorgestellt, allerdings fehlt eine weitergehende Dokumentation (z.B. Umgang mit leeren Absätzen, Aufzählungslisten und den Zeilenumbruchzeichen Carriage Return und Line Feed)[336]. Darüber hinaus sind XPointer-Verweise nicht stabil bezogen auf Änderungen des Textes. Verschiebt sich bei einer Textänderung der Annotationskontext bezogen auf den Text, bleibt die Positionsangabe im XPointer unverändert, so dass die Annotation nach der Veränderung möglicherweise auf einen falschen Kontext verweist, ohne dass dies vom Autor der Annotation bemerkbar ist.

Prinzipiell ist die gewählte Implementierung mit dem XPointer-Standard eine Möglichkeit, die Annotationen mit ihrem Kontext zu verbinden. Dadurch sind sowohl die erzeugten Individuen als Instanzen von Konzepten der Ontologie als auch deren Properties mit dem Kontext verbunden. Diese Verbindung kann als Basis für die Verfolgbarkeit von Individuen zu ihrem Ursprung in Elementen der Texte von Rechtsvorschriften dienen.

3.3.2 Manuelle Kodierung der RDFa-Annotationen

Der neuere RDFa-Standard wird von dem im Vorangegangenen eingesetzten Werkzeug OntoMat nicht unterstützt, da die Werkzeugentwicklung 2004 endete. Andere verfügbare Annotations-Werkzeuge, die RDFa prinzipiell unterstützen, wiesen im Rahmen dieser Arbeit Kompatibilitäts- und Stabilitätsprobleme beim Einsatz benutzerspezifischer Ontologien auf. Deshalb wird in diesem Abschnitt die Verwendung von RDFa anhand manuell kodierter Annotationen dargestellt, für die ein einfacher XML-Editor verwendet wurde. Den Ausgangspunkt für das nachfolgende Beispiel bildet der folgende Auszug aus dem StVG und der VwV-StVO, in dem die Ziele des Verwaltungshandelns festgelegt werden.

> „Das Bundesministerium für Verkehr, Bau und Stadtentwicklung wird ermächtigt, Rechtsverordnungen mit Zustimmung des Bundesrates zu erlassen über [...] die Beschränkung des Haltens und Parkens zugunsten der Bewohner städtischer Quartiere mit erheblichem Parkraummangel sowie die Schaffung von Parkmöglichkeiten für Schwerbehinderte mit außergewöhnlicher Gehbehinderung und Blinde, insbesondere in unmittelbarer Nähe ihrer Wohnung oder ihrer Arbeitsstätte; [...]."
>
> „[...] X. Sonderparkberechtigung für Bewohner städtischer Quartiere mit erheblichem Parkraummangel (Bewohnerparkvorrechte)
>
> 1. Die Anordnung von Bewohnerparkvorrechten ist nur dort zulässig, wo mangels privater Stellflächen und auf Grund eines erheblichen allgemeinen Parkdrucks die Bewohner des städtischen Quartiers regelmäßig keine ausreichende Möglichkeit haben, in ortsüblich fußläufig zumutbarer Entfernung von ihrer Wohnung einen Stellplatz für ihr Kraftfahrzeug zu finden. [...]"

Das XHTML-Dokument mit dem Text dieses Abschnittes wird manuell mit RDFa-Annotationen versehen, die in Listing 16 dargestellt sind. Dazu müssen im Kopf des Dokuments die entsprechenden Namensräume für OWL, RDF und RDFS definiert werden.

[336] Vgl. [Oren et al., 2006], [Handschuh, 2005], S. 16 ff.

Es ist auch möglich, die Namensräume der Ontologie und der Annotationen anzugeben, um die Werteausprägungen der Annotationsattribute zu verkürzen.

RDFa kann u. a. mit dem span-Element des HTML-Sprachumfangs verwendet werden, das es um spezielle Eigenschaften erweitert. Durch Verwendung der Eigenschaft typeof in Zeile 19 wird ausgedrückt, dass der zwischen dem Beginn und dem Ende des Span-Elements stehende Text eine Instanz der OWL-Klasse VwZielZweck ist. Mit der about-Eigenschaft wird der Instanz eine eindeutige Bezeichnung zugewiesen. Seinen Namen (rdfs:label) erhält die Instanz in Zeile 19 durch Verwendung der property-Eigenschaft. Sie drückt aus, das der vom span-Element umschlossene Text dem rdfs:label entspricht. In den Zeilen 21 und 27 sind durch Annotation anderer Textpassagen weitere Instanzen der Klasse VwZielZweck dargestellt.

In Zeile 28 ist dargestellt, wie Beziehungen zwischen Instanzen ausgedrückt werden. Die about-Eigenschaft des span-Elements bezeichnet die Instanz, die durch Annotation in der Zeile 27 definiert ist. Mit der rel-Eigenschaft wird der Typ der Beziehung zwischen beiden Instanzen bezeichnet (govobjects-ontology:konkretisiertZiel). Die resource-Eigenschaft verweist auf die Instanz in Zeile 19. Das sich dadurch ergebende Tripel besagt, dass das Ziel, „in ortsüblich fußläufig zumutbarer Entfernung von ihrer Wohnung [einen Parkplatz zu finden]“ das Ziel „Beschränkung des Haltens und Parkens zugunsten der Bewohner städtischer Quartiere mit erheblichem Parkraummangel“ im Sinne der Ontologie des Verwaltungshandelns „konkretisiert“.

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML+RDFa 1.0//EN"
(3)       "http://www.w3.org/MarkUp/DTD/xhtml-rdfa-1.dtd">
(4)   <html xmlns="http://www.w3.org/1999/xhtml"
(5)       xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(6)       xmlns:owl="http://www.w3.org/2002/07/owl#"
(7)       xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(8)       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(9)       xmlns:bewohnerparken="http://govobjects.thomasoff.de/Bewohnerparken#"
(10)      version="XHTML+RDFa 1.0">
(11)
(12)  <head>
(13)  <meta http-equiv="content-type" content="application/xhtml+xml; charset=UTF-
      8"/>
(14)      <title>Straßenverkehrsgesetz (StVG) und Allgemeine Verwaltungsvorschrift
      zur
            Straßenverkehrs-Ordnung (VwV-StVO)</title>
(15)    </head>
(16)    <body>
(17)      <!-- ... -->
(18)      <p>14. die
(19)        <span typeof="govobjects-ontology:VwZielZweck"
      about="http://govobjects.thomasoff.de/bewohnerparken#Beschr_nkung_des_Halten
      s_und_
      Parkens_zugunsten_der_Bewohner_st_dtischer_Quartiere_mit_erheblichem_Parkrau
      mmangel" property="rdfs:label">Beschränkung des Haltens und Parkens
      zugunsten der Bewohner städtischer Quartiere mit erheblichem
      Parkraummangel</span>
(20)        sowie die
(21)        <span typeof="govobjects-ontology:VwZielZweck"
      about="http://govobjects.thomasoff.de/bewohnerparken#Schaffung_von_Parkm_gli
      chkeiten_f_r_schwerbehinderte_Menschen" property="rdfs:label">
(22)  Schaffung von Parkmöglichkeiten für schwerbehinderte Menschen</span>
(23)  mit außergewöhnlicher Gehbehinderung, mit beidseitiger Amelie oder
      Phokomelie oder vergleichbaren Funktionseinschränkungen sowie für blinde
      Menschen, insbesondere in unmittelbarer Nähe ihrer Wohnung oder
      Arbeitsstätte;</p>
(24)      <p/>
(25)      <!-- ... -->
(26)      <p>1. Die Anordnung von Bewohnerparkvorrechten ist nur dort zulässig, wo
      mangels privater Stellflächen und auf Grund eines erheblichen allgemeinen
      Parkdrucks die Bewohner des städtischen Quartiers regelmäßig keine
      ausreichende Möglichkeit haben, in
```

```
(27)        <span typeof="govobjects-ontology:VwZielZweck"
      about="http://govobjects.thomasoff.de/bewohnerparken#in_orts_blich_fu_l_ufig
      _zumutbarer_Entfernung_von_ihrer_Wohnung" property="rdfs:label">ortsüblich
      fußläufig zumutbarer Entfernung von ihrer Wohnung</span>
(28)        <span
      about="http://govobjects.thomasoff.de/bewohnerparken#in_orts_blich_fu_l_ufig
      _zumutbarer_Entfernung_von_ihrer_Wohnung" rel="govobjects-
      ontology:konkretisiertZiel"
      resource="http://govobjects.thomasoff.de/bewohnerparken#Beschr_nkung_des_Hal
      tens_und_Parkens_zugunsten_der_Bewohner_st_dtischer_Quartiere_mit_erhebliche
      m_Parkraummangel"/>
(29)        einen Stellplatz für ihr Kraftfahrzeug zu finden.
(30)      </p>
(31)      <!-- ... -->
(32)    </body>
(33)  </html>
```

Listing 16 – RDFa-Annotation zur Instanziierung der Ontologie des Verwaltungshandelns

Die RDFa-Annotation sind innerhalb der Browserdarstellung des HTML-Dokuments standardmäßig nicht sichtbar. Mittels einer Browsererweiterung wie dem *RDFa Developer*[337], der für den Browser *Firefox*[338] verfügbar ist, können die RDFa-Annotationen innerhalb des Dokuments angezeigt werden. Im unteren Teil des Browserfensters (Abbildung 43) sind die Annotationen dargestellt. Sie können ausgewählt werden und heben im oberen Teil des Fensters die korrespondierende Textpassage hervor.

[337] Die Homepage des Open Source-Projektes RDFa Developer ist erreichbar unter: http://rdfadev.sourceforge.net/ (letzter Aufruf: 08.08.2011.)

[338] Die Hompage der Mozilla Foundation, die u. a. den Firefox-Browser zur Verfügung stellt, ist erreichbar unter: http://www.mozilla.org/ (letzter Aufruf: 08.08.2011).

Abbildung 43 – Darstellung der RDFa-Annotationen im Firefox-Browser mit dem Add-on „RDFa Developer“

Es ist das Ziel des RDFa-Standards einen Mechanismus bereitzustellen, der eine direkte Verbindung zwischen den für den Benutzer sichtbaren Informationen und maschinell verarbeitbaren Informationen herstellt.[339] Deshalb erfolgen RDFa-Annotationen in Form von Attributen, die den Elementen des HTML-Sprachumfangs zugewiesen werden. Damit haben RDFa-Annotationen in der Regel immer einen unmittelbaren Bezug zur annotierten

[339] Vgl. [W3C RDF, 2004a].

Textpassage.[340] Die Verfolgbarkeit der Annotation zu ihrem Annotationskontext ist somit gewährleistet. Werden Änderungen am HTML-Dokument vorgenommen, bleibt (im Gegensatz zur Verwendung des XPointer-Standards beim OntoMat) der Zusammenhang zwischen der Annotation und der annotierten Textpassage erhalten. Allerdings setzt die Hinzufügung von RDFa-Annotationen die Möglichkeit voraus, das HTML-Dokument bearbeiten zu können. Dies ist bei Dokumenten (z. B. Rechtsvorschriften), die auf Webservern bereitgestellt werden, standardmäßig nicht der Fall. Deshalb müssen diese Dokumente (zuvor heruntergeladen und) lokal gespeichert werden.

3.4 Zusammenfassung

In diesem Abschnitt wurde zunächst die im Rahmen dieser Arbeit entwickelte Ontologie des Verwaltungshandelns vorgestellt, die einerseits auf rechtstheoretische Bestandteile in Normtexten zurückgeht und andererseits verwaltungsspezifische Konzepte umfasst. Ihr Einsatz praktischer Einsatz im Kontext des Semantic Web wurde anhand des Werkzeugs OntoMat und der manuellen Kodierung von RDFa-Annotationen demonstriert. Dadurch wurde gezeigt, dass die Ontologie direkt zur Annotation von Rechtsvorschriften, die als lokal gespeicherte HTML-Dokumente vorliegen oder im Internet zugreifbar sind, eingesetzt werden kann.

Für die Annotation konnten Verwaltungskonzepte instanziiert werden, die sich über Rechtskonzepte aus Rechtssätzen ableiten lassen, da Rechtssätze auf definierte Bestandteile in den Texten von Rechtsnormen zurückgeführt werden können. Aber auch Konzepte, die nicht über diesen Weg als Bestandteil einer Rechtsvorschrift vorausgesetzt werden konnten (z.B. Ziele und Zwecke des Verwaltungshandelns), wurden exemplarisch identifiziert, um ihre durch verwaltungswissenschaftliche Methoden begründete Existenz zu illustrieren. Die während der Identifikation der Konzepte getroffenen Entwurfsentscheidungen, fachlichen Beschreibungen und Namen wurden als Eigenschaftsausprägungen der Instanzen und Beziehungen zwischen Instanzen definiert.

Im Ergebnis liegen Instanzen der Ontologie vor, die über ihren Annotationskontext einen direkten Bezug zum Text der Rechtsvorschrift haben. Bei Verwendung des Werkzeugs OntoMat kann diese Beziehung über die entsprechende Funktion im OntoMat hergestellt werden, da ihre Implementierung werkzeugspezifisch ist und auf einer weitgehend undokumentierten Nutzung des XPointer-Standards beruht. Bei Verwendung des RDFa-Standards ist der Bezug der Annotation zum Kontext unmittelbar gegeben. Durch das bisher erreichte Ergebnis erfüllt sich die Anforderung Nr. 1, die an den zu entwickelnden Ansatz gestellt wird (vgl. Seite 7), weil die Annotation von Rechtsvorschriften unmittelbar auf deren Texte anwendbar ist, ohne dass von der Existenz eines initialen Modells ausgegangen werden muss. Die Annotationen können weiterverarbeitet werden und haben gleichzeitig einen direkten Bezug zu den annotierten Textpassagen, so dass die Verfolgbarkeit möglich ist und eine Ausgangsbasis für die modellgetriebene Softwareentwicklung geschaffen werden kann. Anforderung Nr. 2 (vgl. Seite 7) ist dadurch ebenfalls erfüllt.

Dem erzielten Zwischenergebnis fehlen noch Elemente und Informationen, die für die Beschreibung der Besonderheiten des Verwaltungshandelns (z.B. im Kontext der Erbringung von Bürgerdiensten) notwendig sind. Sie konnten durch die Betrachtung der Rechtsvorschriften noch nicht identifiziert werden und bleiben damit Aufgabe der nachfolgenden Schritte (Abschnitt 4).

[340] Sollten Ausnahmen von dieser Regel notwendig sein, lassen sich diese durch Kombination von property- und content-Eigenschaften konstruieren.

4 Modell des Verwaltungshandelns und seine Anwendung

Durch die Annotation der Rechtsvorschriften liegen instanziierte Konzepte der Ontologie in Form von OWL-Individuen vor, die einen direkten Bezug zum Text der jeweiligen Vorschrift haben. Diese intanziierten Konzepte sind für die Transformationsprozesse der MDA nicht direkt nutzbar, da sie nicht als Modell vorliegen. Sie werden deshalb durch Anwendung des ODM (vgl. Abschnitt 2.3) verfolgbar in ein initiales MOF-konformes Modell überführt. In diesem Modell können die so gewonnenen, initialen Modellelemente weiter verfeinert und ergänzt werden. Dabei können auch die Besonderheiten des Verwaltungshandelns im Modell berücksichtigt werden, die sich nicht direkt aus dem Text von Rechtsvorschriften ableiten lassen. Weil die Überführung der OWL-Individuen in das Modell verfolgbar ist, haben dessen Modellelemente einerseits einen direkten Bezug zu Texten von Rechtsvorschriften und können anderseits für die Modellierung der Besonderheiten des Verwaltungshandelns genutzt werden.

Das Modell formalisiert Wissen aus dem Anwendungsbereich, ohne dass explizite Bezüge zu Anforderungen an Prozesse oder unterstützende Informationstechnik existieren. Deshalb ordnen sich die Modellierungsaktivitäten und das Modell nicht in die klassischen Phasen und korrespondierenden Modelltypen der MDA ein. Im Rahmen dieser Arbeit wird deshalb ein neuer Modelltyp – das Pre-Requirements Model (PRM, dt. Modell im Vorfeld der Anforderungsspezifikation) – für die MDA definiert, um diesen konzeptionellen Unterschied zu berücksichtigen.

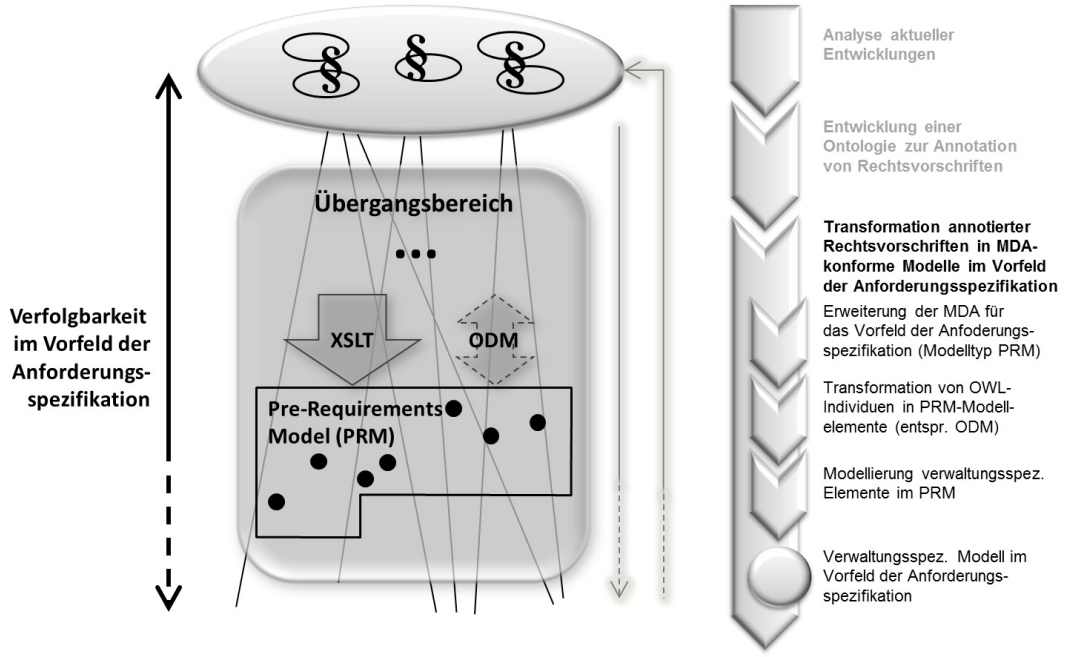


Abbildung 44 – Modell des Verwaltungshandelns im Vorfeld der Anforderungsspezifikation

Der Abschnitt führt zunächst das PRM ein (4.1), um dann die Erzeugung initialer Modelle durch Transformation der annotierten Rechtsvorschriften vorzustellen (4.2). Abschließend wird gezeigt, wie die bisher noch fehlenden Informationen und Besonderheiten des Verwaltungshandelns in die weitere Modellierungsarbeit mit den PRM am Beispiel einfließen können (4.3).

4.1 Pre-Requirements Model (PRM)

Die Darstellung der MDA beschränkt sich auf die Softwareentwicklung im engeren Sinne, d.h. die Modellierung von der Anforderungsspezifikation bis zum lauffähigen System. Das

Vorfeld der Anforderungsspezifikation wird nicht beachtet, was für die Verfolgbarkeit von Anforderungen in diesem Bereich allerdings unerlässlich ist. Deshalb wird im Rahmen dieser Arbeit eine MDA-Erweiterung eingeführt und auf das Vorfeld der Anforderungsspezifikation von E-Government-Anwendungen angewendet.

4.1.1 MDA für das Vorfeld der Anforderungsspezifikation

Die MDA sieht den informationstechnikunabhängigen Standpunkt dafür vor, das System aus einer Perspektive darzustellen, die die Umgebung des Systems und die Anforderungen an das System zeigt. Alle weitere Standpunkte konkretisieren die Sicht auf das System schrittweise hinsichtlich umsetzungsorientierter und technischer Aspekte. Für Pre-Requirements Traceability ist es nicht ausreichend, sich auf die dokumentierten Anforderungen zu beschränken (vgl. Abschnitt 2.1). Vielmehr muss die Verbindung zum Vorfeld der Anforderungsspezifikation hergestellt werden, was allein durch die klassische Perspektive des informationstechnikunabhängigen Standpunktes nicht möglich ist. Um deutlich zu machen, dass hierfür eine Erweiterung der Perspektive (und damit auch der MDA) stattfinden muss, wird für diese Arbeit ein neuer Standpunkt vorgeschlagen, der das Vorfeld der Anforderungsspezifikation repräsentiert. Er stellt ein System aus einer Perspektive dar, die Ursprünge von Anforderungen, deren Zusammenhänge und Aspekte zeigt, aus denen Anforderungen an das System abgeleitet werden können, die selbst aber (noch) keine Anforderungen darstellen. Ein solcher Standpunkt würde beispielsweise dazu dienen, die Ursprünge von Anforderungen in Rechtsvorschriften zu identifizieren und hält Antworten auf die Frage nach dem Warum von Anforderungen bereit. Dieser neu eingeführte Standpunkt wird entsprechend als *Standpunkt im Vorfeld der Anforderungsspezifikation (Pre-Requirements Viewpoint)* bezeichnet.

Korrespondierend zu den Standpunkten werden von der MDA Sichten in Form von Modelltypen definiert.[341] Konkrete Modelle eines bestimmten Typs beschreiben immer ein System und ggf. dessen Umgebung aus der Perspektive, die durch den Standpunkt vorgegeben wird. Die Modelle einer Anforderungsspezifikation stellen das System und seine Umgebung von einem informationstechnikunabhängigen Standpunkt aus vereinfacht dar und sind deshalb vom Typ CIM. Für den im Rahmen dieser Arbeit eingeführten Standpunkt im Vorfeld der Anforderungsspezifikation wird ebenfalls ein eigener Modelltyp definiert, das *Pre-Requirements Specification Model (PRM)*, als „Modell im Vorfeld der Anforderungen". Konkrete Modelle dieses Typs umfassen Elemente und deren Beziehungen, die ein System von Aspekten vereinfacht beschreiben, die im Vorfeld einer Anforderungsspezifikation relevant sind und Ursprünge von Anforderungen darstellen können.

Um den neu eingeführten Modelltyp nutzen zu können, müssen Syntax und Semantik im Sinne der MDA definiert werden, was durch die Einführung eines PRM-Metamodells erreicht wird. In Abbildung 45 (auf Seite 120) werden die Erweiterungen des MDA-Ansatzes (dargestellt durch graue Modellelemente) illustriert.

Das Ziel ist, MDA-konforme Transformationsprozesse zu verwenden, um die Elemente von Modellen im Vorfeld der Anforderungsspezifikation in Elemente von informationstechnik-unabhängigen Modellen zu transformieren. In der gängigen Darstellung der MDA konzentriert sich ihr Einsatz auf die Transformation von plattformunabhängigen Modellen in plattformspezifische Modelle. Die Transformation von informationstechnikunabhängigen Modellen untereinander, ihre Transformation in plattformunabhängige Modelle sowie deren Transformation untereinander werden in der Darstellung der MDA in der Regel

[341] Die Begriffsdefinitionen in der MDA unterscheiden hierbei nicht in dem sonst üblichen Maße zwischen Typen und konkreten Modellen. So benutzt die OMG in [OMG, 2003b] Modelltypen beispielsweise zur Illustration, obwohl es sich hierbei um konkrete Modelle handeln müsste. Im Rahmen der vorliegenden Arbeit wird dem nicht gefolgt, sondern deutlich zwischen den Typen und konkreten Modellen unterschieden.

vernachlässigt.[342] Allerdings ist grundsätzlich die Transformation von beliebigen MDA-konformen Modellen ineinander möglich, somit auch die Transformation des neu eingeführten PRM in CIM.[343]

[342] Es findet sich in [Kleppe et al., 2003] der Hinweis darauf, dass eine Transformation von CIM in PIM nicht möglich sei (vgl. [Kleppe et al., 2003], S. 19). Als Argument wird angeführt, dass die Entscheidung, welcher Teil des CIM durch ein Softwaresystem unterstützt wird und welcher manuell zu bearbeiten ist, nicht automatisch getroffen werden kann. Allerdings ist es für eine MDA-konforme Transformation nicht notwendige Voraussetzung, dass sie vollautomatisch durchgeführt wird (vgl. [OMG, 2003b], S. 3-7 und S. 4-2). Zum anderen sieht Kleppe auch für die Durchführung einer PIM/PSM-Transformation immer die Möglichkeit vor, Entscheidungen, die während der Transformation getroffen werden müssen, durch Parameter zu steuern, die zuvor manuell festgelegt wurden (vgl. [Kleppe et al., 2003], S. 100). Dieser Ansatz der parametergesteuerten Transformation lässt sich auch auf eine CIM-PIM-Transformation übertragen. Dort könnte beispielsweise über Parameter gesteuert werden, wie welcher Teil des CIM durch das Softwaresystem unterstützt wird und damit in das PIM einfließen soll. Die Aussage, dass CIM-PIM-Transformationen grundsätzlich nicht möglich sind, kann auch vor dem Hintergrund der laufenden Forschungen zur CIM-Transformation (vgl. u. a. [Zhang et al., 2005], [Rodríguez et al., 2007a], [Rodríguez et al., 2007b], [Rodríguez et al., 2008], [Kherraf et al. 2008], [Suarez et al., 2008], [Koch et al., 2008], [Kardoš&Drozdová, 2010]) nicht aufrechterhalten werden.

[343] Vgl. [OMG, 2003b], S. 5-2 ff.

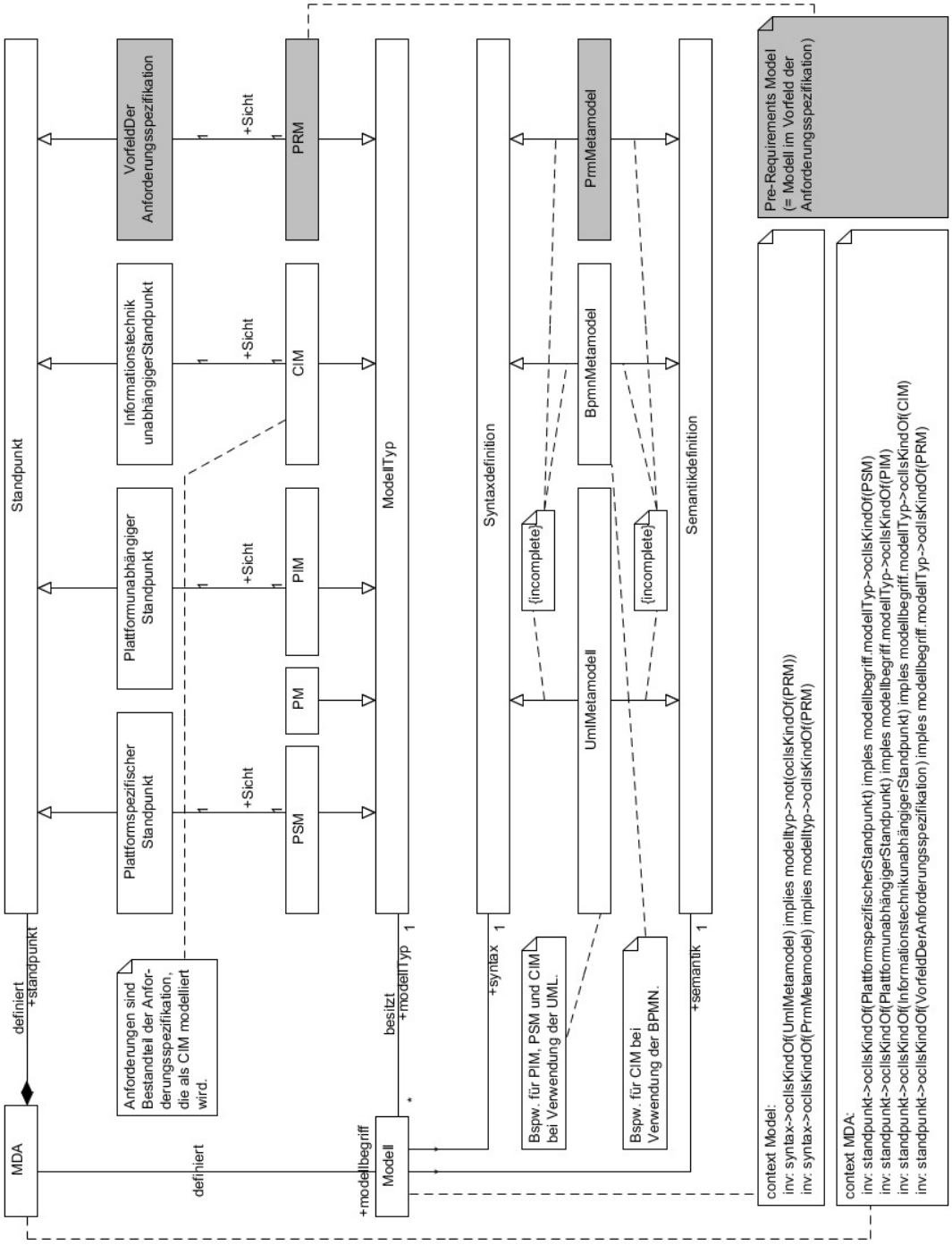


Abbildung 45 – Erweiterung der MDA um Aspekte des Vorfelds der Anforderungsspezifikation

4.1.2 Verwaltungshandeln als Pre-Requirements Model

Mit dem PRM steht ein Modelltyp bereit, der für die Modellierung des Vorfeldes der Anforderungsspezifikation verwendet werden kann. Das Vorfeld der Anforderungsspezifikation, wie es in dieser Arbeit für E-Government-Anwendungssysteme und die damit verbundene Verfolgbarkeit zu Aspekten in Rechtsvorschriften eingegrenzt wird, ist nur ein mögliches und entsprechend spezielles Einsatzgebiet. Deshalb wird im Folgenden die Bezeichnung *PRM des Verwaltungshandelns* benutzt, um es von anderen Einsatzgebieten abzugrenzen.

Ein PRM des Verwaltungshandelns beschreibt ein System von Aspekten vereinfacht, die sich aus dem Text von Rechtsvorschriften ergeben und im Vorfeld einer Anforderungs-

spezifikation relevant sind. Dazu umfasst es Elemente und deren Beziehungen, die ihren Ursprung in Annotationen von Rechtsvorschriften haben. Da dieses Modell Elemente aus den annotierten Rechtvorschriften enthält, die auf einer Ontologie des Verwaltungshandeln basieren, werden für das PRM die Syntax- und Semantik-Definition des Ontology Definition Metamodel (ODM) verwendet.

Die sich dadurch ergebenden Erweiterungen sind in der nachfolgenden Abbildung als Spezialisierungen des PRM und des PRM-Metamodells dargestellt. Vor dem Hintergrund, dass die Ontologie des Verwaltungshandelns auch unter Beachtung relevanter Aspekte aus dem E-Government-Kontext erstellt wurde, stellt ein PRM des Verwaltungshandelns, basierend auf dem ODM, das Vorfeld der Anforderungsspezifikation von E-Governement-Anwendungen dar.

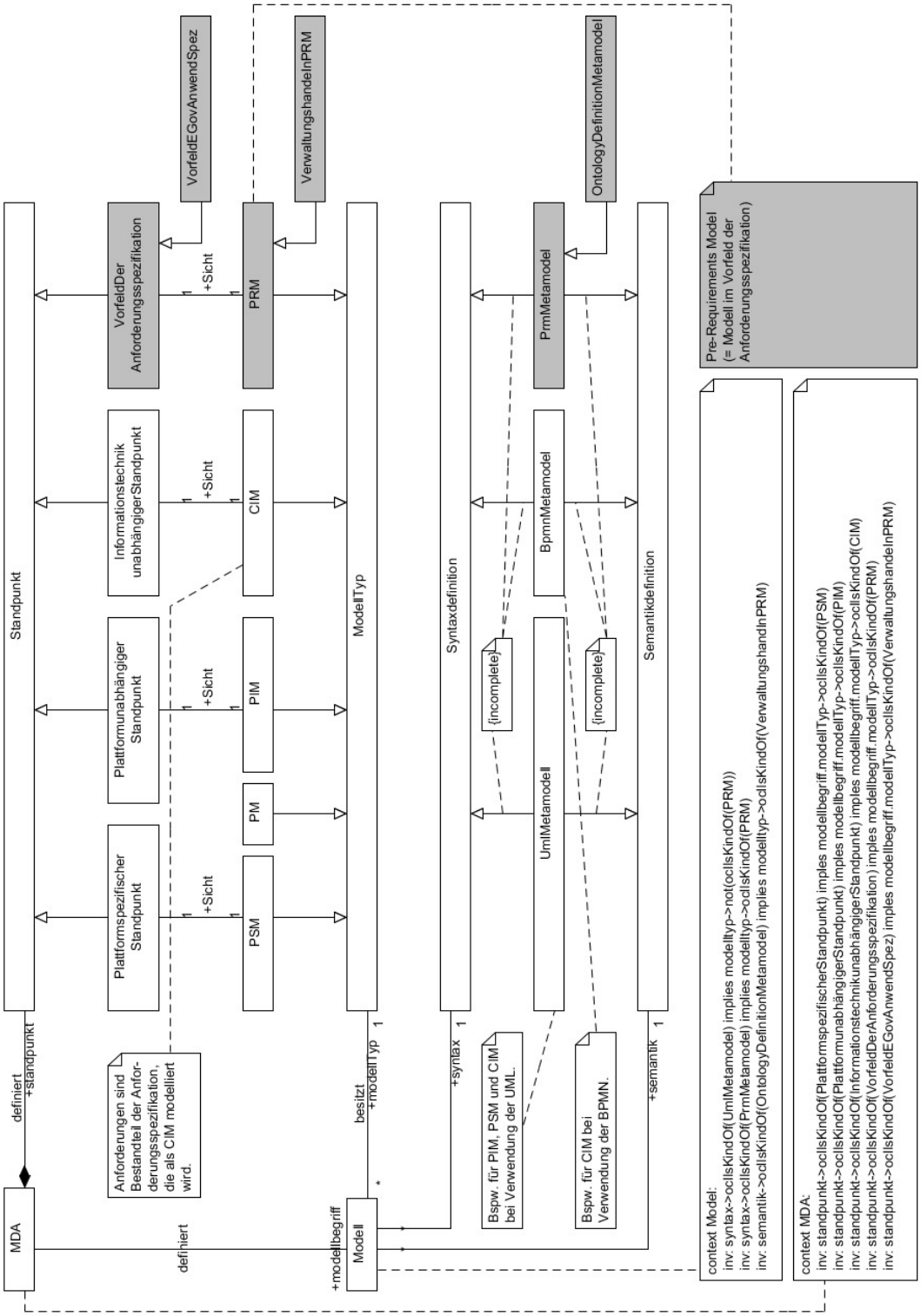


Abbildung 46 – Spezialisierung des Vorfelds der Anforderungsspezifikation für das Verwaltungshandeln

4.1.3 Pre-Requirements Model basierend auf ODM UML-Profil

Um das ODM in Werkzeugen für die Modellierung eines PRM des Verwaltungshandelns nutzen zu können, müssen die vom ODM definierten UML-Profile für RDF und OWL[344] hinterlegt werden. Die Elemente dieser UML-Profile (z.B. Stereotypen «owlClas»,

[344] Vgl. [OMG ODM,2009], S. 131 ff.

«objectProperty») werden, basierend auf dem zugrunde liegenden Meta-Metamodell (MOF), modelliert. Einen Ausschnitt aus diesen Modellen zeigt die nachfolgende Abbildung 47 für die RDF- und OWL-Konzepte. Zusätzlich werden für die Anwendung als Pre-Requirements Model und dessen weitere Verfeinerung auch die Standardstereotypen (z.B. «refine» und «trace») der UML benötigt. Sie sind ebenfalls im Werkzeug als Profil hinterlegt.[345] Im Anhang C sind die für den Ansatz dieser Arbeit modellierten Profildiagramme vollständig dargestellt.

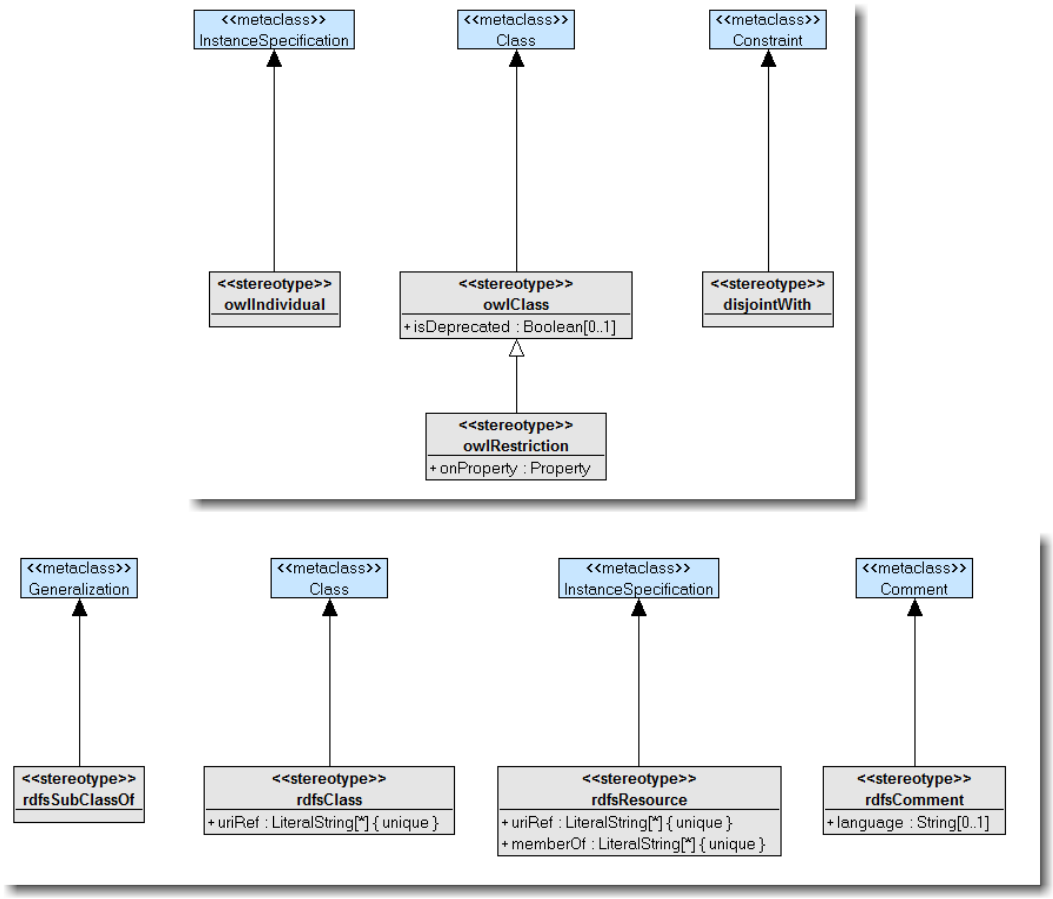


Abbildung 47 – Auszüge aus dem verwendeten OWL- und RDF-Profil für UML

Anschließend wird das Profil über eine automatische Generierung, die Teil des EMF-Projektes von Eclipse ist, in eine Form gebracht, die es möglich macht, von anderen auf Eclipse basierenden Werkzeugen verwendet zu werden. Technisch entspricht diese Form einem Eclipse-Plugin, so dass mehrere JAR-Dateien durch den Generator erzeugt und im Plugins-Verzeichnis der Installation bereitgestellt werden (Abbildung 48). Im Ergebnis kann das ODM in Form eines UML-Profils für die Modellierung in Eclipse verwendet werden.

[345] Vgl. [OMG UML, 2009a], S. 695 ff. (Für die Umsetzung mit dem Werkzeug TOPCASED war, abweichend vom UML-Standard, die zusätzliche Definition von «refine» und «trace» als Stereotyp für uml::Dependency notwendig, weil uml::Abstraction in der verwendeten TOPCASED-Version keine grafische Darstellung im Diagramm hatte.)

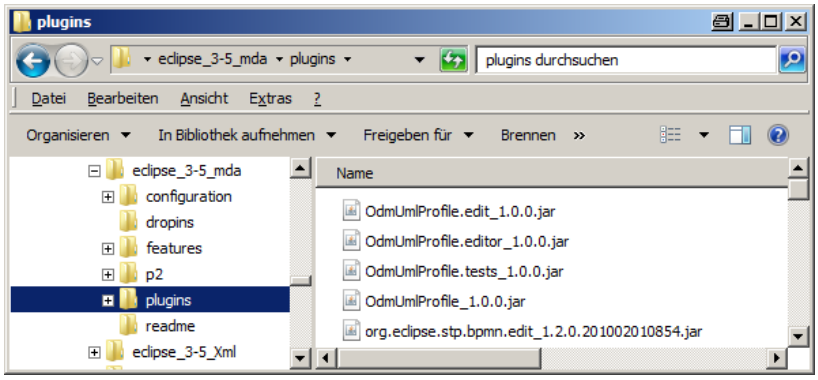


Abbildung 48 – JAR-Dateien des ODM UML-Profils installiert im Plugins-Verzeichnis von Eclipse

4.2 Erzeugung initialer Modelle

Zwischen den durch Annotationen instanziierten Konzepten in Form von OWL-Individuen einerseits und der Repräsentation der Konzepte in Form von OWL-Klassen und OWL-Properties andererseits existiert eine formale Beziehung. Diese Beziehung definiert die Zugehörigkeit eines Individuums zu einer OWL-Klasse oder einem OWL-Property. In den annotierten Rechtsvorschriften sind keine Informationen über diese OWL-Klassen und -Properties enthalten. Um ein Modell, ausgehend von den Annotationen, konstruieren zu können, müssen sowohl die OWL-Ontologie mit Klassen und Properties als auch die OWL-Individuen in ein gemeinsames Modell transformiert werden. In diesem Abschnitt wird daher in einem ersten Schritt die OWL-Ontology des Verwaltungshandelns in ein UML-Modells, basierend auf dem ODM, transformiert. Dadurch können in einem zweiten Schritt die OWL-Individuen, die durch Annotation der Rechtsvorschriftentexte erzeugt wurden, ebenfalls ein UML-Modell transformiert werden.

4.2.1 Transformation der OWL-Ontologie

Die mit Protegé erstellte OWL-Ontologie des Verwaltungshandelns liegt als XML-Datei im OWL-Format vor. Das zuvor, basierend auf dem ODM-Standard, erstellte UML-Profil für RDF und OWL wird verwendet, um die Ontologie in ein MOF-konformes Modell auf M1-Ebene zu überführen.[346] Dazu wird eine Transformation definiert, die aus dem OWL-Format der Ontologie des Verwaltungshandelns (*.owl-Datei) ein UML-Modell im XMI-Format (*.uml-Datei) erzeugt, das auf dem zuvor im Werkzeug hinterlegten UML-Profil für RDF und OWL basiert. Bei beiden Formaten handelt es sich um spezielle XML-Formate, so dass eine Transformation mittels *XSL Transformation (XSLT)* grundsätzlich die geeignete und angemessene technische Umsetzungsform zur Überbrückung der Werkzeuglücke ist.[347] Dazu werden die in den nachfolgenden Tabellen durch das ODM UML-Profil weitgehend vorgegebenen Abbildungsregeln als Transformationsschritte in XSLT implementiert.[348]

[346] Für den Ansatz dieser Arbeit ist die Verwendung eines UML-Profils ausreichend, wodurch auch die Integration in das eingesetzte Modellierungswerkzeug TOPCASED einfacher möglich ist, als durch die alternative Implementierung des OWL-Metamodells innerhalb von Eclipse.

[347] Da im Ausgangsformat keine Informationen über die grafische Anordnung der enthaltenen Konzepte vorhanden sind, ist diese Information auch nicht in dem durch die Transformation erzeugten XMI enthalten. Durch eine Werkzeugunterstützung (z.B. durch TOPCASED oder die UML2 Tools des Eclipse MDT-Projektes) können die Modellelemente nach der Transformation quasi zufällig im Diagramm positioniert werden.

[348] Die Tabelle illustriert nur die Abbildungsregeln des ODM UML-Profils, die auf die Ontologie des Verwaltungshandelns tatsächlich angewendet werden. Regeln, für die es in der Ontologie kein Ausgangselement gibt, werden nicht dargestellt.

ID	XSLT_OWL-ONTOLOGY-PRM-01
Name	UmlModell
Beschreibung	Die Ontologie (owl:Ontology) wird in ein uml:Model transformiert, wobei der Name des Modells durch den Namen der Ontologie (bzw. den Dateinamen, wenn der Name der Ontologie leer ist) und das Suffix „-Modell" gebildet wird. Es wird mit der Transformation der Ontologie in ein UML-Package fortgefahren (XSLT_OWL-ONTOLOGY-PRM-02).

Tabelle 14 – Definition der Abbildungsregel "UmlModell"(XSLT_OWL-ONTOLOGY-PRM-01)

ID	XSLT_OWL-ONTOLOGY-PRM-02
Name	UmlPackage
Beschreibung	Die Ontologie (owl:Ontology) wird in ein uml:Package transformiert, wobei der Name des Packages durch den Namen der Ontologie (bzw. den Dateinamen, wenn der Name der Ontologie leer ist) und das Suffix „-Package" gebildet wird. Dem Package wird der Stereotyp «owlOntology» aus dem OWL UML-Profil zugewiesen. Es wird mit dem Anlegen des Stereotyps für dieses Elements fortgefahren (XSLT_OWL-ONTOLOGY-PRM-03).

Tabelle 15 – Definition der Abbildungsregel "UmlPackage"(XSLT_OWL-ONTOLOGY-PRM-02)

ID	XSLT_OWL-ONTOLOGY-PRM-03
Name	UmlPackageStereotyp
Beschreibung	Dem Package wird der Stereotyp «owlOntology» aus dem OWL UML-Profil zugewiesen.

Tabelle 16 – Definition der Abbildungsregel "UmlPackageStereotyp"(XSLT_OWL-ONTOLOGY-PRM-03)

ID	XSLT_OWL-ONTOLOGY-PRM-04
Name	UmlKlasse
Beschreibung	Jedes Element vom Typ owl:Class wird in eine UML-Klasse (uml:Class) transformiert, wobei der Klassenname aus der rdfs:label-Eigenschaft (wenn diese nicht gesetzt aus der rdf:about-Eigenschaft) übernommen wird. Jedes Element vom Typ rdfs:comment innerhalb einer owl:Class wird in eine ecore:eAnnotation der uml:Class als Dokumentation für die Verwendung im Werkzeug TOPCASED transformiert. Es wird mit dem Anlegen der Attribute der Klasse (XSLT_OWL-ONTOLOGY-PRM-06) und ihres Stereotyps fortgefahren (XSLT_OWL-ONTOLOGY-PRM-05).

Tabelle 17 – Definition der Abbildungsregel "UmlKlasse"(XSLT_OWL-ONTOLOGY-PRM-04)

ID	XSLT_OWL-ONTOLOGY-PRM-05
Name	UmlKlasseStereotyp
Beschreibung	Der UML-Klasse wird der Stereotyp «owlClass» aus dem OWL UML-Profil zugewiesen.

Tabelle 18 – Definition der Abbildungsregel "UmlKlasseStereotyp"(XSLT_OWL-ONTOLOGY-PRM-05)

ID	XSLT_OWL-ONTOLOGY-PRM-06
Name	UmlAttribut
Beschreibung	Jedes Element vom Typ owl:DatatypeProperty, dessen rdfs:domain mit der rdf:about-Eigenschaft der zuvor transformierten owl:Class übereinstimmt, wird als Attribut der erzeugten uml:Class angelegt, wobei der Name des Attributs aus der rdfs:label-Eigenschaft (wenn diese nicht gesetzt aus der rdf:about-Eigenschaft) des DatatypeProperty übernommen wird. Die Beschreibung des Attributes wird aus dem rdfs:comment und sein Datentyp aus dem rdfs:range übernommen. Anschließend wird mit dem Anlegen des Stereotyp fortgefahren (XSLT_OWL-ONTOLOGY-PRM-07).

Tabelle 19 – Definition der Abbildungsregel "UmlAttribut"(XSLT_OWL-ONTOLOGY-PRM-06)

ID	XSLT_OWL-ONTOLOGY-PRM-07
Name	UmlAttributStereotyp
Beschreibung	Dem UML-Attribut wird der Stereotyp «datatypeProperty» aus dem OWL UML-Profil zugewiesen.

Tabelle 20 – Definition der Abbildungsregel "UmlAttributStereotyp"(XSLT_OWL-ONTOLOGY-PRM-07)

ID	XSLT_OWL-ONTOLOGY-PRM-08
Name	UmlGeneralisierung
Beschreibung	Jedes Element vom Typ rdfs:subClassOf, das innerhalb einer owl:Class eine Ressource bezeichnet, wird in eine Generalisierungsbeziehung der durch die owl:Class erzeugten uml:Class und der aus der Ressource erzeugten uml:Class transformiert.

Tabelle 21 – Definition der Abbildungsregel "UmlGeneralisierung"(XSLT_OWL-ONTOLOGY-PRM-08)

ID	XSLT_OWL-ONTOLOGY-PRM-09
Name	UmlConstraint
Beschreibung	Jedes Element vom Typ owl:disjointWith innerhalb einer owl:Class wird in einen UML-Constraint transformiert und mit den beteiligten Klassen in Beziehung gesetzt. Anschließend wird mit dem Anlegen des Stereotyp fortgefahren (XSLT_OWL-ONTOLOGY-PRM-10).

Tabelle 22 – Definition der Abbildungsregel "UmlConstraint"(XSLT_OWL-ONTOLOGY-PRM-09)

ID	XSLT_OWL-ONTOLOGY-PRM-10
Name	UmlConstraintStereotyp
Beschreibung	Dem UML-Constraint wird der Stereotyp «disjointWith» zugewiesen.

Tabelle 23 – Definition der Abbildungsregel "UmlConstraintStereotyp"(XSLT_OWL-ONTOLOGY-PRM-10)

ID	XSLT_OWL-ONTOLOGY-PRM-11
Name	UmlAssociation
Beschreibung	Jedes Element vom Typ owl:Restriction innerhalb einer owl:Class und jedes owl:ObjectPropterty wird eine in eine uml:Association transformiert und mit den beteiligten Klassen in Beziehung gesetzt. Anschließend wird mit dem Anlegen des Stereotyps fortgefahren (XSLT_OWL-ONTOLOGY-PRM-12).

Tabelle 24 – Definition der Abbildungsregel "UmlAssociation"(XSLT_OWL-ONTOLOGY-PRM-11)

ID	XSLT_OWL-ONTOLOGY-PRM-12
Name	UmlAssociationStereotyp
Beschreibung	Der UML-Assoziation wird der Stereotyp «objectProperty» zugewiesen.

Tabelle 25 – Definition der Abbildungsregel "UmlAssociationStereotyp"(XSLT_OWL-ONTOLOGY-PRM-12)

Die Abbildungsregeln werden technisch mit einem XSL-Stylesheet implementiert. Generell werden XSL-Stylesheets mittels eines XSLT-Prozessors verarbeitet, der eine Quelldatei im XML-Format in eine Zieldatei transformiert. Im vorliegenden Ansatz wird die Ontologie des Verwaltungshandelns im XML-basierten OWL-Format als Quelldatei verwendet und in ein UML-Modell im XML-basierten XMI-Format transformiert. Die folgenden zwei Beispiele verdeutlichen die Funktionsweise der Abbildungsregeln XSLT_OWL-ONTOLOGY-PRM-04 und XSLT_OWL-ONTOLOGY-PRM-08 anhand von Auszügen der Ausgangsdatei, der zu erzeugenden Zieldatei und dem verwendeten XSL-Stylesheet.

Das Ausgangsmodellelement `Einzelfallfestsetzung` in Listing 17 (Zeilen 3 bis 9) ist die Darstellung des Konzepts als Klasse im OWL-Format. Es wird durch die Abbildungsregel XSLT_OWL-ONTOLOGY-PRM-04 in eine UML-Klasse überführt, die auf dem OWL UML-Profil basiert. Das Listing 18 zeigt in den Zeilen 4 bis 12 die entsprechende Klasse und in Zeile 16 die Anwendung des Stereotyps aus dem Profil innerhalb des UML-Modells, das Ziel der Transformation ist. Mit der Abbildungsregel XSLT_OWL-ONTOLOGY-PRM-08 wird die Subklassenbildung in OWL in eine Generalisierungsbeziehung zwischen Klassen im UML-Modell überführt. Die Zeile 6 in Listing 17 zeigt, dass die OWL-Klasse `Einzelfallfestsetzung` eine Subklasse der Klasse `Rechtshandlung` ist. Im Ergebnis der Transformation wird eine Generalisierung (`generalization`) in Zeilen 10 bis 11 (Listing 18) angelegt.

```
(1)   <rdf:RDF (...)>
(2)     <!-- ...-->
(3)     <owl:Class rdf:about="#Einzelfallfestsetzung">
(4)       <rdfs:label rdf:datatype="&xsd;string"
(5)         >Einzelfallfestsetzung</rdfs:label>
(6)       <rdfs:subClassOf rdf:resource="#RechtsHandlung"/>
(7)       <!-- ... -->
(8)       <rdfs:comment rdf:datatype="&xsd;string">Ein Schritt der
      Rechtsanwendung, in dessen Rahmen aus einer abstrakten Rechtfolge durch
      Konkretisierung vor dem Hintergrund des vorliegenden Einzelfalls eine
      konkrete Ma&#223;nahme abgeleitet und festgesetzt wird.</rdfs:comment>
(9)     </owl:Class>
(10)
(11)  </rdf:RDF>
```

Listing 17 – Beispiel des Konzepts „Rechtshandlung" als RDF-Subclass

```
(1)   <xmi:XMI (...)>
(2)    <uml:Model (...)>
(3)     <!-- ... -->
(4)     <packagedElement xmi:id="einzelfallfestsetzung_class" xmi:type="uml:Class"
(5)       name="Einzelfallfestsetzung">
(6)       <eAnnotations xmi:id="einzelfallfestsetzung_class_annotation"
(7)           source="http://www.topcased.org/documentation">
```

```
(8)           <details xmi:id="einzelfallfestsetzung_class_annotation_details"
                 key="documentation" value="Ein Schritt der Rechtsanwendung, in
      dessen Rahmen aus einer abstrakten Rechtfolge durch Konkretisierung vor dem
      Hintergrund des vorliegenden Einzelfalls eine konkrete Maßnahme abgeleitet
      und festgesetzt wird."/>
(9)       </eAnnotations>
(10)      <generalization xmi:id="einzelfallfestsetzung_class_is-
      a_rechtshandlung_class"
(11)          general="rechtshandlung_class"/>
(12)    </packagedElement>
(13)    <!-- ... -->
(14)   </uml:Model>
(15)   <!-- ... -->
(16)   <OwlUmlProfile:owlClass xmi:id="d3e1023_owlClass"
          base_Class="einzelfallfestsetzung_class"/>
(17)   <!-- ... -->
(18)  </xmi:XMI>
```

Listing 18 – Beispiel des Konzepts„Einzelfallfestsetzung" als OWL-Klasse im OWL UML-Profil (XMI-Format)

Das für die Transformation verwendete XSL-Stylesheet ist in Listing 19 in Auszügen dargestellt. Beim Entwurf des Stylesheets wurde darauf geachtet, dass eine Abbildungsregel einem XSL-Template innerhalb des Stylesheets entspricht, was aufgrund der Umsetzungskomplexität der Transformationen nicht durchgängig sinnvoll gelingen konnte. Ein Kommentar (eingeleitet durch <!–– und abschlossen mit ––>, vgl. Abschnitt 1.6) vor jedem Template stellt den Bezug zur entsprechenden Abbildungsregel her. Der vollständige Code des XSLT-Stylesheets befindet sich im Anhang D.1.

Die Abbildungsregel XSLT_OWL-ONTOLOGY-PRM-04 wird durch die Templates `packagedElementClasses` (Zeilen 64 bis 76) und `packagedElementClass` (Zeilen 11 bis 75) umgesetzt. Innerhalb des Templates `packagedElementClasses` werden über eine Schleife alle OWL-Klassen ermittelt (Zeile 72 bis 75). Für jede einzelne Klasse wird das Template `packagedElementClass` aufgerufen (Zeile 74). Innerhalb des Templates `packagedElementClass` wird dann die UML-Klasse angelegt (Zeile 16 bis 56), ihre Dokumentation aus dem `rdfs:comment` der OWL-Klasse übernommen, durch Aufruf des Templates `ownedAttributes` in Zeile 49 das Anlegen der Attribute und durch Aufruf des Templates `generalizations` (in den Zeilen 52 bis 54) das Anlegen der Generalisierung angestoßen. Das Template `generalizations` ist die Umsetzung der Abbildungsregel XSLT_OWL-ONTOLOGY-PRM-08 (Zeilen 83 bis 108). Das Template legt in einer Schleife für jede `rdfs:subClassOf` (Zeile 85 bis 106) eine UML-Generalisierung an (Zeilen 91 bis 94), für die es die ID der generalisierten Klasse vermerkt (Zeile 93).

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet ... >
(3)    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
(4)    <!-- ... -->
(5)    <!--
(6)     Abbildungsregel
(7)     ID:  XSLT_OWL-ONTOLOGY-PRM-04a
(8)     Name: UmlKlasseSingle
(9)     Hinweis:Bestandteil der Abbildungsregel XSLT_OWL-ONTOLOGY-PRM-04
(10)   -->
(11)   <xsl:template name="packagedElementClass">
(12)    <xsl:variable name="name" select="govobj:name(.)"/>
(13)    <xsl:variable name="id" select="concat(govobj:id(.), '_class')"/>
(14)    <xsl:variable name="ref" select="govobj:ref(.)"/>
(15)
(16)    <xsl:element name="packagedElement">
(17)     <xsl:attribute name="xmi:id" select="$id"/>
(18)     <xsl:attribute name="xmi:type" select="'uml:Class'"/>
(19)     <xsl:attribute name="name" select="$name"/>
(20)
(21)     <xsl:element name="eAnnotations">
(22)      <xsl:attribute name="xmi:id" select="concat($id,'_annotation')"/>
(23)      <xsl:attribute name="source"
      select="'http://www.topcased.org/documentation'"/>
(24)      <xsl:element name="details">
(25)       <xsl:attribute name="xmi:id"
      select="concat($id,'_annotation_details')"/>
```

```
(26)       <xsl:attribute name="key" select="'documentation'"/>
(27)       <xsl:attribute name="value">
(28)        <xsl:for-each select="rdfs:comment">
(29)         <xsl:value-of select="."/>
(30)         <xsl:if test="position()!=last()">
(31)          <xsl:text> </xsl:text>
(32)         </xsl:if>
(33)        </xsl:for-each>
(34)       </xsl:attribute>
(35)      </xsl:element>
(36)     </xsl:element>
(37)
(38)     <!--  Tracing -->
(39)     <xsl:call-template name="traceTransformation">
(40)      <xsl:with-param name="transName" select="$transName"/>
(41)      <xsl:with-param name="transRule" select="'UmlClass'"/>
(42)      <xsl:with-param name="srcName" select="'srcOwlClass'"/>
(43)      <xsl:with-param name="dstName" select="'dstUmlClass'"/>
(44)      <xsl:with-param name="srcElement" select="$ref"/>
(45)      <xsl:with-param name="dstElement" select="$id"/>
(46)     </xsl:call-template>
(47)
(48)     <!--  Attribute der Klasse anlegen -->
(49)     <xsl:call-template name="ownedAttributes"/>
(50)
(51)     <!-- Generalisierung -->
(52)     <xsl:call-template name="generalizations">
(53)      <xsl:with-param name="id-cls" select="$id"/>
(54)     </xsl:call-template>
(55)
(56)    </xsl:element>
(57)   </xsl:template>
(58)
(59)   <!--
(60)    Abbildungsregel
(61)    ID:  XSLT_OWL-ONTOLOGY-PRM-04
(62)    Name: UmlKlasse
(63)   -->
(64)   <xsl:template name="packagedElementClasses">
(65)    <!--  Kommentar zur Orientierung in der XMI-Datei -->
(66)    <xsl:text>&#x0A;&#x9;</xsl:text>
(67)    <xsl:comment>
(68)     <xsl:text>owl:Classes</xsl:text>
(69)    </xsl:comment>
(70)    <xsl:text>&#x0A;&#x9;</xsl:text>
(71)
(72)    <xsl:for-each select="owl:Class">
(73)     <!-- Klasse anlegen -->
(74)     <xsl:call-template name="packagedElementClass"/>
(75)    </xsl:for-each>
(76)   </xsl:template>
(77)
(78)   <!--
(79)    Abbildungsregel
(80)    ID:  XSLT_OWL-ONTOLOGY-PRM-08
(81)    Name: UmlGeneralisierung
(82)    -->
(83)   <xsl:template name="generalizations">
(84)    <xsl:param name="id-cls"/>
(85)    <xsl:for-each select="rdfs:subClassOf[@rdf:resource]">
(86)     <xsl:if test="@rdf:resource">
(87)      <xsl:variable name="ref" select="lower-case(replace(substring-
      after(@rdf:resource, '#'),':','_'))"/>
(88)      <xsl:variable name="gen" select="concat(govobj:replace-umlaute($ref),
      '_class')"/>
(89)      <xsl:variable name="id" select="concat($id-cls, '_is-a_', $gen)"/>
(90)
(91)      <xsl:element name="generalization">
(92)       <xsl:attribute name="xmi:id" select="$id"/>
(93)       <xsl:attribute name="general" select="$gen"/>
(94)      </xsl:element>
(95)
(96)      <!--  Tracing -->
(97)      <xsl:call-template name="traceTransformation">
(98)       <xsl:with-param name="transName" select="$transName"/>
```

```
(99)       <xsl:with-param name="transRule" select="'UmlGeneralisierung'"/>
(100)      <xsl:with-param name="srcName" select="'srcOwlDisjointWith'"/>
(101)      <xsl:with-param name="dstName" select="'dstUmlConstraint'"/>
(102)      <xsl:with-param name="srcElement" select="$ref"/>
(103)      <xsl:with-param name="dstElement" select="$id"/>
(104)     </xsl:call-template>
(105)    </xsl:if>
(106)   </xsl:for-each>
(107)
(108)  </xsl:template>
(109)
(110)   <!-- ... -->
(111)
(112)  </xsl:template>
(113) </xsl:stylesheet>
```

Listing 19 – Auszug aus dem XSLT-Stylesheet zur Transformation der OWL-Ontologie in ein ODM UML-Profil der Ontologie

Im Ergebnis der XSL-Transformation liegt die in Abschnitt 3.1.4 definierte Ontologie inhaltlich unverändert als ein MOF-konformes UML-Modell, basierend auf dem ODM UML-Profil im hier eingesetzten Modellierungswerkzeug TOPCASED, vor. Einen Auszug aus dem entsprechenden UML-Modell zeigt die Abbildung 49 (auf Seite 131).

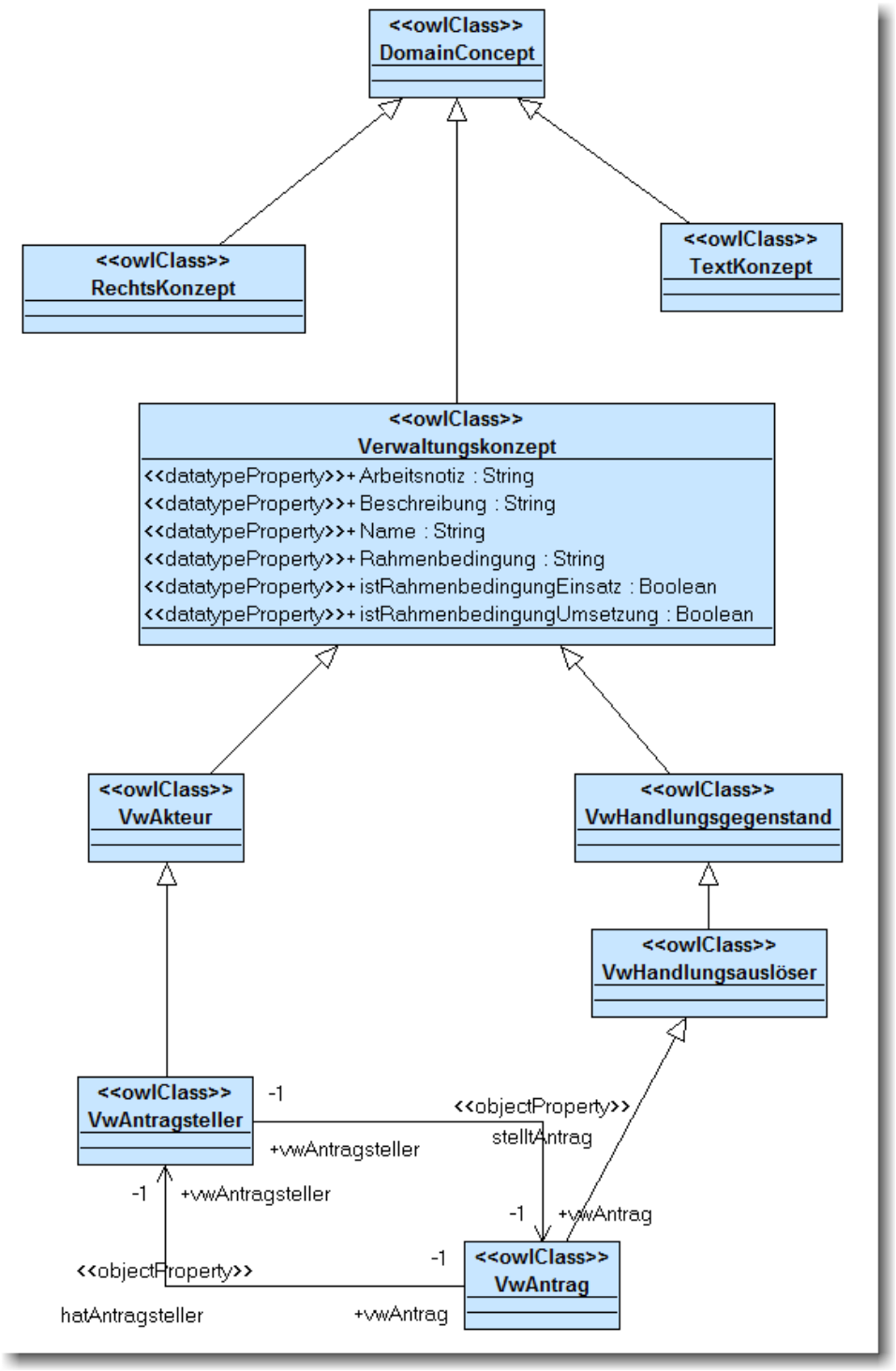


Abbildung 49 – Allgemeine Konzepte der Ontologie in ihrer Umsetzung im Modell

4.2.2 Transformation annotierter Rechtsvorschriften

Die Transformation der annotierten Rechtsvorschriften in ein UML-Modell erfolgt in vergleichbarer Weise wie die zuvor beschriebene Transformation der OWL-Ontologie. Dazu wird das im Vorangegangenen erstellte UML-Profil für RDF und OWL verwendet, um ein auf ODM basierendes Modell zu erzeugen.

Dazu muss auf die in den annotierten Rechtsvorschriften enthaltenen Individuen zugegriffen werden, was abhängig von der für die Annotation gewählten Werkzeugunterstützung ist. Für

das Werkzeug OntoMat muss eine andere Variante gewählt werden als für die Verarbeitung von RDFa-Annotationen.

Beim Einsatz des Werkzeugs OntoMat sind die OWL-Inidividuen innerhalb eines Kommentars am Ende des HTML-Dokuments kompakt in das Dokument eingebettet.[349] Weil der OntoMat nicht für die Verarbeitung von XHTML, sondern nur für HTML ausgelegt ist, können sich im praktischen Einsatz Inkompatibilitäten bei Sonderzeichen (z.B. den deutschen Umlauten in Verbindung mit wechselnden Zeichencodierungen) ergeben. Weiterhin ergeben sich zwei mögliche Varianten, wie die OWL-Annotationen und RDF-Tripel zur Weiterverarbeitung extrahiert werden können. Für HTML-Dokumente ist ein Zwischenschritt notwendig, für den im Rahmen dieser Arbeit in Java ein Konverter entwickelt wurde (Listing im Anhang B.1) Für die Verarbeitung von XHTML-Dokumenten wurde alternativ ein XSL-Stylesheet entwickelt, das die Extraktion und eine ggf. notwendige Konvertierung der Umlaute vornimmt (Listing in Anhang B.2). Ergebnis beider Schritte ist, dass die Annotationen aus dem XHTML bzw. HTML extrahiert wurden, so dass sie als gültige XML-Datei vorliegen.

Bei Einsatz von RDFa sind die Annotationen als RDF-Tripel innerhalb der Elemente des XHTML-Sprachumfangs enthalten. Da es sich bei XHTML-Dokumenten um gültige XML-Dokumente handelt, kann auf die RDF-Tripel direkt mittels einer XSLT zugegriffen werden. Dieses Prinzip nutzt beispielsweise der GRDDL-Standard, mit dem das XHTML-Dokument mit einem Stylesheet verbunden wird, um es bei Bedarf (bei Nutzung eines so genannten *GRDDL aware agent*) in ein RDF-Dokument zu überführen. Dieser Zwischenschritt zur Extraktion der RDF-Tripel in ein eigenständiges Dokument ist für den Ansatz dieser Arbeit nicht erforderlich. Da es hier das Ziel ist, ein PRM im XMI-Format zu erzeugen, bringt ein RDF-Dokument keine Vorteile gegenüber der direkten Transformation des XHTML-Dokuments. Auf den Einsatz von GRDDL für die Implementierung der Transformation wird deshalb verzichtet.

Für die Transformation der Annotationen sind die entsprechenden Regeln durch das ODM weitgehend vorgegeben. Sie werden in den nachfolgenden Tabellen zusammengefasst.

ID	XSLT_OWL-ANNOTATION-PRM-01
Name	UmlModell
Beschreibung	Die Ontologie (owl:Ontology) wird in ein uml:Model transformiert, wobei der Name des Modells durch den Namen der Ontologie und das Suffix „-Modell" gebildet wird. Es wird mit der Transformation der Ontologie in ein UML-Package fortgefahren (XSLT_OWL-ANNOTATION-PRM-02).

Tabelle 26 – Definition der Abbildungsregel "UmlModell"(XSLT_OWL-ANNOTATION-PRM-01)

ID	XSLT_OWL-ANNOTATION-PRM-02
Name	UmlPackage
Beschreibung	Die Ontologie (owl:Ontology) wird in ein uml:Package transformiert, wobei der Name des Packages durch den Namen der Ontologie und das Suffix „-Package" gebildet wird. Dem Package wird der Stereotyp «owlOntology» aus dem OWL UML-Profil zugewiesen. Es wird mit dem Anlegen des Stereotyps für dieses Element fortgefahren (XSLT_OWL-ANNOTATION-PRM-03).

Tabelle 27 – Definition der Abbildungsregel "UmlPackage"(XSLT_OWL ANNOTATION-PRM-02)

[349] Alternativ können mit OntoMat-Annotizer die Individuen auch direkt in der Ontologie gespeichert werden. Dadurch geht aber der Zusammenhang zum annotierten Text verloren, da dieser nur innerhalb des Kommentars in der HTML-Datei gespeichert wird. Aus diesem Grund wird für den Ansatz dieser Arbeit die (X)HTML-Datei als Ausgangspunkt für die weitere Verarbeitung gewählt.

ID	XSLT_OWL-ANNOTATION-PRM-03
Name	UmlPackageStereotyp
Beschreibung	Dem Package wird der Stereotyp «owlOntology» aus dem OWL UML-Profil zugewiesen.

Tabelle 28 – Definition der Abbildungsregel "UmlPackageStereotyp"(XSLT_OWL-ANNOTATION-PRM-03)

ID	XSLT_OWL-ANNOTATION_PRM-04
Name	UmlInstanceSpecification
Beschreibung	Jede Instanz einer Klasse der Ontologie wird in eine uml:InstanceSpecification transformiert, wobei ihr Name aus der rdf:label-Eingenschaft übernommen (und seine Schreibweise nach CamelCase/PascalCase transformiert) wird. Die Attributsausprägungen der Klasse werden übernommen. Es wird mit dem Anlegen des Stereotyps für dieses Element fortgefahren (XSLT_OWL-ANNOTATION-PRM-05).

Tabelle 29 – Definition der Abbildungsregel "UmlInstanceSpecification"(XSLT_OWL-ANNOTATION_PRM-04)

ID	XSLT_OWL-ANNOTATION_PRM-05
Name	UmlInstanceSpecificationStereotyp
Beschreibung	Der UML-Instance Specification wird der Stereotyp «owlIndividual» aus dem OWL UML-Profil zugewiesen.

Tabelle 30 – Definition der Abbildungsregel "UmlInstanceSpecificationStereotyp"(XSLT_OWL-ANNOTATION_PRM-05)

ID	XSLT_OWL-ANNOTATION_PRM-06
Name	UmlInstanceSpecificationLink
Beschreibung	Jede Instanz eines Object Property der Ontologie wird in einen UML-Instance Specification Link als Instanz einer Assoziation transformiert, wobei 1. der Name aus dem Typ des Elements und den rdf:label-Elementen der beteiligten Konzepte übernommen, 2. die Beziehung zwischen den beteiligten Konzepten angelegt und 3. aus der Instanz der Typ der Beziehung abgeleitet wird.

Tabelle 31 – Definition der Abbildungsregel "UmlInstanceSpecificationLink"(XSLT_OWL-ANNOTATION-PRM-07)

Die Anwendung der Abbildungsregeln XSLT_OWL-ANNOTATION_PRM-04, XSLT_OWL-ANNOTATION_PRM-05 und XSLT_OWL-ANNOTATION_PRM-06 illustrieren Listing 20 und Listing 21 für Annotationen, die mit dem OntoMat erzeugt wurden. Ausgangspunkt des Beispiels ist die Instanz eines Antragsstellers mit der Bezeichnung „Bewohner“ (Listing 20, Zeilen 3 bis 14). Sie wird durch Anwendung der Abbildungsregel in eine Instance Specification der UML transformiert (Listing 21, Zeilen 4 bis 6), die Instanz der Klasse `VwAntragsteller` ist. Weiterhin wird der Instanz durch Anwendung der Abbildungsregel XSLT_OWL-ANNOTATION_PRM-05 der entsprechende Stereotyp aus dem OWL UML-Profil (Zeile 25) zugewiesen. Für die Anwendung der Abbildungsregel XSLT_OWL-ANNOTATION_PRM-06 ist eine Instanz des Object Property `stelltAntrag`, das die beiden OWL-Individuen `Bewohner` und `AntragBewohnerparken` in Beziehung setzt (Listing 20, Zeilen 5 bis 12), der Ausgangspunkt. Das Object Property `stelltAntrag` wird in eine Instanz einer UML-Assoziation

vwant15_stell12_vwant8 (Listing 21, Zeile 9) transformiert. Die Enden dieser Instanz werden mit den zuvor angelegten Instanzen „Bewohner“ (d4e138_individual) und „AntragBewohnerparken“ (d4e142_individual) verbunden (Listing 21, Zeile 12 und 18).

```
(1)   <rdf:RDF (...)>
(2)     <!-- ... -->
(3)     <govobjects-ontology:VwAntragsteller
(4)       rdf:about="http://govobjects.thomasoff.de/bewohnerparken#Bewohner">
(5)       <govobjects-ontology:stelltAntrag>
(6)         <govobjects-ontology:VwAntrag
(7)           rdf:about="http://govobjects.thomasoff.de/bewohnerparken#Antrag">
(8)           <rdfs:label>Antrag</rdfs:label>
(9)           <govobjects-ontology:Name
      rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
(10)            AntragBewohnerparkausweis</govobjects-ontology:Name>
(11)        </govobjects-ontology:VwAntrag>
(12)      </govobjects-ontology:stelltAntrag>
(13)      <rdfs:label>Bewohner</rdfs:label>
(14)    </govobjects-ontology:VwAntragsteller>
(15)    <!-- ... -->
(16)  </rdf:RDF>
```

Listing 20 – „Bewohner“ als Beispiel für ein OWL-Individual der Klasse „Anwohner“

```
(1)   <xmi:XMI (...)>
(2)    <uml:Model (...)>
(3)     <!-- ... -->
(4)     <packagedElement xmi:id="d4e138_individual"
      xmi:type="uml:InstanceSpecification"
          name="Bewohner">
(5)       <classifier xmi:type="uml:Class"
            href="../Model/prm_ontology-govobj-
      ontologie.uml#vwantragsteller_class"/>
(6)     </packagedElement>
(7)     <!-- ... -->
(8)     <packagedElement xmi:type="uml:InstanceSpecification"
      xmi:id="d4e140_instance-specification-link"
      name="Bewohner(vwantragsteller)_stelltAntrag_Antrag(vwantrag)">
(9)       <classifier xmi:type="uml:Association" href="../Model/prm_ontology-
      govobj-ontologie.uml#vwant15_stell12_vwant8"/>
(10)      <slot xmi:id="d4e140_slot1">
(11)        <definingFeature xmi:type="uml:Property"
      href="../Model/prm_ontology-govobj-
      ontologie.uml#vwant15_stell12_vwant8_vwant15_ownedEnd-1"/>
(12)        <value xmi:type="uml:InstanceValue" xmi:id="d4e140_slot1_value"
      instance="d4e138_individual">
(13)          <type xmi:type="uml:Class" href="../Model/prm_ontology-govobj-
      ontologie.uml#vwantragsteller_class"/>
(14)        </value>
(15)      </slot>
(16)      <slot xmi:id="d4e140_slot2">
(17)        <definingFeature xmi:type="uml:Property"
      href="../Model/prm_ontology-govobj-
      ontologie.uml#vwant15_stell12_vwant8_vwant8_ownedEnd-2"/>
(18)        <value xmi:type="uml:InstanceValue" xmi:id="d4e140_slot2_value"
      instance="d4e142_individual">
(19)          <type xmi:type="uml:Class" href="../Model/prm_ontology-govobj-
      ontologie.uml#vwantrag_class"/>
(20)        </value>
(21)      </slot>
(22)    </packagedElement>
(23)    <!-- ... -->
(24)   </uml:Model>
(25)   <OwlUmlProfile:owlIndividual xmi:id="d1e28"  base_InstanceSpecification="
      d4e138_individual "/>
(26)   <!-- ... -->
(27)  </xmi:XMI>
```

Listing 21 – „Bewohner“ als uml::InstanceSpecification der UML-Klasse „Antragsteller“

Die Implementierung dieser Abbildungsregeln erfolgt als XSL-Stylesheet. Es ist in Listing 22 in Auszügen dargestellt, sein vollständiger Code befindet sich im Anhang D.2. Der Auszug zeigt die Umsetzung der Abbildungsregeln XSLT_OWL-ANNOTATION_PRM-04 und

XSLT_OWL-ANNOTATION_PRM-05. Die Abbildungsregel XSLT_OWL-ANNOTATION_PRM-04 wird durch die Templates `instanceSpecifications` und `instanceSpecification` umgesetzt. Das Template `instanceSpecifications` ermittelt in einer Schleife alle Annotationen und ruft für jede Annotation das Template `instanceSpecification` auf. In diesem Template wird die entsprechende UML-Instance Specification angelegt und die Referenz auf ihre UML-Klasse (aus dem UML-Modell der Ontologie) gesetzt. Sind in der Annotation Ausprägungen von Datatype Properties vorhanden, werden diese als Werteausprägungen von Attributen zur UML-Instance Specification generiert.

Die Abbildungsregel XSLT_OWL-ANNOTATION_PRM-05 ist durch das Template owlIndividual umgesetzt. Dieses Template bearbeitet in einer Schleife alle Annotationen. Es weist jeder Annotation den Stereotyp owlIndividual aus dem OWL UML-Profil (OwlUmlProfile) zu, indem es ein entsprechendes Modellelement anlegt und den Attributwert von base_InstanceSpecification auf die ID des in Abbildungsregel XSLT_OWL-ANNOTATION_PRM-04 erzeugten UML-Instance Specification setzt.

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0" (...) >
(3)
(4)     <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
(5)
(6)     <!-- ... -->
(7)     <!--
(8)       Abbildungsregel
(9)       ID:    XSLT_OWL-ANNOTATION_PRM-04a
(10)      Name:  UmlInstanceSpecificationSingle
(11)      Hinweis:Bestandteil der Abbildungsregel XSLT_OWL-ANNOTATION_PRM-04
(12)     -->
(13)    <xsl:template name="instanceSpecification">
(14)      <xsl:variable name="instance" select="govobj:id(.)"/>
(15)      <xsl:variable name="instance-ref" select="govobj:ref(.)"/>
(16)      <xsl:variable name="class" select="local-name()"/>
(17)      <xsl:variable name="instance-id" select="concat(govobj:generate-id(.),
      '_individual')"/>
(18)
(19)      <!--  Tracing -->
(20)      <xsl:call-template name="traceTransformation">
(21)        <xsl:with-param name="transName" select="$transName"/>
(22)        <xsl:with-param name="transRule" select="'UmlInstanceSpecification'"/>
(23)        <xsl:with-param name="srcName" select="'srcOwlIndividual'"/>
(24)        <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecification'"/>
(25)        <xsl:with-param name="srcElement" select="$instance-ref"/>
(26)        <xsl:with-param name="dstElement" select="$instance-id"/>
(27)      </xsl:call-template>
(28)
(29)    <!-- ... -->
(30)
(31)      <!--  uml:InstanceSpecification -->
(32)      <xsl:element name="packagedElement">
(33)        <xsl:attribute name="xmi:id" select="$instance-id"/>
(34)        <xsl:attribute name="xmi:type" select="'uml:InstanceSpecification'"/>
(35)        <xsl:attribute name="name" select="govobj:pascalize($instance)"/>
(36)
(37)        <!-- Referenz auf Classifier -->
(38)        <xsl:element name="classifier">
(39)          <xsl:attribute name="xmi:type" select="'uml:Class'"/>
(40)          <xsl:attribute name="href" select="concat($ontology, '#', lower-
      case($class), '_class')"/>
(41)        </xsl:element>
(42)
(43)        <!-- uml:Slots -->
(44)        <xsl:for-each select="govobjects-ontology:*[@rdf:datatype][string-
      length(normalize-space(.))>0]">
(45)          <!-- Variablen -->
(46)          <xsl:variable name="slot-id" select="concat(replace(generate-
      id(),':','_'),'_slot',position())"/>
(47)          <xsl:variable name="property" select="lower-case(local-name())"/>
(48)          <xsl:variable name="property-id" select="concat($instance-ref, '/',
      $property)"/>
```

```
(49)          <xsl:variable name="value" select="normalize-space(.)"/>
(50)
(51)          <!--  Tracing -->
(52)          <xsl:call-template name="traceTransformation">
(53)            <xsl:with-param name="transName" select="$transName"/>
(54)            <xsl:with-param name="transRule"
      select="'UmlInstanceSpecificationSlot'"/>
(55)            <xsl:with-param name="srcName" select="'srcOwlIndividual'"/>
(56)            <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecificationSlot'"/>
(57)            <xsl:with-param name="srcElement" select="$property-id"/>
(58)            <xsl:with-param name="dstElement" select="$slot-id"/>
(59)            <xsl:with-param name="check-sum" select="round(string-
      length($value)*1000 div functx:word-count($value))"/>
(60)          </xsl:call-template>
(61)
(62)          <!-- Element -->
(63)          <xsl:element name="slot">
(64)            <xsl:attribute name="xmi:id" select="$slot-id"/>
(65)            <xsl:element name="definingFeature">
(66)              <xsl:attribute name="xmi:type"  select="'uml:Property'"/>
(67)              <xsl:attribute name="href" select="concat($ontology, '#',
      $property, '_attribute')"/>
(68)            </xsl:element>
(69)            <xsl:element name="value">
(70)              <xsl:attribute name="xmi:type">
(71)                <!--  Weitere Datentypen analog zu diesem Muster -->
(72)                <xsl:choose>
(73)                  <xsl:when test="ends-with(@rdf:datatype, 'string')">
(74)                    <xsl:text>uml:LiteralString</xsl:text>
(75)                  </xsl:when>
(76)                  <xsl:when test="ends-with(@rdf:datatype, 'boolean')">
(77)                    <xsl:text>uml:LiteralBoolean</xsl:text>
(78)                  </xsl:when>
(79)                  <xsl:otherwise>
(80)                    <xsl:text>uml:LiteralString</xsl:text>
(81)                  </xsl:otherwise>
(82)                </xsl:choose>
(83)              </xsl:attribute>
(84)              <xsl:attribute name="xmi:id" select="concat($slot-
      id,'_value')"/>
(85)              <xsl:attribute name="value" select="$value"/>
(86)            </xsl:element>
(87)          </xsl:element>
(88)        </xsl:for-each>
(89)      </xsl:element>
(90)    </xsl:template>
(91)    <!-- ... -->
(92)
(93)    </xsl:template>
(94)
(95)    <!--
(96)      Abbildungsregel
(97)      ID:    XSLT_OWL-ANNOTATION_PRM-04
(98)      Name:  UmlInstanceSpecification
(99)     -->
(100)   <xsl:template name="instanceSpecifications">
(101)
(102)     <!-- Kommentar zur Orientierung im erzeugten XMI -->
(103)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(104)     <xsl:comment>InstanceSpecifications</xsl:comment>
(105)     <xsl:text>&#9;&#9;</xsl:text>
(106)
(107)     <!-- Jeweils eine Instanz als Element anlegen -->
(108)     <xsl:for-each select="//govobjects-ontology:*[@rdf:about]">
(109)       <xsl:call-template name="instanceSpecification"/>
(110)     </xsl:for-each>
(111)
(112)   </xsl:template>
(113)
(114)   <!-- ... -->
(115)
(116)   <!--
(117)     Abbildungsregel
(118)     ID:    XSLT_OWL-ANNOTATION_PRM-05
```

```
(119)     Name:  UmlInstanceSpecificationStereotyp
(120)    -->
(121)   <xsl:template name="owlIndividual">
(122)     <xsl:for-each select="//govobjects-ontology:*[@rdf:about]">
(123)       <xsl:variable name="id" select="concat(replace(generate-id(),':','_'),
      '_individual')"/>
(124)       <xsl:variable name="ref" select="govobj:ref(.)"/>
(125)
(126)       <xsl:element name="OwlUmlProfile:owlIndividual">
(127)         <xsl:attribute name="xmi:id" select="concat(replace(generate-
      id(),':','_'), '_owl-individual')"/>
(128)         <xsl:attribute name="base_InstanceSpecification" select="$id"/>
(129)       </xsl:element>
(130)
(131)       <!--  Tracing -->
(132)       <xsl:call-template name="traceTransformation">
(133)         <xsl:with-param name="transName" select="$transName"/>
(134)         <xsl:with-param name="transRule" select="'UmlIndividualStereotyp'"/>
(135)         <xsl:with-param name="srcName" select="'srcOwlIndividual'"/>
(136)         <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecificationStereotype'"/>
(137)         <xsl:with-param name="srcElement" select="$ref"/>
(138)         <xsl:with-param name="dstElement" select="$id"/>
(139)       </xsl:call-template>
(140)     </xsl:for-each>
(141)   </xsl:template>
(142)
(143)   <!--
(144)     ODM-Stereotypen zuweisen
(145)    -->
(146)   <xsl:template name="ProfileRef">
(147)
(148)     <!-- Stereotyp owl:Ontology für das UML-Package -->
(149)     <xsl:call-template name="owlOntology"/>
(150)
(151)     <!-- Stereotyp owl:Individual für die UML-Instanzen-->
(152)     <xsl:call-template name="owlIndividual"/>
(153)
(154)   </xsl:template>
(155)   <!-- ... -->
(156) </xsl:stylesheet>
```

Listing 22 – Auszug aus dem XSLT-Stylesheet zur Transformation der OntoMat-Annotationen

Die Transformation der RDFa-Annotationen setzt direkt an dem annotierten XHTML-Dokument an, unterscheidet sich aber nur geringfügig von der Implementierung der zuvor dargestellten Stylesheets für den OntoMat. Diese Unterschiede liegen im Wesentlichen in angepassten XPath-Ausdrücken, um auf die relevanten Annotationselemente und ihre Eigenschaftenausprägungen zuzugreifen. Das Listing 23 hebt exemplarisch die Unterschiede in den Abbildungsregeln XSLT_OWL-ANNOTATION_PRM-04 in den Zeilen 10 bis 13, 58 und in der Regel XSLT_OWL-ANNOTATION_PRM-05 in Zeile 72 hervor. Das vollständige Listing ist in Anhang D.3 enthalten.

```
(1)     <!-- ... -->
(2)
(3)     <!--
(4)       Abbildungsregel
(5)       ID:    XSLT_OWL-ANNOTATION_PRM-04a
(6)       Name:  UmlInstanceSpecificationSingle
(7)       Hinweis:Bestandteil der Abbildungsregel XSLT_OWL-ANNOTATION_PRM-04
(8)      -->
(9)     <xsl:template name="instanceSpecification">
(10)      <xsl:variable name="instance" select="substring-after(@about, '#')"/>
(11)      <xsl:variable name="instance-ref" select="govobj:generate-id(.)"/>
(12)      <xsl:variable name="instance-name" select="normalize-space(.)"/>
(13)      <xsl:variable name="class" select="substring-after(@typeof, ':')"/>
(14)      <xsl:variable name="instance-id" select="concat(govobj:generate-id(.),
      '_individual')"/>
(15)
(16)    <!-- ... -->
(17)
(18)      <!-- Kommentar zur Orientierung im erzeugten XMI -->
(19)      <xsl:text>&#x0A;</xsl:text>
```

```
(20)      <xsl:comment>
(21)        <xsl:value-of select="$instance"/>
(22)        <xsl:text> (</xsl:text>
(23)        <xsl:value-of select="$class"/>
(24)        <xsl:text>)</xsl:text>
(25)      </xsl:comment>
(26)      <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(27)
(28)      <!--  uml:InstanceSpecification -->
(29)      <xsl:element name="packagedElement">
(30)        <xsl:attribute name="xmi:id" select="$instance-id"/>
(31)        <xsl:attribute name="xmi:type" select="'uml:InstanceSpecification'"/>
(32)        <xsl:attribute name="name" select="govobj:pascalize-on-
      space($instance-name, ' ')"/>
(33)
(34)        <!-- Referenz auf Classifier -->
(35)        <xsl:element name="classifier">
(36)          <xsl:attribute name="xmi:type" select="'uml:Class'"/>
(37)          <xsl:attribute name="href" select="concat($ontology, '#', lower-
      case($class), '_class')"/>
(38)        </xsl:element>
(39)
(40)        <!-- uml:Slots -->
(41)        <!-- ... -->
(42)      </xsl:element>
(43)    </xsl:template>
(44)
(45)    <!--
(46)      Abbildungsregel
(47)      ID:    XSLT_OWL-ANNOTATION_PRM-04
(48)      Name:  UmlInstanceSpecification
(49)     -->
(50)    <xsl:template name="instanceSpecifications">
(51)
(52)      <!-- Kommentar zur Orientierung im erzeugten XMI -->
(53)      <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(54)      <xsl:comment>InstanceSpecifications</xsl:comment>
(55)      <xsl:text>&#9;&#9;</xsl:text>
(56)
(57)      <!-- Jeweils eine Instanz als Element anlegen -->
(58)      <xsl:for-each select="//html:*[@typeof]">
(59)        <xsl:call-template name="instanceSpecification"/>
(60)      </xsl:for-each>
(61)
(62)    </xsl:template>
(63)
(64)  <!-- ... -->
(65)
(66)    <!--
(67)      Abbildungsregel
(68)      ID:    XSLT_OWL-ANNOTATION_PRM-05
(69)      Name:  UmlInstanceSpecificationStereotyp
(70)     -->
(71)    <xsl:template name="owlIndividual">
(72)      <xsl:for-each select="//html:*[@typeof]">
(73)        <xsl:variable name="id" select="concat(replace(generate-id(),':','_'),
      '_individual')"/>
(74)        <xsl:variable name="ref" select="govobj:ref(.)"/>
(75)
(76)        <xsl:element name="OwlUmlProfile:owlIndividual">
(77)          <xsl:attribute name="xmi:id" select="concat(replace(generate-
      id(),':','_'), '_owl-individual')"/>
(78)          <xsl:attribute name="base_InstanceSpecification" select="$id"/>
(79)        </xsl:element>
(80)
(81)        <!-- ... -->
(82)
(83)      </xsl:for-each>
(84)    </xsl:template>
(85)
(86)  <!-- ... -->
```

Listing 23 – Auszug aus dem XSLT-Stylesheet zur Transformation der RDFa-Annotationen

Im Ergebnis der XSL-Transformation liegen die Annotationen aus dem Text der Rechtsvorschriften in Form von Instance Specifications und Instance Specification Links als MOF-konformes UML-Modell, basierend auf dem ODM UML-Profil, vor. Die Instance Specifications unterscheiden sich in der grafischen Darstellung voneinander durch ihren Namen und die Klasse, von der sie eine Instanz sind.[350] Dadurch kann, insbesondere bei großen Modellen, die Übersichtlichkeit leiden. Für das PRM werden deshalb in Anlehnung an den Ansatz von Coad et al.[351] unterschiedliche Farben für die Darstellung der Instance Specifications, abhängig von ihrer Klasse, verwendet. Handlungsgegenstände und ihre Spezialisierungen werden in grün, Handlungen und ihre Spezialisierungen in rot sowie Akteure und ihre Spezialisierungen in gelb modelliert. Das Ergebnis dieser initialen Modellerzeugung für das Bewohnerparken ist in Abbildung 50 dargestellt. Der Name der Instance Specifications entspricht der Textpassage, die während der Erzeugung der Annotation markiert wurde.[352] Deshalb können sich im Folgenden grammatisch irritierende Formulierungen ergeben, wenn auf den Namen einer Instanz im Fließtext Bezug genommen wird.

In der Abbildung 50 (auf Seite 140) ist exemplarisch das aus den Annotationen zum Bewohnerparken generierte Modell dargestellt. In diesem Modell sind beispielsweise die Instanzen `Bewohner` als Instanz der Klasse `VwAntragsteller` und `AntragBewohnerparken` als Instanz der Klasse `VwAntrag` erkennbar. Im eingeblendeten Properties-Fenster ist ersichtlich, dass die markierte Beziehung zwischen diesen Instanzen den Namen `Bewohner(VwAntrag-steller)_stelltAntrag_AntragBewohnerparken(VwAntrag)` hat und eine Instanz der Assoziation `stelltAntrag` ist.[353]

[350] Wurden gleiche Textpassagen mit unterschiedlichen Annotationen versehen, so unterscheiden sich diese lediglich durch ihre Klasse.

[351] [Coad et al., 1999]

[352] Es wurden während der Transformation aus technischen Gründen lediglich Umlaute ersetzt und Leerzeichen zugunsten der CamelCase-Schreibweise entfernt.

[353] Wegen eines bekannten Fehlers (Nr. 3602) im Werkzeug TOPCASED ist es nicht möglich, aus UML-Modellen, in denen InstanceSpecification Links verwendet werden, über die TOPCASED-Funktion automatisch UML-Diagramme zu generieren. Deshalb wurde im Rahmen dieser Arbeit eine weitere XSLT entwickelt, die das dargestellte Diagramm aus dem UML-Modell erzeugt. Anschließend kann mit dem Diagramm und dem Modell ohne Probleme in TOPCASED gearbeitet werden.

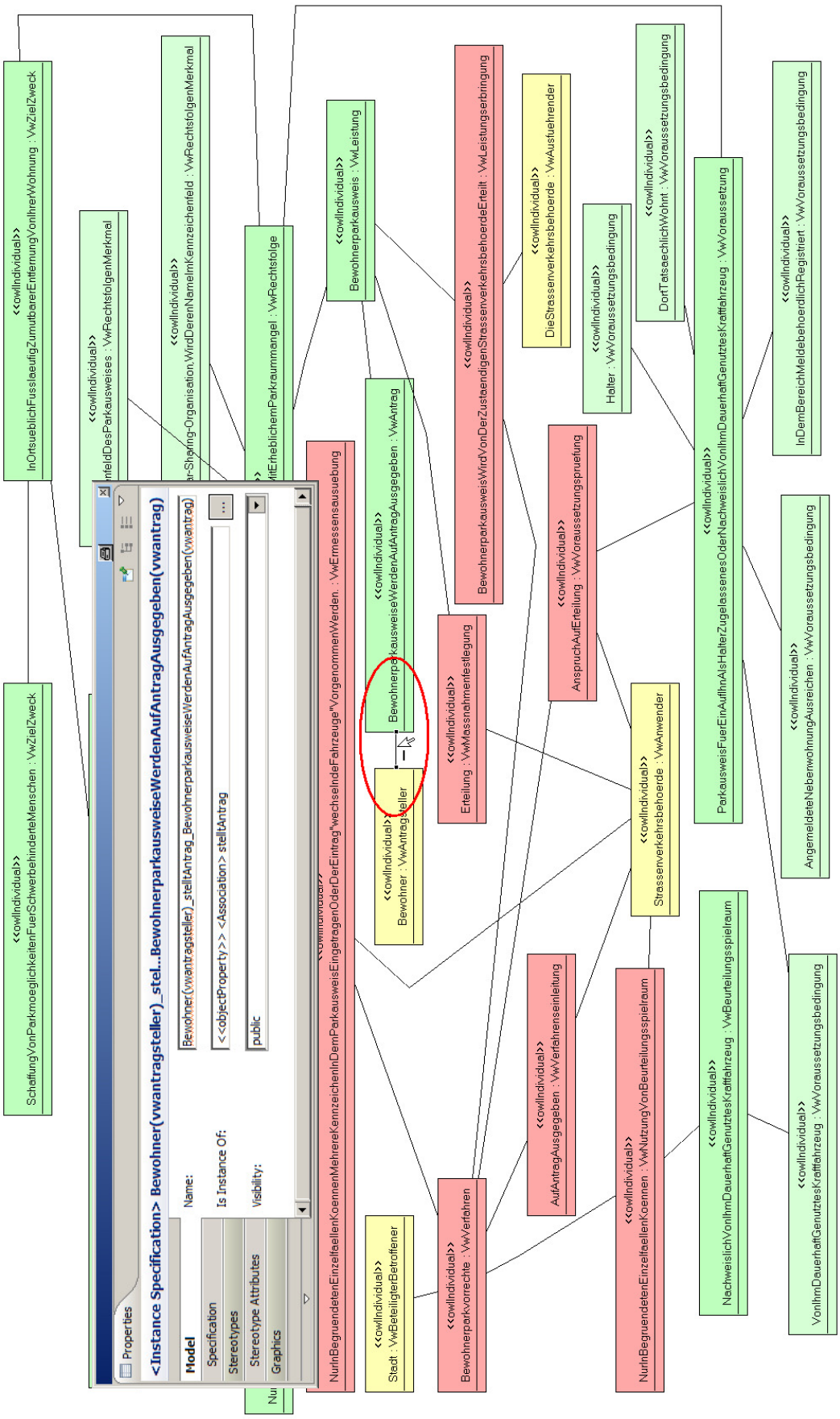


Abbildung 50 – Auszug aus dem Modell der Annotationen des Bewohnerparken-Beispiels

4.2.3 Verfolgbarkeit der initialen Modellerzeugung

Es gibt kein standardisiertes Verfahren, wie Verfolgbarkeitsinformationen für eine XSL-Transformation aufgezeichnet werden können. Es sind jedoch Ansätze dokumentiert, die für Debugging-Zwecke Message-Elemente verwenden, um allgemeine Informationen auf der Konsole auszugeben.[354] Dieses Verfahren wird im Rahmen dieser Arbeit abgewandelt, um Verfolgbarkeitsinformationen in einer ähnlichen Form aufzuzeichnen, wie sie auch bei QVT-Transformationen erzeugt werden. Der QVT-Standard sieht vor, dass zwei Arten von Informationen aufgezeichnet werden: die Verfolgbarkeitsklassen und die Verfolgbarkeitsinstanzen (vgl. Abschnitt 2.2). Im Rahmen dieser Arbeit wurden zwei XSL-Stylesheets entwickelt, mit denen diese Informationen aufgezeichnet werden können. Das Ergebnis, das unter Verwendung dieser Stylesheets erzielt wird, ist Abbildung 51 in innerhalb der Projektstruktur von Eclipse dargestellt. In Anlehnung an den QVT-Standard werden jeweils Dateipaare aus Verfolgbarkeitsinstanzen und Klassen erzeugt, deren Zusammenhang sich erkennbar aus ihrem Dateinamen ergibt. Beispielsweise bilden die `trace.rdfa2prm_annotation` und die `rdfa2prm_annotation.ecore` ein zusammengehöriges Dateipaar. Die Datei `trace.rdfa2prm_annotation` enthält die Verfolgbarkeitsinstanzen und die `rdfa2prm_annotation.ecore` die Verfolgbarkeitsklassen.

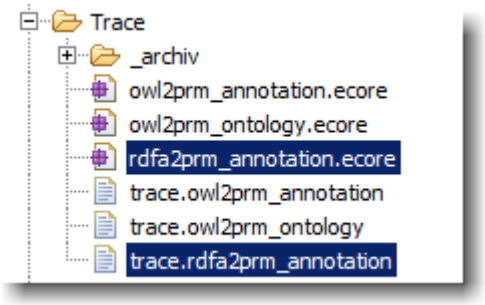


Abbildung 51 – Beispiel aufgezeichneter Verfolgbarkeitsinformationen der XSL-Transformationen

Das XSL-Stylesheet `xsltrace.xsl` wurde entwickelt, um Verfolgbarkeitsinstanzen aufzuzeichnen. Es ist in Listing 24 vollständig dargestellt. Dieses Stylesheet muss in das jeweilige XSL-Stylesheet, das zur eigentlichen Transformation der OWL-Ontologie und der OWL-/RDFa-Annotationen in UML-Modelle dient, als Erweiterung eingebunden und an den entsprechenden Stellen aufgerufen werden. Die bereits zuvor in Auszügen dargestellten Listings der XSL-Stylesheets enthalten die Erweiterung und den Aufruf (siehe beispielsweise Listing 22 in Zeilen 20 bis 27 oder 52 bis 60).

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <!--
(3)
(4)    Erzeugt Tracing in Anlehnung an QvtTraces auf der Konsole
(5)
(6)    Hinweis:
(7)    Damit der ausgegebene Text innerhalb der Message tatsächlich unkodiert
      erfolgt, reicht
(8)    die Eigenschaft disable-output-escaping="yes" nicht aus. Zusätzlich muss
      der Aufruf von
(9)    Saxon im Textmodus erfolgen, was durch den Parameter -m
      net.sf.saxon.event.TEXTEmitter
(10)   erreicht wird. (Siehe
      http://sourceforge.net/apps/mediawiki/saxon/index.php?title=Xslmessage)
(11)
(12)  -->
(13)  <xsl:stylesheet version="2.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(14)    xmlns:dbg="http://govobjects.thomasoff.de/xslt">
(15)    <xsl:output method="xml" indent="yes" name="trace"/>
(16)
(17)    <xsl:param name="tracingOn" select="true()"/>
```

[354] Vgl. u.a. [Mangano, 2002], S. 527ff.

```
(18)
(19)    <!--
(20)      Trancing initialisieren
(21)
(22)      Der Aufruf dieses Templates muss erfolgen, bevor die erste
      Abbildungsregel
(23)      aufgezeichnet werden soll.
(24)     -->
(25)    <xsl:template name="initTracing">
(26)      <xsl:param name="transName" select="'Unknown'"/>
(27)      <xsl:if test="$tracingOn">
(28)        <xsl:message>
(29)          <xsl:text disable-output-escaping="yes">&lt;?xml
      version=&quot;1.0&quot; encoding=&quot;UTF-8&quot;?&gt;</xsl:text>
(30)          <xsl:text>&#x0A;&lt;xmi:XMI xmi:version=&quot;2.0&quot;
      &#x0A;&#x9;xmlns:xmi=&quot;http://www.omg.org/XMI&quot;
      &#x0A;&#x9;</xsl:text>
(31)          <xsl:text>xmlns:xsi=&quot;http://www.w3.org/2001/XMLSchema-
      instance&quot; &#x0A;&#x9;</xsl:text>
(32)          <xsl:text>xmlns:</xsl:text>
(33)          <xsl:value-of select="$transName"/>
(34)          <xsl:text>=&quot;urn:</xsl:text>
(35)          <xsl:value-of select="$transName"/>
(36)          <xsl:text>.ecore&quot;&gt;&#x0A;</xsl:text>
(37)        </xsl:message>
(38)      </xsl:if>
(39)    </xsl:template>
(40)
(41)    <!--
(42)      Aufzeichnung von Abbildungsregeln
(43)     -->
(44)    <xsl:template name="traceTransformation">
(45)      <xsl:param name="transName" select="'Unknown'"/>
(46)      <xsl:param name="transRule" select="'Unknown'"/>
(47)      <xsl:param name="srcName" select="'srcElement'"/>
(48)      <xsl:param name="dstName" select="'dstElement'"/>
(49)      <xsl:param name="srcElement" select="'../unknownModel/#Unknown'"/>
(50)      <xsl:param name="dstElement" select="'../unknownModel/#Unknown'"/>
(51)      <xsl:param name="check-sum" select="-1"/>
(52)
(53)      <xsl:if test="$tracingOn">
(54)        <xsl:message>
(55)          <xsl:text disable-output-escaping="yes">&#x0A;&#9;&lt;</xsl:text>
(56)            <xsl:value-of select="$transName"/>
(57)            <xsl:text>:</xsl:text>
(58)            <xsl:value-of select="$transRule"/>
(59)          <xsl:text>&gt;&#x0A;</xsl:text>
(60)
(61)            <xsl:text disable-output-escaping="yes">&#9;&#9;&lt;</xsl:text>
(62)            <xsl:value-of select="$srcName"/>
(63)            <xsl:text> href=&quot;</xsl:text><xsl:value-of
      select="concat(replace($srcMdl, '\\', '/'), '#', $srcElement)"/><xsl:text
      disable-output-escaping="yes">&quot;/&gt;&#x0A;</xsl:text>
(64)            <xsl:text disable-output-escaping="yes">&#9;&#9;&lt;</xsl:text>
(65)            <xsl:value-of select="$dstName"/>
(66)            <xsl:text> href=&quot;</xsl:text><xsl:value-of
      select="concat(replace($dstMdl, '\\', '/'), '#', $dstElement)"/>
(67)            <xsl:if test="not($check-sum = -1)">
(68)              <xsl:text disable-output-escaping="yes">&quot;
      checksum=&quot;</xsl:text><xsl:value-of select="$check-
      sum"/><xsl:text>&quot;</xsl:text>
(69)            </xsl:if>
(70)            <xsl:text>/&gt;&#x0A;</xsl:text>
(71)
(72)          <xsl:text disable-output-escaping="yes">&#9;&lt;/</xsl:text>
(73)            <xsl:value-of select="$transName"/>
(74)            <xsl:text>:</xsl:text>
(75)            <xsl:value-of select="$transRule"/>
(76)          <xsl:text>&gt;&#x0A;</xsl:text>
(77)        </xsl:message>
(78)      </xsl:if>
(79)    </xsl:template>
(80)
(81)    <!--
(82)      Trancing abschließen
```

```
(83)
(84)      Der Aufruf dieses Templates muss erfolgen, nachdem die letzte
      Abbildungsregel
(85)      aufgezeichnet wurde.
(86)     -->
(87)    <xsl:template name="finalizeTracing">
(88)      <xsl:if test="$tracingOn">
(89)        <xsl:message>
(90)          <xsl:text disable-output-
      escaping="yes">&#x0A;&lt;/xmi:XMI&gt;&#x0A;</xsl:text>
(91)        </xsl:message>
(92)      </xsl:if>
(93)    </xsl:template>
(94)
(95)  </xsl:stylesheet>
```

Listing 24 – XSL-Stylesheet zur Ausgabe von Verfolgbarkeitsinstanzen (xsltrace.xsl)

Durch diese Integration in das Stylesheet zur Transformation der OWL-Ontologie und in das Stylesheet zur Transformation der OWL-/RDFa-Annotationen wird jeweils eine XML-Datei mit Verfolgbarkeitsinformationen erzeugt (`trace.owl2prm_ontology` und `trace.owl2-prm_annotation` bzw. `trace.rdfa2prm_annotation`). Die Verfolgbarkeitsinformationen orientieren sich an den bei der Ausführung der QVTs erzeugten Informationen (siehe Abschnitt 5.4). Das Listing 25 zeigt ein Beispiel für Informationen, die bei der Transformation der Ontologie aufgezeichnet wurden. In den Zeilen 18 bis 21 ist beispielsweise innerhalb der Transformation owl2prm_`ontology` die Anwendung der Regel `UmlClass` (entspricht XSLT_OWL-ONTOLOGY-PRM-04 in Tabelle 17) auf das Ausgangselement `VwAntrag` innerhalb des Ausgangsmodells `../Input/govobj-ontologie.owl` dokumentiert (Zeile 19). Das erzeugte Zielelement mit der `xmi:id`-Eigenschaftenausprägung `vwantrag_class` im Zielmodell `../Model/prm_ontology-govobj-ontologie.uml` ist in Zeile 20 dokumentiert.

In Listing 26 sind die Informationen dargestellt, die während der Durchführung einer Transformation der OWL-Annotationen aufgezeichnet wurden. In den Zeilen 8 bis 11 ist aufgezeichnet worden, dass die Abbildungsregel owl2prm_annotation:UmlInstance-Specification (entspricht XSLT_OWL-ANNOTATION_PRM-05 in Tabelle 29) ausgeführt wurde. Im Rahmen ihrer Ausführung wurde die Annotation `Bewohner` in das Modellelement mit dem Wert der `xmi:id`-Eigenschaft `d4e138_individual` transformiert. Die Zeilen 13 bis 16 zeigen die Anwendung dieser Regel auf die Annotation `Antrag`. Am Beispiel dieser Annotation ist in den Zeilen 18 bis 21 erkennbar, wie welche Informationen bei der Transformation der Ausprägung eines Data Properties in eine Attributsausprägung der UML-Instanz aufgezeichnet werden. Zusätzlich zu den Angaben über Ausgangs- und Zielelement wird für den Wert des Data Property eine Prüfsumme berechnet und gespeichert.[355] Basierend auf der Prüfsumme (Attribut `checksum` in Zeile 20) können Veränderungen der Attributsausprägungen eines Elements im Vergleich zur Annotation ermittelt werden.

Auch bei der Transformation von OWL-Object Properties in UML Instance Specification Links werden Verfolgbarkeitsinformationen aufgezeichnet (Zeilen 25 bis 28). Für die Identifikation der Instanzen von Object Properties ist die Verwendung einer eindeutigen ID oder URI, ausgehend von den Annotationen, nicht möglich, weil sie mehrfach instanziiert werden und nur im Kontext der instanziierten Konzepte eindeutig sind. Deshalb wurde für ihre Referenzierung eine Notation gewählt, die die beteiligten Konzepte im Sinne von Subjekt-Prädikat-Objekt verbindet (z. B. `#Bewohner/stelltAntrag/#Antrag` in Zeile 26).

Die bei einer OntoMat-Transformation aufgezeichneten Verfolgbarkeitsinformationen unterscheiden sich von denen einer RDFa-Transformation lediglich im Namensraum der

[355] In der prototypischen Implementierung dieser Verfolgbarkeitsinformation wurde zu Demonstrationszwecken ein sehr einfacher Algorithmus gewählt, der die Prüfsumme als Quotient von Zeichenanzahl und Wortanzahl ermittelt und deshalb Beschränkungen hinsichtlich der Eindeutigkeit hat. Er kann für die praktische Anwendung durch komplexe Algorithmen (z.B. MD5) ersetzt werden.

Transformation und in der Werteausprägung der `href`-Eigenschaft des Quellelementes (Listing 27). Als Namensraum der Transformation wird `rdfa2prm_annotation` anstelle von `owl2prm_annotation` verwendet. Die `href`-Eigenschaft verwendet die aus dem XHMTL-Dokument mit der XSL-Funktion `generate-id()` erzeugte eindeutige Identifikation für das Ausgangselement (Zeile 17).

In den XML-Dateien sind auch für die weiteren zuvor definieren Abbildungsregeln Verfolgbarkeitsinformationen aufgezeichnet, so dass für die gesamte Transformation Informationen vorliegen.

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xmi:XMI xmi:version="2.0"
(3)     xmlns:xmi="http://www.omg.org/XMI"
(4)     xmlns:owl2prm_ontology="urn:owl2prm_ontology.ecore">
(5)
(6)     <owl2prm_ontology:UmlModel>
(7)       <srcOwlOntology href="../Input/govobj-
      ontologie.owl#Verwaltungshandeln"/>
(8)       <dstUmlModel href="../Model/prm_ontology-govobj-
      ontologie.uml#verwaltungshandeln_model"/>
(9)     </owl2prm_ontology:UmlModel>
(10)
(11)    <owl2prm_ontology:UmlPackage>
(12)      <srcOwlOntology href="../Input/govobj-
      ontologie.owl#Verwaltungshandeln"/>
(13)      <dstUmlPackage href="../Model/prm_ontology-govobj-
      ontologie.uml#verwaltungshandeln_package"/>
(14)    </owl2prm_ontology:UmlPackage>
(15)
(16)  <!-- ... -->
(17)
(18)    <owl2prm_ontology:UmlClass>
(19)      <srcOwlClass href="../Input/govobj-ontologie.owl#VwAntrag"/>
(20)      <dstUmlClass href="../Model/prm_ontology-govobj-
      ontologie.uml#vwantrag_class"/>
(21)    </owl2prm_ontology:UmlClass>
(22)
(23)    <owl2prm_ontology:UmlGeneralisierung>
(24)      <srcRdfsubClassOf href="../Input/govobj-ontologie.owl#VwAntrag"/>
(25)      <dstUmlGeneralization href="../Model/prm_ontology-govobj-
      ontologie.uml#vwantrag_class_is-a_vwhandlungsausloeser_class"/>
(26)    </owl2prm_ontology:UmlGeneralisierung>
(27)
(28)    <owl2prm_ontology:UmlClass>
(29)      <srcOwlClass href="../Input/govobj-ontologie.owl#VwAntragsteller"/>
(30)      <dstUmlClass href="../Model/prm_ontology-govobj-
      ontologie.uml#vwantragsteller_class"/>
(31)    </owl2prm_ontology:UmlClass>
(32)
(33)    <owl2prm_ontology:UmlGeneralisierung>
(34)      <srcRdfsubClassOf href="../Input/govobj-ontologie.owl#VwAntragsteller"/>
(35)      <dstUmlGeneralization href="../Model/prm_ontology-govobj-
      ontologie.uml#vwantragsteller_class_is-a_vwakteur_class"/>
(36)    </owl2prm_ontology:UmlGeneralisierung>
(37)
(38)    <!-- ... -->
(39)
(40)    <owl2prm_ontology:UmlAssociation>
(41)      <srcOwlObjectProperty href="../Input/govobj-
      ontologie.owl#hatAntragsteller"/>
(42)      <dstUmlAssociation href="../Model/prm_ontology-govobj-
      ontologie.uml#vwant8_hatan16_vwant15"/>
(43)    </owl2prm_ontology:UmlAssociation>
(44)
(45)    <!-- ... -->
(46)
(47)  </xmi:XMI>
```

Listing 25 – Aufgezeichnete Verfolgbarkeitsinformationen der Ontologie-Transformation

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xmi:XMI xmi:version="2.0"
(3)     xmlns:xmi="http://www.omg.org/XMI"
```

```
(4)     xmlns:owl2prm_annotation="urn:owl2prm_annotation.ecore">
(5)
(6)     <!-- ... -->
(7)
(8)   <owl2prm_annotation:UmlInstanceSpecification>
(9)       <srcOwlIndividual href="../Rdf/stvg_vwv-stvo.rdf#Bewohner"/>
(10)      <dstUmlInstanceSpecification href="../Model/prm_stvg_vwv-
      stvo.uml#d4e138_individual"/>
(11)    </owl2prm_annotation:UmlInstanceSpecification>
(12)
(13)    <owl2prm_annotation:UmlInstanceSpecification>
(14)      <srcOwlIndividual href="../Rdf/stvg_vwv-stvo.rdf#Antrag"/>
(15)      <dstUmlInstanceSpecification href="../Model/prm_stvg_vwv-
      stvo.uml#d4e142_individual"/>
(16)    </owl2prm_annotation:UmlInstanceSpecification>
(17)
(18)    <owl2prm_annotation:UmlInstanceSpecificationSlot>
(19)      <srcOwlIndividual href="../Rdf/stvg_vwv-stvo.rdf#Antrag/name"/>
(20)      <dstUmlInstanceSpecificationSlot href="../Model/prm_stvg_vwv-
      stvo.uml#d4e147_slot1" checksum="25000"/>
(21)    </owl2prm_annotation:UmlInstanceSpecificationSlot>
(22)
(23)    <!-- ... -->
(24)
(25)    <owl2prm_annotation:UmlInstanceSpecificationLink>
(26)      <srcOwlIndividualProperty href="../Rdf/stvg_vwv-
      stvo.rdf#Bewohner/stelltAntrag/#Antrag"/>
(27)      <dstUmlInstanceSpecificationLink href="../Model/prm_stvg_vwv-
      stvo.uml#d4e140_instance-specification-link"/>
(28)    </owl2prm_annotation:UmlInstanceSpecificationLink>
(29)
(30)    <!-- ... -->
(31)
(32)  </xmi:XMI>
```

Listing 26 – Aufgezeichnete Verfolgbarkeitsinformationen der Annotations-Transformation (OntoMat)

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xmi:XMI xmi:version="2.0"
(3)     xmlns:xmi="http://www.omg.org/XMI"
(4)     xmlns:rdfa2prm_annotation="urn:rdfa2prm_annotation.ecore">
(5)
(6)     <rdfa2prm_annotation:UmlModel>
(7)       <srcOwlOntology href="../Rdfa/stvg_vwv-stvo_rdfa.xhtml#d4e1_ontology"/>
(8)       <dstUmlModel href="../Model/prm_stvg_vwv-stvo_rdfa.uml#d4e1_model/>
(9)     </rdfa2prm_annotation:UmlModel>
(10)
(11)    <rdfa2prm_annotation:UmlPackage>
(12)      <srcOwlOntology href="../Rdfa/stvg_vwv-stvo_rdfa.xhtml#d4e1_ontology"/>
(13)      <dstUmlModel href="../Model/prm_stvg_vwv-stvo_rdfa.uml#d4e1_package/>
(14)    </rdfa2prm_annotation:UmlPackage>
(15)
(16)    <rdfa2prm_annotation:UmlInstanceSpecification>
(17)      <srcOwlIndividual href="../Rdfa/stvg_vwv-
      stvo_rdfa.xhtml#d4e52_individual"/>
(18)      <dstUmlInstanceSpecification href="../Model/prm_stvg_vwv-
      stvo_rdfa.uml#d4e52_individual/>
(19)    </rdfa2prm_annotation:UmlInstanceSpecification>
(20)
(21)  <!-- ... -->
(22)
(23)  </xmi:XMI>
```

Listing 27 – Aufgezeichnete Verfolgbarkeitsinformationen der Annotations-Transformation (RDFa)

Mit dem XSL-Stylesheet in Listing 28 werden die Verfolgbarkeitsklassen erzeugt. Dazu wird das XSL-Stylesheet auf die XSL-Stylesheets zur Transformation der OWL-Ontologie und der OWL-/RDFa-Annotationen in UML-Modelle angewendet.[356] Zunächst wird aus dem XSL-Stylesheet der Name der Transformation ermittelt (Zeilen 23 bis 30) und als Dateiname

[356] Da XSL-Stylesheets vollständige und gültige XML-Dokumente sind, ist es prinzipiell möglich, ein XSL-Stylesheet als Eingabedokument einer XSL-Transformation zu verwenden und mit Hilfe eines anderen XSL-Stylesheets in ein Zieldokument zu transformieren.

für das zu erzeugende Zieldokument verwendet (Zeile 33). Dann wird für jeden Aufruf des Templates zur Aufzeichnung der Verfolgbarkeitsinstanzen eine Verfolgbarkeitsklasse angelegt (Zeilen 58 bis 106). Der Verfolgbarkeitsklasse werden für das Ausgangselement und das Zielelement Attribute hinzugefügt (Zeilen 63 bis 71 und 74 bis 104). Im Ergebnis wird pro XSL-Stylesheet eine Datei erzeugt, die die Verfolgbarkeitsklassen als Ecore-Modell repräsentiert. Die Abbildung 52 (auf Seite 149) zeigt einen entsprechenden Ausschnitt des Ecore-Modells innerhalb von Eclipse als Struktur (links oben), als Diagramm unter Verwendung der *Eclipse Ecore Tools*[357] (links oben) und als entsprechendes XML-Dokument (unten) dar.

Damit wird insgesamt ein Ergebnis erzielt, dass den Informationen entspricht, die auch im Rahmen einer QVT-Transformation aufgezeichnet werden. Die weitergehende Diskussion dieses Ergebnisses erfolgt in Abschnitt 5.4, um dem Ergebnis der QVT-Transformationen aus Abschnitt 5 nicht vorzugreifen.

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <!--
(3)    Konvertiert die XSLTs für die Transformation
(4)    der OWL-Ontologie und der OWL-/RDFa-Annotationen
(5)    in Verfolgbarkeitsklassen eines Ecore-Modells
(6)   -->
(7)   <xsl:stylesheet version="2.0"
(8)    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(9)    xmlns:xs="http://www.w3.org/2001/XMLSchema"
(10)   xmlns:fn="http://www.w3.org/2005/xpath-functions"
(11)   xmlns:xmi="http://www.omg.org/XMI"
(12)   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(13)   xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore">
(14)
(15)   <xsl:output name="xml" method="xml" version="1.0" encoding="UTF-8"
       indent="yes"/>
(16)
(17)   <!--  Parameter zur Angabe des Zielverzeichnisses (vorbelegt mit dem das
       QVT-Trace-Verzeichnis) -->
(18)   <xsl:param name="dir" select="'../Trace'"/>
(19)
(20)   <!-- Wurzelelement Stylesheet verarbeiten -->
(21)   <xsl:template match="/xsl:stylesheet">
(22)
(23)    <!-- Name der Transformation (inkl. ihn umschließender Anführungszeichen)
       -->
(24)    <xsl:variable name="transname"
       select="xsl:variable[@name='transName']/@select"/>
(25)    <!-- Name der Transformation (ohne Anführungszeichen) -->
(26)    <xsl:variable name="transformation"
       select="substring(substring($transname, 2), 1, string-length($transname)-
       2)"/>
(27)    <!-- Dateiname und Pfade des zu erzeugenden Ecore-Modells -->
(28)    <xsl:variable name="result-filename" select="concat($transformation,
       '.ecore')"/>
(29)    <xsl:variable name="result-path" select="concat(replace($dir,'\\', '/'),
       '/')"/>
(30)    <xsl:variable name="result" select="concat($result-path, $result-
       filename)"/>
(31)
(32)    <!-- Erzeugen des Ecore-Modells mit dem zuvor ermittelten Namen und Pfad -
       ->
(33)    <xsl:result-document format="xml" href="{$result}">
(34)     <!--  Wurzelknoten des Ecore-Modells (inkl. Namespaces) -->
(35)     <xsl:element name="ecore:EPackage">
(36)      <xsl:attribute name="xmi:version" select="'2.1'"/>
(37)      <xsl:namespace name="xmi" select="'http://www.omg.org/XMI'"/>
(38)      <xsl:namespace name="xsi" select="'http://www.w3.org/2001/XMLSchema-
       instance'"/>
```

[357] Die Ecore Tools stellen den hier verwendeten Ecore Diagram Editor bereit. Sie haben als Teilprojekt des Projektes *Eclipse Modeling Framework Technology (EMFT)* ihre Homepage unter der folgenden Adresse: http://www.eclipse.org/modeling/emft/?project=ecoretools (letzter Aufruf: 21.08.2011).

```
(39)      <xsl:namespace name="ecore"
      select="'http://www.eclipse.org/emf/2002/Ecore'"/>
(40)      <xsl:attribute name="name" select="$transformation"/>
(41)      <xsl:attribute name="nsURI" select="concat('urn:', $transformation,
      '.ecore')"/>
(42)      <xsl:attribute name="nsPrefix" select="$transformation"/>
(43)
(44)      <!-- Jeden Aufruf des traceTransformation-templates aus der xsltrace.xsl
      bearbeiten -->
(45)      <xsl:for-each select="//xsl:call-template[@name='traceTransformation']">
(46)       <!-- Bezeichner des Quellelements (inkl. ihn umschließender
      Anführungszeichen) -->
(47)       <xsl:variable name="src" select="xsl:with-
      param[@name='srcName']/@select"/>
(48)       <!-- Bezeichner des Zielelements (inkl. ihn umschließender
      Anführungszeichen) -->
(49)       <xsl:variable name="dst" select="xsl:with-
      param[@name='dstName']/@select"/>
(50)       <!-- Bezeichner des Abbildungsregel (inkl. ihn umschließender
      Anführungszeichen) -->
(51)       <xsl:variable name="trule" select="xsl:with-
      param[@name='transRule']/@select"/>
(52)       <!--  Bezeichner des Quell-, Zielelementes und der Abbildungsregel -->
(53)       <xsl:variable name="srcName" select="substring(substring($src, 2), 1,
      string-length($src)-2)"/>
(54)       <xsl:variable name="dstName" select="substring(substring($dst, 2), 1,
      string-length($dst)-2)"/>
(55)       <xsl:variable name="transRule" select="substring(substring($trule, 2),
      1, string-length($trule)-2)"/>
(56)
(57)       <!-- Ecore-Klasse (Verfolgbarkeitsklasse) mit dem Namen der
      Abbildungsregel -->
(58)       <xsl:element name="eClassifiers">
(59)        <xsl:attribute name="xsi:type" select="'ecore:EClass'"/>
(60)        <xsl:attribute name="name" select="$transRule"/>
(61)
(62)        <!-- Attribut für das Quellelement mit entsprechendem Namen anlegen --
      >
(63)        <xsl:element name="eStructuralFeatures">
(64)         <xsl:attribute name="xsi:type" select="'ecore:EReference'"/>
(65)         <xsl:attribute name="name" select="$srcName"/>
(66)         <!-- Typ des Quellelementes hat in Ecore keine Entsprechung, deshalb
      EObject -->
(67)         <xsl:element name="eType">
(68)          <xsl:attribute name="xsi:type" select="'ecore:EClass'"/>
(69)          <xsl:attribute name="href"
      select="'http://www.eclipse.org/emf/2002/Ecore#//EObject'"/>
(70)         </xsl:element>
(71)        </xsl:element>
(72)
(73)        <!-- Attribut für das Zielelement mit entsprechendem Namen anlegen -->
(74)        <xsl:element name="eStructuralFeatures">
(75)         <xsl:attribute name="xsi:type" select="'ecore:EReference'"/>
(76)         <xsl:attribute name="name" select="$dstName"/>
(77)         <!--  Datentyp des Elements per Namenskonvention ermitteln -->
(78)         <xsl:element name="eType">
(79)          <xsl:attribute name="xsi:type" select="'ecore:EClass'"/>
(80)          <xsl:attribute name="href">
(81)           <xsl:choose>
(82)            <xsl:when test="ends-with($dstName, 'Model')">
(83)             <xsl:text>http://www.eclipse.org/uml2/3.0.0/UML#//Model</xsl:text
      >
(84)            </xsl:when>
(85)            <xsl:when test="ends-with($dstName, 'Package')">
(86)             <xsl:text>http://www.eclipse.org/uml2/3.0.0/UML#//Package</xsl:te
      xt>
(87)            </xsl:when>
(88)            <xsl:when test="ends-with($dstName, 'InstanceSpecification')">
(89)             <xsl:text>http://www.eclipse.org/uml2/3.0.0/UML#//InstanceSpecifi
      cation</xsl:text>
(90)            </xsl:when>
(91)            <xsl:when test="ends-with($dstName, 'InstanceSpecificationLink')">
(92)             <xsl:text>http://www.eclipse.org/uml2/3.0.0/UML#//InstanceSpecifi
      cation</xsl:text>
(93)            </xsl:when>
```

```
(94)            <xsl:when test="ends-with($dstName, 'Stereotype')">
(95)             <xsl:text>http://www.eclipse.org/uml2/3.0.0/UML#//InstanceSpecifi
      cation</xsl:text>
(96)            </xsl:when>
(97)            <xsl:otherwise>
(98)
        <xsl:text>http://www.eclipse.org/emf/2002/Ecore#//EObject</xsl:text>
(99)             </xsl:otherwise>
(100)           </xsl:choose>
(101)         </xsl:attribute>
(102)        </xsl:element>
(103)       <!-- End: eStructuralFeatures -->
(104)       </xsl:element>
(105)      <!--  End: eClassifiers -->
(106)      </xsl:element>
(107)     <!-- End: xsl:call-template -->
(108)     </xsl:for-each>
(109)    <!--  End: ecore:EPackage -->
(110)    </xsl:element>
(111)
(112)   </xsl:result-document>
(113)
(114)   <!-- Informationsmeldung auf der Konsole ausgeben -->
(115)   <xsl:message>
(116)    <xsl:text disable-output-escaping="yes">Erzeugt wurde: </xsl:text>
(117)    <xsl:value-of select="$result-filename"/>
(118)    <xsl:text disable-output-escaping="yes">.</xsl:text>
(119)   </xsl:message>
(120)  </xsl:template>
(121)
(122) </xsl:stylesheet>
```

Listing 28 – XSL-Stylesheet zur Erzeugung von Verfolgbarkeitsklassen

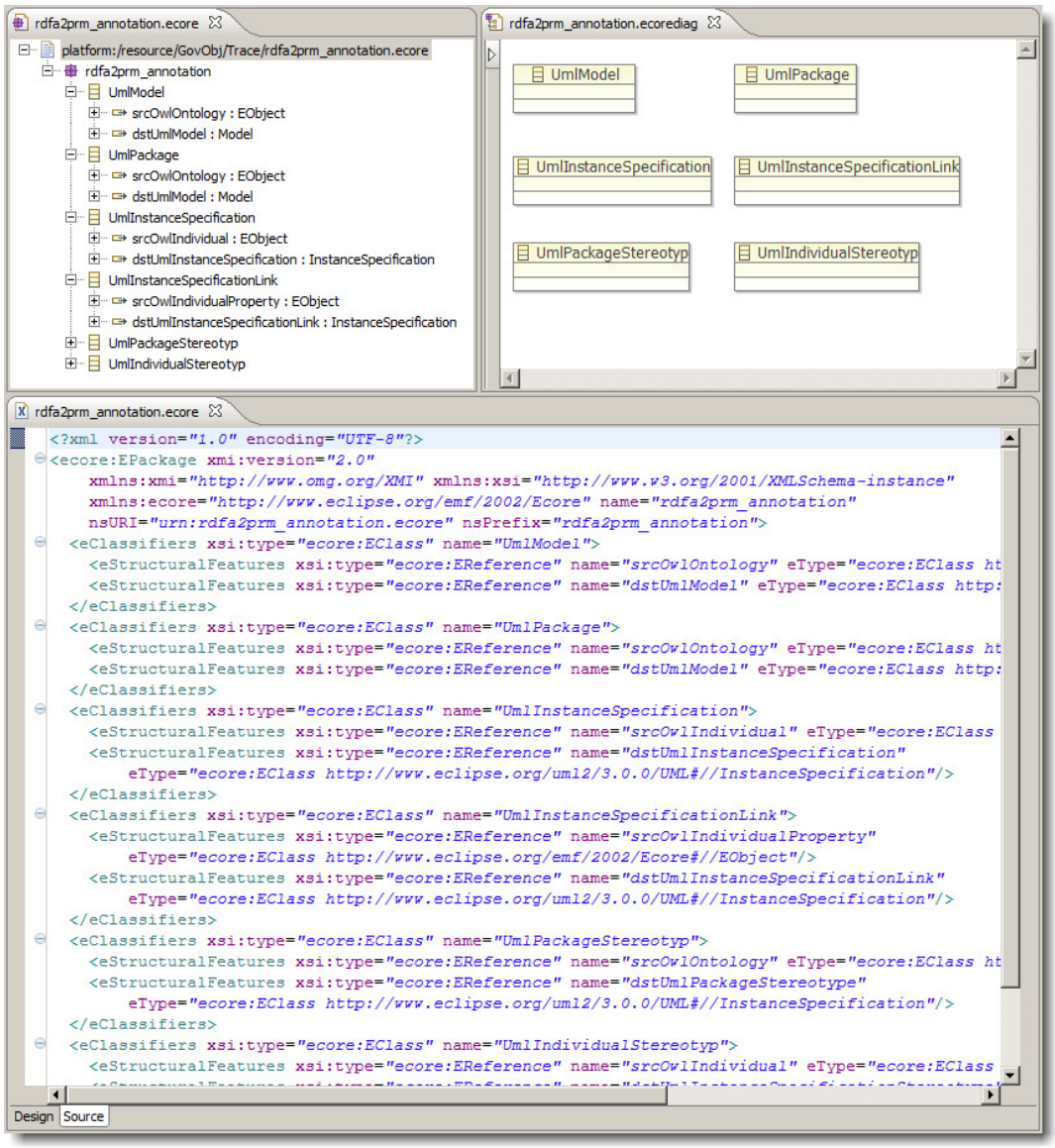


Abbildung 52 – Verfolgbarkeitsklassen aus der Transformation „rdfa2prm_annotation“ (Auszug)

4.3 Anwendung als Pre-Requirements Model

Das zuvor erzeugte Pre-Requirements Model (PRM) des Verwaltungshandelns, das Aspekte dieses Handelns in Form von Modellelementen und Beziehungen zwischen ihnen, basierend auf dem Text einer Rechtsvorschrift, formalisiert, wird als Ausgangspunkt für die weitere Modellierung verwendet. Es stellt die Instanzen von Klassen als Instance Specification mit dem Stereotyp «owlIndividual» dar, die durch Instanzen von Assoziationen (Instance Specification Link) verbunden sind.

Im Modell fehlen aufgrund der Rechtsvorschriften inhärenten Eigenschaften (vgl. Abschnitt 1.2) noch Informationen, um es im Sinne von MDA-Transformationen aus fachlicher Sicht unverändert direkt weiterverarbeiten zu können. Darüber hinaus müssen gegebenenfalls noch Besonderheiten des Verwaltungshandelns im Modell berücksichtigt werden. Um diese fehlenden Informationen zu ergänzen und alle relevanten Aspekte des Vorfeldes der Anforderungsspezifikation darzustellen, wird in einem notwendigen manuellen Schritt das ODM-konforme Modell des Verwaltungshandelns weiterentwickelt:

- Vorhandene Modellelemente können um zusätzliche Informationen ergänzt oder vorhandene Informationen können geändert werden.

- Es können neue Modellelemente angelegt und beschrieben werden, die keinen Bezug zu Annotationen in Rechtsvorschriften haben.
- Vorhandene Modellelemente mit Bezug zu Annotationen können gelöscht werden.

4.3.1 Ergänzung vorhandener Elemente

Ergänzungen vorhandener Modellelemente sind durch Bearbeitung von Attributsausprägungen und durch die Dokumentation mit Hilfe der Werkzeugfunktionen möglich. Die `uml:InstanceSpecifications` im PRM können bereits Ausprägungen von Data Properties besitzen, die in der Auseinandersetzung mit dem Text als Annotation angelegt wurden. Wurden die Ausprägungen während der Auseinandersetzung mit dem Text nicht abschließend bearbeitet oder wurden keine Ausprägungen angelegt, können diese nun im Modell bearbeitet oder angelegt werden. Die Abbildung 53 zeigt ein Beispiel, in dem für das «`owlIndividual`» `MitgliedEinerCar-SharingOrganisation,WirdDerenNameImKennzeichenfeld`, das eine Instanz der UML-Klasse `VwRechtsfolgenMerkmal` ist, die Ausprägung des Attributes `Arbeitsnotiz` ergänzt wurde. Dieses Attribut wird durch die Klasse `Verwaltungskonzept` definiert, von der `VwRechtsfolgenMerkmal` im UML-Modell (analog zur Ontologie) eine Spezialisierung ist. Für die Arbeitsnotiz kann deshalb exemplarisch die Erläuterung des Begriffs Car Sharing (z.B. „Car sharing ist die organisisierte ...“, nach Wikipedia) eingefügt werden, um weiteres Wissen über das Vorfeld der Anforderungsanalyse zu formalisieren.

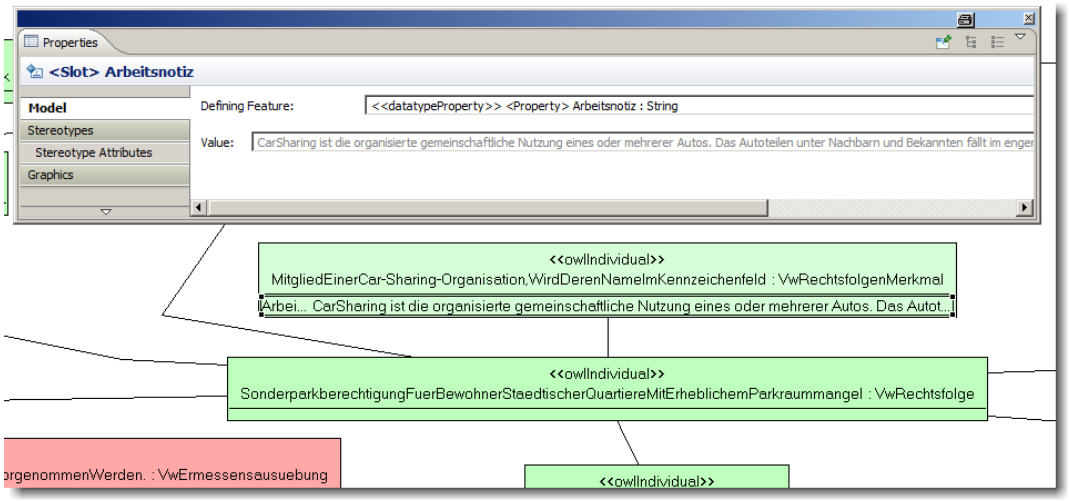


Abbildung 53 – Beispiel für die Ergänzung einer Attributsausprägung

Darüber hinaus ist es möglich, die Dokumentationsmöglichkeiten des eingesetzten Modellierungswerkzeugs zu nutzen, um ergänzende Informationen zu den Elementen zu erfassen. In Abbildung 54 ist ein Beispiel dargestellt, wie zum «`owlIndividual`» `DortTatsaechlichWohnt` eine solche Ergänzung vorgenommen wurde, die auf eine Abhängigkeit zu einer anderen Voraussetzungsbedingung hinweist.

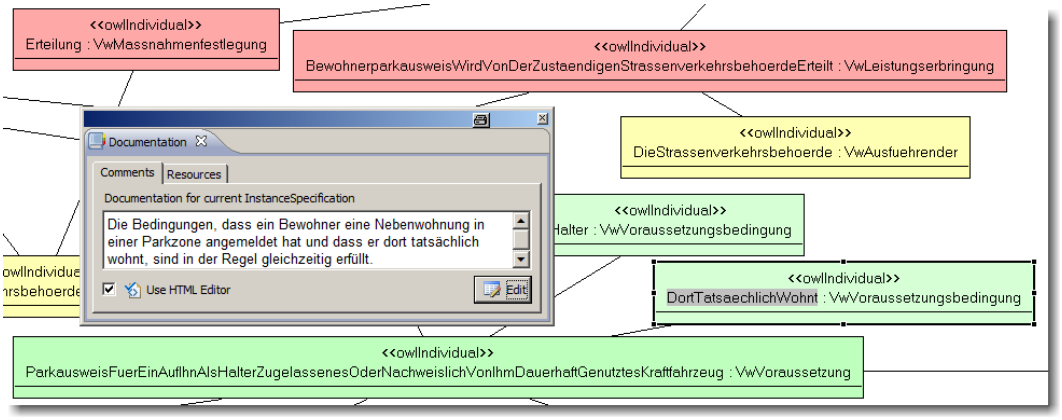


Abbildung 54 – Beispiel einer Ergänzung durch Dokumentation mit Werkzeugfunktionen

4.3.2 Hinzufügen neuer Elemente

Durch Anlegen einer neuen Instanz eines Konzepts aus der Ontologie in Form einer Instance Specification mit dem Stereotyp «owlIndividual» oder durch Herstellen von Beziehungen in Form von Instance Specification links kann das PRM um neue Elemente ergänzt werden. Es ist prinzipiell auch möglich, neue Diagramme innerhalb des initial erzeugten Modells anzulegen.

Da in diesem Modellierungsschritt nicht mehr die Auseinandersetzung mit dem Text der Rechtsvorschrift im Vordergrund steht, können prinzipiell Instanzen aller UML-Klassen der Ontologie angelegt werden. Die Beschränkung auf solche Instanzen, die sich nachvollziehbar aus dem Text der Rechtsvorschrift ergeben, gilt hier nicht mehr. Allerdings gilt beim Anlegen einer Instanz im Sinne einer Modellierungsvorgabe, dass diese zwingend einer UML-Klasse aus der Ontologie zugeordnet werden muss, damit sie über eine definierte PRM-Semantik verfügt.

Ein Beispiel hierfür zeigt die Abbildung 55. Es wurde die Instanz «`owlIndividual`» `SonderparkberechtigungFuerBewohnerStaedtischerQuartiereMitErheblichemParkraum-mangel` der Klasse `VwRechtsfolge` mit einer Instanz von `VwHinweisInformation` und diese mit zwei Instanzen der Klasse `VwLebenslage` in Beziehung gesetzt. Der entsprechende Instanz Specification link zwischen den Instanzen von `VwRechtsfolge` und `VwHinweisInformation` ist eine Instanz der Assoziation `moeglichesVerwaltungsprodukt`, und der link zwischen den Instanzen von `VwHinweisInformation` und `VwLebenslage` eine Instanz der Assoziation `zugehoerigerHinweis`. Um für die Instanz Bewohner deutlicher zu machen, dass es sich um den Bewohner einer Parkzone handelt, wurde die Instanz BewohnerParkzone eingeführt. Sie ist eine Verfeinerung der Instanz Bewohner, was durch eine Abhängigkeitsbeziehung mit dem UML-Standardstereotyp «`refine`» dargestellt wird. Dadurch wird hier exemplarisch ausgedrückt, dass einem Bürger bei Vorliegen der Lebenslage Umzug oder Einzug in eine Nebenwohnung ein Hinweis bzw. eine Information gegeben werden sollte, dass für ihn ein Bewohnerparkausweis sinnvoll oder möglich wäre, wenn seine Wohnung in einer Bewohnerparkzone liegt. Diese Modellierungsentscheidung kann mit Hilfe der Standarddokumentationsfunktion des Modellierungswerkzeugs dokumentiert werden.

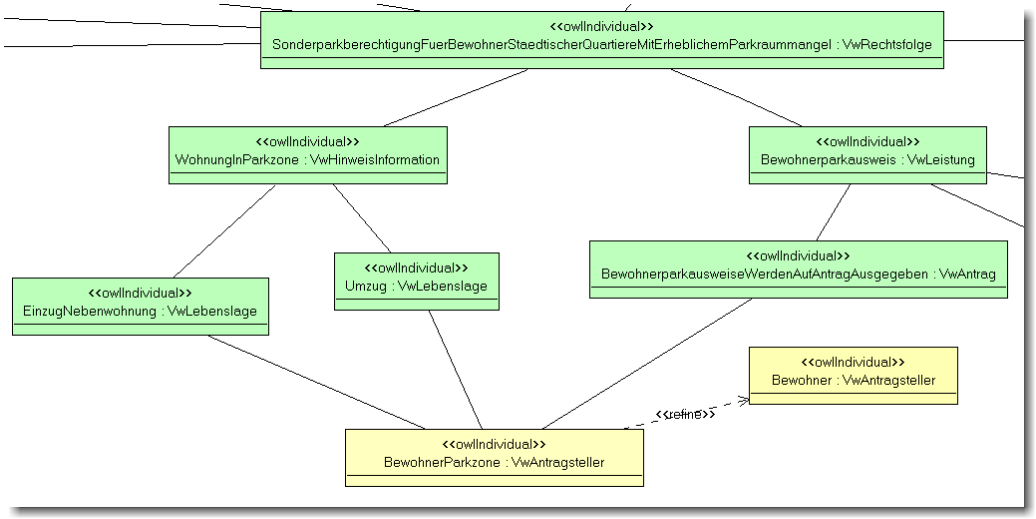


Abbildung 55 – Erweiterung des PRM um neue Modellelemente

4.3.3 Löschen vorhandener Elemente

In der Verfeinerung des PRM kann der Modellierer auch die Entscheidung treffen, dass Modellelemente, die aus dem Text einer Rechtsvorschrift gewonnen wurden, im Vorfeld der Anforderungsanalyse nicht relevant sind. Beispielsweise wurde in Abbildung 54 (Abschnitt 4.3.1) dokumentiert, dass die Instanzen «owlIndividual» `DortTatsaechlichWohnt` und «owlIndividual» `AngemeldeteNebenwohnungAusreichen`, den gleichen Sachverhalt ausdrücken. Hier wird deshalb angenommen, dass die `angemeldete Nebenwohnung`-Instanz für die weitere Modellierung nicht relevant ist und deshalb gelöscht werden kann.

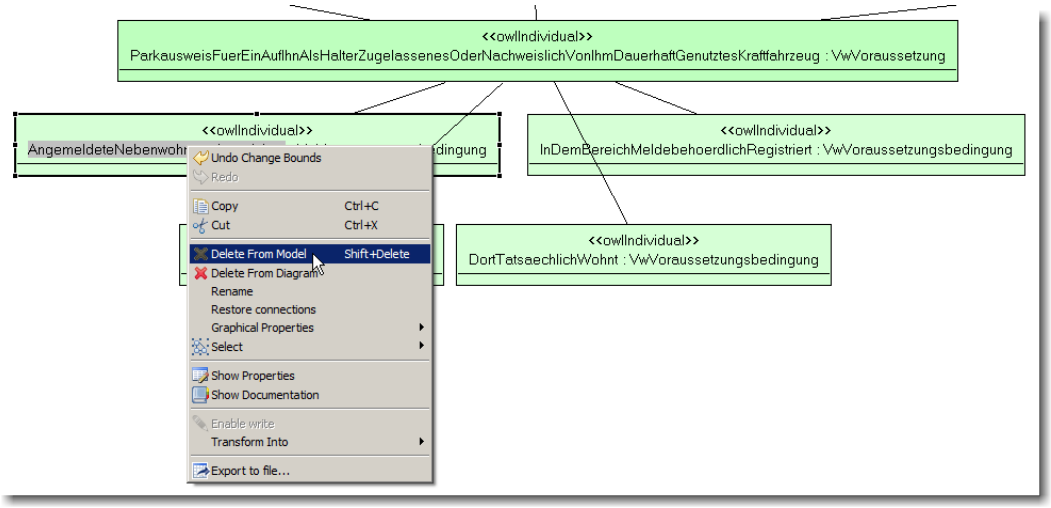


Abbildung 56 – Löschen von Modellelementen im PRM

4.3.4 Verfolgbarkeit der PR-Modellierung

Ausgehend von den möglichen Aktivitäten der Modellierung des PRM, die in den vorangegangenen Abschnitten 4.3.1 bis 4.3.3 anhand von Beispielen vorgestellt wurden, ergeben sich unterschiedliche Möglichkeiten ihrer Verfolgbarkeit.

Wurden vorhandene Modellelemente um zusätzliche Attributsausprägungen ergänzt, so haben sie in den Annotationen der Rechtsvorschriften keine Entsprechung. Um dies festzustellen, können die bei der initialen Modellerzeugung aufgezeichneten Verfolgbarkeitsinformationen ausgewertet werden. Dort sind diese Attributsausprägungen nicht enthalten und können deshalb als neu hinzugefügt identifiziert werden. Ergänzungen mit Hilfe der Standarddokumentationsfunktion des Werkzeugs sind ebenfalls als neu

hinzugefügt identifizierbar, da sie keine Entsprechung in den Verfolgbarkeitsinformationen haben können.

Veränderungen an den vorhandenen Attributsausprägungen der Modellelemente können identifiziert werden, weil innerhalb der aufgezeichneten Verfolgbarkeitsinformationen eine entsprechende Attributsausprägung bereits existiert. Durch Auswertung der Prüfsumme kann die Veränderung identifiziert werden, die Einfluss auf die inhaltliche Qualität dieser Information haben kann. Deshalb ist es sinnvoll, bei Auswertung der Verfolgbarkeitsinformationen diese Änderungen und ihre zugehörige Annotation identifizieren zu können.

Als Instance Specification oder Instance Specification link neu angelegte Modellelemente haben in den Verfolgbarkeitsinformationen, die während der initialen Erzeugung der Modelle erstellt wurden, keine Entsprechung. Deshalb können sie prinzipiell als neu hinzugefügt identifiziert werden. Durch Verwendung des Standardstereotyps «refine» kann für neu hinzugefügte Modellelemente ausgedrückt werden, dass diese ein Modellelement verfeinern, dass initial erzeugt wurde. Diese Information kann durch Auswertung des entsprechenden Pre-Requirements Model gewonnen und unter Verwendung der aufgezeichneten Verfolgbarkeitsinformationen bis auf die verfeinerte Annotation zurückgeführt werden. Somit sind auch Verfeinerungen von Modellelementen zu den Ursprüngen in Rechtsvorschriften verfolgbar.

Wurden während der Modellierungsaktivitäten initial angelegte Modellelemente gelöscht, so existieren in den aufgezeichneten Verfolgbarkeitsinformationen die entsprechenden Verfolgbarkeitsbeziehungen, ohne dass es im PRM ein entsprechendes Modellelement gibt. Dadurch ist es auch möglich, gelöschte Elemente zu identifizieren. Um das Fehlen des Modellelements nachvollziehbar zu machen, muss der Modellierer zusätzliche Informationen im Modell (z.B. bei anderen, noch vorhandenen Modellelementen oder als Notiz) erfassen.

Insgesamt ist für die Ergebnisse von Modellierungsaktivitäten innerhalb des PRM verfolgbar, auf welche Individuen in Form von Annotationen in Texten von Rechtsvorschriften sie sich beziehen.

4.4 Zusammenfassung

Ausgangspunkt für die Konstruktion und Anwendung der Modelle des Verwaltungshandelns in diesem Abschnitt waren die Annotation von Rechtsvorschriften. Sie repräsentieren instanziierte Konzepte der Ontologie in Form von OWL-Individuen, die über ihren Annotationskontext einen direkten Bezug zum Text der jeweiligen Vorschrift haben. Um sie im Rahmen der MDA zu nutzen, wurden sie in einem ersten Schritt anhand der durch das ODM UML-Profil weitgehend definierten Regeln in ein initiales Modell überführt, dass als neuer MDA-Modelltyp für das Vorfeld der Anforderungsanalyse (Pre-Requirements Model, PRM) eingeführt wurde, um es insbesondere von informationstechnikunabhängigen Modellen der Anforderungsspezifikation (CIMs) zu unterscheiden. Im PRM können die aus den Annotationen gewonnenen Modellelemente um die Besonderheiten des Verwaltungshandelns und sonstige Aspekte, die sich nicht direkt aus dem Text von Rechtsvorschriften ableiten lassen, verfeinert und ergänzt werden. Weil die Überführung der OWL-Individuen in das Modell verfolgbar ist, haben dessen Modellelemente einerseits einen direkten Bezug zu Texten von Rechtsvorschriften und können anderseits für die Modellierung der Besonderheiten des Verwaltungshandelns genutzt werden. Das PRM basiert dabei einerseits auf den Annotationen, andererseits auf den Klassendefinitionen der OWL-Ontologie, die ebenfalls unter Anwendung des ODM UML-Profils in ein MOF-konformes Modell transformiert wurden. Dadurch ist es möglich, während der Verfeinerung des PRM sowohl ontologische Beziehungen (z.B. zwischen Instanzen von Lebenslage und Verwaltungsprodukt), als auch Standardbeziehungen der UML (z.B. die Verfeinerungsbeziehung zwischen `Strassenverkehrsbehoerde` und `BuergerService`) zu nutzen.

Die Verfolgbarkeit der Transformation aus den annotierten Rechtsvorschriften in das initiale PRM wurde mit Mitteln der eingesetzten XSL-Transformationstechnik in Anlehnung an QVT-Verfolgbarkeit (Abschnitt 5.4) sichergestellt. Für die Erweiterung und Verfeinerung des PRM kann die Verfolgbarkeit über Standardmodellierungstechniken sichergestellt werden.

5 Anforderungsspezifikationsmodelle für E-Government

Das Pre-Requirements Modell (PRM) des Verwaltungshandelns bildet für dieses Kapitel den Ausgangspunkt. Es wird in ein informationstechnikunabhängiges Modell (CIM) transformiert, das im Rahmen dieser Arbeit als initiale Anforderungsspezifikation eines E-Government-Anwendungssystems dient (Abbildung 57). In der sich anschließenden Anforderungsanalyse kann das CIM verfeinert und vervollständigt werden.[358] Um bei dieser Aufgabe die Verfolgbarkeit sicherzustellen, gibt es etablierte Methoden und Techniken der Softwareentwicklung (vgl. Post-RS Traceability in Abschnitt 2.1). Somit kann davon ausgegangen werden, dass mit der Transformation des PRM in das CIM der Anschluss an die klassischen Aktivitäten der Softwareentwicklung gefunden ist und ebenfalls durchgängige Verfolgbarkeit erreicht werden kann.

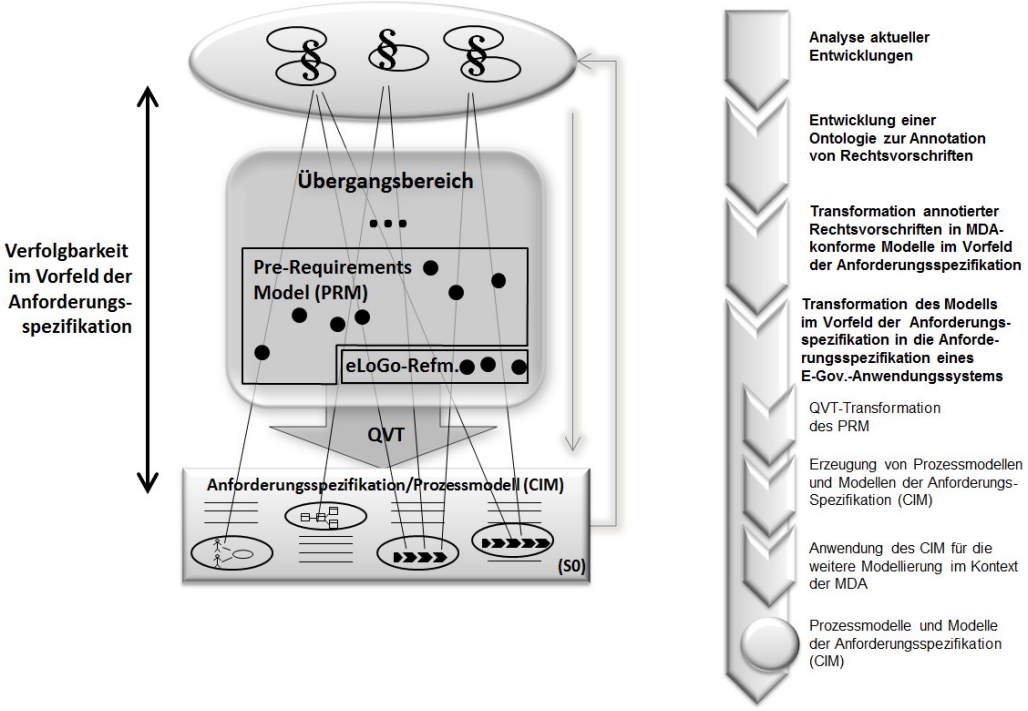


Abbildung 57 – Transformation des PRM in CIM unter Verwendung der eLoGo-Referenzmodelle

Zwischen den Elementen eines PRM und eines CIM gibt es konzeptionelle Unterschiede, aus denen sich eine Lücke zwischen beiden Modellen ergibt. Während das PRM auf Konzepten der Ontologie basiert, werden im CIM Anforderungen an das zu entwickelnde System formalisiert. Die Transformation von PRM nach CIM muss die Lücke zwischen den Konzepten überbrücken. In Anlehnung an die Funktion des Plattformmodells (PM) bei einer PIM-PSM-Transformation, bei der ebenfalls eine Lücke zu überbrücken ist (vgl. Abschnitt 2.2), wird hier der Einsatz von Referenzmodellen vorgeschlagen. Aufgabe dieser Modelle ist es, die Elemente des PSM soweit zu konkretisieren, dass die Lücke zu den Elementen im CIM überbrückt wird.

Anhand eines PRM des Verwaltungshandelns wird gezeigt, wie für den Ansatz dieser Arbeit die eLoGo-Referenzmodelle für E-Government verwendet werden können. In einer MDA-

[358] Die weitere Anforderungsanalyse basiert auf der Analyse zusätzlicher Anforderungsquellen (z.B. Zielen, Ideen und Meinungen der Beteiligten, Funktionsumfang eines abzulösenden Altsystems). Weil diese Quellen spezifisch für den jeweiligen Projektkontext sind und über die Betrachtung von Ursprüngen von Anforderungen in Rechtsvorschriften hinausgehen, ist dieser Schritt nicht mehr Bestandteil des Ansatzes der vorliegenden Arbeit.

Transformation wird aus den Elementen und den Beziehungen zwischen ihnen ein CIM im Sinne einer initialen Anforderungsspezifikation eines E-Government-Anwendungssystems erzeugt. Im Ergebnis entstehen dadurch Prozessmodelle als BPMN-Diagramme und Modelle der Anforderungsanalyse in Form von UML-Diagrammen mit erläuterndem Text als Beschreibung.

Das Kapitel beginnt mit einer Darstellung der Voraussetzungen und Rahmenbedingungen des PRM-CIM-Transformationsprozesses, schlägt den Lückenschluss, basierend auf Referenzmodellen analog zu Plattformmodellen, vor und stellt für das Beispiel des Verwaltungshandelns dieses Prinzip anhand der eLoGo-Referenzmodelle dar (Abschnitt 5.1). Anschließend wird die Transformation des vorliegenden PRM des Verwaltungshandelns in Prozessmodelle (Abschnitt 5.1.4) und Modelle der Anforderungsanalyse (Abschnitt 5.3) beschrieben. Während der Transformation werden Verfolgbarkeitsinformationen aufgezeichnet, deren Struktur und Inhalt in Abschnitt 5.4 vorgestellt werden.

5.1 Voraussetzungen und Rahmenbedingungen

Mit dem Übergang vom PRM zum CIM ist ein Wechsel des Standpunktes verbunden, denn beide Modelltypen stellen verschiedene Sichten auf das System dar. Damit verbunden sind Unterschiede zwischen den in den Modellen formalisierten Konzepten, den genutzten Modellelementen und deren Semantik. Für die Durchführung der Transformation des PRM in ein CIM sind deshalb bestimmte Voraussetzungen zu schaffen und bestimmte Rahmenbedingungen zu beachten, die in diesem Abschnitt beschrieben werden.

5.1.1 Standpunktwechsel vom PRM zum CIM

Das PRM wurde in Abschnitt 4.1 als MDA-konformer Modelltyp für das Vorfeld der Anforderungsspezifikation eingeführt. Konkrete Modelle dieses Typs umfassen Elemente und deren Beziehungen, die ein System von Aspekten vereinfacht beschreiben, die im Vorfeld einer Anforderungsspezifikation relevant sind. Bei diesen Aspekten handelt es sich nicht um die Anforderungen selbst, sondern um Ursprünge möglicher Anforderungen. Modelle des Typs CIM stellen hingegen das System vom informationstechnikunabhängigen Standpunkt aus einer Perspektive dar, die die Umgebung des Systems und die Anforderungen an das System zeigt. PRM- und CIM-Modelle formalisieren somit unterschiedliche Konzepte und nutzen dafür auch unterschiedliche Modellelemente. Diese Modellelemente werden durch das jeweils zugrunde liegende Metamodell definiert. Entsprechend werden für das PRM Modellelemente verwendet, die durch das ODM definiert sind. Beispielsweise sind im PRM Klassen von Konzepten einer geeigneten Ontologie und deren Instanzen modelliert. Für eine Anforderungsspezifiktion in Form eines CIM werden nach dem Stand der Technik die UML und/oder vergleichbare Notationen für spezielle Aspekte verwendet (z.B. BPMN für Prozessmodelle). In UML-Modellen des CIM werden beispielsweise Akteure, Anwendungsfälle, Klassen und ihre Beziehungen zur Spezifikation von Anforderungen verwendet.

Im Rahmen dieser Arbeit wird von einem speziellen PRM des Verwaltungshandelns und einem speziellen CIM eines E-Government-Anwendungssystems ausgegangen. Das PRM des Verwaltungshandelns basiert auf der Ontologie des Verwaltungshandelns und umfasst vier wesentliche Typen von Modellelementen. Zum einen umfasst es die aus Annotationen generierten Instanzen (`uml::InstanceSpecifications`) und die Klassen der Ontologie, deren Instanz sie sind. Zum anderen enthält es die aus den Annotationen generierten `Instance Specification links` und deren auf Object Properties der Ontolgoie zurückgehende Assoziationen. Mit den Instanzen wird das Vorkommen eines bestimmten Aspekts des Verwaltungshandelns im Text einer konkreten Rechtsvorschrift ausgedrückt. Diese Aspekte bilden in Form des PRM ein Modell, dass eine vereinfachte Sicht auf das Vorfeld der Anforderungsanalyse eines E-Government-Anwendungssystems darstellt.

Im Gegensatz dazu muss ein CIM, das für die Modellierung von Anforderungen an E-Government-Anwendungssysteme verwendet wird, einerseits die Anforderungen an

Verwaltungsprozesse und andererseits die Anforderungen an die unterstützende Anwendungssoftware formalisieren. Verwaltungsprozesse lassen sich in Form von Geschäftsprozessmodellen mit einer geeigneten Notation ausdrücken (vgl. Abschnitt 2.2). Die von der OMG standardisierte BPMN ist auch zur Darstellung von Verwaltungsprozessen geeignet.[359] Prozessdiagramme sind eine wesentliche Darstellungsform von Prozessen in BPMN.[360] Sie ziegen u.a. den Prozess, die Beteiligten, auszuführende Aktivitäten, bearbeitete Daten sowie einen Kontrollfluss, der die Aktivitäten verbindet und Verzweigungen enthalten kann. Zusätzlich zur Prozesssicht sind auch die Anforderungen an das E-Government-Anwendungssystem zu formalisieren. Es ist Stand der Technik, die funktionalen Anforderungen mit Hilfe von UML-Anwendungsfalldiagrammen auszudrücken. Diese Diagramme zeigen Akteure, die durch Kommunikationsbeziehungen mit Anwendungsfällen (Use Cases) verbunden sind sowie Beziehungen zwischen Anwendungsfällen. Für die Akteure, ihre Kommunikationsbeziehungen und den Anwendungsfall werden weitere Details als Fließtext, als strukturierter Text oder durch weitere Diagramme (z.B. Interaktions- oder Aktivitätendiagramme) beschrieben.

Ausgehend von diesen Beispielen für Konzepte, die einerseits in den Diagrammen von PRMs und andererseits in CIMs formalisiert werden können, zeigen sich auch hier Unterschiede, die beim Versuch, Modellelemente des PRM auf Elemente des CIM abzubilden, hinderlich sind. Beispielsweise erscheint es für Instanzen der Ontologie-Klasse `Verwaltungsakteur` noch nahe liegend, diese auf Rollen in Prozessdiagrammen oder Akteure in UML-Anwendungsfalldiagrammen abzubilden. Offen blieben dann aber Fragen der Beziehung des Akteurs zu bearbeiteten Prozessaktivitäten oder Anwendungsfällen. Für Instanzen der `Verwaltungshandlung` könnte die Abbildung auf Aktivitäten in Prozessdiagrammen oder auf Anwendungsfälle gelingen. Offen blieben dabei Fragen der Reihenfolge von Aktivitäten oder der inhaltlichen Beschreibung. Wie Instanzen von `Verwaltungshandlungsgegenstand` in diesen Diagrammen berücksichtigt werden können, ist nicht offensichtlich.

Diese Beispiele deuten auf eine Lücke zwischen dem PRM des Verwaltungshandelns und dem CIM eines E-Government-Anwendungssystems hin, die für die Durchführung der Transformation geschlossen werden muss. Im Rahmen dieser Arbeit wird nicht davon ausgegangen, dass durch Betrachtung der Ursprünge von Anforderungen im Vorfeld einer Anforderungsspezifikation alle Informationen gewonnen werden können, die für eine Anforderungsspezifikation relevant sind. Deshalb kann bzw. soll die Lücke nicht durch inhaltliche Erweiterung des PRM geschlossen werden. Andererseits ist die Semantik der CIM-Elemente als gegeben zu betrachten, so dass ein anderer Weg zum Lückenschluss gefunden werden muss.

[359] Schuppan schätzt ein, dass insbesondere die Schwimmbahnen-Darstellung, die einen wesentlichen Unterschied zur in Deutschland verbreiteten EPK-Notation darstellt, Optimierungspotenziale im Verwaltungshandeln gut erkennen lässt. (Freundliche Auskunft von Tino Schuppan am 09.08.2011.)

[360] Es gibt verschiedene Arten von BPMN-Diagrammen, von denen die Prozessdiagramme für den im Rahmen dieser Arbeit entwickelten Ansatz ausreichend sind. Die anderen Diagramme stellen spezielle Aspekte des Prozesses deutlicher dar: Die Zusammenarbeit zwischen mehreren Prozessbeteiligten kann mit Hilfe von Kollaborationsdiagrammen (Collaboration Diagrams) dargestellt werden. Sie zeigen Beteiligte (z.B. als Pools) mit den ausgetauschten Nachrichten (Message Flow) je nach Blickwinkel mit oder ohne der Darstellung des Prozessablaufs und den zugehörigen Aktivitäten. Durch Abstraktion weiterer Details dieser Zusammenarbeit können Konversationsdiagramme (Conversation Diagrams) erstellt werden, die im wesentlichen zusammengefasste Nachrichtenflüsse zwischen den Beteiligten darstellen. In Choreographiediagrammen werden Elemente eines Prozesses in den Vordergrund gerückt, die am Austausch einer oder mehrerer Nachrichten beteiligt sind. Dadurch kann die Abfolge der Nachrichten mit eventuell notwendigen Bedingungen und Schleifen dargestellt werden.

5.1.2 Referenzmodellunterstützte Transformation

Bei der in Abschnitt 2.2 vorgestellten PIM-PSM-Transformationen gab es ebenfalls eine Lücke zwischen den plattformunabhängigen Konzepten des PIM und den plattformspezifischen Konzepten des PSM. Diese Lücke wird durch das Konzept der Plattform geschlossen, das die Konkretisierung der PIM-Elemente bei der Transformation in das PSM erlaubt. Idealerweise fließt die Plattform in Form eines Plattformmodells in die PIM-PSM-Transformation ein, wobei stattdessen die Nutzung plattformspezifischer Abbildungsregeln verbreiteter ist. Die Plattform und das Plattformmodell liefern aber einen Ansatzpunkt, dessen Übertragbarkeit auf die Transformation des PRM in ein CIM zu untersuchen ist.

Für die Transformation von PRM nach CIM würde analog zum Plattformmodell ein Modell benötigt, das (in Anlehnung an die Definition des Plattformmodells in Abschnitt 2.2) eine Menge von zusammengehörigen fachlichen Modellierungskonzepten umfasst, die innerhalb von Modellen der Anforderungsanalyse (CIM) verwendet werden können, um Anforderungen an die Anwendung auszudrücken. Als Ergebnis einer Transformation unter Verwendung eines solchen Modells würde ein CIM einerseits Elemente wie Aktivitäten, Anwendungsfälle und Akteure umfassen. Andererseits würde es aber auch Elemente enthalten, die, basierend auf dem zusätzlichen Modell, die Semantik der Standardelemente weitergehend konkretisieren. Abbildung 58 stellt diesen Ansatz analog zum „Model Merging“ (vgl. Abschnitt 2.2) vereinfacht dar.

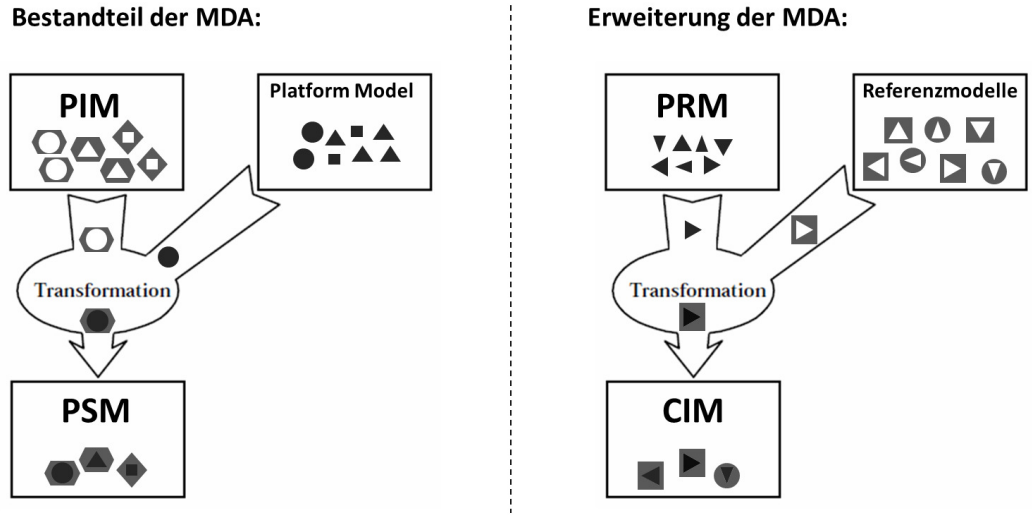


Abbildung 58 – Informelle Darstellung des Zusammenhangs zwischen PRM, Referenzmodell und CIM im Vergleich zu PIM, PSM und PM[361]

Modelle, die bei der Konstruktion anderer Modelle hilfreich sind und deshalb dafür herangezogen werden, wurden in Abschnitt 2.5 als Referenzmodelle definiert. Würde man Referenzmodelle analog zu Plattformmodellen für den Übergang vom PRM zum CIM einsetzen, so müssten sie einerseits auf einem Abstraktionsniveau bereitgestellt werden, dass ihnen aus dem Vorfeld der Anforderungsspezifikation heraus die Abbildung auf Elemente in informationstechnikunabhängige Modelle der Anforderungsspezifikation ermöglicht. Dazu müssten ihre Modellelemente mit den Elementen eines PRM kombinierbar sein. Andererseits müssten die Referenzmodelle bei der Entwicklung von Modellen der Anforderungsspezifikation nützlich sein. Dies erfordert bei Anwendung der MDA, dass die Referenzmodelle innerhalb von Transformationsprozessen zur Erstellung von Modellen des Typs CIM verwendet werden können. Mit Referenzmodellen, die diesen beiden

[361] Notation in Anlehnung an das „Model Merging“-Muster in [OMG MDA, 2003b], S. 3-12.

Bedingungen genügen, stünde ein Pendant zum Plattformmodell zur Verfügung, das in PRM-CIM-Transformationen verwendet werden kann.

Im Rahmen dieser Arbeit werden spezielle Modelle des Typs PRM und spezielle Modelle des Typs CIM betrachtet. Entsprechend bezieht sich der Einsatz von Referenzmodellen in dieser Arbeit auf Modelle, die zur Unterstützung einer PRM-CIM-Transformation für PRMs des Verwaltungshandelns in CIMs eines E-Government-Anwendungssystems verwendet werden können. Es gibt mit den eLoGo-Referenzmodellen für E-Government (siehe Abschnitt 2.5) solche Modelle, die auch für diesen Zweck eingesetzt werden können.[362] Das eLoGo-Referenzprozessmodell für E-Government umfasst eine Semantikdefinition und die Konkretisierung für die gängigen Elemente von Verwaltungsprozessen und die Beziehungen dieser Elemente zueinander. Analog dazu enthält das eLoGo-Referenzanforderungsmodell eine weitergehende Semantikdefinition und die Konkretisierung für die gängigen Elemente von Anwendungsfallmodellen eines E-Government-Anwendungssystems sowie die Beziehungen zwischen diesen Elementen. Beide Modelle bieten diese Eigenschaften aufgrund ihres Referenzcharakters, da sie für die Erstellung und Modellierung anderer Modelle nützlich sein sollen. Den Einsatz der eLoGo-Referenzmodelle im Kontext einer PRM-CIM-Transformation illustriert Abbildung 59.

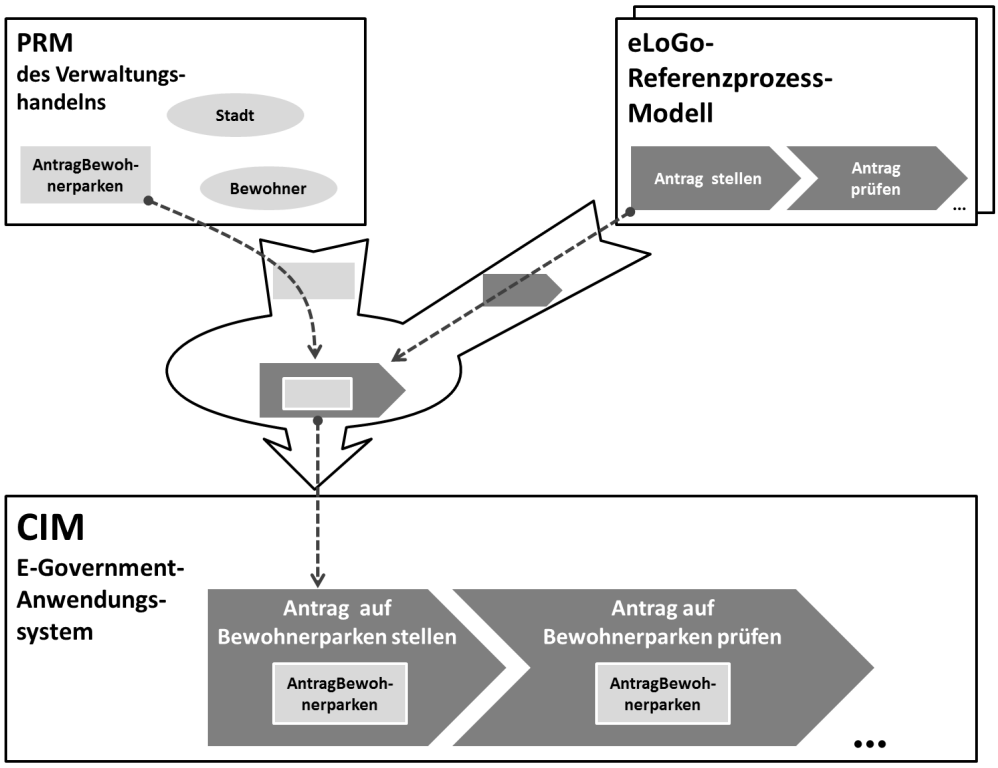


Abbildung 59 – Informelle Darstellung des Einsatzes der eLoGo-Referenzmodelle für die PRM-CIM-Transformation[363]

[362] Von den dokumentierten Ansätzen sind vor allem die eLoGo-Referenzmodelle geeignet, da sie für die Ebene des CIM ein Referenzprozessmodell und ein Referenzanforderungsmodell umfassen. Weiterhin sind die in diesen Modellen formalisierten Konzepte kompatibel zu den Inhalten der Ontologie des Verwaltungshandelns. Darüber hinaus ist die Konkretisierung der Referenzprozesssicht über das Referenzanforderungsmodell bei den eLoGo-Referenzmodellen einmalig ([Hinkelmann et al., 2005] S. 359).

[363] Notation in Anlehnung an [OMG MDA, 2003], S. 2-7 ff.

5.1.3 Integration der eLoGo-Referenzmodelle für E-Government

Die Transformation eines PRM in ein CIM wird im Rahmen dieser mittels des von der OMG dafür vorgesehenen QVT-Standards technisch umgesetzt (vgl. Abschnitt 2.2). Das PRM bildet in der Transformation das Ausgangsmodell und das CIM das Zielmodell. Für ihre Modellelemente werden in QVT Abbildungsregeln (Relationen) zwischen Ausgangs- und Zielelementen definiert. Das hierzu notwendige Vorgehen ist im QVT-Standard gut dokumentiert und vergleichsweise trivial. Im Gegensatz dazu muss geklärt werden, wie Referenzmodelle mit ihren Modellelementen in eine Transformation einfließen können, in der sie selbst nicht direkt transformiert werden. Auch hier könnte die erneute Orientierung am Plattformmodell hilfreich sein, doch Nolte weist dabei auf folgenden Widerspruch hin: Obwohl das Plattform-Konzept auch für die Transformation mittels QVT von großer Bedeutung ist, geht die QVT-Spezifikation hierauf nicht ein.[364] Dies kann ein Grund dafür sein, dass in den dokumentierten PIM-PSM-Transformationen der Einsatz eines Plattformmodells nur in Ansätzen zu finden ist. Solche Ansätze sind beispielsweise die von der MDA vorgeschlagenen plattformspezifischen UML-Profile für das Zielmodell.[365] Petrasch, Meimberg und Nolte sehen es darüber hinaus als nahe liegend an, dass Plattformmodelle zusätzlich zu UML-Profilen auch in Form von Metamodellen vorliegen und in dieser Form in die Transformation einfließen können.[366] Im Rahmen dieser Arbeit wird darüber hinaus die Meinung vertreten, dass Plattformmodelle nicht nur auf einer Meta-Ebene (wie UML-Profile oder Metamodelle), sondern auch als Modelle der gleichen Ebene wie die zu transformierenden Ausgangs- und Zielmodelle in die Transformation einfließen können.

Der Einsatz als Metamodell erscheint besonders dann geeignet, wenn dem Zielmodell dieses Metamodell zugrunde gelegt werden kann, so dass die Transformation die Lücke zwischen Ausgangs- und Zielmodell durch Instanziierung der Metamodellelemente eines Plattformmodells schließt. Entweder das Zielmodell verwendet dann ausschließlich Elemente dieses plattformbezogenen Metamodells oder (über einen geeigneten Erweiterungsmechanismus) Elemente von mehreren Metamodellen. Denkbar wäre hierfür beispielsweise, dass das als Metamodell bereitgestellte Plattformmodell auch die Standard-Elemente des UML-Metamodells importiert, so dass Elemente beider Metamodelle im Zielmodell verwendet werden können. Vergleichbar mit dem Einsatz eines Metamodells ist der Einsatz eines UML-Profils zur Repräsentation des Plattformmodells. In diesem Fall würde die Transformation die Lücke zwischen Ausgangs- und Zielmodell dadurch schließen, dass Elemente des Profils auf die Elemente des Zielmodells angewendet werden. Beispielsweise würde einer Klasse des Zielmodells ein Stereotyp zugewiesen, der die Semantik dieser Klasse weiter konkretisiert. Nachteilig kann sich bei diesem Verfahren auswirken, dass der UML-Erweiterungsmechanismus vergleichsweise eingeschränkte Möglichkeiten bietet. So können zwar Stereotypen definiert und mit Informationen (z.B. einer konkretisierenden Semantik) versehen werden, es ist aber nicht möglich, zwischen Stereotypen Beziehungen zu definieren. Deshalb ließe sich beim Einsatz eines UML-Profils beispielsweise nicht ausdrücken, dass ein Element des Zielmodells mit dem Stereotyp A aufgrund des ihm zugewiesenen Stereotyps mit einem anderen Element des Zielmodells, das den Stereotyp B hat, in Beziehung steht. Die Beziehung kann (nur) durch die Transformation explizit angelegt und ihr beispielsweise Stereotyp C des Profils zugewiesen werden. Der Einsatz des Plattformmodells als zusätzliches Ausgangsmodell für die Transformation schlägt einen bisher nicht beachteten Weg ein, der im Rahmen der MDA als

[364] Freundliche Auskunft von Siegfried Nolte, Verfasser von [Nolte, 2009] (per E-Mail am 13.02.2011).
[365] Vgl. u.a. [OMG MDA, 2003b], S. 2-6 und S. 4-2.
[366] Vgl. [Petrasch&Meimberg, 2006], S. 103, [Nolte, 2009], S. 141 f.

„Model Merge“ angedeutet wird.[367] Die Transformation würde dann geeignete Elemente beider Ausgangsmodelle kombinieren, um die Lücke zum Zielmodell zu schließen.[368]

Der MDA-Guide weist darauf hin, dass es für die PIM-PSM-Transformation einen prinzipiellen Zusammenhang zwischen dem Plattformmodell und der gewählten Transformationsvariante (Modelltypabbildung, Markierungstechnik, vorlagenbasierter Ansatz) gibt: „The platform model will determine the nature of the mapping.“[369] Basierend auf den vorangegangenen Überlegungen bedeutet dies, dass ein Plattformmodell in Form eines Metamodells oder UML-Profils gut geeignet ist, um in einer Modelltypabbildung oder der Markierungstechnik verwendet zu werden. Ein als zusätzliches Ausgangsmodell bereitgestelltes Plattformmodell scheint hingegen besser für einen vorlagenbasierten Transformationsansatz geeignet.

Überträgt man diese bisher auf das Plattformmodell bezogenen Überlegungen auf die PRM-CIM-Transformation von PRMs des Verwaltungshandeln in CIMs von E-Government-Anwendungssystemen, müssen die Auswirkungen der verschiedenen Bereitstellungsformen der eLoGo-Referenzmodelle auf den Ansatz untersucht werden.

Würden die eLoGo-Referenzmodelle im Rahmen des Ansatzes dieser Arbeit als Metamodelle bereitgestellt, wäre innerhalb der gewählten Eclipse-Werkzeugumgebung ein spezieller Modelleditor zu entwickeln, der die Verwendung der Instanzen von Metamodellelementen des eLoGo-Referenzmodells und der Instanzen von Metamodellelementen des UML-Metamodells (z.B. mit einer kombinierten Notation) als Modellelemente ermöglicht. Hiermit wäre nicht nur ein vergleichsweise hoher Entwicklungsaufwand verbunden, diese spezifische Umsetzung würde das Ziel des nahtlosen Anschlusses an Standardentwicklungswerkzeuge der Softwareentwicklung gefährden. Um dieses Ziel zu erreichen, ist eine stärkere Orientierung an der UML besser, so dass die für die Modellierung eingesetzten Werkzeuge (z.B. das hier verwendete TOPCASED) die Zielmodelle direkt verarbeiten können. Die Bereitstellung der eLoGo-Referenzmodelle als UML-Profil ermöglicht die Verwendung von Standardwerkzeugen, weil anstelle einer Metamodellerweiterung der Erweiterungsmechanismus der UML verwendet werden kann. Er wird von allen gängigen Standardwerkzeugen unterstützt. Es lassen sich durch den Einsatz eines UML-Profils die Elemente des Zielmodells mit einer konkretisierten Semantik versehen, indem die Informationen aus dem Profil zum Schließen der Lücke zwischen Ausgangs- und Zielmodell verwendet werden. Das Zielmodell umfasst dann Instanzen des UML-Metamodells, die mit Stereotypen versehen sind und ggf. auch über *Tagged Values* verfügen. Voraussetzung hierfür ist, dass das Zielmodell auf dem UML-Metamodell oder einem vergleichbaren Metamodell basiert, das einen Erweiterungsmechanismus unterstützt. Sollen Zielmodelle erzeugt werden, deren Metamodelle keinen Erweiterungsmechanismus unterstützen, können die Referenzmodelle in Form von zusätzlichen Ausgangsmodellen in die Transformation einfließen. Innerhalb der Transformation werden dann die Informationen aus dem eigentlichen Ausgangsmodell mit den Informationen aus den Referenzmodellen kombiniert, um die semantische Lücke bei der Transformation zum Zielmodell zu schließen. Die Referenzmodelle bilden hierbei eine Vorlage (im Sinne eines vorlagenbasierten Transformationsansatzes), die durch Informationen aus dem PRM zu füllen ist. Zusammenfassend ergibt sich für den Ansatz dieser Arbeit, dass die Bereitstellung der eLoGo-Referenzmodelle in Form von UML-Profilen im Vergleich zu Metamodellen besser geeignet erscheint. Die Bereitstellung als zusätzliches Ausgangsmodell ist eine weitere Alternative.

[367] Vgl. [OMG MDA, 2003b], S. 3-12.

[368] Wenn die Transformation die Konsistenz der Ausgangs- und Zielmodelle (in beide Richtungen) sicherstellen soll, muss darauf geachtet werden, dass das Plattformmodell, obwohl es ein weiteres Ausgangsmodell ist, nicht modifiziert werden darf, da die Plattform als gegeben anzusehen ist.

[369] [OMG MDA, 2003b], S. 3-2.

5.1.3.1 Referenzprozessmodell als weiteres Ausgangsmodell

Für die Bereitstellung des Teils der Referenzmodelle, der das eLoGo-Referenzprozessmodell umfasst, ergibt sich die Schwierigkeit, dass die BPMN in ihrer aktuellen Version 2 zwar einen mit dem UML-Profil vergleichbaren Erweiterungsmechanismus vorsieht, sich aber die Entwicklung eines BPMN 2-Werkzeugs auf Basis von Eclipse und Ecore zum Zeitpunkt der Erstellung dieser Arbeit noch in der Vorbereitungsphase befindet.[370] Das verfügbare BPMN-Werkzeug für Eclipse unterstützt nur die Version 1.0, die keinen Erweiterungsmechanismus kennt und ausgehend vom Standard kein MOF-konformes Metamodell besitzt. Allerdings gibt es eine Ecore-Repräsentation des Metamodells, die als Teil des Werkzeugs ausgeliefert wird. Für die Verwendung der BPMN im Rahmen dieser Arbeit ergeben sich somit drei Möglichkeiten.[371] Zum einen kann die BPMN in Version 1 verwendet und auf die Nutzung eines Erweiterungsmechanismus verzichtet werden. Zum anderen könnten die vorliegende Werkzeugunterstützung und das damit verbundene Metamodell der BPMN 1 um zusätzliche Modellelemente erweitert werden. Die dritte Möglichkeit ist die Verwendung der BPMN 2, indem der aktuell laufenden Entwicklungsarbeit im Rahmen dieser Arbeit durch Implementierung eines eigenen BPMN 2-Editors vorgegriffen wird. Die beiden Möglichkeiten 2 und 3 sind mit erheblichem Entwicklungsaufwand verbunden, der vor dem Hintergrund der zu erwartenden neuen BPMN 2-Werkzeugunterstützung und im Verhältnis zum zusätzlichen Erkenntnisgewinn für die Verfolgbarkeit bewertet werden muss. Weil neben der Bereitstellung der Referenzmodelle auf einer Metaebene auch die Verwendung als zusätzliches Ausgangsmodell eine Alternative ist, ist die BPMN 1 für diese Arbeit prinzipiell ausreichend. Dadurch entfällt die Notwendigkeit einen Erweiterungsmechanismus zu nutzen und es kann die vorhandene Werkzeugunterstützung verwendet werden. Gleichzeitig ergibt sich der Vorteil, dass die gewonnenen Erkenntnisse und erzielten Ergebnisse mit dem Einsatz von UML-Profilen in Abschnitt 5.4 verglichen werden können, weil diese für die eLoGo-Referenzanforderungsmodelle verwendet werden (siehe unten).

Das eLoGo-Referenzprozessmodell wurde im Rahmen des eLoGo-Projektes an der Universität Potsdam ursprünglich nicht in der damals noch wenig verbreiteten BPMN, sondern in einer an die EPK-Notation angelehnten Notation erarbeitet, um einen breiten Adressatenkreis anzusprechen.[372] In einem vorbereitenden Schritt muss das Modell deshalb in das BPMN-Format überführt werden. Um als zusätzliches Ausgangsmodell innerhalb der Transformation verarbeitet werden zu können, muss es auf einem definierten Metamodell basieren. Hier wird das BPMN 1-Metamodell in der durch die Werkzeugunterstützung bereitgestellten Ecore-Repräsentation verwendet.

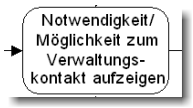
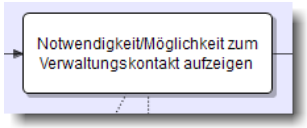
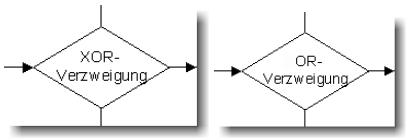
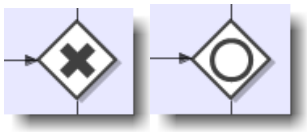
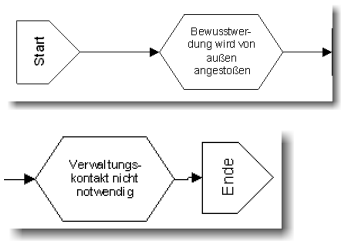
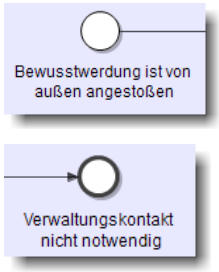
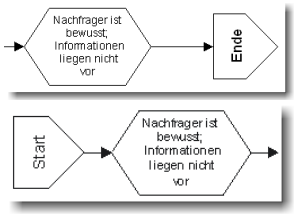
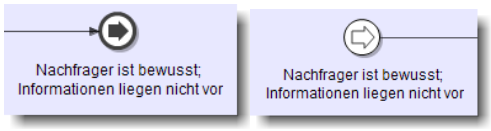
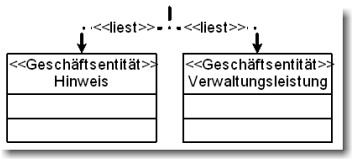
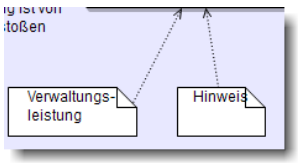
Zur konkreten Überführung der EPK-Modelle in BPMN-Modelle gibt es verschiedene Ansätze. Decker et al. geben einen zusammenfassenden Überblick über die vorhandenen Ansätze in Verbindung mit einem eigenen Vorschlag für eine pragmatische

[370] Zum Zeitpunkt der Erstellung dieser Arbeit sind verschiedene Werkzeuge verfügbar, die BPMN 2-Diagramme unterstützen (z.B. BOC ADONIS, MID Innovator for Business Analysts, BizAgi Process Modeler) und diese auch in XML-basierten Austauschformaten bereitstellen. Für den Ansatz dieser Arbeit ist die Integration in Transformationen mit QVT auf Basis der hier verwendeten Eclipse-basierten Werkzeugkombination von Bedeutung. Deshalb wird ein Kompromiss eingegangen und die BPMN in Version 1 verwendet, da diese (wie unten beschrieben) prinzipiell ausreichend und der weitergehende Funktionsumfang fortgeschrittener BPMN 2-Werkzeuge nicht erforderlich ist.

[371] Vor dem Hintergrund des bei der OMG zum Zeitpunkt der Erstellung dieser Arbeit in Vorbereitung befindlichen UML-Profil für BPMN könnten hier UML-Aktivitätsdiagramme als weitere Möglichkeit gesehen werden (vgl. [OMG BPMN, 2010b]). Aufgrund der zunehmenden praktischen Bedeutung der BPMN verwendet diese Arbeit bewusst die BPMN mit ihren grafischen Notationselementen, da sich in den öffentlich verfügbaren Informationen zum Stand des Standardisierungsprozesses bis zum Zeitpunkt der Fertigstellung dieser Arbeit noch keine Ergebnisse abzeichnen (vgl. http://www.omg.org/schedule/, letzter Aufruf: 21.08.2011).

[372] Die eLoGo-Referenzmodelle wurden im Jahr 2003 entwickelt. Die BPMN war seit 2002 zwar verfügbar, hatte allerdings noch keine große Verbreitung gefunden. Sie wurde erst im Jahr 2005 durch die OMG in den Standardisierungsprozess übernommen.

Migrationsstrategie.[373] Ihr pragmatischer Ansatz wird auch hier für die Migration des eLoGo-Referenzprozessmodells verwendet. Die dazu angewendeten Migrationsregeln sind in Tabelle 32 anhand von Beispielen dargestellt.

Modellelement in EPK	Modellelement in BPMN
 Aktivität	 Task
 Exklusive Oder-Verzweigung und Oder-Verzweigung (analog Vereinigungen)	 Exclusive Data-based Gateway und Inclusive Data-based Gateway (analog Vereinigungen)
 Start-Ereignis und Ende-Ereignis	 Start Event und End Event
 Ende-Ereignis und korrespondierendes Startereignis	 Linked End Event und Linked Start Event
 Geschäftsentität mit Angabe des Zugriffs an der Assoziation zur Aktivität	 Data Object mit gerichteter Assoziation zum Task (hier per Konvention: ausgehend = lesender Zugriff, eingehend = schreibender Zugriff, beide = lesender und schreibender Zugriff)

[373] [Decker et al., 2009]

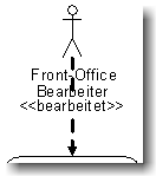
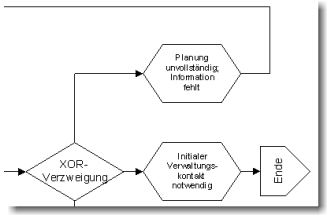
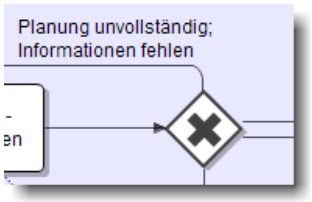
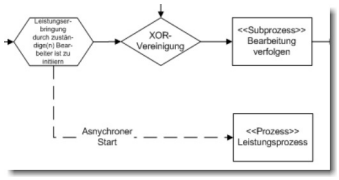
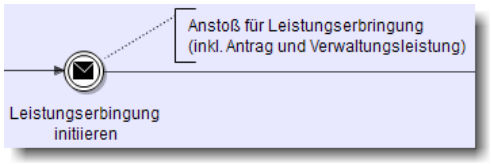
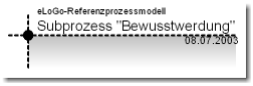
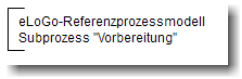
	
Rolle	Lane innerhalb eines Pools
Ohne Symbol Unterscheidung der übergeordneten Prozesse in Anliegenprozess und Leistungsprozess	 Pool
 Zwischenereignis	 Flow Connector mit Bezeichnung
 Asynchroner Aufruf des Leistungsprozesses	 Message Intermediate Event mit Text-Annotation
 Notiz	 Text-Annotation

Tabelle 32 – Illustration der Migrationsregeln für Modellelemente der EPK-artigen Notation in BPMN[374]

Die Migration der einzelnen Modellelemente entsprechend der vorgestellten Migrationsregeln stellt keine besondere Herausforderung dar. Vergleichsweise aufwändig ist es, den Bezug zu den ursprünglichen Modellen, zu ihrer Beschreibung und Dokumentation herzustellen. Deshalb wurden die einzelnen eLoGo-Subprozesse jeweils als eigenes BPMN-Diagramm modelliert und dabei weitgehend an die graphische Anordnung der EPK-Modelle angelehnt.[375] Einen Überblick über den im Ergebnis der Migration erstellten Gesamtablauf in BPMN gibt Abbildung 60.

[374] In Anlehnung an [Decker et al., 2009].

[375] Aufgrund der Beschränkungen des Werkzeugs ist eine Verknüpfung dieser BPMN-Diagramme in einem übergeordneten Gesamtablauf nicht möglich. Zwar lassen sich Subprozesse als Elemente

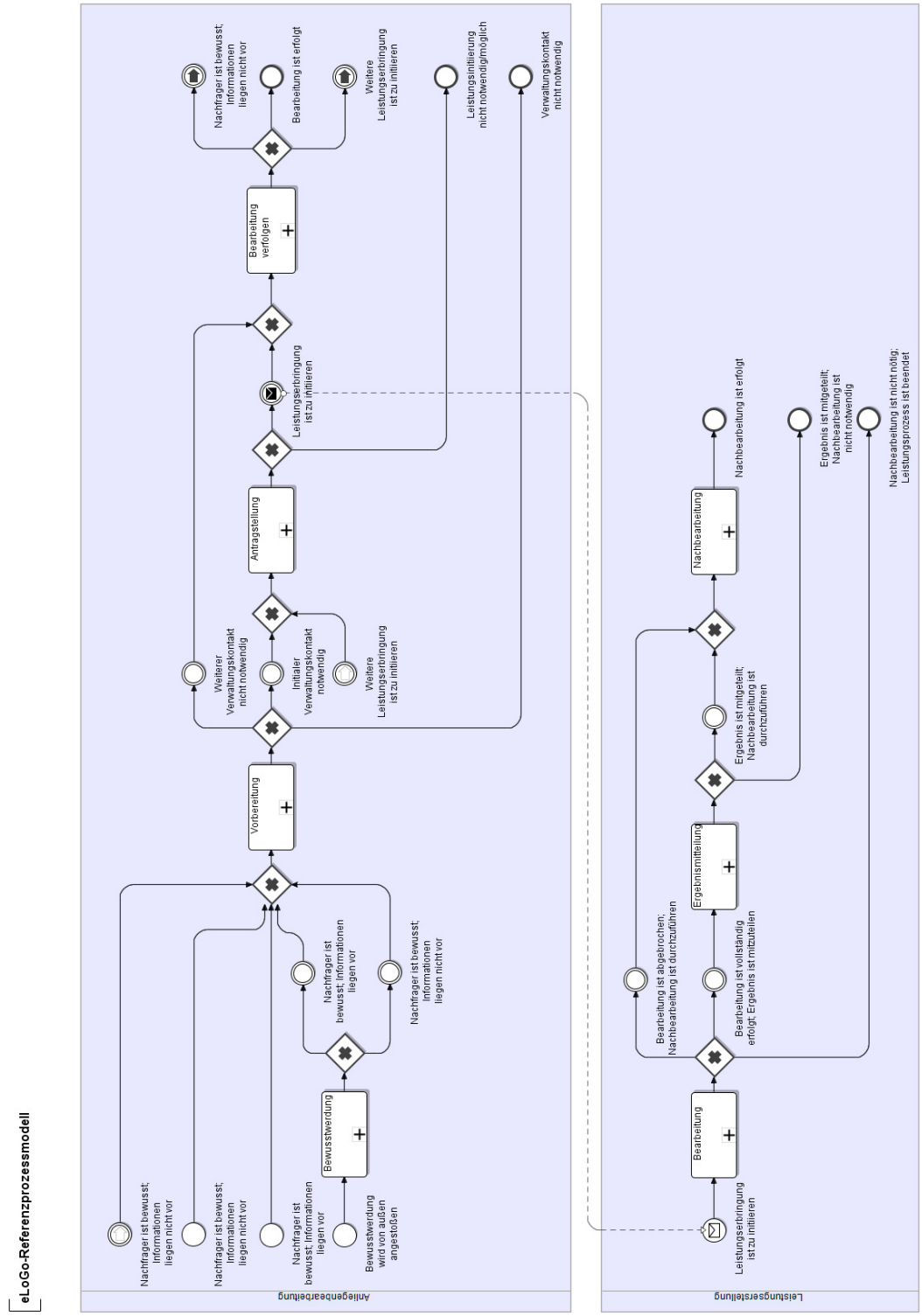


Abbildung 60 – Gesamtablauf der eLoGo-Referenzmodelle in BPMN

Der Ablauf des eLoGo-Anliegenprozesses und -Leistungsprozesses ist in der Gesamtdarstellung visuell zu erkennen. Darüber hinaus ergibt sich ihr Ablauf aus der Aufrufreihenfolge der zusammenhängenden Subprozesse, die über BPMN-Zwischenereignisse des Typs `Link Event` verbunden sind. Um die Zugehörigkeit eines

darstellen, jedoch nicht mit den in separaten Diagrammen modellierten Subprozessen verknüpfen. Dies muss bei den Transformationsprozessen beachtet werden.

Subprozesses zu einem eLoGo-Anliegen- oder Leistungsprozess auszudrücken, wurden in den BPMN-Diagrammen der Subprozesse Pools und Schwimmbahnen (Lanes) verwendet. Der Pool `Anliegenbearbeitung` umfasst alle Tasks, die als Aktivitäten im Rahmen des eLoGo-Anliegenprozesses von der Rolle `Nachfrager` oder der Rolle `Bearbeiter Front Office` bearbeitet wurden. Entsprechend sind in dem Pool `Leistungserstellung` Tasks enthalten, die vom Bearbeiter Back Office bzw. Bearbeiter Front Office im Rahmen des Leistungsprozesses bearbeitet wurden.

Ein Beispiel für das Ergebnis der Migration wird nachfolgend anhand des Subprozesses „Bewusstwerdung“ dargestellt.[376] Der Prozessgraph (Abbildung 61) zeigt die Anordnung der Modellelemente, die soweit möglich an die ursprüngliche Darstellung angelehnt ist, um die Wiedererkennung zu unterstützen (vgl. Abbildung 15 auf Seite 49).

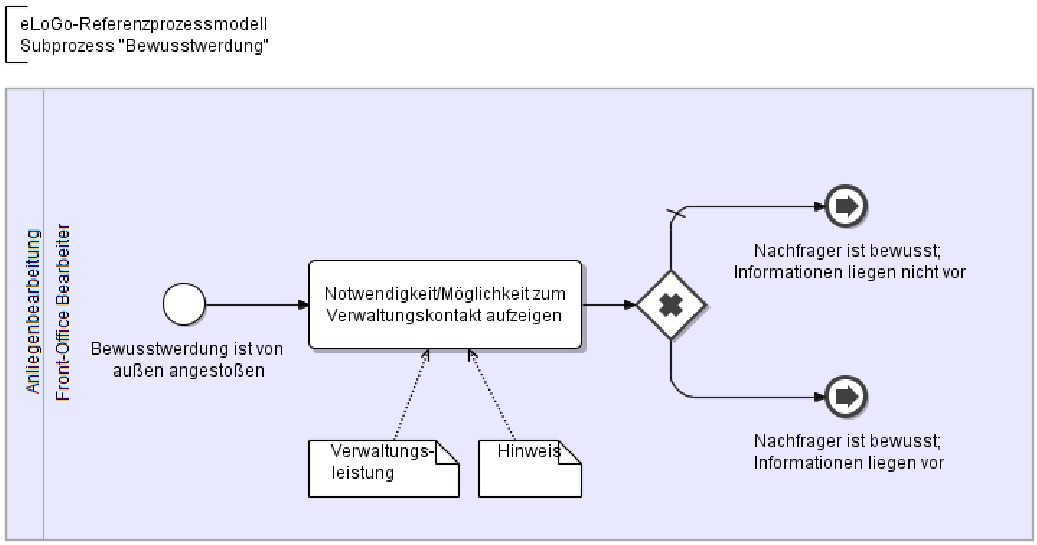


Abbildung 61 – BPMN-Notation der eLoGo-Referenzmodelle am Beispiel des Subprozesses „Bewusstwerdung“

Außer in der graphischen Darstellung liegen die Prozessmodelle nach der Überführung in BPMN auch in einem XML-Format vor. Dieses Format ist ein Mapping des im Werkzeug implementierten Metamodells auf XML-Elemente.[377] Im Listing 29 ist exemplarisch der Subprozess „Bewusstwerdung“ aus der Abbildung 61 im XML-Format dargestellt. In Zeile 2 beginnt das BPMN-Diagramm als Wurzelknoten des XML-Dokuments. Unterhalb dieses Wurzelknotens sind der Pool „Anliegenbearbeitung“ (Zeile 5) und die Text-Annotation mit dem Namen des Prozesses (Zeile 3) enthalten. Das Pool-Element enthält seinerseits die Artefakte (z.B. BPMN-DataObjects) in den Zeilen 7 bis 12, die BPMN-Aktivitäten in den Zeilen 14 bis 18 und die verbindenden Sequenzflüsse (Zeilen 20-23). Ebenfalls Bestandteil des Pools ist die Schwimmbahn „Front-Office Bearbeiter“, die in ihrer `activities`-Eigenschaft auf die in ihr enthaltenen Tasks referenziert (Zeile 25).

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <bpmn:BpmnDiagram xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
      xmlns:bpmn="http://stp.eclipse.org/bpmn" xmi:id="_a1" iD="_a1">
(3)     <artifacts xmi:type="bpmn:TextAnnotation" xmi:id="_a2" iD="_a3"
      name="eLoGo-Referenzprozessmodell&#xD;&#xA;Subprozess
      &quot;Bewusstwerdung&quot;" ncname="eLoGoNotiz"/>
(4)
(5)     <pools xmi:type="bpmn:Pool" xmi:id="_a4" iD="_a5"
      name="Anliegenbearbeitung">
(6)
```

[376] Die Modelle der weiteren Subprozesse sind im Anhang E.1 dargestellt.

[377] Der Standard der BPMN in den Version 1.0, 1.1 und 1.2 umfasst keine Definition eines Austauschformates (vgl. [BPMI, 2004], S. 20; [OMG BPMN, 2006], S. 2; [OMG BPMN, 2008], S. 1; [OMG BPMN, 2009], S. 1), weshalb hier mit dem werkzeugspezifischen XML-Format gearbeitet wird.

```
(7)       <artifacts xmi:type="bpmn:DataObject" xmi:id="_a6" iD="_a7"
      name="Hinweis" ncname="Hinweis">
(8)         <associations xmi:type="bpmn:Association" xmi:id="_a8"
      direction="From"/>
(9)       </artifacts>
(10)      <artifacts xmi:type="bpmn:DataObject" xmi:id="_a9" iD="_b1"
      name="Verwaltungs-&#xD;&#xA;leistung" ncname="Verwaltungsleistung">
(11)        <associations xmi:type="bpmn:Association" xmi:id="_b2"
      direction="From"/>
(12)      </artifacts>
(13)
(14)      <vertices xmi:type="bpmn:Activity" xmi:id="_b3" iD="_b4"
      associations="_a8 _b2" outgoingEdges="_b5" incomingEdges="_b6"
      name="Notwendigkeit/Möglichkeit zum Verwaltungskontakt aufzeigen"
      ncname="BewusstwerdungAufzeigen" lanes="_b7" activityType="Task"/>
(15)      <vertices xmi:type="bpmn:Activity" xmi:id="_d1" iD="_d2"
      outgoingEdges="_b6" name="Bewusstwerdung ist von&#xD;&#xA; außen angestoßen"
      lanes="_b7" activityType="EventStartEmpty"/>
(16)      <vertices xmi:type="bpmn:Activity" xmi:id="_b8" iD="_b9"
      outgoingEdges="_c1 _c2" incomingEdges="_b5" lanes="_b7"
      activityType="GatewayDataBasedExclusive"/>
(17)      <vertices xmi:type="bpmn:Activity" xmi:id="_c3" iD="_c4"
      incomingEdges="_c1" name="Nachfrager ist bewusst; &#xD;&#xA;Informationen
      liegen nicht vor" ncname="Informationen" lanes="_b7"
      activityType="EventEndLink"/>
(18)      <vertices xmi:type="bpmn:Activity" xmi:id="_c5" iD="_c6"
      incomingEdges="_c2" name="Nachfrager ist bewusst; &#xD;&#xA;Informationen
      liegen vor" ncname="KeineInformationen" lanes="_b7"
      activityType="EventEndLink"/>
(19)
(20)      <sequenceEdges xmi:type="bpmn:SequenceEdge" xmi:id="_b6" iD="_c7"/>
(21)      <sequenceEdges xmi:type="bpmn:SequenceEdge" xmi:id="_b5" iD="_c8"/>
(22)      <sequenceEdges xmi:type="bpmn:SequenceEdge" xmi:id="_c1" iD="_c9"
      conditionType="Default" isDefault="true"/>
(23)      <sequenceEdges xmi:type="bpmn:SequenceEdge" xmi:id="_c2" iD="_d3"/>
(24)
(25)      <lanes xmi:type="bpmn:Lane" xmi:id="_b7" iD="_d4" name="Front-Office
      Bearbeiter" activities="_b8 _d1 _c3 _c5 _b3"/>
(26)
(27)    </pools>
(28)
(29)  </bpmn:BpmnDiagram>
```

Listing 29 – BPMN-Subprozess „Bewusstwerdung" im XML-Format

Bestandteil der BPMN-Modelle ist auch die Dokumentation der Referenzmodelle, wie sie in [Horn&Off, 2004b] in Form von Fließtext und Tabellen dargestellt ist. Diese Dokumentation wurde als Wert der Eigenschaft `documentation` übernommen. Exemplarisch ist die Dokumentation für den Task „Notwendigkeit/Möglichkeit zum Verwaltungskontakt aufzeigen" in der Werkzeugdarstellung und im XMI-Format nachfolgend dargestellt (Abbildung 62 und Listing 30, Zeile 8).

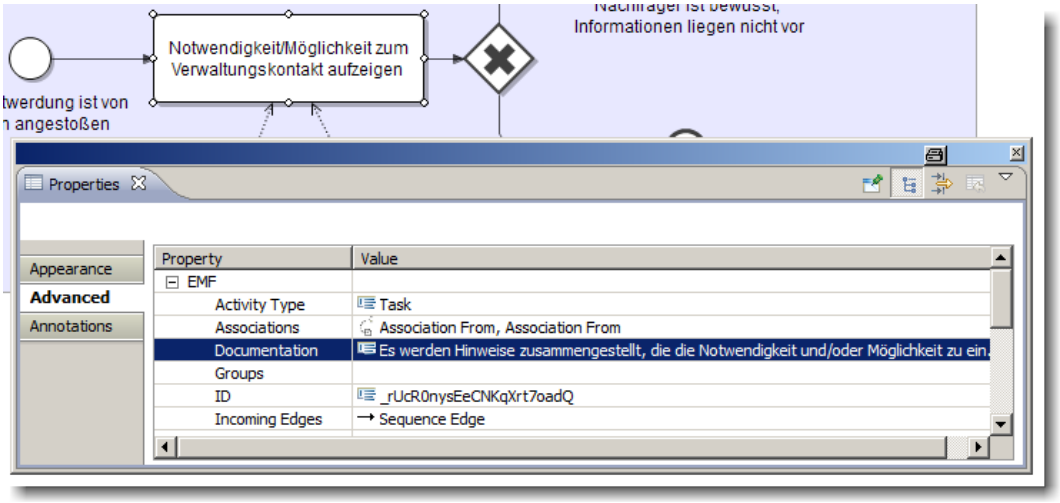


Abbildung 62 – Dokumentation zum Task „Notwendigkeit/Möglichkeit zum Verwaltungskontakt aufzeigen" im Properties-Fenster

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <bpmn:BpmnDiagram xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
      xmlns:bpmn="http://stp.eclipse.org/bpmn" xmi:id="_a1" iD="_a2"
      documentation="Der Subprozess ... ">
(3)    <!-- ... -->
(4)    <pools xmi:type="bpmn:Pool" xmi:id="_a3" iD="_a4"
      name="Anliegenbearbeitung">
(5)      <!-- ... -->
(6)      <vertices xmi:type="bpmn:Activity" xmi:id="_a5" iD="_a6"
      associations="_a7 _a8"
(7)          outgoingEdges="_a9" incomingEdges="_b1"
(8)          documentation="Es werden Hinweise zusammengestellt, die die
      Notwendigkeit und/oder
             Möglichkeit zu einem Verwaltungskontakt aufzeigen. Der Bearbeiter
      dieser Aktivität
             informiert den Nachfrager über die Hinweise und ggf. über weitere
      Informationen,
             die in Zusammenhang mit den Hinweisen stehen (z.B.: über mögliche
             Verwaltungsleistungen zur aktuellen Lebenslage). Der Nachfrager nimmt
      die ihm
             gegebenen Hinweise und Informationen zur Kenntnis."
(9)          name="Notwendigkeit/Möglichkeit zum&#xD;&#xA;Verwaltungskontakt
      aufzeigen"
(10)         ncname="BewusstwerdungAufzeigen" lanes="_b3" activityType="Task"/>
(11)     <!-- ... -->
(12)     <lanes xmi:type="bpmn:Lane" xmi:id="_b2" iD="_b3" documentation="Front-
      Office Bearbeiter sind Personen, die ... " activities="_a6 ..."/>
(13)     <!-- ... -->
(14)    </pools>
(15)  </bpmn:BpmnDiagram>
```

Listing 30 – Dokumentation zum Task „Notwendigkeit/Möglichkeit zum Verwaltungskontakt aufzeigen" im XML-Format

Eine Besonderheit ergibt sich für die Bezeichnung der grafischen Elemente. In der Bezeichnung einiger Elemente wurden bedarfsgerecht Zeilenumbruchzeichen oder Bindestriche ergänzt, um die Lesbarkeit der Grafik zu verbessern. Diese zusätzlichen Zeichen sind dadurch auch im Namen der Elemente enthalten (z.B. Zeilenumbruch `
` in Listing 30, Zeile 9). Um innerhalb der Abbildungsregeln die Elemente des Modells unabhängig hiervon identifizieren zu können, ist deshalb die Eigenschaft `name` weniger gut geeignet. Hier wird deshalb die Eigenschaft `ncname` verwendet, der eine eindeutige Kurzbezeichnung der Aktivität (Listing 30, Zeile 10) zugewiesen wird, die unabhängig von der grafischen Darstellung ist. Im Ergebnis der Migration und entsprechend der dargestellten Überlegungen liegen die eLoGo-Referenzprozessmodelle in Form vom BPMN-Diagrammen in Eclipse vor, die auf dem Ecore-Metamodell des BPMN 1-Modellierungswerkzeugs basieren (Abbildung 63). Ihre vollständige grafische Darstellung zeigt Anhang E.1.

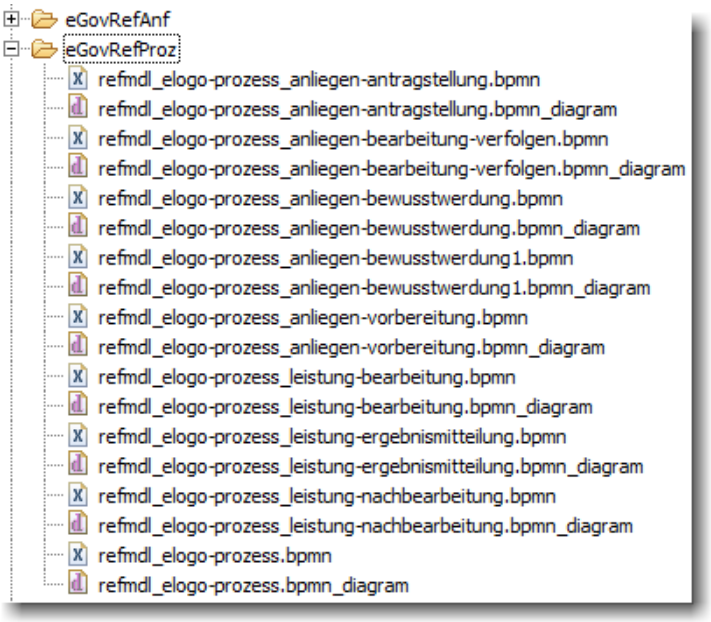


Abbildung 63 – eLoGo-Referenzprozessmodell im BPMN-Format in Eclipse

5.1.3.2 Referenzanforderungsmodell als UML-Profil

Der Teil der eLoGo-Referenzmodelle, der das Referenzanforderungsmodell umfasst, kann als UML-Profil für die Nutzung in den Transformationen bereitgestellt werden. Weil das Zielmodell die UML und deren Metamodell verwendet, ist der Einsatz des UML-Erweiterungsmechanismus in Form eines UML-Profils möglich. Das Profil umfasst Stereotypen, die bei der Transformation eines PRM des Verwaltungshandelns in ein CIM eines E-Government-Anwendungssystem den Elementen des CIM zugewiesen werden. Dadurch wird die Semantik der CIM-Elemente soweit konkretisiert, dass das entstehende Modell eine Konkretisierung des entsprechenden Referenzmodells darstellt.

Das eLoGo-Referenzanforderungsmodell wurde im Rahmen des eLoGo-Projektes in Form von Anwendungsfalldiagrammen und Domänenobjektmodellen (Klassendiagrammen) sowie erläuterndem Text (strukturierter Fließtext, Tabellen) erstellt. Um dieses Modell in ein Profil zu überführen, sind Migrationsschritte notwendig. Im Rahmen dieser Arbeit wurden die Anwendungsfallmodelle und die Domänenobjektmodelle hierzu analysiert. Es wurden dann abstrakte Stereotypen als Erweiterung der Metaklassen `UseCase`, `Actor` und `Class` definiert. Sie sind die Oberklasse `GovObjUseCase`, `GovObjActor` und `GovObjClass`[378] für weitere Spezialisierungen durch die im Profil zu nutzenden Stereotypen. Den Oberklassen wurden die Attribute `Arbeitsnotiz`, `Beschreibung` und `Quelle` zugeordnet, die zu den Data Properties von `DomainConcept` der Ontologie des Verwaltungshandelns (vgl. Abschnitt 3.1.4) korrespondieren. Dadurch können Informationen aus den Instanzen des PRM in Tagged Values (siehe unten) des Stereotyps eines Modellelements im CIM übernommen werden.

Für jeden Anwendungsfall und jeden Akteur der Anwendungsfallmodelle wurde ein Stereotyp als Spezialisierung des `GovObjUseCase` bzw. `GovObjActor` im Profil definiert. Zusätzlich wurde für jede Domänenklasse, die im Klassendiagramm zum jeweiligen Anwendungsfall dargestellt ist, ein Stereotyp als Spezialisierung des `GovObjClass` definiert. Die Abbildung 64 zeigt einen Ausschnitt des im Rahmen dieser Arbeit auf Basis der eLoGo-Referenzmodelle erstellten UML-Profils für die Modelle der Anforderungsanalyse. Das vollständige UML-Profil ist im Anhang E.2 dargestellt.

[378] Das Präfix „GovObj“ (für Government Objects) soll den Bezug der Elemente zum Verwaltungshandeln ausdrücken.

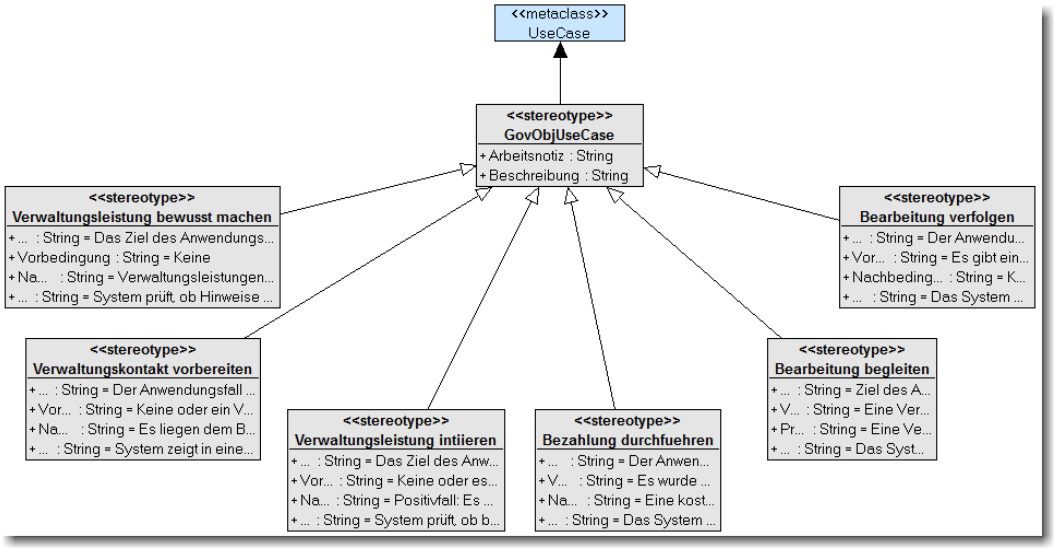


Abbildung 64 – Auszug aus den Elementen zum Subsystem „Anliegen" des UML-Profils für das eLoGo-Referenzanforderungsmodell

Für diese spezialisierten Stereotypen wurden abhängig von der Oberklasse des Elements Attribute definiert. Die Vorbelegung dieser Attribute (Default Value) wurde aus der Beschreibung des Elements im Referenzanforderungsmodell übernommen, wie sie in [Horn&Off, 2004b] in Form von Fließtext und Tabellen enthalten ist. Für Anwendungsfälle wurden auf diese Weise beispielsweise Ziel, Vorbedingung, Nachbedingung und eine Beschreibung ihres Ablaufs definiert. Für Akteure wurden das vom Akteur verfolgte Ziel und sein Profil (z.B. ob regelmäßiger oder gelegentlicher Benutzer) definiert.[379] Weil ihre Vorbelegungswerte unterschiedlich sind, war es nicht möglich, diese Attribute in der Oberklasse zu definieren. Die Abbildung 65 zeigt den Stereotyp für den Anwendungsfall `Verwaltungsleistung bewusst machen` und dessen Attribut `Ziel`. Dargestellt ist die Vorbelegung des Attributes mit dem Text „Das Ziel des Anwendungsfalls ist es, den Benutzer (als potenziellen Nachfrager) einer Verwaltungsleistung ...", der aus der Anwendungsfallbeschreibung übernommen wurde (vgl. Tabelle 3 auf Seite 51).

[379] Während der Aufbereitung als Profil mussten geringfügige Änderungen an den Inhalten der Referenzmodelle vorgenommen werden. Beispielsweise können die Bezeichnungen von Stereotypen aufgrund von Werkzeugbeschränkungen eine bestimmte Zeichenzahl nicht überschreiten. Deshalb mussten die Namen der Elemente gekürzt werden, bevor sie in das Profil übernommen werden konnten. Die Interaktionsbeschreibung zum Anwendungsfall musste aus einer Tabellenform in die Form eines Fließtextes überführt werden, um als Default-Value für Properties verwendet werden zu können. Alternativ hätte ihre Modellierung als Aktivitätendiagramm erfolgen können. Dadurch hätte sich der Umfang der Transformation weiter erhöht, da ein zusätzliches Modell zu berücksichtigen wäre. Für die Konstruktion von Verfolgbarkeitsbeziehungen ergäbe sich jedoch kein Vorteil, weshalb die Umsetzung als Fließtext erfolgte. Darüber hinaus wurden kleinere Anpassungen der Terminologie vorgenommen, um die Inhalte der Referenzmodelle besser in die Ergebnisse dieser Arbeit einbinden zu können.

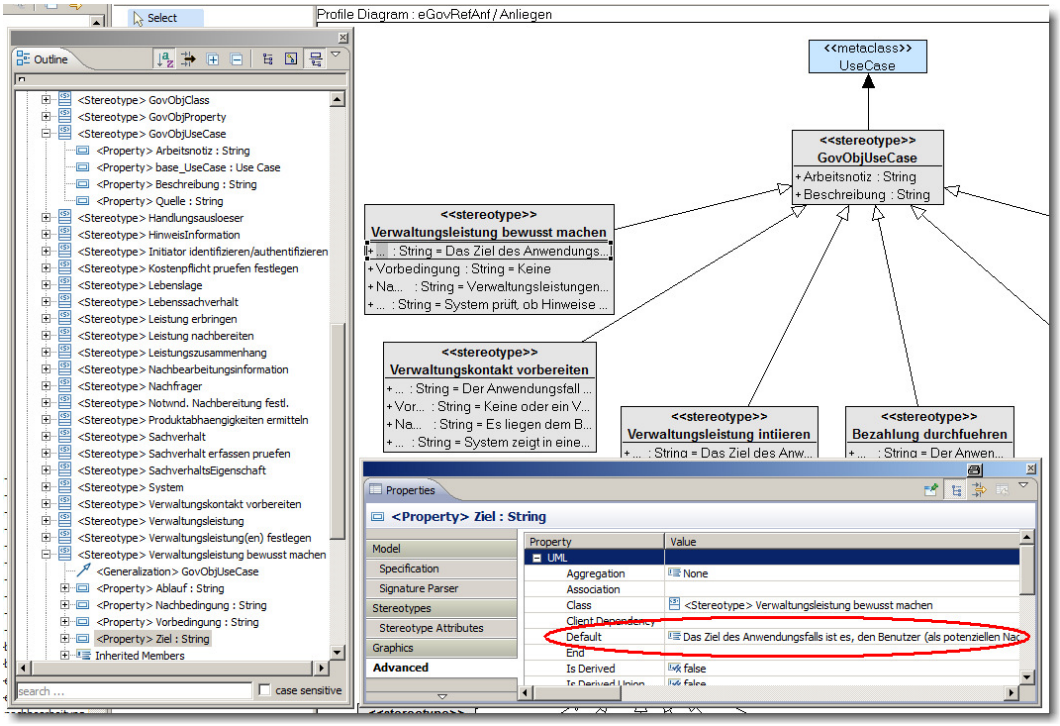


Abbildung 65 – Definition von Default-Werten für Tag Definitions basierend auf der Beschreibung des eLoGo-Modells

Die Attribute der Stereotypen bilden bei der Anwendung des jeweiligen Stereotyps auf ein Element im CIM so genannte T*ag Definitions*, die mit Werteausprägungen in Form von *Tagged Values* versehen werden können. Dadurch kann in der Transformation durch Anwendung des Stereotyps die Semantik des jeweiligen Zielelementes konkretisiert werden, so dass das resultierende Zielmodell eine Konkretisierung des Referenzanforderungsmodells ist.

Um das UML-Profil technisch in Transformationen nutzen zu können, wurde es (analog zur Beschreibung in Abschnitt 4.1.3) als Eclipse-Plugin (`eGovRefAnf*.jar`) generiert. Dadurch kann es insbesondere innerhalb des Werkzeugs mediniQVT als `eGovRefAnf.ecore` registriert und genutzt werden (Abbildung 66).

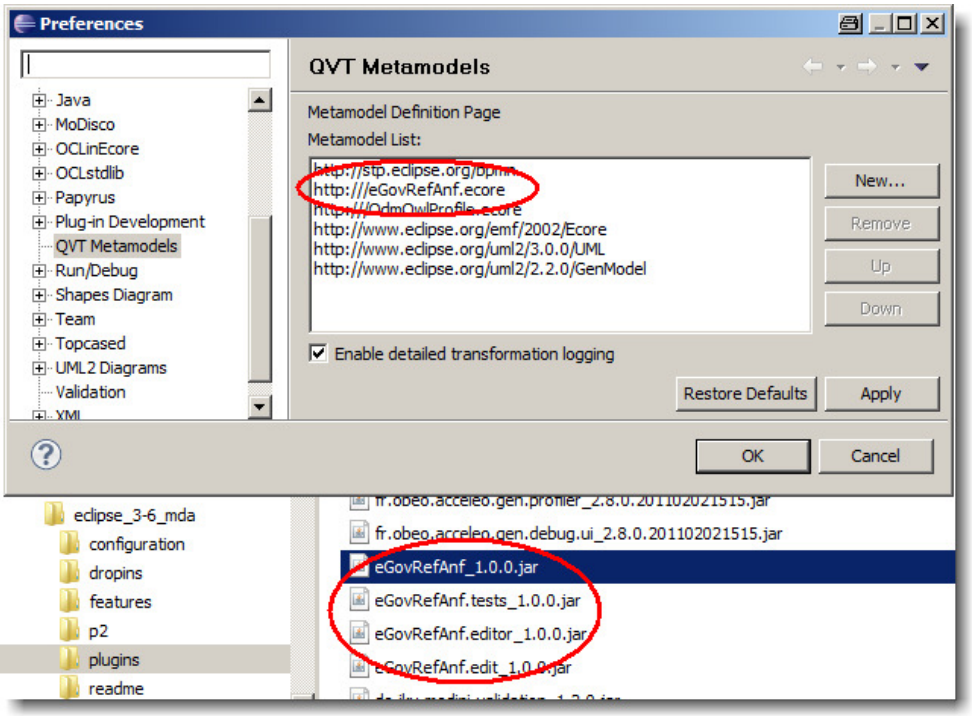


Abbildung 66 – Im Werkzeug mediniQVT u.a. registriertes Metamodell für die eLoGo-Referenzmodelle

5.1.4 Integration nicht-funktionaler Anforderungen

Von den zahlreichen praxisbezogenen und wissenschaftlichen Ansätzen[380], die seit Jahren beschreiben, wie nicht-funktionale Anforderungen in Modelle einer Anforderungsspezifikation integriert werden können, hebt sich der in US-Patent Anmeldung Nummer 2011/0113402 A1 und der EU-Patent Anmeldung Nr. 09175788.0 dokumentierte Ansatz ab. Das im Mai 2011 beantragte Patent soll zukünftig unter anderem die Rechte an der Verwendung eines UML-Profils zur Modellierung nicht-funktionaler Anforderungen in Anwendungsfalldiagrammen und dessen Implementierung auf Basis der Eclipse-Plattform schützen. Die nachfolgende Abbildung 67 zeigt einen Ausschnitt aus dieser Erfindung. Im oberen Teil der Abbildung ist ein Anwendungsfalldiagramm dargestellt, das um Notizen bzw. Constraints mit speziellen Stereotypen zu nicht-funktionalen Aspekten und um einen Testfall der entsprechenden nicht-funktionalen Anforderung mit einem Stereotyp erweitert wurde. Der untere Teil der Abbildung zeigt die durch das Patent zu schützende Implementierung dieses Ansatzes durch ein UML-Profil (in der Abbildung mit Ziffer 400 bezeichnet) und die zur Modellierung mit dem Profil notwendigen weiteren Plug-in-Komponenten (Ziffer 402) innerhalb der Eclipse-Plattform. Unabhängig von einer Einschätzung zur tatsächlichen Neuartigkeit dieses Ansatzes kann er bezogen auf die Nutzung von UML-Profilen in Eclipse doch mindestens als Stand der Technik gelten.

[380] Ansätze zur Spezifikation von nicht-funktionalen Anforderungen mit UML sind u.a. in [Cysneiros&Leite, 2001], [Tonu, 2006] und [Zuh&Gorton, 2007] dokumentiert. Für einen Überblick über weitere Ansätze vgl. [Matoussi&Laleau, 2008].

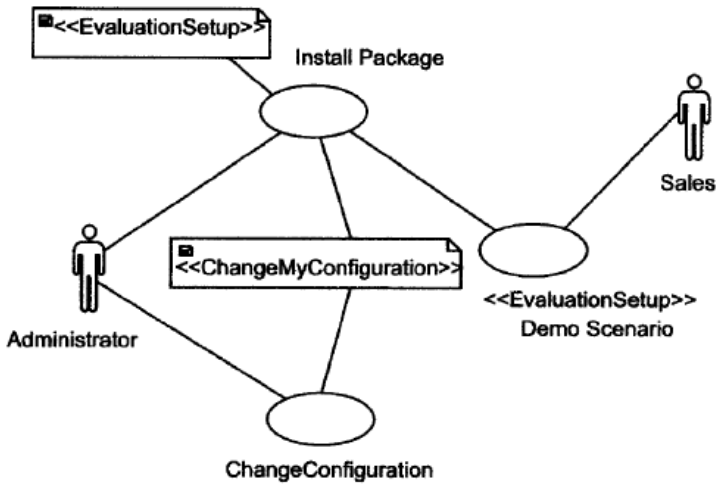


Fig. 3

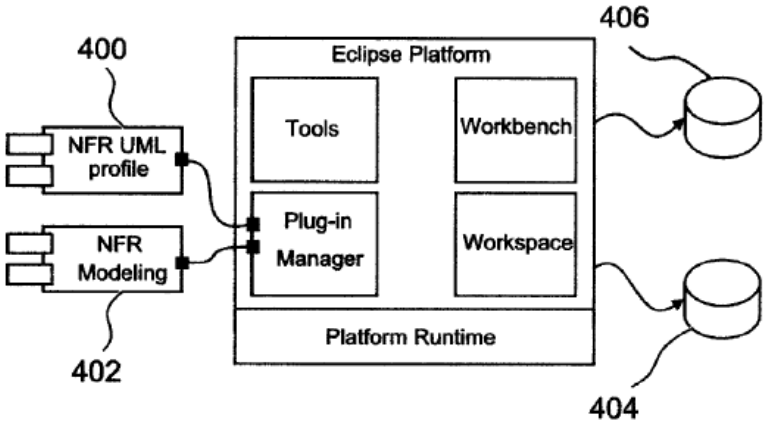


Fig. 4

Abbildung 67 – Ausschnitt aus „US-Patent Application Nummer 2011/0113402 A1" (entnommen aus [Ciano et al., 2010])

Um im Rahmen dieser Arbeit durch die PRM-CIM-Transformation Anschluss an diesen Stand der Technik zu finden, wird aus pragmatischen Gründen (und patentrechtlichen Erwägungen) anstelle eines eigenständigen Profils zur Modellierung nicht-funktionaler Anforderungen das eLoGo-Referenzanforderungsmodell um Stereotypen für die Metaklasse Comment (Kommentar) erweitert.[381]

[381] Kommentare werden in der UML in der Regel verwendet, um Informationen zu Modellelementen hinzuzufügen, die für einen Bearbeiter des Modells interpretierbar sein sollen. Kommentare unterscheiden sich hierdurch vom verwandten Element des Constraint (Einschränkung), weil Constraints Bedingungen oder Voraussetzungen definieren, die durch das System stets erfüllt sein müsen. Für die Formulierung von Contraints ist der Einsatz einer Sprache mit wenig Interpretations-

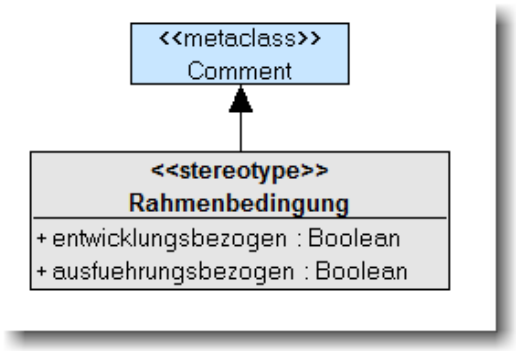


Abbildung 68 – Stereotyp „Rahmenbedingung" zur Formalisierung von Ursprüngen nicht-funktionaler Anforderungen

Im CIM können durch den im Rahmen dieser Arbeit eingeführten Stereotyp «Rahmenbedingung» Notizen gekennzeichnet werden, die allgemeine Rahmenbedingungen für die Umsetzung der Modellelemente ausdrücken, auf die sich die Notiz bezieht. Die Tag Definitions des Stereotyps ermöglichen bei seiner Anwendung im CIM durch Tagged Values eine Unterscheidung, ob die Rahmenbedingung für den Erstellungsprozess und die Entwicklung des Anwendungssystems und/oder für seinen Betrieb gelten. Mit derartigen Notizen ist es möglich, Ursprünge von nicht-funktionalen Anforderungen im CIM zu repräsentieren.

Im Rahmen der sich anschließenden Verfeinerung und Erweiterung des CIM können diese allgemeinen Rahmenbedingungen durch Anwendung der zuvor beschriebenen patentierten Erfindung in nicht-funktionale Anforderungen auf Basis von Contraints mit Stereotypen eines speziellen UML-Profil für nicht-funktionale Anforderungen überführt und zu den Modellelementen im CIM in Bezug gesetzt werden.

5.2 Erzeugung initialer Prozessmodelle

Die Transformation eines PRM des Verwaltungshandelns in ein CIM als initiale Anforderungsspezifikation eines E-Government-Anwendungssystems muss als einen Bestandteil das Prozessmodell des Verwaltungshandelns erzeugen. Die Transformation besteht aus einzelnen Abbildungsregeln, die jeweils eine Abbildung von Elementen der Ausgangsmodelle auf Elemente der Zielmodelle vornehmen. Ausgangsmodelle sind das PRM des Verwaltungshandelns und das eLoGo-Referenzprozessmodell. Zielmodell ist ein Prozessmodell des Verwaltungsprozesses, der durch ein E-Government-Anwendungssystem zu unterstützen ist. Zunächst werden die Abbildungsregeln informell beschrieben, bevor sie in der QVT-Relation Language formalisiert werden.

5.2.1 Beschreibung der Transformation

Die Transformation des PRM in ein CIM muss für jeden Subprozess des Referenzprozess-modells einzeln erfolgen, da diese jeweils in eigenen BPMN-Diagrammen dargestellt sind, die aufgrund von Beschränkungen des verwendeten Werkzeugs keinen in den QVT-Regeln nutzbaren Zusammenhang besitzen (vgl. Abschnitt 5.1.3.1). Weil durch die Transformation aus dem PRM ein Modell erzeugt werden soll, dass auf den verwendeten Referenzmodellen basiert und ihnen deshalb hinsichtlich der Struktur ähnlich ist, kann für die QVT das von Nolte vorgestellte und von ihm als *rekursiver Abstieg* (in Entlehnung des Begriffs aus dem Compilerbau) bezeichnete QVT-Muster benutzt werden.[382] Dazu wird ausgehend vom Wurzelknoten des jeweiligen Subprozesses für jedes dort mögliche Element eine eigene Abbildungsregel angewendet. Dem Muster folgend, wird dann „hinabgestiegen" und geprüft, ob im aktuell bearbeiteten Element weitere Elemente enthalten sind, um in diesem Fall mit

spielraum empfehlenswert, da Constraints auch durch Werkzeuge verarbeitet werden sollen. Häufig wird deshalb hierfür die OCL eingesetzt.

[382] Vgl. [Nolte, 2009], S. 98.

den enthaltenen Elementen fortzufahren. Wenn es keine enthaltenen Elemente gibt, wird die Bearbeitung des aktuellen Elements abgeschlossen. Entsprechend dieses Musters ergeben sich aus der in Listing 29 (Seite 167) vorgestellten Struktur des XML-basierten BPMN-Formates folgende acht Arten von Abbildungsregeln:

- Transformation von Diagrammen und der in ihnen enthaltenen Pools und Elemente: Die Transformation überführt Diagramme (`bpmn:BpmnDiagram`) des Referenzprozessmodells in Diagramme des Zielmodells. Dabei werden auch die in den Diagrammen enthaltenen Pools (`pool`) und weiteren Elemente (`artifacts`) bearbeitet, wobei es in den Diagrammen der Referenzprozessmodelle als weitere Elemente lediglich Text-Annotationen (`bpmn:TextAnnotation`) gibt.

- Transformation der in Diagrammen enthaltenen Pools: Ein Pool innerhalb eines Diagramms des Ausgangsmodells wird in einen Pool des Zielmodells transformiert, in dem auch die enthaltenen Aktivitäten (`vertices`), weiteren Elemente (`artifacts`), verbindenden Sequenzflüsse (`sequenceEdges`) und Schwimmbahnen (`lanes`) transformiert werden.

- Transformation der in Pools enthaltenen Aktivitäten: Die Aktivitäten vom Typ `bpmn:Activity` des Ausgangsmodells werden inklusive ihres jeweiligen Subtyps (z.B. `Task`, `GatewayDataBasedExclusive`, `EventEndEmpty`, `EventStartLink`) in Aktivitäten des Zielmodells überführt. Dabei werden die eingehenden und ausgehenden Kanten des Sequenzflusses angelegt. Anschließend wird mit der Konkretisierung der Semantik fortgefahren.

- Transformation der in Pools enthaltenen Datenobjekte: Die Datenobjekte (`artifacts` vom Typ `bpmn:DataObject`) und ihre Beziehungen (`bpmn:Association`) zu Aktivitäten werden im Zielmodell entsprechend der im Ausgangsmodell modellierten Zusammenhänge angelegt. Anschließend wird mit der Konkretisierung der Semantik fortgefahren.

- Transformation der in Pools enthaltenen Sequenzflüsse: Sequenzflüsse im Ausgangsmodell werden im Zielmodell zwischen den vorhandenen Elementen angelegt. Wurde ein Element des Ausgangsmodells (z.B. aufgrund von Zusammenhängen im PRM) nicht ins Zielmodell übernommen, wird auch dessen Sequenzfluss nicht angelegt.

- Transformation der in Pools enthaltenen Schwimmbahnen: Die Schwimmbahnen im Ausgangsmodell werden im Zielmodell zusammen mit Referenzen auf die enthaltenen Aktivitäten, Datenobjekte und sonstigen Elemente angelegt. Anschließend wird mit der Konkretisierung der Semantik fortgefahren.

- Transformation der in Diagrammen enthaltenen Text-Annotationen: Text-Annotationen des Ausgangsmodells werden im Zielmodell angelegt. Anschließend wird mit der Konkretisierung der Semantik fortgefahren.

- Konkretisierung der Semantik von Aktivitäten, Datenobjekten, Schwimmbahnen und Text-Annotationen durch PRM-Elemente: Aktivitäten und Datenobjekte, die aufgrund der im PRM modellierten Zusammenhänge eine Konkretisierung ermöglichen, werden durch die Elemente des PRM ergänzt oder modifiziert (z.B. die Dokumentation, der Name sowie ggf. ausgehende Beziehungen).

Einen Überblick über diese Arten von Abbildungsregeln und ihre Zusammenhänge gibt die Abbildung 69.

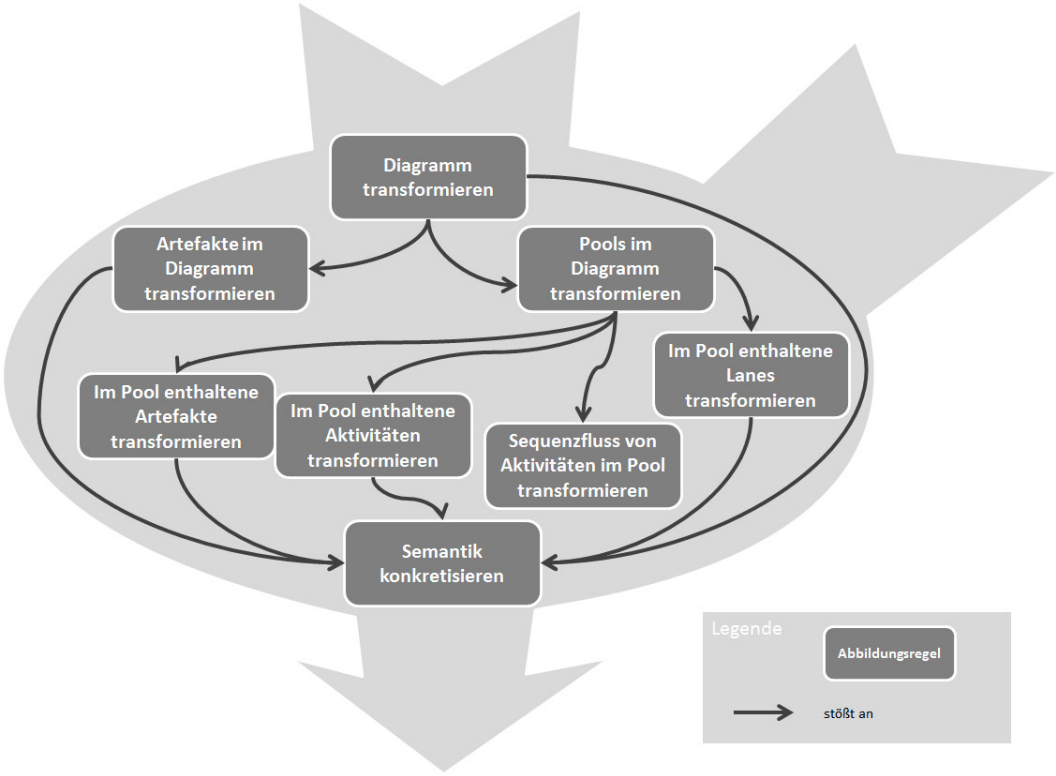


Abbildung 69 – Informelle Darstellung des Zusammenhang zwischen Abbildungsregeln

Die Konkretisierung der Semantik erfolgt durch Auswertung der Informationen im PRM anhand des aktuellen Kontexts im zu transformierenden Subprozess. Die Auswertung berücksichtigt dabei nicht nur die Instanzen, sondern auch deren Beziehungen, die im PRM durch Instance Specification Links repräsentiert verwendet. So lassen sich semantische Abhängigkeiten zwischen den Elementen berücksichtigen, die bei der Annotation der Rechtsvorschriften als Object Properties angelegt wurden.

Es wird beispielsweise bei der Transformation der Aktivität „Möglichkeit/Notwendigkeit zum Verwaltungskontakt aufzeigen“ im PRM eine Instanz des Konzepts „Verwaltungsleistung“ ermittelt und dann zur Konkretisierung der Aktivität des Referenzprozessmodells verwendet. Die prototypische Umsetzung zeigt die Konkretisierung anhand der Beschreibung und des Namens der Aktivität. In Abbildung 70 ist ein Ausschnitt des Referenzprozessmodells dargestellt, der durch die Anwendung der Abbildungsregeln konkretisiert wird. Das Ergebnis dieser Konkretisierung stellt Abbildung 71 dar.

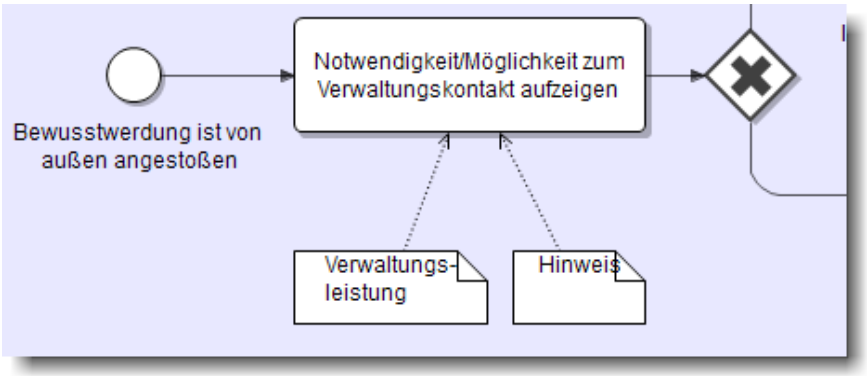


Abbildung 70 – Konkretisierung der Semantik für Aktivität und Data Objects am Beispiel der Bewusstwerdung (Referenzprozess)

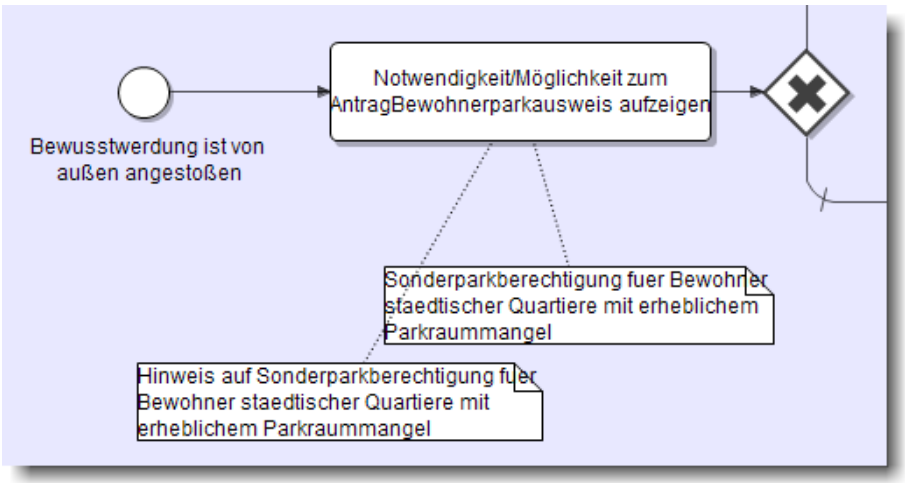


Abbildung 71 – Konkretisierung der Semantik für Aktivität und Data Objects am Beispiel der Bewusstwerdung (Zielmodell)

Außer durch die zuvor beschriebene Möglichkeit mit Instanzen des Konzepts `VwHandlungsgegenstand` und dessen Spezialisierungen Aktivitäten des Referenzanforderungsmodells zu konkretisieren, ist eine Konkretisierung auch durch Instanzen des Konzepts Verwaltungshandlung möglich. Dieses Vorgehen bietet sich für die Aktivitäten zur Leistungserstellung (z.B. Sachverhalt prüfen, Maßnahmen festlegen) an, die eine Entsprechung in den Konzepten der Ontologie haben. Weil jedoch in Rechtsvorschriften die Darstellung des Endergebnisses dominiert und der Weg zu diesem Endergebnis ebenso wie die zeitliche Dimension vernachlässigt werden, ist im PRM nicht zwangsläufig von der Existenz von Instanzen des `VwHandlung`-Konzepts (und dessen Spezialisierungen) auszugehen. Hierauf muss in den Transformationen flexibel reagiert werden, beispielsweise durch einen alternativen Transformationsansatz, der dem zuvor vorgestellten Muster folgt.

Bei Anwendung der Abbildungsregeln ist es nicht nur möglich, Elemente des Referenzanforderungsmodells zu konkretisieren. Es kann anhand der im PRM modellierten Informationen auch entschieden werden, welcher Teil der Referenzprozessmodelle überhaupt relevant ist. So ist es möglich, bei der Annotation der Rechtsvorschriften oder bei der Verfeinerung des PRM festzulegen, dass eine bestimmte Verwaltungsleistung kostenfrei ist. Im Rahmen einer Transformation kann diese Information ausgewertet werden. Aktivitäten und Subprozesse zur Zahlungsabwicklung werden dann nicht in das Zielmodell transformiert. In den nachfolgenden Tabellen sind die Abbildungsregeln für den Subprozess „Bewusstwerdung“ und seine Elemente informell beschrieben.

ID	QVT_PRM2CIM_PROZ-01
Name	Diagramm
Beschreibung	Für jedes im Referenzprozess enthaltene Diagramm wird ein Diagramm im Zielmodell angelegt. Zur Konkretisierung der Semantik wird Regel „QVT_PRM2CIM_PROZ_ALG-01“ auf das neu angelegte Diagramm angewendet.

Tabelle 33 – Abbildungsregel "Diagramm" (QVT_PRM2CIM_PROZ-01)

ID	QVT_PRM2CIM_PROZ-02
Name	DiagrammArtefakt
Beschreibung	Für jedes im Diagramm des Referenzprozessmodells enthaltene Artefakt wird ein Artefakt im Diagramm des Zielmodells angelegt. Zur Konkretisierung der Semantik wird Regel „QVT_PRM2CIM_PROZ_ALG-02" auf das neu angelegte Diagramm angewendet.

Tabelle 34 – Abbildungsregel "DiagrammArtefakt" (QVT_PRM2CIM_PROZ-02))

ID	QVT_PRM2CIM_PROZ-03
Name	DiagrammPool
Beschreibung	Für jeden im Diagramm des Referenzprozessmodells enthaltenen Pool wird ein Pool im Diagramm des Zielmodells angelegt. Name und Eigenschaften des angelegten Pools werden aus dem Pool des Referenzprozessmodells übernommen.

Tabelle 35 – Abbildungsregel "DiagrammPool" (QVT_PRM2CIM_PROZ-03)

ID	QVT_PRM2CIM_PROZ-04
Name	DiagrammPoolArtefakt
Beschreibung	Für jedes im Pool des Referenzprozessmodells enthaltene Artefakt wird ein Artefakt im Pool des Zielmodells angelegt. Anschließend wird die Semantik des Artefakts konkretisiert (Regeln QVT_PRM2CIM_PROZ_BWSTW-02 und QVT_PRM2CIM_PROZ_BWSTW-03).

Tabelle 36 – Abbildungsregel "DiagrammPoolArtefakt" (QVT_PRM2CIM_PROZ-04)

ID	QVT_PRM2CIM_PROZ-05
Name	DiagrammPoolAktivität
Beschreibung	Für jede im Pool des Referenzprozessmodells enthaltene Aktivität wird eine Aktivität im Pool des Zielmodells inkl. eingehender bzw. ausgehender Sequenzflüsse und Referenzen auf ihre zugehörige Schwimmbahn angelegt. Anschließend wird die Semantik der Aktivität konkretisiert (Regel QVT_PRM2CIM_PROZ_BWSTW-01).

Tabelle 37 – Abbildungsregel "DiagrammPoolAktivität" (QVT_PRM2CIM_PROZ-05)

ID	QVT_PRM2CIM_PROZ-06
Name	DiagrammPoolSchwimmbahn
Beschreibung	Für jede im Pool des Referenzprozessmodells enthaltene (nicht leere) Schwimmbahn wird eine Schwimmbahn im Pool des Zielmodells inkl. Referenzen auf die ihr zugeordneten Aktivitäten angelegt. Anschließend wird die Semantik der Schwimmbahn konkretisiert (Regeln QVT_PRM2CIM_PROZ_ALG-03, QVT_PRM2CIM_PROZ_ALG-04 und QVT_PRM2CIM_PROZ_ALG-05 werden angewendet).

Tabelle 38 – Abbildungsregel "DiagrammPoolSchwimmbahn" (QVT_PRM2CIM_PROZ-06)

ID	QVT_PRM2CIM_PROZ_ALG-01
Name	Diagramm_eLoGo
Beschreibung	Für ein Diagramm, das einen Subprozess des eLoGo-Referenzprozessmodells zeigt, wird im Namen des Subprozesses der Text „eLoGo-Referenzprozessmodell" durch den Namen des PRM-Package und den Zusatz „-Prozessmodell" ersetzt. Die sonstige Beschreibung bleibt unverändert.

Tabelle 39 – Abbildungsregel "Diagramm_eLoGo" (QVT_PRM2CIM_PROZ_ALG-01)

ID	QVT_PRM2CIM_PROZ_ALG-02
Name	DiagrammArtefakt_eLoGoNotiz
Beschreibung	Wenn es sich beim Artefakt um eine TextAnnotation („eLoGoNotiz") handelt, wird der Text „eLoGo-Referenzprozessmodell" durch den Namen des PRM-Package und den Zusatz „-Prozessmodell" ersetzt. Die sonstige Beschreibung bleibt unverändert.

Tabelle 40 – Abbildungsregel "DiagrammArtefakt_eLoGoNotiz " (QVT_PRM2CIM_PROZ_ALG-02)

ID	QVT_PRM2CIM_PROZ_ALG-03
Name	DiagrammPoolSchwimmbahn_BearbeiterFrontOffice
Beschreibung	Für eine Schwimmbahn, die den Front Office-Bearbeiter im Referenzmodell repräsentiert, wird im PRM die Instanz der Klasse VwBearbeiter gesucht. (Die prototypische Implementierung verzichtet auf die weitergehende Prüfung der Eindeutigkeit des Modellelements.) Gibt es von dieser Instanz eine Verfeinerung, die ihrerseits eine Instanz der Klasse VwBearbeiterFrontOffice ist, so werden ihr Name und ihre Beschreibung zur Konkretisierung der Semantik der Schwimmbahn verwendet. Andernfalls werden Name und Beschreibung der Instanz von VwBearbeiter zur Konkretisierung der Semantik der Schwimmbahn verwendet.

Tabelle 41 – Abbildungsregel "DiagrammPoolSchwimmbahn_BearbeiterFrontOffice" (QVT_PRM2CIM_PROZ_ALG-03)

ID	QVT_PRM2CIM_PROZ_ALG-04
Name	DiagrammPoolSchwimmbahn_BearbeiterBackOffice
Beschreibung	Für eine Schwimmbahn, die den Back Office-Bearbeiter im Referenzmodell repräsentiert, wird im PRM die Instanz der Klasse VwBearbeiter gesucht. (Die prototypische Implementierung verzichtet auf die weitergehende Prüfung der Eindeutigkeit des Modellelements.) Gibt es von dieser Instanz eine Verfeinerung, die ihrerseits eine Instanz der Klasse VwBearbeiterFrontOffice ist, so werden ihr Name und ihre Beschreibung zur Konkretisierung der Semantik der Schwimmbahn verwendet. Andernfalls werden Name und Beschreibung der Instanz von VwBearbeiter zur Konkretisierung der Semantik der Schwimmbahn verwendet.

Tabelle 42 – Abbildungsregel "DiagrammPoolSchwimmbahn_BearbeiterBackOffice" (QVT_PRM2CIM_PROZ_ALG-04)

ID	QVT_PRM2CIM_PROZ_ALG-05
Name	DiagrammPoolSchwimmbahn_Nachfrager
Beschreibung	Für eine Schwimmbahn, die den Nachfrager im Referenzmodell repräsentiert, wird im PRM die Instanz der Klasse VwAntragsteller gesucht. (Die prototypische Implementierung verzichtet auf die weitergehende Prüfung der Eindeutigkeit des Modellelements.) Es werden Name und Beschreibung dieser Instanz zur Konkretisierung der Semantik der Schwimmbahn verwendet.

Tabelle 43 – Abbildungsregel "DiagrammArtefakt_Nachfrager " (QVT_PRM2CIM_PROZ_ALG-05)

ID	QVT_PRM2CIM_PROZ_BWSTW-01
Name	DiagrammPoolAktivität_BewusstwerdungAufzeigen
Beschreibung	Für eine Aktivität, die das Aufzeigen der Möglichkeit oder Notwendigkeit zu einem Verwaltungskontakt im Subprozess „Bewusstwerdung" des Referenzmodells repräsentiert, wird im PRM die Instanz der Klasse Verwaltungsleistung gesucht. (Die prototypische Implementierung verzichtet auf die weitergehende Prüfung der Eindeutigkeit des Modellelements.) Es werden Name und Beschreibung dieser Instanz zur Konkretisierung der Semantik der Aktivität verwendet.

Tabelle 44 – Abbildungsregel "DiagrammPoolAktivität_BewusstwerdungAufzeigen" (QVT_PRM2CIM_PROZ_BWSTW-01)

ID	QVT_PRM2CIM_PROZ_BWSTW-02
Name	DiagrammPoolArtefakt_Verwaltungsleistung
Beschreibung	Für ein DataObject, das die Verwaltungsleistung im Subprozess „Bewusstwerdung" des Referenzmodells repräsentiert, wird im PRM die Instanz der Klasse Verwaltungsleistung gesucht. (Die prototypische Implementierung verzichtet auf die weitergehende Prüfung der Eindeutigkeit des Modellelements.) Es werden Name und Beschreibung dieser Instanz zur Konkretisierung der Semantik des DataObjects verwendet.

Tabelle 45 – Abbildungsregel "DiagrammPoolArtefakt_Verwaltungsleistung" (QVT_PRM2CIM_PROZ_BWSTW-02)

ID	QVT_PRM2CIM_PROZ_BWSTW-03
Name	DiagrammPoolArtefakt_Hinweis
Beschreibung	Für ein DataObject, das die Verwaltungsleistung im Subprozess „Bewusstwerdung" des Referenzmodells repräsentiert, wird im PRM die Instanz der Klasse Verwaltungsleistung gesucht. (Die prototypische Implementierung verzichtet auf die weitergehende Prüfung der Eindeutigkeit des Modellelements.) Es wird dann die mit der Instanz in Beziehung stehende Instanz der Klasse Hinweis ermittelt. Von dieser Instanz werden Name und Beschreibung dieser Instanz zur Konkretisierung der Semantik des DataObjects verwendet.

Tabelle 46 – Abbildungsregel "DiagrammPoolArtefakt_Verwaltungsleistung" (QVT_PRM2CIM_PROZ_BWSTW-02)

5.2.2 Formalisierung der Transformation

Die zuvor informell beschriebenen Regeln für die Transformation allgemeiner Elemente und der Elemente des Subprozesses „Bewusstwerdung" wurden im Rahmen dieser Arbeit weitergehend formalisiert und prototypisch in QVT implementiert. Die Implementierung orientiert sich an der zuvor eingeführten Struktur und verfeinert sie. Um dabei die Übersichtlichkeit zu gewährleisten, wurden die implementierten Regeln auf mehrere Quellcode-Dateien aufgeteilt. Sie werden in einer übergeordneten Transformation importiert

und dort verwendet. Die nachfolgende Abbildung 72 zeigt die insgesamt verwendeten Quellcode-Dateien. Im Listing 32 ist in Zeile 24 exemplarisch dargestellt, wie eine Transformationen aus der Quellcode-Datei `prm2cim_proz_allgemein.qvt` (in Listing 31) aufgerufen wird.

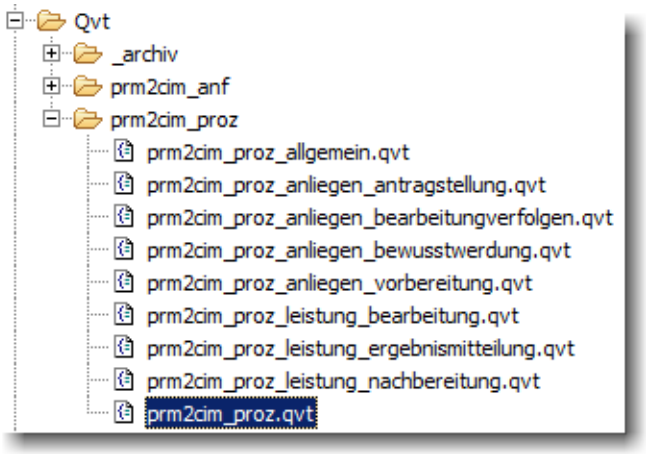


Abbildung 72 – Implementierung der QVT-Regeln in getrennten Dateien

```
(1)   transformation prm2cim_proz_allgemein(refmdl : bpmn, prm : uml, target :
      bpmn) {
(2)
(3)      /*
(4)       * Semantische Konkretisierung der im Diagramm enthaltenen eLoGo-Notiz
(5)       */
(6)      relation diagramArtifactTextAnnotation_eLoGoNotiz {
(7)         -- ...
(8)      };
(9)   }
```

Listing 31 – Transformation prm2cim_proz_allgemein in Quellcode-Datei „prm2cim_proz_allgemein.qvt“ (Auszug)

```
(1)   import prm2cim_proz_allgemein;
(2)
(3)   import prm2cim_proz_anliegen_bewusstwerdung;
(4)   import prm2cim_proz_anliegen_vorbereitung;
(5)   import prm2cim_proz_anliegen_antragstellung;
(6)   import prm2cim_proz_anliegen_bearbeitungverfolgen;
(7)
(8)   import prm2cim_proz_leistung_bearbeitung;
(9)   import prm2cim_proz_leistung_ergebnismitteilung;
(10)  import prm2cim_proz_leistung_nachbereitung;
(11)
(12)  /*
(13)   * Transformation von PRM nach CIM basierend auf dem Referenzmodell
(14)   * für die Prozess-Sicht einer Anforderungsspezifikation
(15)   */
(16)  transformation prm2cim_proz(refmdl : bpmn, prm : uml, target : bpmn) {
(17)     -- ...
(18)     /*
(19)      * Transformation der im Diagramm enthaltenen Artefakte
(20)      */
(21)     relation diagramArtifact {
(22)        -- ...
(23)        -- Semantische Konkretisierung der im Diagramm enthaltenen eLoGo-Notiz
(24)        prm2cim_proz_allgemein::diagramArtifactTextAnnotation_eLoGoNotiz(refa,
      pack, a);
(25)        -- ...
(26)     }
(27)  -- ...
(28)  }
```

Listing 32 – Import von Transformationen und ihre Verwendung in der Haupttransformation „prm2cim_proz“ (Auszug)

Die QVT-Spezifikation der OMG umfasst auch die graphische Notation zur Darstellung von QVT-Transformationen und -Relationen. Außer einem Hinweis, dass sich die Transformationen prinzipiell als UML-Klassendiagramme darstellen lassen, beschreibt der Standard die Notation des neuen *Transformationsdiagramms*. Dieser Notation fehlt eine Möglichkeit, die Abhängigkeiten zwischen Relationen auszudrücken. Aus diesen Abhängigkeiten resultiert jedoch ein wesentlicher Anteil der Komplexität der Gesamttransformation in der vorliegenden Arbeit. Deshalb sind Transformationsdiagramme hier nicht geeignet. Stattdessen werden die QVT-Transformationen als Packages und die Relationen als Klassen in einem Klassendiagramm dargestellt, die mit Abhängigkeitsbeziehungen (Dependency) verbunden sind. Für die Packages, Klassen und Abhängigkeitsbeziehungen wurden Stereotypen definiert, die das repräsentierte QVT-Konzept bezeichnen. Packages sind mit dem Stereotyp «transformation» versehen, um Transformationen zu repräsentieren. Klassen tragen den Stereotyp «top-relation» bzw. «relation» in Abhängigkeit davon, welche Art von Relation sie repräsentieren. Der Stereotyp der Abhängigkeitsbeziehung gibt an, welcher Art die Abhängigkeit ist. Eine Beziehung mit dem Stereotyp «where» steht für den Aufruf einer Relation aus der Where-Klausel einer anderen Relation, «when» entsprechend für die When-Klausel. Eine Abhängigkeitsbeziehung mit dem Stereotyp «import» zwischen zwei Packages drückt aus, dass eine Transformation eine andere Transformation importiert. Die Abbildung 73 stellt die Formalisierung der Transformationen und ihrer Regeln in dieser Notation dar. Sie beschränkt sich auf die Details der Transformation des Subprozesses „Bewusstwerdung“. Die Transformationen der anderen Subprozesse folgen prinzipiell dem gleichen Muster und sind deshalb ohne weitere Details lediglich als Packages dargestellt.

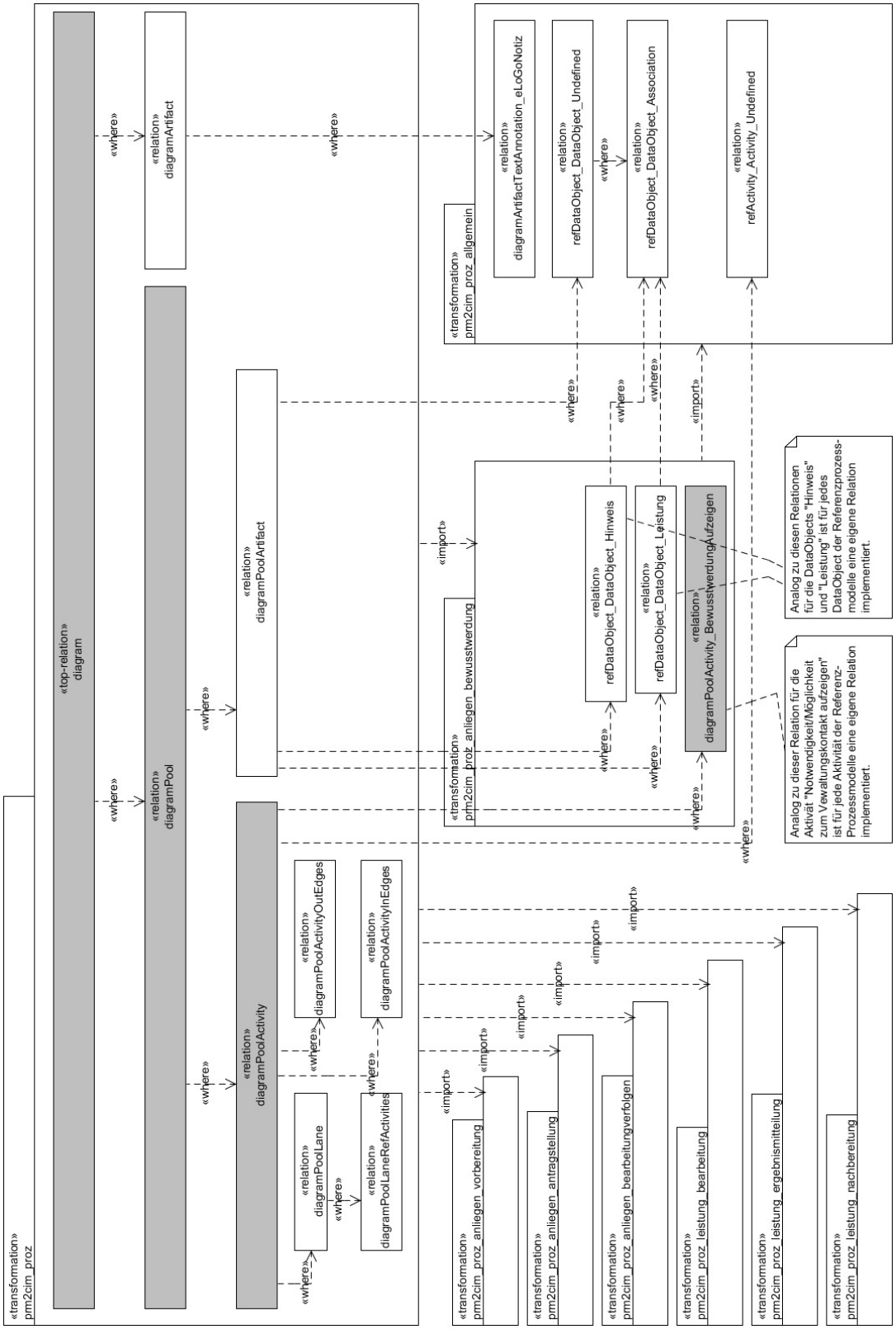


Abbildung 73 – Überblick über die in QVT implementierten Abbildungsregeln (Ausschnitt)

Im nachfolgenden Listing werden die in Abbildung 73 grau hervorgehobenen Regeln in Auszügen vorgestellt. Ihre komplette Darstellung erfolgt im Anhang F.1. Zunächst werden innerhalb der Transformation `prm2cim_proz` in Zeile 1 die notwendige Transformation `prm2cim_proz_bewusstwerdung` und weitere im gekürzten Listing nicht dargestellte

Transformationen importiert. Die Transformation prm2cim_proz_bewusstwerdung importiert ihrerseits die Transformation prm2cim_proz_allgemein, deren Relationen daher auch in prm2cim_proz verfügbar sind.[383] In Zeile 7 ist ersichtlich, dass die Transformation mit drei Modellen aufgerufen wird. Das BPMN-Modell refmdl und das UML-Modell prm sind die Ausgangsmodelle, das BPMN-Modell target ist das Zielmodell der Transformation. Diese Modelle und ihre Elemente stehen in den Abbildungsregeln der Transformation zur Verfügung. Die Zeilen 18 bis 41 zeigen die Abbildungsregel höchster Stufe (top relation) mit dem Namen diagram. Die Abbildungsregel diagram legt im Zielmodell target ein neues BPMN-Diagramm mit dem Namen des PRM-Modells an. Hier zeigt sich bereits eine sehr einfache Überführung von Inhalten aus dem PRM in das Zielmodell unter Verwendung des Referenzmodells. Anschließend wird mit der Transformation der im Diagramm des Referenzprozessmodells enthaltenen Pools fortgefahren (Zeile 39). Die Pools werden in der Abbildungsregel diagramPool aus dem Referenzprozessmodell in das Ausgangsmodell übernommen (Zeilen 45-64). Anschließend wird in Zeile 64 mit der Transformation der in den Pools enthaltenen Aktivitäten fortgefahren. Dazu dient die Abbildungsregel diagramPoolActivity (Zeilen 69-123). In dieser Regel werden die Aktivitäten zunächst angelegt, es erfolgt ihre Zuordnung zu einer Schwimmbahn des Pools sowie die Verbindung mit eingehenden und ausgehenden Sequenzflüssen. Weiterhin wird von hier in die semantische Konkretisierung der angelegten Aktivitäten verzweigt. In der Zeile 104 ist der Aufruf der Abbildungsregel für die Aktivität „Möglichkeit/Notwendigkeit zum Verwaltungskontakt aufzeigen“ dargestellt. Die Regel wird dabei unter Angabe ihrer Transformation prm2cim_proz_anliegen_bewusstwerdung, die zuvor importiert wurde, aufgerufen. Für Aktivitäten, für die keine semantische Konkretisierung erfolgen muss, ist die Abbildungsregel diagramPoolActivity_Undefined in der Transformation prm2cim_proz_allgemein implementiert (Aufruf in Zeile 121).

```
(1)   import prm2cim_proz_anliegen_bewusstwerdung;
(2)   -- ...
(3)   /*
(4)    * Transformation von PRM nach CIM basierend auf dem Referenzmodell
(5)    * für die Prozess-Sicht einer Anforderungsspezifikation
(6)    */
(7)   transformation prm2cim_proz(refmdl : bpmn, prm : uml, target : bpmn) {
(8)
(9)      -- Schlüssel für Identität der Elemente festlegen
(10)     key bpmn::SequenceEdge{iD};
(11)     key bpmn::Lane{iD};
(12)     key bpmn::Activity{iD};
(13)     key bpmn::DataObject{iD};
(14)
(15)     /*
(16)      * Transformation des Diagramms
(17)      */
(18)     top relation diagram{
(19)        modelName : String;
(20)
(21)        checkonly domain refmdl refd : bpmn::BpmnDiagram {
(22)           pools = refpool : bpmn::Pool {}
(23)        };
(24)
(25)        checkonly domain prm mdl : uml::Model {
(26)            name = modelName,
(27)            packagedElement = pack : uml::Package { }
(28)        };
(29)
(30)        enforce domain target diag : bpmn::BpmnDiagram {
```

[383] Die anderen Transformationen für die weiteren Subprozesse importieren ihrerseits ebenfalls die prm2cim_proz_allgemein, was in der Abbildung nicht dargestellt ist. Mit der zum Zeitpunkt der Erstellung dieser Arbeit verfügbaren Version des eingesetzten Werkzeugs mediniQVT führte der Import der prm2cim_proz_allgemein aufgrund eines Importproblems zu einem technischen Fehler (siehe Ticket-Nr. 44 unter http://projects.ikv.de/qvt/ticket/44, letzter Aufruf: 21.08.2011). Um dieses Problem zu umgehen, muss der Inhalt der prm2cim_proz_allgemein redundant in die anderen Transformationen kopiert werden, um ausführbar zu sein.

```
(31)           name = modelName,
(32)           pools = pool : bpmn::Pool {}
(33)        };
(34)
(35)        where {
(36)           -- Transformation der im Diagramm enthaltenen Artefakte
(37)           -- ...
(38)           -- Transformation der im Diagramm enthaltenen Pools
(39)           diagramPool(refpool, pack, pool);
(40)        }
(41)     }
(42)     /*
(43)      * Transformation der im Diagramm enthaltenen Pools inkl. ihrer Elemente
(44)      */
(45)     relation diagramPool{
(46)        pname : String;
(47)
(48)        checkonly domain refmdl refp : bpmn::Pool {   };
(49)
(50)        checkonly domain prm pack : uml::Package {
(51)              name = pname
(52)        };
(53)
(54)        enforce domain target p : bpmn::Pool {
(55)           name = pname
(56)        };
(57)
(58)        where {
(59)           -- Transformation der im Pool enthaltenen Artefakte
(60)           -- ...
(61)           -- Transformation der im Pool enthaltenen Aktivitäten
(62)           diagramPoolActivity(refp,pack,p);
(63)        }
(64)     }
(65)     /*
(66)      * Transformation der im Pool enthaltenen Aktivitäten inkl. ihrer
      eingehenden
(67)      * und ausgehenden Kanten sowie Referenzen auf ihre Schwimmbahn (Lane)
(68)      */
(69)     relation diagramPoolActivity{
(70)        ident : String;
(71)        t : bpmn::ActivityType;
(72)        asso : OrderedSet(bpmn::Association);
(73)
(74)        checkonly domain refmdl refp : bpmn::Pool {
(75)           vertices = refv : bpmn::Activity {
(76)              iD = ident,
(77)              activityType = t
(78)           }
(79)        };
(80)
(81)        checkonly domain prm pack : uml::Package {
(82)
(83)        };
(84)
(85)        enforce domain target p : bpmn::Pool {
(86)           vertices = v : bpmn::Activity {
(87)              iD = ident,
(88)              activityType = t,
(89)              associations = asso->select(a | refv.associations->exists(refa|
                                                        refa.source.iD =
      a.source.iD))
(90)           }
(91)        };
(92)
(93)        where {
(94)           asso =
      p.artifacts.associations.oclAsType(OrderedSet(bpmn::Association));
(95)
(96)           -- Eingehenden und ausgehenden Sequenzfluss der Aktivität
      transformieren
(97)           -- ...
(98)
(99)           -- Transformation der Schwimmbahnen des Pools
(100)          -- ...
```

```
(101)
(102)          -- Konkretisierung der Semantik von Aktivitäten
(103)          -- Anliegen - Bewusstwerdung
(104)          prm2cim_proz_anliegen_bewusstwerdung::diagramPoolActivity_
                                                        BewusstwerdungAufzeigen(ref
      v, pack, v);
(105)          -- Anliegen - Antragstellung
(106)          -- ...
(107)          -- Anliegen - Vorbereitung
(108)          -- ...
(109)          -- Anliegen - Bearbeitung verfolgen
(110)          -- ...
(111)          -- Leistung - Bearbeitung
(112)          -- ...
(113)          -- Leistung - Ergebnismitteilung
(114)          -- ...
(115)          -- Leistung - Nachbereitung
(116)          -- ...
(117)          -- Gesamtprozess
(118)          -- ...
(119)
(120)          -- Transformation von Aktivitäten ohne Konkretisierung der Semantik
(121)          prm2cim_proz_allgemein::diagramPoolActivity_Undefined(refv, pack,
      v);
(122)       }
(123)    }
(124)    -- ...
(125) }
```

Listing 33 – Transformation „prm2cim_proz" (Auszug)

Das Listing 34 zeigt die Transformation prm2cim_proz_anliegen_bewusstwerdung. Sie wird ebenfalls mit dem Referenzmodell und dem PRM als Ausgangs- und dem CIM als Zielmodell aufgerufen. In ihrer Abbildungsregel diagramPoolActivity_Bewusstwerdung-Aufzeigen (ab Zeile 5) wird im PRM die Instanz der Klasse VwAntrag ermittelt. Anschließend wird für die Aktivität des Referenzprozessmodells, deren Eigenschaft ncname den Wert „BewusstwerdungAufzeigen" hat, im Text des Namens und der Dokumentation das Wort „Verwaltungskontakt" durch den Namen der zuvor gefundenen Instanz ersetzt. Diese einfache prototypische Umsetzung der Konkretisierung des Referenzprozessmodells illustriert, wie innerhalb der Abbildungsregel die Inhalte des PRM kontextbezogen ausgewertet werden können (Zeile 27) und für die Konkretisierung der Semantik von Elementen des Zielmodells herangezogen werden können (Zeilen 21 und 22). Prinzipiell sind im Rahmen der Abbildungsregeln auch komplexe Konkretisierungsschritte möglich (z.B. durch Implementierung eigener Funktionen und Einbettung in die QVT). Für die Entwicklung eines Verfolgbarkeitsansatzes im Rahmen dieser Arbeit ist es jedoch nicht erforderlich, so dass sich diese Arbeit auf einfache Transformationen beschränkt.

```
(1)   import prm2cim_proz_allgemein;
(2)
(3)   transformation prm2cim_proz_anliegen_bewusstwerdung(refmdl : bpmn, prm :
      uml, target : bpmn) {
(4)
(5)      relation diagramPoolActivity_BewusstwerdungAufzeigen {
(6)         n : String;
(7)         d : String;
(8)         r : String;
(9)
(10)        checkonly domain refmdl refa : bpmn::Activity {
(11)           name = n,
(12)           documentation = d,
(13)           ncname = 'BewusstwerdungAufzeigen'
(14)        };
(15)
(16)        checkonly domain prm pack : uml::Package {
(17)
(18)        };
(19)
(20)        enforce domain target a: bpmn::Activity {
(21)           name = n.replace('Verwaltungskontakt', r),
(22)           documentation = d.replace('Verwaltungskontakt', r),
```

```
(23)            ncname = 'BewusstwerdungAufzeigen'
(24)        };
(25)
(26)        where {
(27)            r = pack.packagedElement->select(inst |
      inst.oclAsType(uml::InstanceSpecification).classifier->exists(clf |
      clf.name='VwAntrag'))->first().name;
(28)
(29)        }
(30)    }
(31)  -- ...
(32)  }
```

Listing 34 – Transformation „prm2cim_proz_anliegen_bewusstwerdung“ (Auszug)

Wie das prototypische Beispiel zeigt, ist in den Abbildungsregeln der volle Zugriff auf die Inhalte des PRM möglich. Somit lassen sich auch die weiteren zuvor informell beschriebenen Regeln ebenso implementieren. Im Ergebnis der Transformationen kann durch Konkretisierung des Referenzprozessmodells mit den Informationen aus dem PRM das Zielmodell erzeugt werden. Abbildung 74 stellt das durch die prototypisch implementierte Transformation erzeugte Zielmodell mit dem Subprozess „Bewusstwerdung“ vollständig dar. Der Vergleich mit Abbildung 61 (auf Seite 166) macht die vorgenommene Konkretisierung des Modells anhand der einfachen Abbildungsregeln deutlich.

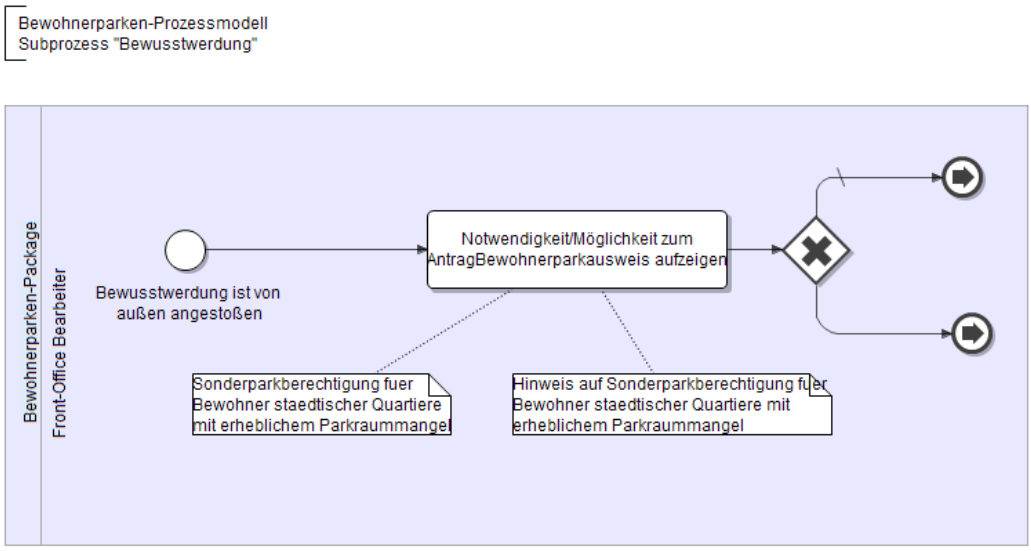


Abbildung 74 – Ergebnis der Transformation des PRM in den Subprozess „Bewusstwerdung“ auf Basis des Referenzprozessmodells

5.3 Erzeugung initialer Modelle der Anforderungsanalyse

Die Transformation eines PRM des Verwaltungshandeln in ein CIM muss neben Prozessmodellen auch Anwendungsfall- und Klassendiagramme liefern, die als initiale Anforderungsspezifikation eines E-Government-Anwendungssystems dienen. Um die semantische Lücke zwischen den Konzepten der beteiligten Modelltypen zu schließen, wird das UML-Profil der eLoGo-Referenzanforderungsmodelle innerhalb der Transformation auf das erzeugte CIM angewendet. Die Transformation eines PRM des Verwaltungshandelns in ein CIM erfolgt wie bei den Prozessmodellen durch eine Menge von Abbildungsregeln, die von der QVT-Laufzeitumgebung ausgeführt werden. Diese Abbildungsregeln werden zunächst informell beschrieben und dann in der QVT-Relation Language implementiert.

5.3.1 Beschreibung der Transformation

Die Transformation wendet einzelne Abbildungsregeln auf das PRM als Ausgangsmodell an und erzeugt ein CIM, das aus Anwendungsfall- und Klassendiagrammen besteht, die sich an den verwendeten Referenzmodellen orientieren. In der Transformationen eines PRM des

Verwaltungshandelns in das CIM eines E-Government-Anwendungssystems wird auf das UML-Profil des eLoGo-Referenzanforderungsmodells zurückgegriffen, um die Elemente und ihre Zusammenhänge zu erzeugen. Die Struktur der erzeugten Elemente ist im Vergleich zu den Prozessmodellen einfacher, weshalb das Muster des rekursiven Abstiegs hier verkürzt eingesetzt werden kann. Die Abbildungsregeln der Transformation erzeugen zunächst das Zielmodell und darin die benötigte Package-Struktur. Dann werden innerhalb dieser Packages durch weitere Abbildungsregeln Modellelemente erzeugt, die ihrerseits aus den Elementen des PRM jeweils für einen konkreten Anwendungsfall alle Bestandteile eines Anwendungsfalldiagramms und eines Klassendiagramms erzeugen.[384] Zu diesen Bestandteilen gehören die Akteure, der Anwendungsfall und ihre Kommunikationsbeziehungen sowie die Klassen und ihre Beziehungen untereinander. Es ergeben sich dadurch die folgenden Arten von Abbildungsregeln:

- Transformation des Modells und Erzeugung der Package-Struktur des Zielmodells: Es wird das Zielmodell angelegt und innerhalb des Zielmodells die Package-Struktur aus den Referenzanforderungsmodellen übernommen.
- Transformation der in den Packages enthaltenen Elemente auf Basis des PRM: Ausgehend von den Elementen im PRM werden in den Packages des Zielmodells die enthaltenen Elemente angelegt.
- Transformation der relevanten Elemente des PRM in Akteure eines Packages: Aus den Verwaltungsakteuren im PRM werden Akteure (im Sinne der UML) im Zielmodell erzeugt. Die Informationen aus dem PRM dienen zusammen mit den Informationen des zugeordneten Stereotyps zur Konkretisierung der UML-Akteure.
- Transformation der relevanten Elemente des PRM in Anwendungsfälle eines Packages: Aus den Verwaltungshandlungen im PRM werden Anwendungsfälle erzeugt, für fehlende Verwaltungshandlungen wird anhand des Referenzmodells und der im PRM enthaltenen Handlungsgegenstände ein Anwendungsfall angelegt. Die Informationen aus dem PRM dienen zusammen mit den Informationen des zugeordneten Stereotyps zur Konkretisierung der Anwendungsfälle.
- Transformation der relevanten Elemente des PRM in Klassen eines Packages: Aus den Verwaltungshandlungsgegenständen werden Klassen erzeugt, die das Domänenobjektmodell bilden. Die Informationen aus dem PRM dienen zusammen mit den Informationen des zugeordneten Stereotyps zur Konkretisierung der Klasse.
- Transformation der Zusammenhänge zwischen den Elementen im PRM in Beziehungen zwischen den Elementen: Es werden alle Beziehungen angelegt, deren beteiligte Elemente im Zielmodell vorhanden sind (z.B. Kommunikationsbeziehungen zwischen Akteur und Anwendungsfall oder Assoziationen zwischen Klassen).
- Transformation der Elemente im PRM in Stereotypen für die erzeugten Elemente des Zielmodells: Ausgehend vom Element im PRM wird der Stereotyp im UML-Profil ermittelt und dem zum PRM-Element gehörigen Element im Zielmodell zugewiesen.
- Zuweisung eines UML-Profils: Der Verweis auf das verwendete UML-Profil des eLoGo-Referenzanforderungsmodells wird im Zielmodell angelegt.

Aus dem Zusammenwirken dieser Arten von Abbildungsregeln in einer Transformation ergibt sich der in Abbildung 75 informell dargestellte Zusammenhang.

[384] Es werden durch die Transformation mit QVT lediglich Modellelemente und keine Diagramme erzeugt. Die Diagrammerzeugung aus dem Zielmodell erfolgt durch Nutzung der entsprechenden Werkzeugfunktion des Modellierungswerkzeugs.

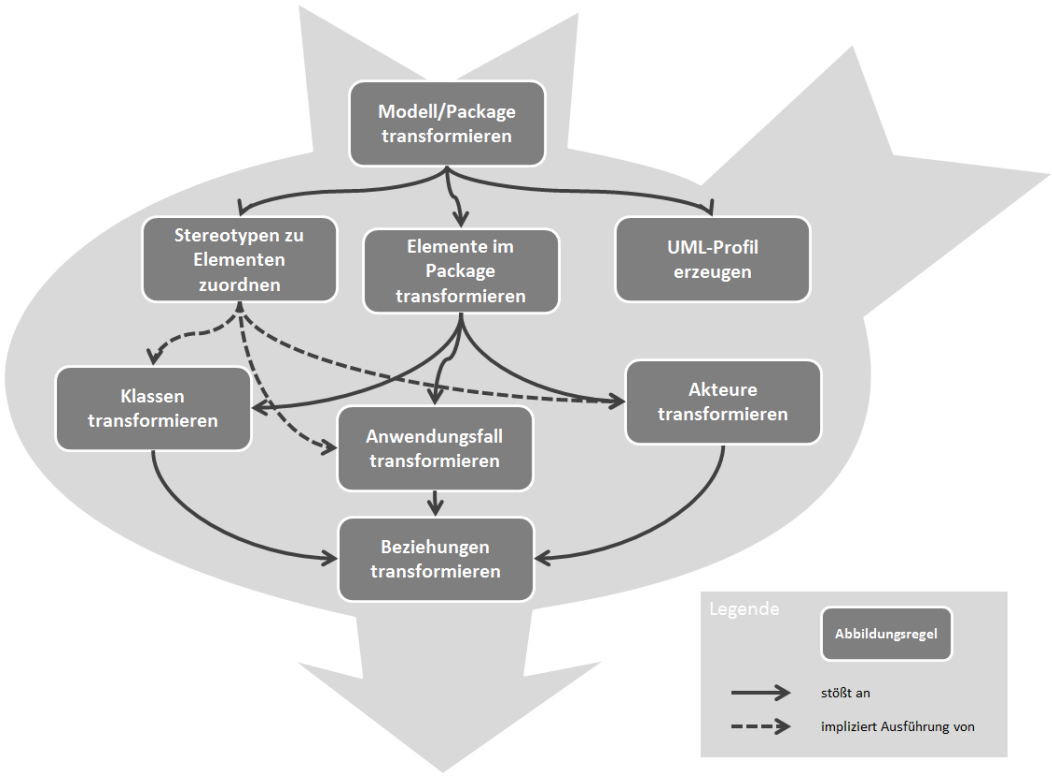


Abbildung 75 – Informelle Darstellung des Zusammenhangs zwischen Abbildungsregeln

Werden diese Abbildungsregeln beispielsweise auf ein PRM zur Erzeugung des Anwendungsfalls „Verwaltungsleistung bewusst machen“ angewendet, wird zunächst ein neues Modell mit den Packages „Anliegen“, „Leistung“ und „Querschnittsfunktionen“ erstellt, sofern diese im Zielmodell nicht bereits existieren. Wenn im PRM des Verwaltungshandelns keine Instanz der Klasse `VwBewusstwerdung` existiert, muss eine Instanz der Klasse `VwLeistung` ermittelt werden. Für sie wird dann ein „Verwaltungsleistung bewusst machen“-Anwendungsfall entsprechend des Referenzanforderungsmodells angelegt. In der prototypischen Umsetzung wird er durch die Bezeichnung und Beschreibung der Instanz einer Verwaltungsleistung und weiterer Informationen aus anderen Instanzen des PRM konkretisiert. Dazu wird beispielsweise das Ziel des Anwendungsfalls mit Angaben zum Verwaltungsprodukt, zur Verwaltungsleistung und zu den Hinweisen/Informationen ergänzt. Wenn ein Anwendungsfall entsprechend im Modell angelegt wurde, wird ihm der Stereotyp «`Verwaltungsleistung bewusst machen`» zugewiesen. Für die Instanzen der Klasse `VwAntragsteller` werden Akteure im Zielmodell angelegt, denen als Name die Bezeichnung der jeweiligen Instanz zugewiesen wird. Jedem dieser Akteure wird der Stereotyp «`NachfragerAkteur`» zugeordnet. Anschließend wird die Kommunikationsbeziehung zwischen dem Akteur und dem Anwendungsfall angelegt. Im Ergebnis dieser Transformation entsteht aus dem PRM die Konkretisierung des eLoGo-Referenzanforderungsmodells in Form eines Anwendungsfalldiagramms. Die Abbildung 76 zeigt einen Ausschnitt dieses Diagramms für den Anwendungsfall „Bewohnerparkausweis bewusst machen“. Im eingeblendeten Dokumentationsfenster ist die konkretisierte Beschreibung des Anwendungsfalls dargestellt. Diese Beschreibung ist durch die Abbildungsregeln auf Basis der Information im PRM und im Stereotyp des UML-Profil erzeugt worden. Im Vergleich zur Beschreibung des entsprechenden Anwendungsfalls im eLoGo-Referenzanforderungsmodell in Tabelle 3 auf Seite 51 ist die Konkretisierung erkennbar.

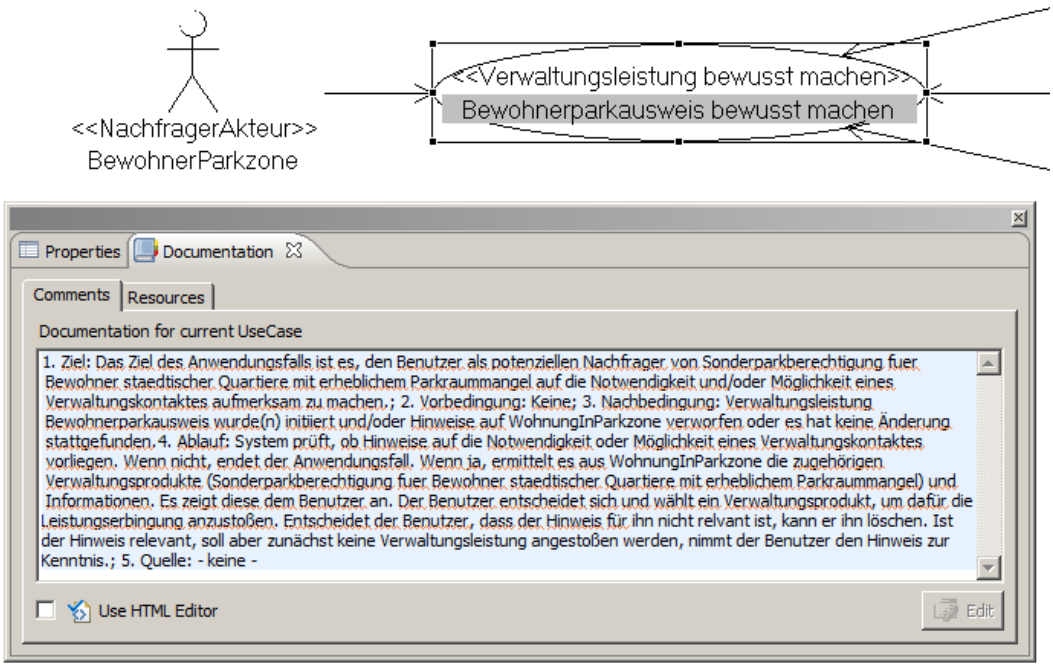


Abbildung 76 – Auszug aus dem Anwendungsfalldiagramm „Bewohnerparkausweis bewusst machen“

Das zum Anwendungsfall gehörende Domänenobjektmodell wird erzeugt, indem Instanzen der Klassen `VwLeistung`, `VwRechtsfolge`, `VwHinweisInformation`, `VwLebenslage` und `VwAntragsteller` im PRM des Verwaltungshandelns auf entsprechende Klassen im Zielmodell abgebildet werden. Dabei werden die Informationen aus dem PRM in die Dokumentation der Klassen übernommen. Jeder dieser Klassen im Zielmodell wird der sich aus der Instanz ergebende Stereotyp zugewiesen (z.B. einer Instanz von `VwRechtsfolge` der Stereotyp «`Verwaltungsprodukt`»). Anschließend wird mit der Transformation der Beziehungen zwischen den Klassen fortgefahren. Im Ergebnis dieser Transformation entsteht aus dem PRM die Konkretisierung des eLoGo-Referenzanforderungsmodells in Form eines Domänenobjektmodells, das als Klassendiagramms zum Anwendungsfall dargestellt werden kann. Die Abbildung 77 stellt einen Auszug aus dem PRM zum Bewohnerparken dar, der entsprechend der zuvor beschriebenen Abbildungsregeln transformiert wurde. Das Ergebnis dieser Transformation ist in Abbildung 78 dargestellt. Im Vergleich zum entsprechenden Diagramm des eLoGo-Referenzanforderungsmodells (in Abbildung 19 auf Seite 52) sind die Klassen entsprechend des PRM konkretisiert und verfügen über Stereotypen aus dem UML-Profil des eLoGo-Referenzanforderungsmodells.

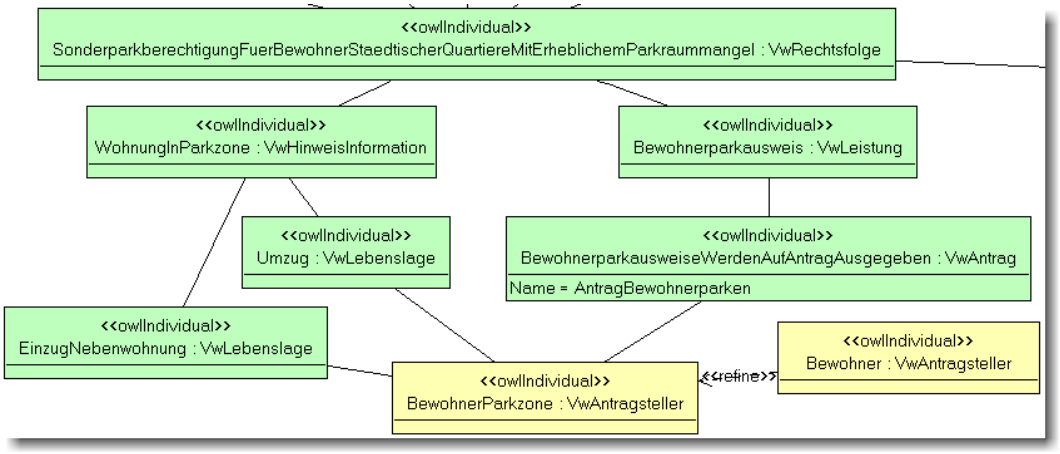


Abbildung 77 – Ausschnitt aus dem PRM zum Bewohnerparken

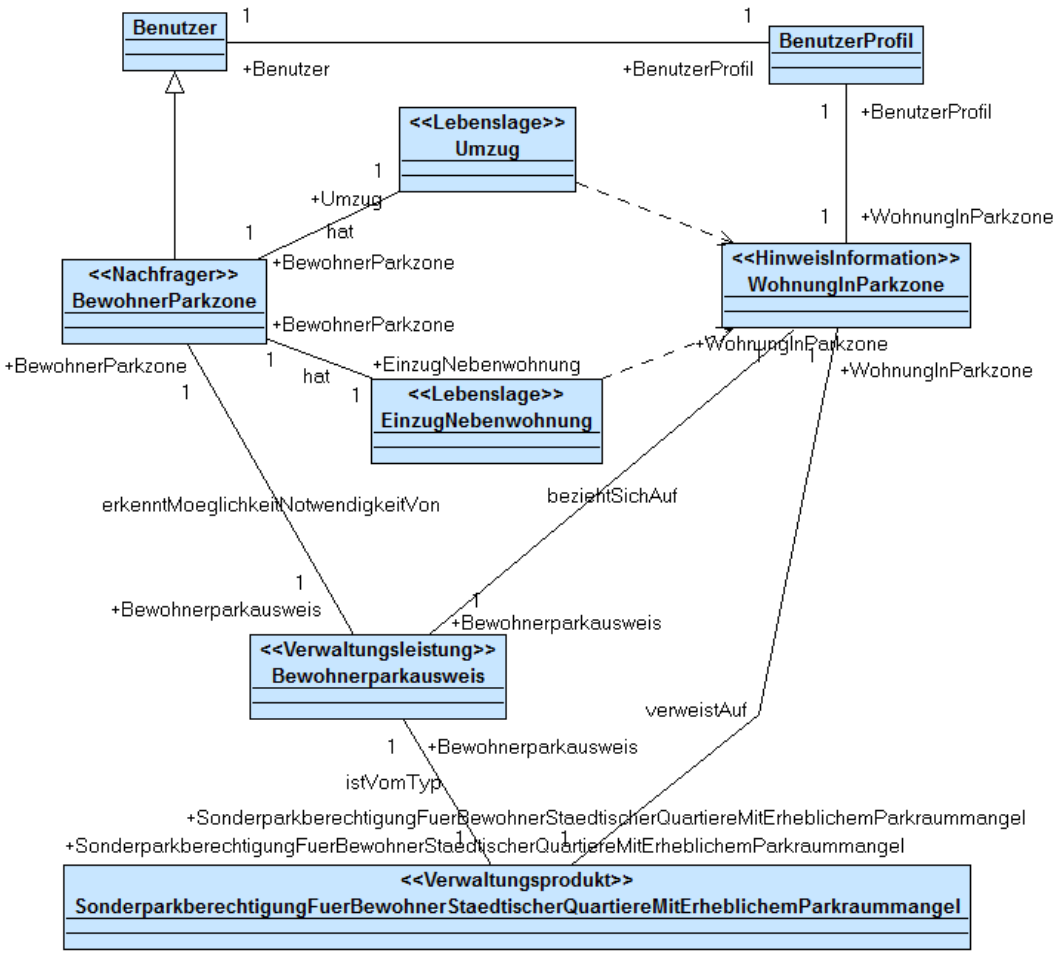


Abbildung 78 – Domänenobjektmodell zum Ausschnitt des PRM in Abbildung 77 als Konkretisierung von Abbildung 19 (S. 52)

Für den zuvor exemplarisch dargestellten Ausschnitt sind in den nachfolgenden Tabellen die Abbildungsregeln der Transformation informell beschrieben. In den Transformationen wird das UML-Profil des eLoGo-Referenzanforderungsmodells (kurz: eGovRefAnf) verwendet. Es werden generell nur solche Elemente berücksichtigt, für die es keine ausgehenden Abhängigkeitsbeziehungen mit dem Standard-UML-Stereotyp «`refine`» gibt. Solche Beziehungen wurden im Rahmen der PRM-Modellierung (Abschnitt 4.3) angelegt, um eine weitergehende Verfeinerung eines Modellelements durch andere (ggf. neu hinzugefügte) Modellelemente auszudrücken. Deshalb werden die Abbildungsregeln nur auf die weiterentwickelten Konzepte angewendet, die Ziel der Verfeinerungsbeziehung sind. Ein Beispiel einer solchen Verfeinerungsbeziehung ist in dem PRM-Ausschnitt in Abbildung 77 zwischen den Individuen `Bewohner` und `BewohnerParkzone` dargestellt. Im Domänenobjektmodell gibt es dann lediglich die Klasse `BewohnerParkzone` mit dem Stereotyp «`Nachfrager`».

Die bisher im Vordergrund stehenden funktionalen Anforderungen können um nicht-funktionale Anforderungen ergänzt werden, wenn bei der Annotation der Rechtsvorschriften oder bei der Verfeinerung des PRM Rahmenbedingungen für die Entwicklung oder für den späteren Betrieb der E-Government-Anwendung festgelegt wurden. Ein Ausschnitt aus dem PRM des Verwaltungshandelns in Abbildung 79 zeigt, wie zum Rechtsfolgenmerkmal `MusterZuVerwenden`, solche Rahmenbedingungen formuliert wurden.

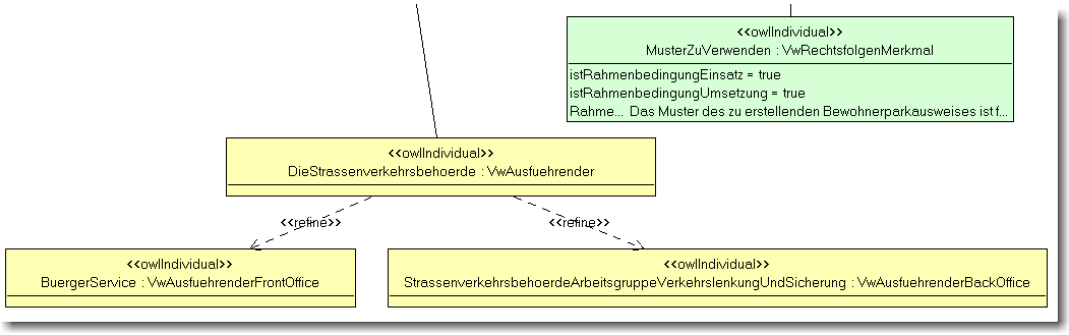


Abbildung 79 – Rahmenbedingungen für Entwicklung und Betrieb als Ausschnitt aus dem PRM des Verwaltungshandeln

Um nicht-funktionale Anforderungen innerhalb der Anwendungsfall- oder Klassendiagramme berücksichtigen zu können, wurde der Stereotyp «Rahmenbedingung» innerhalb des eGovRefAnf-Profils definiert. Durch Anwendung der Abbildungsregeln wird geprüft, ob die an der Transformation beteiligten Elemente im PRM Attributsausprägungen für `istRahmenbedingungEinsatz` oder ist `RahmenbedingungUmsetzung` mit dem Wahrheitswert `true` haben. Ist dies der Fall, wird zu den in CIM erzeugten Elementen eine Notiz generiert. Ihr wird der Text aus der Ausprägung des Attributes `Rahmenbedingung` zugewiesen. Sie erhält darüber hinaus den Stereotyp «Rahmenbedingung». Diesem Stereotyp werden für die Tagged Values `Entwicklungsbezogen` und `Ausfuehrungsbezogen` die entsprechenden Werte aus `istRahmenbedingungUmsetzung` und `istRahmenbedingungEinsatz` zugewiesen. Das Ergebnis dieser Transformation zeigt Abbildung 80.

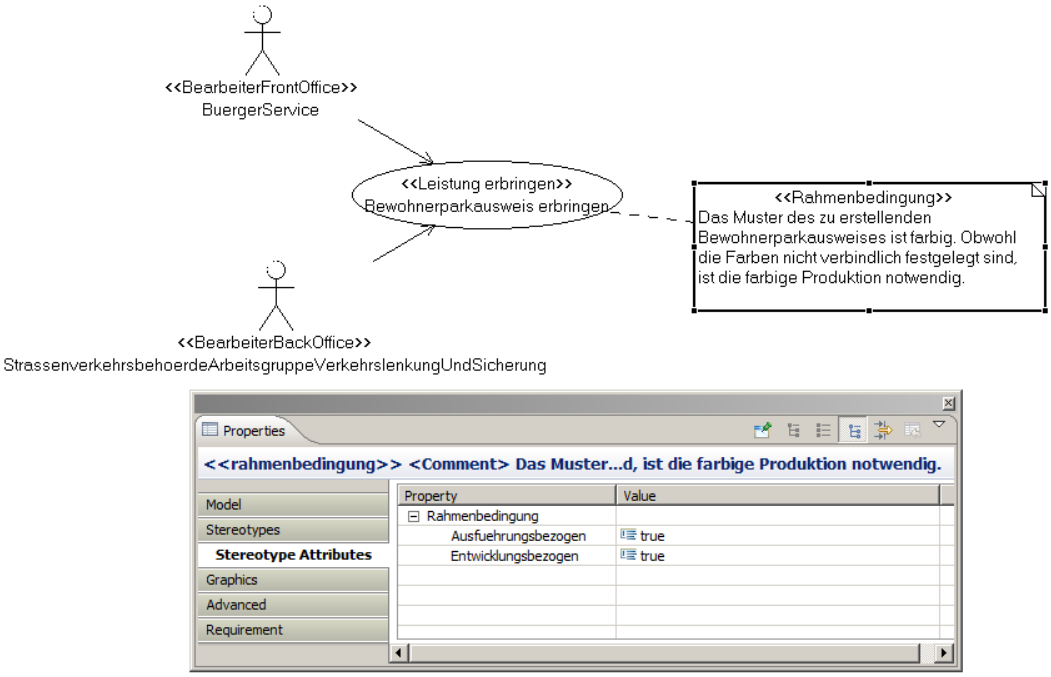


Abbildung 80 – Berücksichtigung nicht-funktionaler Anforderungen im CIM

In der Transformation bilden zwei Abbildungsregeln für das Modell und die Anwendung des UML-Profils die Grundlage, so dass Abbildungsregeln für weitere Modellelemente bearbeitet werden können (Tabelle 47 und Tabelle 48). In Tabelle 49 bis Tabelle 51 sind Abbildungsregeln informell beschrieben, die Stereotypen für den Akteur „Nachfrager“, den Anwendungsfall „Verwaltungsleistung bewusst machen“ und die Klasse „Verwaltungsprodukt“ anlegen. Die Tabelle 52 beschreibt die Abbildungsregel zur Transformation der Ursprünge nicht-funktionaler Anforderungen. In den anschließenden

Abbildungsregeln werden die Modellelemente Akteur, Anwendungsfall und Klasse sowie ihre Beziehungen zu anderen Elementen angelegt (Tabelle 53 bis Tabelle 56).

ID	QVT_PRM-CIM_ANF-01
Name	Modell
Beschreibung	Für jedes im PRM enthaltene Modell wird im CIM ein Modell mit den drei Packages „Anliegen", „Verwaltungsleistungen" und „Querschnittsfunktionen" angelegt. Der Name der Packages wird durch den Namen des PRM ergänzt. Anschließend werden im PRM-Modell enthaltenen Elemente durch Anwendung der weiteren Abbildungsregeln bearbeitet (QVT_PRM-CIM_ANF-02, QVT_PRM-CIM_ANF-03, QVT_PRM-CIM_ANF-04, QVT_PRM-CIM_ANF-05, QVT_PRM-CIM_ANF-06).

Tabelle 47 – Abbildungsregel "Modell" (QVT_PRM-CIM_ANF-01)

ID	QVT_PRM-CIM_ANF-02
Name	Profil
Beschreibung	Dem CIM wird das UML-Profil des eLoGo-Referenzanforderungsmodells zugewiesen und auf das Modell angewendet.

Tabelle 48 – Abbildungsregel "Profil" (QVT_PRM-CIM_ANF-02)

ID	QVT_PRM-CIM_ANF-03
Name	StereotypNachfragerAkteur
Beschreibung	Für jede im PRM-Modell enthaltene uml::InstanceSpecification mit dem Stereotyp «owlIndividual», die keine ausgehenden Verfeinerungsbeziehungen zu anderen Elementen hat und die Instanz der Klasse "VwAntragsteller" ist, wird im CIM ein eGovRefAnf-Stereotyp «Nachfrager» angelegt. Es wird mit der Erzeugung des Akteurs fortgefahren, dem der Stereotyp zugeordnet ist (QVT_PRM-CIM_BWST-01).

Tabelle 49 – Abbildungsregel "Stereotyp NachfragerAkteur" (QVT_PRM-CIM_ANF-03)

ID	QVT_PRM-CIM_ANF-04
Name	StereotypVerwaltungsleistungBewusstMachen
Beschreibung	Für jede im PRM-Modell enthaltene uml::InstanceSpecification mit dem Stereotyp «owlIndividual», die keine ausgehenden Verfeinerungsbeziehungen zu anderen Elementen hat und die Instanz der Klasse „VwLeistung" ist, wird im CIM ein eGovRefAnf-Stereotyp «Verwaltungsleistung bewusst machen» angelegt. Es wird mit der Erzeugung des Anwendungsfalls fortgefahren, dem der Stereotyp zugeordnet ist (QVT_PRM-CIM_BWST-02).

Tabelle 50 – Abbildungsregel "Stereotyp Verwaltungsleistung bewusst machen" (QVT_PRM-CIM_ANF-04)

ID	QVT_PRM-CIM_ANF-05
Name	StereotypVerwaltungsprodukt
Beschreibung	Für jede im PRM-Modell enthaltene uml::InstanceSpecification mit dem Stereotyp «owlIndividual», die keine ausgehenden Verfeinerungsbeziehungen zu anderen Elementen hat und die Instanz der Klasse „`VwRechtsfolge`" ist, wird im CIM ein eGovRefAnf-Stereotyp «Verwaltungsprodukt» angelegt. Es wird mit der Erzeugung des Anwendungsfalls fortgefahren, dem der Stereotyp zugeordnet ist (QVT_PRM-CIM_BWST-03).

Tabelle 51 – Abbildungsregel "Stereotyp Verwaltungsprodukt" (QVT_PRM-CIM_ANF-05)

ID	QVT_PRM-CIM_ANF-06
Name	Rahmenbedingung
Beschreibung	Für jede im PRM-Modell enthaltene uml::InstanceSpecification mit dem Stereotyp «owlIndividual», deren Attribut „Rahmenbedingung" eine Werteausprägung hat, wird im CIM ein eGovRefAnf-Stereotyp «Rahmenbedingung» angelegt. Anschließend wird mit der Erzeugung des Elements fortgefahren, dem der Stereotyp zugeordnet ist (QVT_PRM-CIM_ALG-06).

Tabelle 52 – Abbildungsregel "Rahmenbedingung" (QVT_PRM-CIM_ANF-06)

ID	QVT_PRM-CIM_BWST-01
Name	AkteurNachfrager
Beschreibung	Für jede im PRM-Modell enthaltene uml::InstanceSpecification mit dem Stereotyp «owlIndividual», die keine ausgehenden Verfeinerungsbeziehungen zu anderen Elementen hat und die Instanz der Klasse "VwAntragsteller" ist, wird im CIM ein Akteur mit dem Namen der Instanz angelegt. Die Attributsausprägungen der Instance Specification für Quelle, Arbeitsnotiz und Beschreibung sowie die Defaultwerte der Attribute des Stereotyps werden der Beschreibung der Klasse zugewiesen. Es wird geprüft, ob die Instance Specification über Links mit weiteren Instance Specifications verfügt, die auf einen Anwendungsfall abzubilden sind. In diesem Fall wird mit der Bearbeitung dieser Instance Specifications fortgefahren.

Tabelle 53 – Abbildungsregel "Akteur Nachfrager" (QVT_PRM-CIM_BWST-01)

ID	QVT_PRM-CIM_BWST-02
Name	AnwendungsfallVerwaltungsleistungBewusstMachen
Beschreibung	Für jede im PRM-Modell enthaltene uml::InstanceSpecification mit dem Stereotyp «owlIndividual», die keine ausgehenden Verfeinerungsbeziehungen zu anderen Elementen hat und die Instanz der Klasse „VwLeistung" ist, wird im CIM ein Anwendungsfall angelegt. Seine Bezeichnung wird aus dem Namen der Instanz zur Klasse „VwLeistung" und dem Namen des Anwendungsfalls „Verwaltungsleistung bewusst machen" aus den eLoGo-Referenzmodellen übernommen. Die Beschreibung des Anwendungsfalls wird aus dem eGovRefAnf-Profil übernommen. Es wird geprüft, ob die Instance Specification über Links mit weiteren Instance Specifications verfügt, die auf einen Akteur abzubilden sind. In diesem Fall wird mit der Bearbeitung dieser Instance Specifications fortgefahren.

Tabelle 54 – Abbildungsregel "Anwendungsfall Verwaltungsleistung bewusst machen" (QVT_PRM-CIM_BWST-02)

ID	QVT_PRM-CIM_BWST-03
Name	KlasseVerwaltungsprodukt
Beschreibung	Für jede im PRM-Modell enthaltene uml::InstanceSpecification mit dem Stereotyp «owlIndividual», die keine ausgehenden Verfeinerungsbeziehungen zu anderen Elementen hat und die Instanz der Klasse „VwRechtsfolge" ist, wird im CIM eine Klasse angelegt. Die Attributsausprägungen der Instance Specification für Quelle, Arbeitsnotiz und Beschreibung sowie die Defaultwerte der Attribute des Stereotyps werden der Beschreibung der Klasse zugewiesen. Es wird geprüft, ob die Instance Specification über Links zu weiteren Instance Specifications verfügt, die Verwaltungshandlungsgegenstände repräsentieren. Ist dies der Fall, wird mit dem Anlegen von Assoziationsbeziehungen zwischen den Klassen fortgefahren.

Tabelle 55 – Abbildungsregel "Klasse Verwaltungsprodukt" (QVT_PRM-CIM_BWST-03)

ID	QVT_PRM-CIM_ALG-01
Name	Rahmenbedingung
Beschreibung	Für jede im PRM-Modell enthaltene uml::InstanceSpecification mit dem Stereotyp «owlIndividual», deren Attribut „Rahmenbedingung" eine Werteausprägung hat, wird im CIM ein Kommentar angelegt. Der Wert des Attributes wird in den Text des Kommentars übernommen. Anschließend wird die Notiz mit dem aus der uml::InstanceSpecification erzeugten Modellelement in Beziehung gesetzt und mit der Bearbeitung des Modellelements fortgefahren (QVT_PRM-CIM_BWST-01, QVT_PRM-CIM_BWST-02, QVT_PRM-CIM_BWST-03).

Tabelle 56 – Abbildungsregel "Rahmenbedingung" (QVT_PRM-CIM_ANF-06)

5.3.2 Formalisierung der Transformation

Die zuvor informell beschriebenen Regeln für die Transformation der Elemente des Anwendungsfalls „Verwaltungsleistung bewusst machen" und weiterer allgemeiner Elemente wurden im Rahmen dieser Arbeit formalisiert und prototypisch in QVT-R implementiert. Diese Implementierung orientiert sich an der zuvor eingeführten informellen Struktur (Abbildung 75 auf Seite 189), teilt die implementierten Regeln aber auf mehrere Quellcode-Dateien mit einzelnen Transformationen pro Anwendungsfall auf. In Abbildung 81 ist die Datei `prm2cim_anf_anliegen_bewusstwerdung.qvt` mit der Transformation des Anwendungsfalls „Verwaltungsleistung bewusst machen" hervorgehoben. Sie wird in der ebenfalls hervorgehobenen übergeordneten Transformation in der Datei `prm2cim_anf.qvt` importiert und dort, wie auch die anderen Transformationen, verwendet.

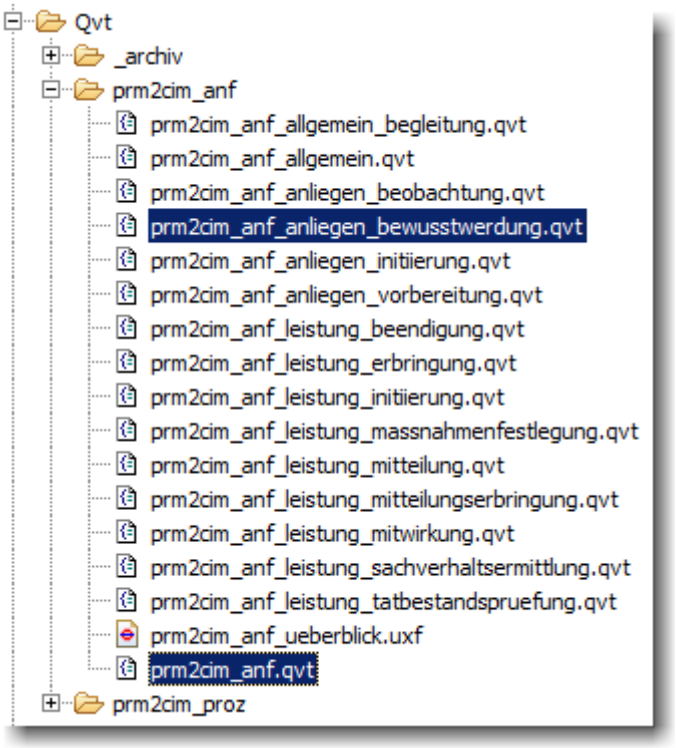


Abbildung 81 – Auszug aus der Projektstruktur in Eclipse mit den implementierten QVT-Transformationen, Ausgangs- und Zielmodellen

Insgesamt ergibt sich aus den informell beschriebenen Arten von Abbildungsregeln und den Transformationen der einzelnen Anwendungsfälle, die in Abbildung 82 dargestellte Struktur. Sie zeigt ausgewählte Details der Transformationen `prm2cim_anf` und `prm2cim_anf_anliegen_bewusstwerdung` sowie Abhängigkeiten zu anderen Transformationen. Die Abbildung nutzt die bereits zuvor in Abschnitt 5.2.2 eingeführte Notation, die jetzt deutlich werden lässt, dass die Komplexität der Gesamttransformation zu einem Großteil aus den Abhängigkeiten zwischen einzelnen Abbildungsregeln resultiert.

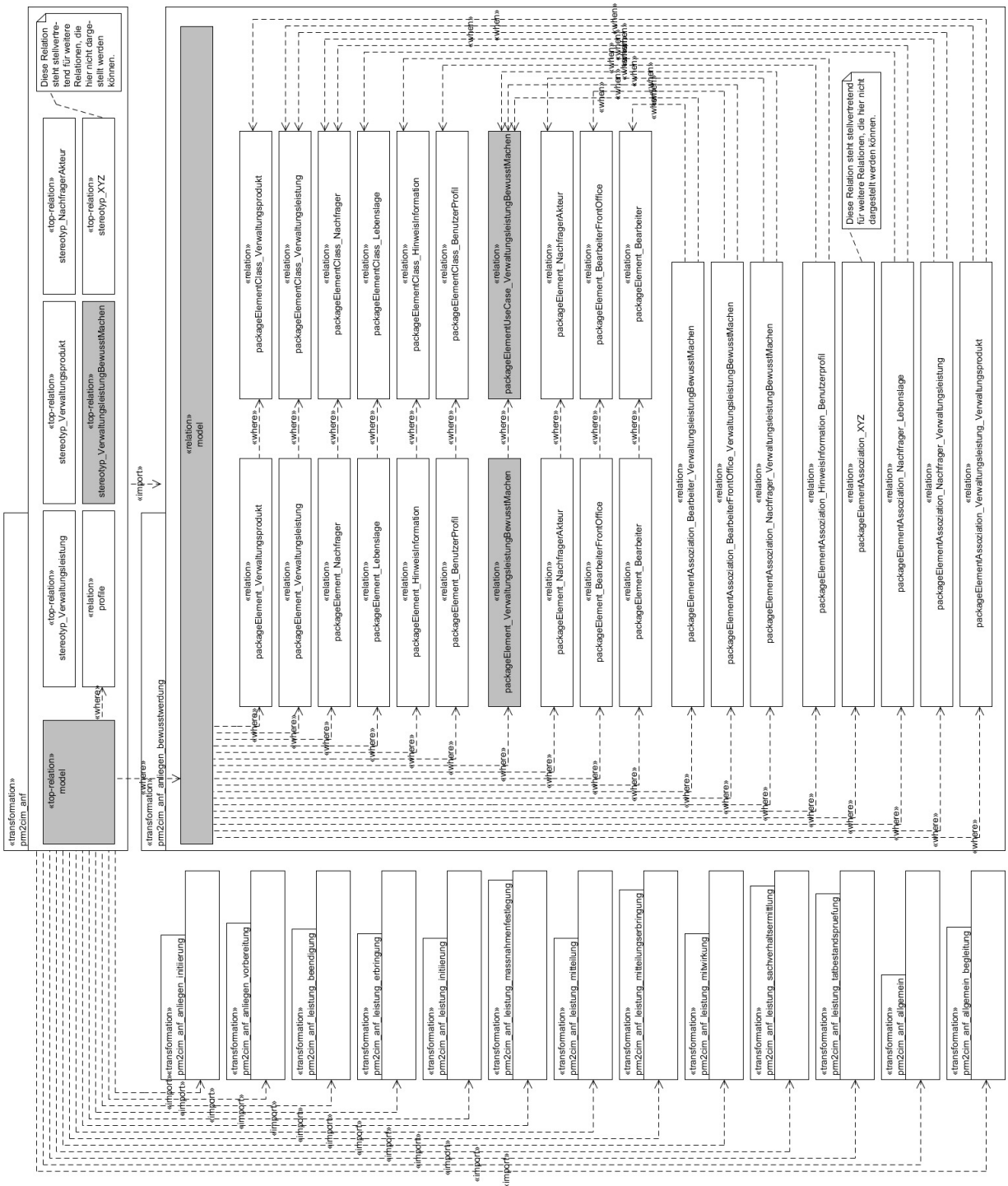


Abbildung 82 –PRM-CIM-Transformation des Anforderungsmodells (Auszug)

Die QVT-Implementierung der folgenden, in Abbildung 82 grau hinterlegten Abbildungsregeln wird im Detail vorgestellt, um die semantische Konkretisierung des PRM, basierend auf dem PRM, zu zeigen:

- `prm2cim_anf::model`: Entspricht der zuvor informell beschriebenen Abbildungsregel „Modell“ (QVT_PRM-CIM_ANF-01) in Tabelle 47 auf Seite 193.
- `prm2cim_anf_bewusstwerdung::model`: Aus technischen Gründen eingeführte neue Abbildungsregel, die Abbildungsregeln des Anwendungsfalls „Verwaltungsleistung bewusst werden“ zusammenfasst.

- `prm2cim_anf_bewusstwerdung::packageElement_VerwaltungsleistungBewusstMachen`: Bildet zusammen mit `prm2cim_anf_bewusstwerdung::packageElement UseCase_Verwaltungsleistung-BewusstMachen` die zuvor informell beschriebene Abbildungsregel „AnwendungsfallVerwaltungsleistungBewusstMachen“ (QVT_PRM-CIM_BWST-02) in Tabelle 54 auf Seite 194.

- `prm2cim_anf_bewusstwerdung::packageElementUseCase_Verwaltungsleistung-BewusstMachen`: Bildet zusammen mit `prm2cim_anf_bewusstwerdung::packageElement_VerwaltungsleistungBewusstMachen` die zuvor informell beschriebene Abbildungsregel „AnwendungsfallVerwaltungsleistungBewusstMachen“ (QVT_PRM-CIM_BWST-02) in Tabelle 54 auf Seite 194.

- `prm2cim_anf::stereotyp_VerwaltungsleistungBewusstMachen`: Entspricht der zuvor informell beschriebenen Abbildungsregel „StereotypVerwaltungsleistungBewusst-Machen“ (QVT_PRM-CIM_ANF-04) in Tabelle 50 auf Seite 193.

Der Haupt-Transformation `prm2cim_anf::model` (Listing 35) werden von der QVT-Laufzeitumgebung das PRM und das UML-Profil des eLoGo-Referenzanforderungsmodells sowie das Standard UML-Profil *L1* als Ausgangsmodell übergeben. Das CIM ist das Zielmodell der Transformation, so dass die Signatur der Transformation die drei Modelle `prm`, `profiles` und `target` umfasst (Zeile 9). Die übergebenen Modelle werden im Eclipse-Konfigurationsdialog zur Ausführung der QVT festgelegt. Eine exemplarische Konfiguration zeigt Abbildung 83. Es ist dargestellt, wie für die Ausführung des QVT-Skripts „prm2cim_anf.qvt“ das Modell „prm_stvg_vwv-stvo.uml“ als Ausgangsmodell der Transformation, die Modelle „eGovRefAnf.uml“ und „Uml2ProfileL1.uml“ als unterstützende Modelle und das Modell „cim_stvg_vsv-stvo.uml“ als Zielmodell der Transformation festgelegt sind.

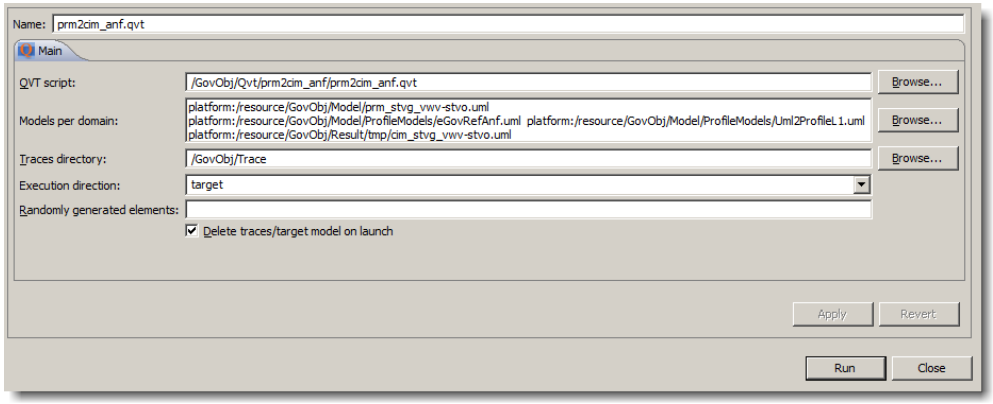


Abbildung 83 – Ausschnitt aus dem Eclipse-Konfigurationsdialog zur Ausführung von QVT

In der Abbildungsregel „model“ (Listing 35, Zeile 13) werden Modell und Package des PRM auf ein Modell und die Package-Struktur des Zielmodells abgebildet. Es werden in den Zeilen 24 bis 32 die Packages Anliegen, Verwaltungsleistung und Querschnittsfunktionen angelegt. In der Where-Klausel dieser Abbildungsregel werden die Transformationen der einzelnen Anwendungsfälle referenziert. In Zeile 38 erfolgt beispielsweise die Einbindung der Abbildungsregel `prm2cim_anf_anliegen_bewusstwerdung::model` aus der in Zeile 1 importierten Transformation. Abschließend wird die Abbildungsregel `profile` referenziert (Zeile 52), die eine Verbindung zum UML-Profil des eLoGo-Referenzanforderungsmodells anlegt.[385]

[385] Wegen einer Beschränkung in der verwendeten Werkzeugunterstützung, basierend auf dem Eclipse M2M-Projekt, und einer Änderungen beim Versionsübergang des EMF-Projektes von 2.5 auf 2.6 ist es nicht möglich, ein UML-Profil innerhalb einer Transformation mit der QVT Relation Language UML-dem Modell zuzuweisen. (Freundliche Auskunft von Edward Willink, dem verantwortlichen

```
(1)   import prm2cim_anf_anliegen_bewusstwerdung;
(2)   -- ...
(3)   /*
(4)    * Transformation von PRM nach CIM, basierend auf dem UML-Profil
(5)    * des eLoGo-Referenzanforderungsmodells für die Anwendungsfall-
(6)    * und Klassendiagramme einer Anforderungsspezifikation
(7)    */
(8)
(9)   transformation prm2cim_anf(prm: uml, profiles: uml, target : uml) {
(10)
(11)    -- ...
(12)
(13)    top relation model {
(14)
(15)      n : String;
(16)
(17)      checkonly domain prm srcMdl :uml::Model{
(18)        name = n,
(19)        packagedElement = srcPack : uml::Package { }
(20)      };
(21)
(22)      enforce domain target dstMdl: uml::Model{
(23)        name = n,
(24)        packagedElement = packAnliegen : uml::Package {
(25)          name = n + ' Anliegen'
(26)        },
(27)        packagedElement = packVerwaltungsleistung : uml::Package {
(28)          name = n + ' Verwaltungsleistung'
(29)        },
(30)        packagedElement = packVerwaltungsleistung : uml::Package {
(31)          name = n + ' Querschnittsfunktionen'
(32)        }
(33)      };
(34)
(35)      where {
(36)        -- Package "Anliegen"
(37)        -- Abbildungsregeln für den Anwendungsfall "Verwaltungleistung bewusst
      machen"
(38)        prm2cim_anf_anliegen_bewusstwerdung::model(srcPack, packAnliegen);
(39)        -- Abbildungsregeln für den Anwendungsfall "Bearbeitung beobachten"
(40)        -- ...
(41)        -- Package "Verwaltungsleistung"
(42)        -- Abbildungsregeln für den Anwendungsfall "Bearbeitung beenden"
(43)        -- ...
(44)        -- Abbildungsregeln für den Anwendungsfall "Leistung erbringen"
(45)        -- ...
(46)
(47)        -- Package "Querschnittsfunktionen"
(48)        -- Abbildungsregeln für den Anwendungsfall "Bearbeitung begleiten"
(49)        -- ...
(50)
(51)        -- Profil
(52)        profile(srcMdl, dstMdl);
(53)      }
(54)    }
(55)    -- ...
(56)    top relation stereotype_NachfragerAkteur {
(57)    -- siehe Listing 39 auf Seite 203
(58)    }
(59)  }
```

Listing 35 – Abbildungsregel „model" (Ausschnitt der QVT-Transformation „prm2cim_anf")

Die in Zeile 38 der prm2cim_anf-Transformation referenzierte Abbildungsregel `prm2cim_anf_anliegen_ bewusstwerdung::model` ist in Listing 36 dargestellt. Ihr werden das Package des Ausgangsmodells und das Anliegen-Package des Zielmodells

Eclipse-Entwickler im Eclipse-Forum, unter http://www.eclipse.org/forums/index.php/mv/msg/210733/677173/#msg_677173, letzter Aufruf: 21.08.2011). Deshalb wird im Ansatz dieser Arbeit die Zuweisung des Profils durch eine XSLT durchgeführt, die auf das Ergebnis der Transformation angewendet wird, bevor das endgültige Modell in TOPCASED (oder einem anderen Modellierungswerkzeug) verwendet werden kann. Das Listing 60 ist in Anhang F.2 dargestellt.

übergeben. Sie referenziert in ihrer Where-Klausel die Abbildungsregeln der weiteren Modellelemente zum Domänenobjektmodell und zum Anwendungsfallmodell. Für die Erzeugung des Anwendungsfalls „Verwaltungsleistung bewusst machen“ ist die Relation packageElement_VerwaltungsleistungBewusstMachen referenziert (Zeile 23).

```
(1)     -- ...
(2)     /*
(3)      * Relation zur Transformation des Modells in den Anwendungsfall
(4)      * "Verwaltungsleistung bewusst machen"
(5)      */
(6)     relation model {
(7)
(8)       checkonly domain prm srcPack :uml::Package{ };
(9)
(10)      enforce domain target packAnliegen : uml::Package { };
(11)
(12)      where {
(13)        -- Domänenobjektmodell
(14)        -- Klassen
(15)        packageElement_Verwaltungsprodukt(srcPack, packAnliegen);
(16)        -- ...
(17)        -- Beziehungen
(18)        packageElementAssoziation_Verwaltungsleistung_Verwaltungsprodukt(srcPa
      ck,
              packAnliegen);
(19)        -- ...
(20)
(21)        -- Anwendungsfallmodell
(22)        -- Anwendungsfall
(23)        packageElement_VerwaltungsleistungBewusstMachen(srcPack,
      packAnliegen);
(24)        -- Akteure
(25)        packageElement_NachfragerAkteur(srcPack, packAnliegen);
(26)        -- ...
(27)        -- Beziehungen
(28)        packageElementAssoziation_Nachfrager_VerwaltungsleistungBewusstMachen(
      srcPack,
              packAnliegen);
(29)        -- ...
(30)
(31)      }
(32)    }
(33)    -- ...
```

Listing 36 – Abbildungsregel „model“ (Ausschnitt der QVT-Transformation „prm2cim_anf_anliegen_bewusstwerdung“)

Die Abbildungsregel packageElement_VerwaltungsleistungBewusstMachen (Listing 37) erzeugt, ausgehend von einer Instanz der Klasse VwLeistung im Ausgangsmodell, den Anwendungsfall „Verwaltungsleistung bewusst machen“. Dazu wird ihr das Package des Ausgangsmodells und das Anliegen-Package des Zielmodells übergeben. Gibt es im Package des Ausgangsmodells eine Instanz der Klasse VwLeistung, wird in der When-Klausel sichergestellt, dass diese keine Verfeinerung hat, die stattdessen zu bearbeiten wäre (Zeile 23). Nur wenn diese Bedingung erfüllt ist, wird in der Where-Klausel durch die Abbildungsregel packageElementUseCase_VerwaltungsleistungBewusstMachen mit der semantischen Konkretisierung des Anwendungsfalls fortgefahren (Zeile 27).

```
(1)     -- ...
(2)     /*
(3)      * "Verwaltungsleistung bewusst machen" als Element (Alternative)
      ausgehend
(4)      * von Verwaltungshandlungsgegenstand "VwLeistung" und Verwaltungsakteur
(5)      * "VwAntragsteller".
(6)      */
(7)     relation packageElement_VerwaltungsleistungBewusstMachen {
(8)
(9)       checkonly domain prm srcPack :uml::Package {
(10)        packagedElement = srcInst1 : uml::InstanceSpecification {
(11)          classifier = clsf1 : uml::Class {
(12)            name = 'VwLeistung'
```

```
(13)          }
(14)        }
(15)      };
(16)
(17)      enforce domain target  pack : uml::Package {
(18)        packagedElement = uc : uml::UseCase { }
(19)
(20)      };
(21)
(22)      when {
(23)        not(hasRefinement(srcInst1));
(24)      }
(25)
(26)      where {
(27)        packageElementUseCase_VerwaltungsleistungBewusstMachen(srcInst1, uc);
(28)      }
(29)    }
(30)  -- ...
```

Listing 37 – Abbildungsregel „packageElement_VerwaltungsleistungBewusstMachen“ (Ausschnitt der QVT-Transformation „prm2cim_anf_anliegen_bewusstwerdung“)

Zur semantischen Konkretisierung des Anwendungsfalls werden der Abbildungsregel packageElementUseCase_VerwaltungsleistungBewusstMachen (Listing 38) die Instanz aus dem PRM-Ausgangsmodell und der angelegte Anwendungsfall übergeben. Zum Anwendungsfall werden Name und Dokumentation mit entsprechenden Werten belegt. Der Name wird in Zeile 23 zugewiesen, indem er aus der Bezeichnung der Instanz aus dem PRM und dem Text „bewusst machen“ zusammengesetzt wird. Die Dokumentation, die hier als EAnnotation (für das Werkzeug TOPCASED) angelegt wird, umfasst das Ziel, die Vor- und Nachbedingungen, den Ablauf des Anwendungsfalls sowie die Quelle der Instanz aus dem PRM. Beispielsweise wird der Ablauf in Zeile 59 aus dem Stereotyp des eGovRefAnf UML-Profils ermittelt und dann durch die im PRM modellierten Hinweise und Verwaltungsprodukte ergänzt. Sie sind in der prototypischen Implementierung in den Zeilen 46 und 47 aus allen Instanzen ermittelt worden. Dann wurden ihre Bezeichnungen zu einer Zeichenkette zusammengefügt und können so in Zeile 60 in den Text zum Ablauf des Anwendungsfalls einfließen.

```
(1)   -- ...
(2)     /*
(3)      * UseCase "Verwaltungsleistung bewusst machen" (Alternative) ausgehend
(4)      * von Verwaltungshandlungsgegenstand "VwLeistung"
(5)      */
(6)
(7)     relation packageElementUseCase_VerwaltungsleistungBewusstMachen {
(8)       nvl, nvp, nvhi : String;
(9)       b, a, q : String;
(10)      vbp, nbp, zlp, abp : String;
(11)      vb, nb, zl, ab : String;
(12)      elem : OrderedSet(uml::Element);
(13)      inst : OrderedSet(uml::InstanceSpecification);
(14)
(15)      checkonly domain prm srcInst1 : uml::InstanceSpecification {
(16)        name = nvl,
(17)        classifier = clsf : uml::Class {
(18)          name = 'VwLeistung'
(19)        }
(20)      };
(21)
(22)      enforce domain target uc : uml::UseCase{
(23)        name = nvl + ' bewusst machen',
(24)        -- Dokumentation
(25)        eAnnotations = doc : ecore::EAnnotation {
(26)          source = 'http://www.topcased.org/documentation',
(27)          eModelElement = uc,
(28)          details = dtls : ecore::EStringToStringMapEntry {
(29)            value = '1. Ziel: ' + zl + '; 2. Vorbedingung: ' + vb + '; 3.
      Nachbedingung: '
                + nb + '4. Ablauf: ' + ab + '; 5. Quelle: ' + q,
(30)            _key = 'documentation'
(31)          }
```

```
(32)        }
(33)      };
(34)
(35)      where {
(36)        -- Quelle, Arbeitsnotiz und Beschreibung aus der Instanz ermitteln
(37)        q = getQuelle(srcInst1);
(38)        a = getArbeitsnotiz(srcInst1);
(39)        b = getBeschreibung(srcInst1);
(40)
(41)        -- Informationen im PRM auswerten (hier prototypischer Zugriff
(42)        -- auf alle Instanzen von HinweisInformation und Produkt
(43)        -- ohne die Beziehungen zwischen den Elementen zu beachten)
(44)        elem = srcInst1.allOwningPackages()->first().allOwnedElements();
(45)        inst = elem->select(e | e.oclIsTypeOf(uml::InstanceSpecification));
(46)        nvhi = concatNames(inst->select(i | i.classifier->exists(c |
                    c.name='VwHinweisInformation')));
(47)        nvp = concatNames(inst->select(i | i.classifier->exists(c |
      c.name='VwRechtsfolge')));
(48)
(49)        -- Ziel
(50)        zlp = getAnwendungsfallZiel('Verwaltungsleistung bewusst machen');
(51)        zl = zlp.replace('einer Verwaltungsleistung', 'von ' + nvp);
(52)        -- Vorbedingung
(53)        vb = getAnwendungsfallVorbedingung('Verwaltungsleistung bewusst
      machen');
(54)
(55)        -- Nachbedingung
(56)        nbp = getAnwendungsfallNachbedingung('Verwaltungsleistung bewusst
      machen');
(57)        nb = nbp.replace('Verwaltungsleistungen wurden', 'Verwaltungsleistung
      ' + nvl +
                  ' wurde(n)').replace('Hinweise', 'Hinweise auf ' + nvhi);
(58)        -- Ablauf
(59)        abp = getAnwendungsfallAblauf('Verwaltungsleistung bewusst machen');
(60)        ab = abp.replace('aus den Hinweisen', 'aus ' + nvhi).replace(
                    'Verwaltungsprodukte und', 'Verwaltungsprodukte (' + nvp + ')
      und');
(61)      }
(62)    }
(63)    -- ...
```

Listing 38 – Abbildungsregel „packageElementUseCase_VerwaltungsleistungBewusstMachen“ (Ausschnitt der QVT-Transformation „prm2cim_anf_anliegen_bewusstwerdung“)

Für die angelegten Modellelemente werden abschließend Stereotypen aus dem UML-Profil des eLoGo-Referenzanforderungsmodells zugeordnet, um die vorgenommene semantische Konkretisierung des Modellelements darzustellen. Das Listing 39 zeigt die Zuordnung des Stereotyps «Verwaltungsleistung bewusst machen» zum Anwendungsfall im Zielmodell (Zeilen 11 und 12), wenn der Anwendungsfall aus einer Instanz der Klasse `VwLeistung` hervorgegangen ist (Zeilen 6 und 7) und bereits im Modell angelegt wurde (`When`-Klausel in Zeile 16). Da die Stereotypen außerhalb des UML-Modells aber innerhalb der XMI-Definition angelegt werden müssen, sind die Relationen zur Zuordnung von Stereotypen als `top relation` implementiert. Im Ergebnis dieser Abbildungsregeln wird der in Abbildung 76 auf Seite 190 dargestellte Anwendungsfall inkl. Stereotyp und Dokumentation im Zielmodell erzeugt.

```
(1)     -- ...
(2)     top relation stereotype_VerwaltungsleistungBewusstMachen {
(3)       n : String;
(4)       checkonly domain prm srcInst : uml::InstanceSpecification {
(5)         name = n,
(6)         classifier = clsf : uml::Class {
(7)           name = 'VwLeistung'
(8)         }
(9)       };
(10)
(11)      enforce domain target bewh :
      eGovRefAnf::Verwaltungsleistungbewusstmachen {
(12)        base_UseCase = uc :uml::UseCase { }
(13)      };
(14)
(15)      when {
```

```
(16)        prm2cim_anf_anliegen_bewusstwerdung::packageElementUseCase_Verwaltungs
      leistungBewusstMachen(srcInst, uc);
(17)
(18)      }
(19)    }
(20)    -- ...
```

Listing 39 – Abbildungsregel „stereotype_VerwaltungsleistungBewusstMachen" (Ausschnitt der QVT-Transformation „prm2cim_anf")

Neben den funktionalen Anforderungen werden in den Transformationen auch nicht-funktionale Anforderungen berücksichtigt. Ein Beispiel hierfür ist die Implementierung der Transformation des Anwendungsfalls „Leistung erbringen" in Listing 40. Ergänzend zu den Abbildungsregeln, die Anwendungsfälle, Akteure und deren Beziehungen zwischen PRM und CIM abbilden, erfolgt auch die Abbildung weiterer relevanter Elemente, für die Rahmenbedingungen formuliert wurden auf Kommentare im CIM. In Zeile 30 ist dargestellt, wie diese Abbildungsregel für Rechtsfolgenmerkmale aufgerufen wird. Die Zeilen 41 bis 99 zeigen die Implementierung dieser Abbildungsregel und der aus ihr aufgerufenen weiteren Regel. Bei der Erzeugung des Kommentars in den Zeilen 90 bis 93 werden über die Hilfsfunktion `getNF()` der beschreibende Text in den Kommentar übernommen und die Beziehung zum Anwendungsfall hergestellt. Die Zuweisung des Stereotyps und der Tagged Values zum Element muss aufgrund der XMI-Struktur in der Haupt-Transformation „prm2cim_anf" erfolgen. Der entsprechende Ausschnitt dieser Transformation ist in Listing 41 dargestellt.

Im Ergebnis werden im PRM des Verwaltungshandeln formulierte Hinweise auf Rahmenbedingungen in Kommentare innerhalb der Modelle der Anforderungsspezifikation generiert, denen ein entsprechender Stereotyp und eine Beschreibung hinzugefügt wird. Ein Beispiel für das Bewohnerparken zeigte Abbildung 80 auf Seite 192.

```
(1)   -- Anwendungsfall "Leistung erbringen"
(2)   transformation prm2cim_anf_leistung_erbringung(prm : uml, target : uml) {
(3)
(4)     -- Schlüssel die die Identität des Elements bestimmen
(5)     key uml::UseCase {name};
(6)
(7)     /*
(8)      * Relation zur Transformation des Modells in den Anwendungsfall
(9)      * "Verwaltungsleistung bewusst machen"
(10)     */
(11)    relation model {
(12)
(13)      checkonly domain prm srcPack :uml::Package{ };
(14)
(15)      enforce domain target packVerwaltungsleistung : uml::Package {  };
(16)
(17)      where {
(18)
(19)        -- Anwendungsfallmodell
(20)        -- Anwendungsfall
(21)        -- ...
(22)
(23)        -- Akteure
(24)        -- ...
(25)
(26)        -- Beziehungen
(27)        -- ...
(28)
(29)        -- Nicht-funktionale Anforderungen
(30)        packageElement_RechtsfolgeMerkmalNF(srcPack, packVerwaltungsleistung);
(31)        -- ...
(32)
(33)      }
(34)    }
(35)
(36)  -- ...
(37)
(38)    /*
(39)     * Nicht-funktionale Anforderungen des RechtsfolgenMerkmal als Element
```

```
(40)     */
(41)    relation packageElement_RechtsfolgeMerkmalNF {
(42)
(43)      checkonly domain prm srcPack :uml::Package {
(44)        packagedElement = srcInst1 : uml::InstanceSpecification {
(45)          classifier = clsf : uml::Class {
(46)            name = 'VwRechtsfolgenMerkmal'
(47)          }
(48)        },
(49)
(50)        packagedElement = srcInst2 : uml::InstanceSpecification {
(51)          classifier = clsf2 : uml::Class {
(52)            name = 'VwLeistung'
(53)          }
(54)        }
(55)      };
(56)
(57)      enforce domain target  pack : uml::Package {
(58)        ownedComment = cmt : uml::Comment {
(59)        }
(60)      };
(61)
(62)      when {
(63)        not(hasRefinement(srcInst1));
(64)        hasDesigntimeNF(srcInst1) or hasRuntimeNF(srcInst1);
(65)      }
(66)
(67)      where {
(68)        packageElementComment_RechtsfolgeMerkmalNF(srcInst1, srcInst2, cmt);
(69)      }
(70)
(71)    }
(72)
(73)    /*
(74)     * * Nicht-funktionale Anforderungen des RechtsfolgenMerkmal als Kommentar
(75)     */
(76)    relation packageElementComment_RechtsfolgeMerkmalNF {
(77)
(78)      checkonly domain prm srcInst1 : uml::InstanceSpecification {
(79)        classifier = clsf1 : uml::Class {
(80)          name = 'VwRechtsfolgenMerkmal'
(81)        }
(82)      };
(83)
(84)      checkonly domain prm srcInst2 : uml::InstanceSpecification {
(85)        classifier = clsf2 : uml::Class {
(86)          name = 'VwLeistung'
(87)        }
(88)      };
(89)
(90)      enforce domain target cmt : uml::Comment{
(91)          _body = getNF(srcInst1),
(92)          annotatedElement = uc : uml::UseCase{ }
(93)      };
(94)
(95)      when {
(96)        packageElementUseCase_LeistungErbringen(srcInst2, uc);
(97)      }
(98)
(99)    }
(100)
(101) -- ...
(102)
(103)   /*************************************************************************
(104)    *
(105)    * Hilfsfunktionen
(106)    *
(107)    *************************************************************************/
(108)   /*
(109)
(110) -- ...
(111)
(112)    * Liefert den Wert des Attributes 'istRahmenbedingungEinsatz'. Wenn es
      kein Attribut mit
```

```
(113)    * diesem Namen gibt oder der Wert nicht gesetzt ist, wird _false
      zurückgeliefert.
(114)    */
(115)   query hasRuntimeNF(inst : uml::InstanceSpecification) :  Boolean {
(116)     if (inst.slot->exists(x : uml::Slot | x.definingFeature.name =
      'istRahmenbedingungEinsatz')) then
(117)      inst.slot->asSequence()->select(x : uml::Slot | x.definingFeature.name
      = 'istRahmenbedingungEinsatz')->first().value->first().booleanValue()
(118)     else
(119)      false
(120)     endif
(121)   }
(122)
(123)   /*
(124)    * Liefert den Wert des Attributes 'istRahmenbedingungUmsetzung'. Wenn es
      kein Attribut mit
(125)    * diesem Namen gibt oder der Wert nicht gesetzt ist, wird _false
      zurückgeliefert.
(126)    */
(127)   query hasDesigntimeNF(inst : uml::InstanceSpecification) :  Boolean {
(128)     if (inst.slot->exists(x : uml::Slot | x.definingFeature.name =
      'istRahmenbedingungUmsetzung')) then
(129)      inst.slot->asSequence()->select(x : uml::Slot | x.definingFeature.name
      = 'istRahmenbedingungUmsetzung')->first().value->first().booleanValue()
(130)     else
(131)      false
(132)     endif
(133)   }
(134)
(135)   /*
(136)    * Liefert den Wert des Attributes 'istRahmenbedingungUmsetzung'. Wenn es
      kein Attribut mit
(137)    * diesem Namen gibt oder der Wert nicht gesetzt ist, wird _false
      zurückgeliefert.
(138)    */
(139)   query getNF(inst : uml::InstanceSpecification) :  String {
(140)     if (inst.slot->exists(x : uml::Slot | x.definingFeature.name =
      'Rahmenbedingung')) then
(141)      inst.slot->asSequence()->select(x : uml::Slot | x.definingFeature.name
      = 'Rahmenbedingung')->first().value->first().stringValue()
(142)     else
(143)      '- keine -'
(144)     endif
(145)   }
(146) }
```

Listing 40 – Verarbeitung von Hinweisen auf nicht-funktionale Rahmenbedingungen als Kommentare

```
(1)   -- ...
(2)   top relation stereotype_RechtsfolgenMerkmalNF {
(3)
(4)       checkonly domain prm srcInst1 : uml::InstanceSpecification {
(5)         classifier = clsf1 : uml::Class {
(6)           name = 'VwRechtsfolgenMerkmal'
(7)         }
(8)       };
(9)
(10)      checkonly domain prm srcInst2 : uml::InstanceSpecification {
(11)        classifier = clsf2 : uml::Class {
(12)          name = 'VwLeistung'
(13)        }
(14)      };
(15)
(16)      enforce domain target bewh : eGovRefAnf::Rahmenbedingung{
(17)        ausfuehrungsbezogen =
      prm2cim_anf_leistung_erbringung::hasRuntimeNF(srcInst1),
(18)        entwicklungsbezogen =
      prm2cim_anf_leistung_erbringung::hasDesigntimeNF(srcInst1),
(19)        base_Comment =  cmt : uml::Comment { }
(20)      };
(21)
(22)      when {
(23)        prm2cim_anf_leistung_erbringung::packageElementComment_RechtsfolgeMerk
      malNF(srcInst1, srcInst2, cmt);
(24)      }
```

```
(25)     }
(26)   -- ...
```

Listing 41 – Zuweisung des Stereotyp «Rahmenbedingung» zu Kommentaren mit Hinweisen auf nicht-funktionale Anforderungen

5.4 Verfolgbarkeit der QVT-Transformationen

Der QVT-Standard stellt die in den Konzepten der MDA festgelegte Verfolgbarkeit von Transformationen mit Hilfe von Verfolgbarkeitsklassen und Verfolgbarkeitsinstanzen sicher. Sie werden während der Durchführung von QVT-Transformationen aufgezeichnet (vgl. Abschnitt 2.2). Zur Umsetzung dieses Standards generiert das in dieser Arbeit eingesetzte Werkzeug mediniQVT in einem konfigurierbaren Verzeichnis (vgl. Abbildung 83 auf Seite 198) Informationen zu ausgeführten Transformationen in getrennten Dateien für die Verfolgbarkeitsklassen und deren Instanzen auf. In Abbildung 84 ist das Ergebnis der in den Abschnitten 5.2 und 5.3 vorgestellten Transformationen zur Erzeugung von Prozess- und Anforderungsmodellen dargestellt. Anhand des Namens der Datei wird ihr Bezug zur entsprechenden Transformation mit gleichem Namen deutlich.

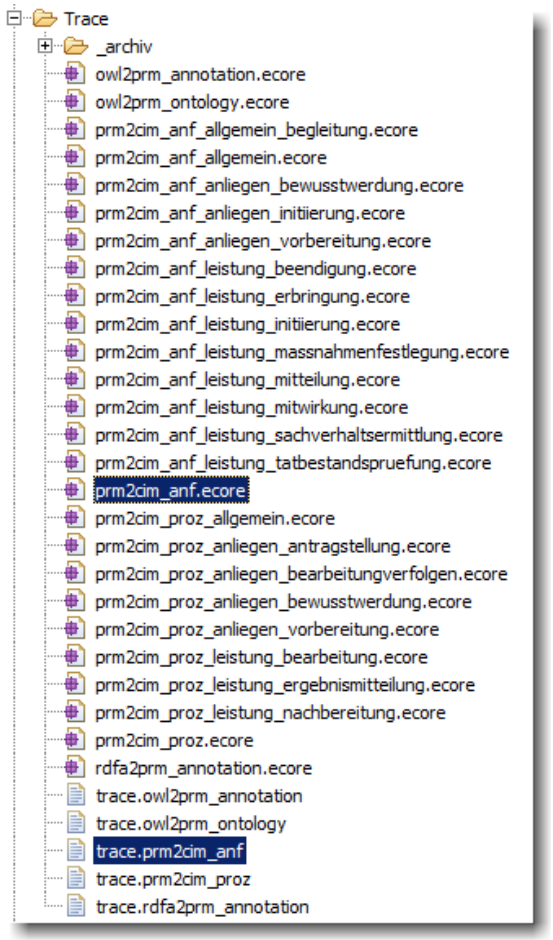


Abbildung 84 – Beispiel aufgezeichneter Verfolgbarkeitsinformationen der QVT-Transformationen

Die in Abbildung 84 markierte Datei `prm2cim_anf.ecore` ist die Darstellung der Verfolgbarkeitsklassen zur Transformation `prm2cim_anf` als Ecore-Modell. Die in diesem Modell enthaltenen Klassen entsprechen den Abbildungsregeln der Transformation. Beispielsweise wird die in Listing 35 dargestellte Regel `model` durch eine entsprechende Klasse `model` repräsentiert. Die Abbildung 85 stellt den entsprechenden Ausschnitt des Ecore-Modells innerhalb von Eclipse als Struktur (links oben), als Ecore-Diagramm (links oben) und als entsprechendes XML-Dokument (unten) dar.

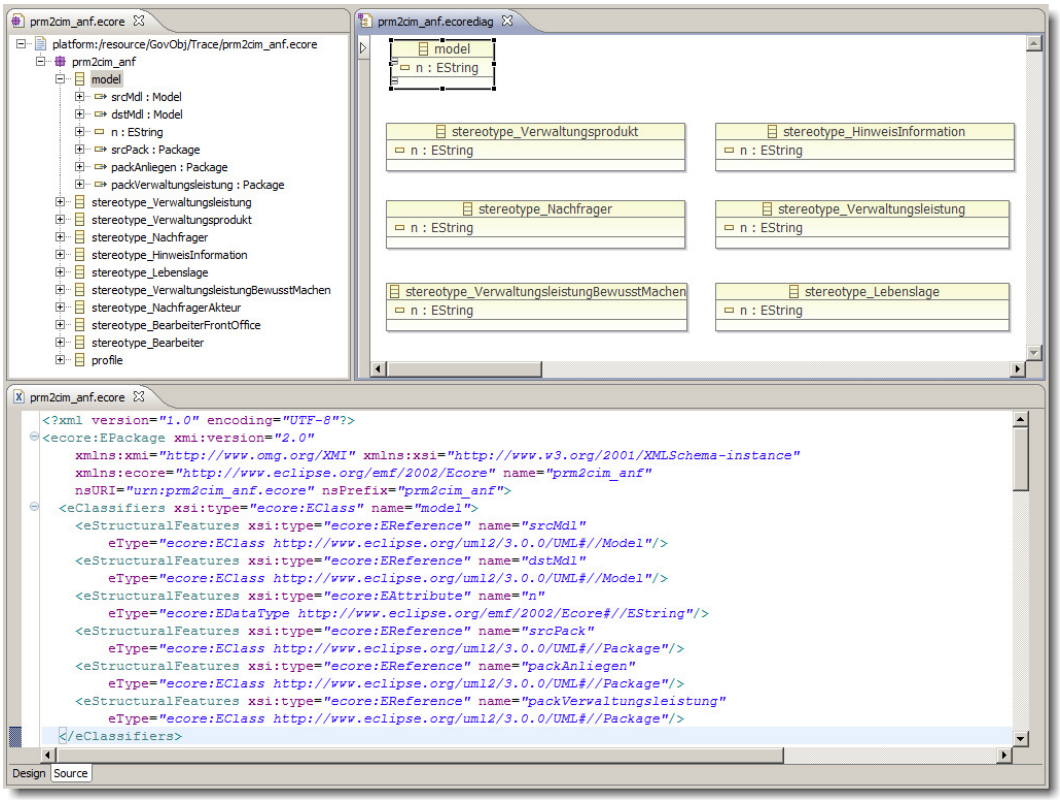


Abbildung 85 – Verfolgbarkeitsklassen aus der Transformation „prm2cim_anf" (Auszug)

Die Verfolgbarkeitsinstanzen, die während der Ausführung einer Transformation erzeugt werden, sind in einer separaten Datei im XML-Format gespeichert. Das Listing 42 zeigt einen Ausschnitt aus der aufgezeichneten Verfolgbarkeitsinformation bei der Transformation der Instanz `BewohnerParkzone` (Klasse `VwAntragsteller`) in die Klasse `BewohnerParkzone` des Domänenobjektmodells. Zeile 2 enthält die Bezeichnung der Verfolgbarkeitsinstanz, mit der auf ihre Klassendefinition referenziert wird. Sie enthält als Unterelemente die Domänen und die in den Domänen verwendeten Elemente. Jedes der Elemente verfügt über eine Referenz (`href`-Eigenschaft) auf das von ihm repräsentierte Element im jeweiligen Ausgangs- und Zielmodell. Die Referenz besteht aus dem Dateinamen des Modells und der xmi:id des Elements.

```
(1)     <!-- ... -->
(2)     <prm2cim_anf_anliegen_bewusstwerdung:packageElement_Nachfrager>
(3)       <srcPack href="../Model/prm_stvg_vwv-stvo.uml#d2e1"/>
(4)       <pack href="../Result/tmp/cim_stvg_vwv-
      stvo.uml#_13JUMJKoEeC0ruDLflD2VQ"/>
(5)       <srcInst href="../Model/prm_stvg_vwv-stvo.uml#_El6bcJBIEeCpsaMFVQMH-g"/>
(6)       <clsf href="../Model/prm_ontology-govobj-
      ontologie.uml#VwAntragsteller_class"/>
(7)       <cls href="../Result/tmp/cim_stvg_vwv-
      stvo.uml#_13hHoJKoEeC0ruDLflD2VQ"/>
(8)     </prm2cim_anf_anliegen_bewusstwerdung:packageElement_Nachfrager>
(9)     <!-- ... -->
```

Listing 42 – Verfolgbarkeitsinstanz „packageElement_Nachfrager" im XML-Format (Auszug aus der Datei „trace.prm2cim_anf")

Um den durch die Verfolgbarkeitsinstanzen ausgedrückten Zusammenhang für den Bearbeiter lesbar und nachvollziehbar zu machen, kann die entsprechende Datei im Eclipse Sample Ecore Model Editor geöffnet werden.[386] Die Abbildung 86 zeigt die geöffnete Datei

[386] In der verwendeten Version von mediniQVT fehlte bei den erzeugten Verfolgbarkeitsinstanzen die Referenz auf das Schema (Schema location), was für die Verfolgbarkeitsklassen bereits behoben wurde (siehe Diskussion Nr. 61 unter http://projects.ikv.de/qvt/discussion/1/61, letzter Aufruf:

und die Verfolgbarkeitsinstanz aus Listing 42 (in abweichender Schreibweise dargestellt). Die Werteausprägungen der Attribute dieser Instanz sind im eingeblendeten Properties-Fenster sichtbar. Ecore Model Editor löst die Referenzen für die Darstellung im Properties-Fenster auf, so dass die referenzierten Modellelemente mit ihrem Stereotyp, Typ und Namen dargestellt werden.

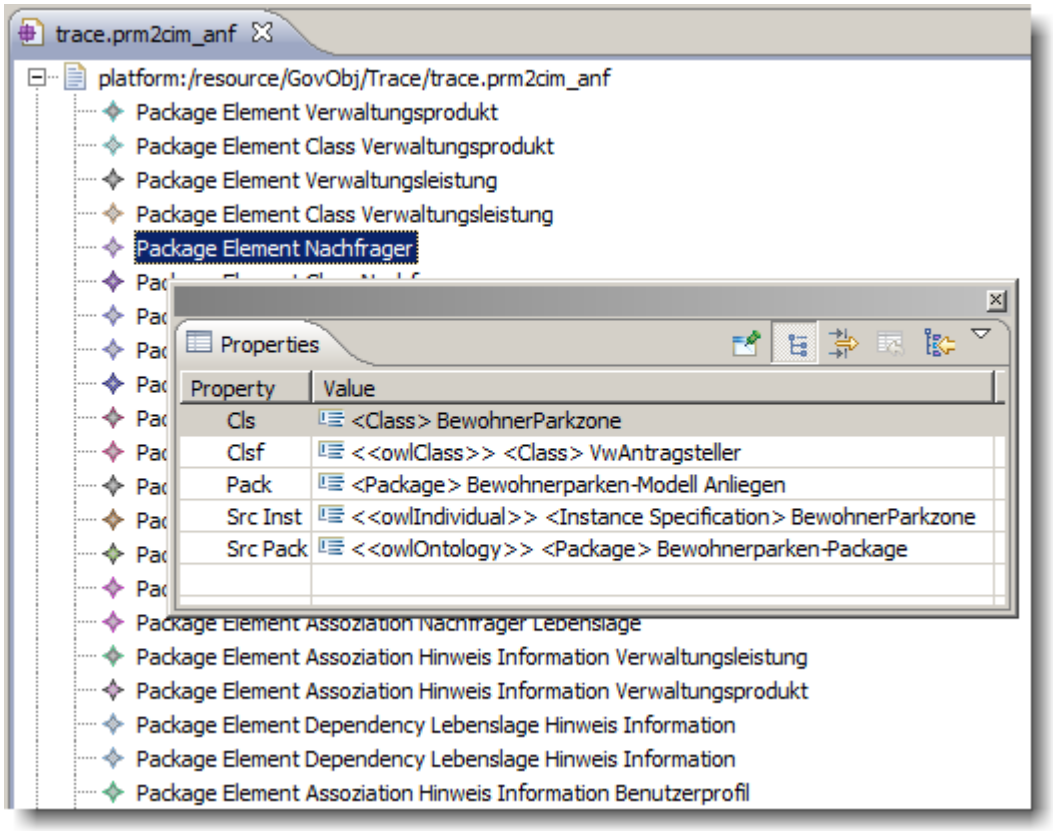


Abbildung 86 – Verfolgbarkeitsinstanz „packageElement_Nachfrager" im Sample Ecore Model Editor

Die Analyse der Verfolgbarkeitsinstanzen im Rahmen dieser Arbeit machte deutlich, dass ihre Aussagekraft für einen Bearbeiter von einer geeigneten Implementierung der Abbildungsregeln abhängt. Aussagekräftige Informationen werden aufgezeichnet, wenn die Ausgangs- und Zielelemente vollständig als Domänen oder Werte in Domänen verwendet werden und ihre Variablenbezeichnungen Aufschluss über den Zweck ihrer Verwendung geben. Für eine Abbildungsregel, die Ausgangs- und Zielelemente nicht vollständig als Domänen oder Werte in Domänen verwendet, werden weniger aussagekräftige Informationen aufgezeichnet. Ein Beispiel einer weniger aussagekräftigen Abbildungsregel ist `diagramPoolActivity_BewusstwerdungAufzeigen` (in Listing 34 auf Seite 187). Entsprechend des für die Erzeugung initialer Prozessmodelle gewählten Ansatzes wird das eLoGo-Referenzprozessmodell als Ausgangsmodell verwendet. Dieses Modell wird in der Transformation durch Informationen aus dem PRM konkretisiert. Diese Informationen des PRM werden in der `Where`-Klausel ermittelt und stehen zwar innerhalb der Transformation zur Erzeugung von Zielelementen zur Verfügung, werden in der resultierenden Verfolgbarkeitsinstanz jedoch nicht aufgezeichnet (Listing 43). Man erkennt lediglich eine Referenz zur Aktivität des Referenzmodells (Zeile 3), zum PRM-Package (Zeile 4) und zur Aktivität des Zielmodells (Zeile 5), während die in der `Where`-Klausel ermittelten Informationen fehlen. Bei Darstellung dieser Verfolgbarkeitsinstanz im Ecore Model Editor

21.08.2011). Für die Verfolgbarkeitsinstanzen muss daher als weitere Eigenschaft im xmi:XMI-Wurzelelement die Eigenschaft xsi:schemaLocation hinzugefügt werden (z.B. xsi:schema-Location="urn:prm2cim_anf.ecore prm2cim_anf.ecore urn:prm2cim_anf_anliegen_bewusst-werdung.ecore prm2cim_anf_anliegen_bewusstwerdung.ecore").

(Abbildung 87) werden die fehlenden Information deutlich, da alle Attribute der Verfolgbarkeitsklasse angezeigt werden.[387] Der Unterschied in der Aussagekraft der Verfolgbarkeitsinformation resultiert auch aus der Tatsache, dass das Referenzprozessmodell als zusätzliches Ausgangsmodell und nicht wie das Referenzanforderungsmodell als Profil bereitgestellt wurde. Durch die Bereitstellung als zusätzliches Ausgangsmodell und den sich daraufhin anbietenden vorlagenbasierten Ansatz wurde eine Implementierung begünstigt, die zu weniger aussagekräftigen Ergebnissen führt.

```
(1)     <!-- ... -->
(2)     <prm2cim_proz_anliegen_bewusstwerdung:diagramPoolActivity_BewusstwerdungAu
      fzeigen>
(3)       <refa href="../Model/eGovRefProz/refmdl_elogo-prozess_anliegen-
      bewusstwerdung1.bpmn#_rUcR03ysEeCNKqXrt7oadQ"/>
(4)       <pack href="../Model/prm_stvg_vwv-stvo.uml#d2e1"/>
(5)       <a href="../Result/cim_stvg_vwv-
      stvo_proz_anliegen_bewusstwerdung.bpmn#_Z3N44ZKYEeCHRsy3654I8A"/>
(6)     </prm2cim_proz_anliegen_bewusstwerdung:diagramPoolActivity_BewusstwerdungA
      ufzeigen>
(7)     <!-- ... -->
```

Listing 43– Verfolgbarkeitsinstanz „diagramPoolActivity_BewusstwerdungAufzeigen“ im XML-Format (Auszug aus der Datei „trace.prm2cim_proz“)

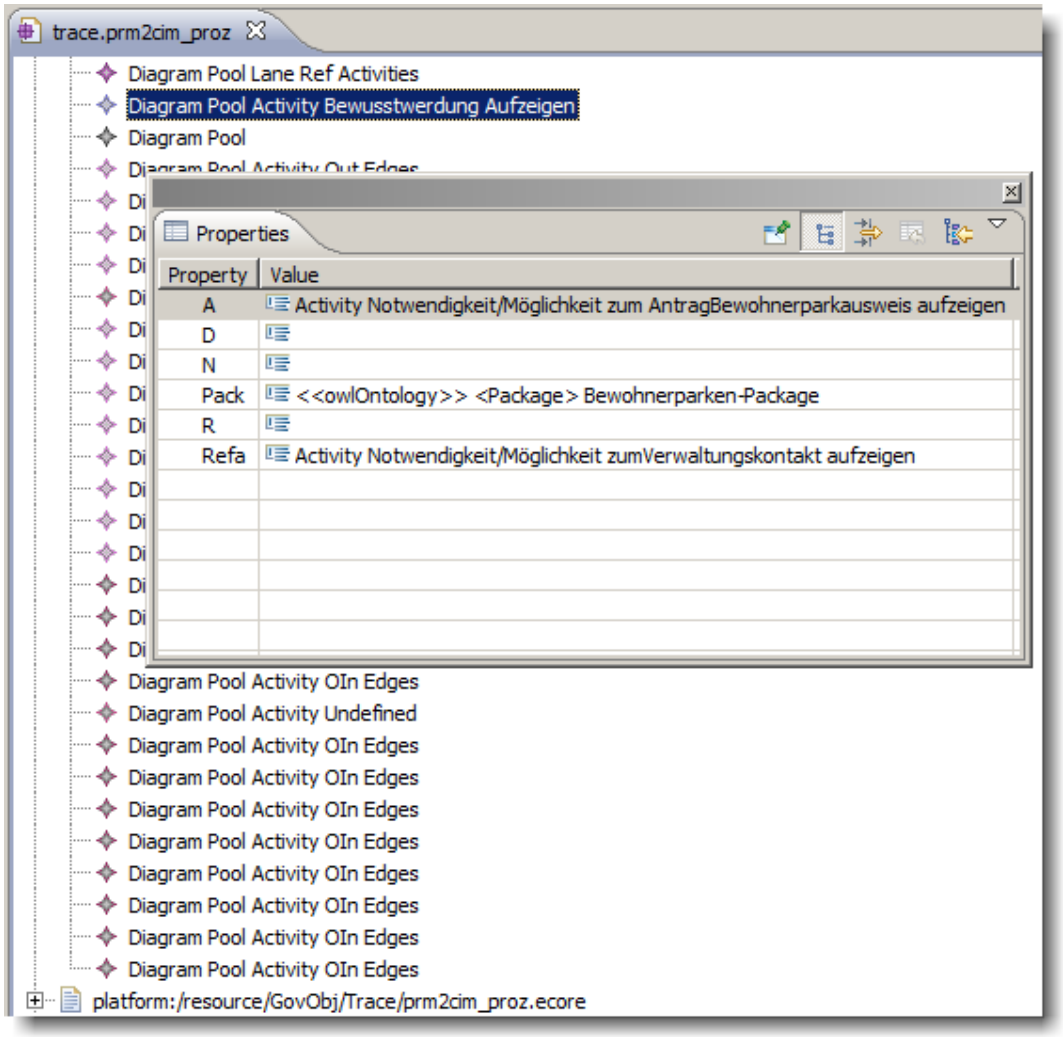


Abbildung 87 – Verfolgbarkeitsinstanz „diagramPoolActivity_BewusstwerdungAufzeigen“ im Sample Ecore Model Editor

[387] Man erkennt auch, dass die Aussagekraft der Verfolgbarkeitsinformation durch die gewählten Variablenbezeichnungen (z. B. D, N, R) leidet.

Weil der QVT-Standard die Nutzung der aufgezeichneten Verfolgbarkeitsklassen und -instanzen für inkrementelle Aktualisierungen der Modelle in den Vordergrund stellt, erfolgt die Aufzeichnung bezogen auf die Ausführung der Transformation für konkrete Dateien, die Ausgangs- und Zielmodelle enthalten. Wird die Transformation erneut, aber mit anderen Dateien ausgeführt, werden zuvor aufgezeichnete Verfolgbarkeitsinstanzen überschrieben. Dieser Effekt tritt bei der Transformation eines PRM des Verwaltungshandelns in die Prozessmodelle des CIM auf, da hier pro Subprozess des eLoGo-Referenzanforderungsmodells eine separate Datei angelegt wurde. Das Problem ist in diesem Fall dadurch lösbar, dass die bereits je Subprozess vorliegenden Transformationen nicht in einer übergeordneten Transformation zusammengefasst, sondern einzeln ausgeführt werden. Dann wird pro ausgeführter Transformation eine eigene Datei mit Verfolgbarkeitsinstanzen angelegt, die durch die Transformation anderer Subprozesse nicht überschrieben wird.

Die Aussagefähigkeit der aufgezeichneten Informationen ist auch dadurch beeinträchtigt, dass bei den Attributen der Verfolgbarkeitsklassen und den Werteausprägungen der Instanzen offen bleibt, ob sie sich auf das Ausgangs- oder das Zielmodell beziehen. In den vorangegangenen Beispielen konnte anhand des in der `href`-Eigenschaft angegebenen Verzeichnisses darauf geschlossen werden, ob es Ausgangs- oder Zielelement ist. Elemente, die sich auf das Verzeichnis „Model“ beziehen, sind Elemente des Ausgangsmodells und die sich auf das Verzeichnis „Result“ beziehenden Elemente gehören zum Zielmodell. Um dies deutlicher zu machen, könnte eine Namenskonvention für die Implementierung der Abbildungsregeln vorgegeben werden (z.B. Elemente des Ausgangsmodells sind mit dem Präfix „src“ als Source zu kennzeichnen).

Nachteilig ist für die Nutzung durch einen Bearbeiter weiterhin, dass durch die Verfolgbarkeitsklassen und ihre Instanzen die ausgeführten Abbildungsregeln in den Vordergrund gestellt werden. Möchte ein Bearbeiter die Verfolgbarkeitsbeziehungen eines konkreten Elements auswerten, muss er stets alle Verfolgbarkeitsinstanzen ermitteln, in denen dieses Element referenziert ist.

Trotz dieser Einschränkungen wird der durch eine Transformation erzeugte Zusammenhang zwischen den Elementen eines PRM des Verwaltungshandelns und den Elementen einer initialen Anforderungsspezifikation eines E-Government-Anwendungssystems entsprechend der im QVT-Standard vorgesehenen Mechanismen umfassend aufgezeichnet. Die Nutzung dieses Zusammenhangs als Verfolgbarkeitsinformation ist für einen Bearbeiter prinzipiell möglich.

5.5 Zusammenfassung

Den Ausgangspunkt dieses Kapitels bildete ein PRM, das instanziierte Konzepte einer Ontologie und deren Beziehungen formalisiert. Dieses Modell wurde in ein informationstechnikunabhängiges Modell (CIM) transformiert, das im Rahmen dieser Arbeit als initiale Anforderungsspezifikation dient. In Vorbereitung dieser Transformation wurden in beiden Modellen Unterschiede in den formalisierten Konzepten, in den verwendeten Modellelementen und in deren unterschiedlicher Semantik herausgearbeitet. Aus den Unterschieden resultierte eine Lücke, für deren Überbrückung in der PRM-CIM-Transformation der Einsatz von Referenzmodellen analog zum Einsatz eines Plattformmodells in der PIM-PSM-Transformation vorgeschlagen wurde.

Anhand eines PRM des Verwaltungshandelns wurde gezeigt, wie das Modell unter Verwendung der eLoGo-Referenzmodelle für E-Government transformiert werden kann. Dazu musste eine geeignete Bereitstellung der eLoGo-Referenzmodelle gefunden werden, um sie in den Transformationsprozess einfließen zu lassen. Das eLoGo-Referenzprozessmodell wurde als zusätzliches Ausgangsmodell im BPMN-Format bereitgestellt. Für das eLoGo-Referenzanforderungsmodell wurde ein entsprechendes UML-Profil entwickelt, das in der Transformation benutzt wurde. Anschließend wurden Regeln definiert, wie PRM-Elemente unter Zuhilfenahme des Referenzmodells und unter Berücksichtigung der zwischen ihnen existierenden Beziehungen auf Elemente des CIM abgebildet werden

können. Diese Regeln wurden prototypisch mit QVT implementiert und ausgeführt. Dabei kamen einfache Mechanismen aus dem Sprachumfang von QVT und OCL zum Einsatz, die im Wesentlichen auf der Ersetzung von Platzhaltern in der Dokumentation der Elemente des Referenzmodells durch Elemente aus dem PRM basierten. Grundsätzlich sind komplexere Mechanismen zur Konkretisierung des Referenzmodells denkbar.

Durch die Transformation wurde ein CIM im Sinne einer initialen Anforderungsspezifikation eines E-Government-Anwendungssystems erzeugt, das aus Prozessmodellen und Modellen der Anforderungsanalyse besteht. Die Prozessmodelle stellen Verwaltungsprozesse in Form von BPMN-Diagrammen dar. Die Modelle der Anforderungsanalyse, die in Form von Anwendungsfall- und Klassendigrammen der UML und ihrer Beschreibung erzeugt wurden, formalisieren funktionale und nicht-funktionale Anforderungen an das Anwendungssystem zur Unterstützung der Verwaltungsprozesse. Damit erfüllt der Ansatz die Anforderung Nr. 4 auf Seite 7, denn der nahtlose Übergang zum standardisierten modellgetriebenen Softwareentwicklungsprozess ist erreicht.[388]

Die durchgeführten Transformationen wurden entsprechend des QVT-Standards in Form von Verfolgbarkeitsklassen und -instanzen von der QVT-Laufzeitumgebung aufgezeichnet. Sie ermöglichen die Verfolgbarkeit der Transformation, indem die ausgeführten Abbildungsregeln mit den verwendeten Objekten und Werten dokumentiert sind. Bei der aufgezeichneten Verfolgbarkeitsdokumentation zeigten sich Auswirkungen der unterschiedlichen Bereitstellungsform der eLoGo-Referenzmodelle. Die vorlagenbasierte Transformation, die das eLoGo-Referenzprozessmodell als weiteres Ausgangsmodell verwendet, zeichnet weniger aussagekräftige Ergebnisse auf als die Transformation, die das eLoGo-Referenzanforderungsmodell als UML-Profil verwendet. Im Folgenden konzentriert sich die Darstellung deshalb auf den Teil des CIM, der die Anforderungsmodelle (Anwendungsfall- und Klassendiagramme) bildet. Die dargestellten Zusamenhänge und Prinzipien sind auch auf die Prozessdiagramme des CIM übertragbar.

Ausgehend von den aufgezeichneten Verfolgbarkeitsinformationen wurde gezeigt, wie bereits eine einfache Werkzeugunterstützung auf Eclipse-Basis es dem Benutzer ermöglicht, diese Informationen einzusehen und zu prüfen. Die weitergehende Nutzung der Verfolgbarkeitsinformationen in anderen Werkzeugen und in einem umfassenden Ansatz zur Verfolgbarkeit wird dadurch prinzipiell ermöglicht.

[388] Das dabei anhand des Beispiels Bewohnerparken erzielte Ergebnis ist mit der von Wolf et al. vorgeschlagenen idealtypischen Servicemodularisierung des Bürgerdienstes „Beantragung eines Anwohnerparkausweises“ vergleichbar (vgl. [Wolf et al., 2011], S. 208 f.).

6 Gesamtansatz zur Verfolgbarkeit

Im Rahmen der zuvor beschriebenen und durchgeführten Transformationsschritte wurden Verfolgbarkeitsinformationen aufgezeichnet. In diesem Kapitel werden die Informationen mit den weiteren entwickelten Elemente zu einem Gesamtansatz kombiniert, um durchgängige Verfolgbarkeit von den Elementen einer Rechtsvorschrift bis zu den Anforderungen in der Anforderungsspezifikation zu erreichen.

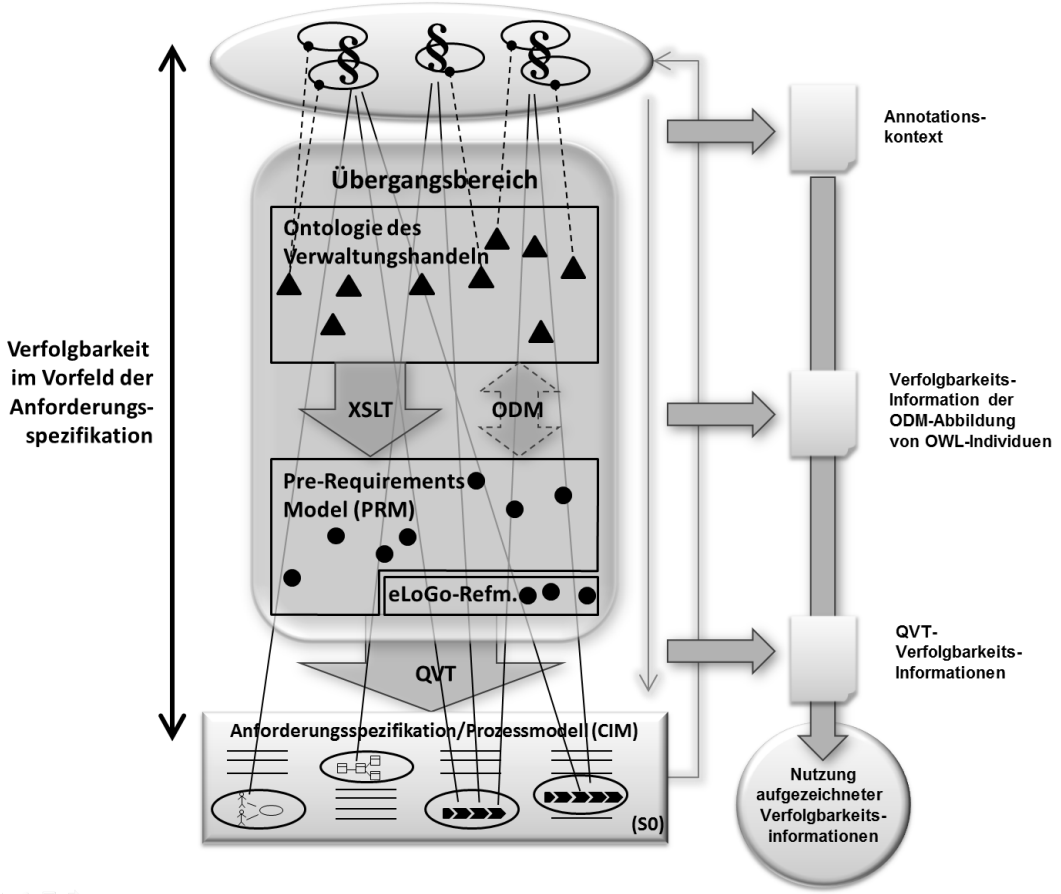


Abbildung 88 – Durchgängige Verfolgbarkeit durch Kombination der aufgezeichneten Verfolgbarkeitsinformationen

6.1 Einordnung der Bestandteile

In den vorangegangenen Kapiteln wurden isoliert zentrale Bestandteile des Ansatzes eingeführt: In Kapitel 3 wurden die im Rahmen dieser Arbeit entwickelte Ontologie des Verwaltungshandelns und die auf ihren Konzepten basierenden Annotationen von Rechtsvorschriften eingeführt. Das ebenfalls im Rahmen dieser Arbeit als Erweiterung der MDA entwickelte Pre-Requirements Model (PRM) und seine Erzeugung durch Transformation von annotierten Rechtsvorschriften wurden in Kapitel 4 beschrieben. Das Kapitel 5 stellte dar, wie mit der Transformation des PRM in das informationstechnikunabhängige Modell (CIM) der Anschluss an die modellgetriebene Softwareentwicklung gefunden werden kann. Anhand der bei den Transformationen aufgezeichneten Informationen wurden bereits Einschätzungen zur prinzipiellen Verfolgbarkeit der jeweiligen Transformation gegeben. Um allerdings für das Vorfeld der Anforderungsspezifikation eine Verbesserung der Verfolgbarkeit aufzeigen zu können, müssen diese Elemente zu einem Gesamtansatz kombiniert werden. Um eine geeignete Kombination herzuleiten, sind neben den einzelnen Bestandteilen, die in Tabelle 57 aus Sicht des Gesamtansatzes beschrieben sind, vor allem ihr nachfolgend überblickartig

beschriebenes Zusammenwirken relevant, um die Ansatz- und Bezugspunkte der Verfolgbarkeit identifizieren zu können.

Bestandteil	Allgemeine Beschreibung	Ihm Rahmen dieser Arbeit verwendete spezielle Form
Ausgangsdokument	Dokument, dessen Text die Ursprünge von Anforderungen an das zu entwickelnde Anwendungssystem enthält.	Ausgangsdokument, das hier eine oder mehrere auf Vollzug in der öffentlichen Verwaltung ausgerichtete Rechtsvorschriften umfasst.
Ontologie	Ontologie zentraler Konzepte, die eine Repräsentation im Text des Ausgangsdokumentes haben und Ursprung von Anforderungen sein können.	Im Rahmen dieser Arbeit entwickelte Ontologie des Verwaltungshandelns, deren Konzepte auf Elemente in Rechtssätzen zurück gehen, so dass vom Vorkommen der Instanzen dieser Konzepte im Text ausgegangen werden kann.
Instanzen der Ontologie-Konzepte als Annotationen	Instanzen der Ontologie-Konzepte, die zur Annotation von Textpassagen und Elementen des Ausgangsdokumentes verwendet werden.	OWL-Individuen der Ontologie des Verwaltungshandelns, in Form von Annotationen im Text der Rechtsvorschriften.
Transformation der Annotationen in ein PRM	Transformation der Annotationen des Ausgangsdokuments in ein Pre-Requirements Model (PRM)	Transformation von Annotationen im Text der Rechtsvorschriften in ein PRM des Verwaltungshandelns.
Verfolgbarkeits-informationen zur Transformation der Annotationen in das PRM	Während der Transformation der Annotationen in das PRM aufgezeichnete Verfolgbarkeitsinformationen.	Verfolgbarkeitsinformationen, die während der Transformation von Annotationen im Text der Rechtsvorschriften in ein PRM des Verwaltungshandelns aufgezeichnet wurden.
Pre-Requirements Model (PRM)	PRM, in dem die Ursprünge von Anforderungen resultierend aus Rechtsvorschriften dargestellt sind. Das Modell kann um weitere für das Vorfeld der Anforderungsspezifikation relevante Aspekte verfeinert werden.	PRM des Verwaltungshandelns, dass die Ursprünge von Anforderungen im Vorfeld der Anforderungsspezifikation formalisiert, die sich aus Rechtsvorschriften ergeben. Die Verfeinerung des Modells kann dazu dienen, verwaltungsspezifische Aspekte in das PRM des Verwaltungshandelns einfließen zu lassen, die über den Text der Rechtsvorschrift hinausgehen.
Transformation des PRM in ein CIM	Transformation des PRM in ein Computation Independent Model (CIM) als initiale Anforderungsspezifikation unter Verwendung von Referenzmodellen.	Transformation des PRM des Verwaltungshandelns unter Verwendung der eLoGo-Referenzmodelle für E-Government in ein CIM, das als initiale Anforderungsspezifikation eines E-Government-Anwendungssystems dient.
Verfolgbarkeitsinformationen zur Transformation des PRM in das CIM	Während der Transformation des PRM in das CIM aufgezeichnete Verfolgbarkeitsinformationen.	Verfolgbarkeitsinformationen, die während der Transformation des PRM des Verwaltungshandelns in das CIM eines E-Government-Anwendungssystems aufgezeichnet wurden.
Computation Independent Model (CIM)	CIM, das als initiale Anforderungsspezifikation dient und in den sich anschließenden Aktivitäten der Anforderungsanalyse weiterentwickelt werden kann.	CIM bestehend aus BPMN-Prozessmodellen und UML-Modellen der Anforderungsanalyse (Anwendungsfallmodell, Klassendiagramm) sowie erläuterndem Text als Beschreibung als initiale Anforderungsspezifikation eines E-Government-Anwendungssystems, die weiterentwickelt werden kann.

Tabelle 57 – Bestandteile des Gesamtansatzes zur Verfolgbarkeit

In einem ersten Schritt wird das Ausgangsdokument mit Annotationen versehen. Im Ansatz dieser Arbeit bilden die relevanten Rechtsvorschriften das Ausgangsdokument. Sie werden mit Annotationen basierend auf der Ontologie des Verwaltungshandelns versehen. Dadurch entstehen Individuen als Instanzen der Ontologie-Klassen in Form von Annotationen. Da die Annotationen über ihren jeweiligen Annotationskontext mit der annotierten Textpassage verbunden sind, kann der Verfolgbarkeitsansatz beim annotierten Ausgangsdokument ansetzen. Die Annotationen und die Ontologie werden entsprechend der Regel des Ontology Definition Metamodels in jeweils ein MDA-konformes Modell auf Basis der UML überführt, das Aspekte des Vorfeldes der Anforderungsspezifikation darstellt. Dieses Pre-Requirements Model (PRM) ist eine neuartige Erweiterung der MDA. Das PRM kann für die Modellierung verwendet werden, um beispielsweise Besonderheiten des Verwaltungshandelns in Form neuer Modellelemente darzustellen. Daran schließt sich eine Transformation des PRM in das CIM an. Das CIM wird in diesem Ansatz als initiale Anforderungsspezifikation einer E-Government-Anwendung verwendet. Beim Transformationsprozess der Annotation in Elemente des PRM und bei der Transformation der Elemente des PRM in Elemente des CIM werden Verfolgbarkeitsinformationen aufgezeichnet. Dieses Zusammenwirken ist im Sinne eines Zwischenstandes in Abbildung 89 illustriert.

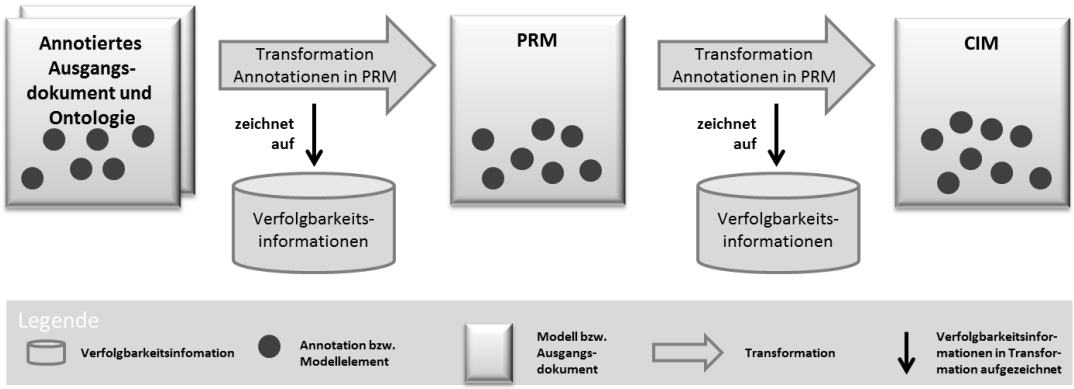


Abbildung 89 – Illustration des Zusammenwirkens der zentralen Bestandteile des Ansatzes (Teil 1)

Die Modellelemente im PRM und im CIM, die Annotationen der Textelemente und die Klassen der Ontologie sind die Bezugspunkte für die Verfolgbarkeitsbeziehungen. Jede aufgezeichnete Verfolgbarkeitsbeziehung verbindet mindestens zwei Elemente unterschiedlicher Modelle oder verbindet mindestens eine Annotation bzw. Ontologie-Klasse mit mindestens einem Modellelement des PRM. Elemente im PRM und im CIM sind durch ihre `xmi:id`-Eigenschaft eindeutig innerhalb ihrer Modelle identifizierbar. Entsprechendes gilt für die Ontologie-Klassen, die durch ihre `rdf:about`-Eigenschaft innerhalb der Ontologie eindeutig identifizierbar sind. Die Annotationen innerhalb des Ausgangsdokumentes sind durch ihre `rdf:label`- bzw. ihre `rdf:about`- Eigenschaft sowie den Annotationskontext identifizierbar. Von ihrer Eindeutigkeit kann hier aber nur ausgegangen werden, wenn das eingesetzte Werkzeug oder der zugrunde liegende Standard die mehrmalige Annotation des gleichen Kontexts verbieten würden. Das eingesetzte Werkzeug OntoMat verbietet Annotationen mit gleicher `rdf:label`- bzw. ihre `rdf:about`-Eigenschaft innerhalb eines Dokuments. Bei der manuellen Kodierung von RDFa-Annotationen muss die Eindeutigkeit manuell sichergestellt werden. Hier wird deshalb von der prinzipiellen Eindeutigkeit der Annotationen bei ihrer Nutzung in Verfolgbarkeitsbeziehungen ausgegangen.

Die Verfolgbarkeitsbeziehungen werden automatisch während der Transformationsprozesse durch Anwendung von Abbildungsregeln in XSLT beim Übergang von annotierten Texten bzw. von der Ontologie zum PRM und in QVT beim Übergang vom PRM zum CIM angelegt. Dadurch werden Beziehungen zwischen Elementen unterschiedlicher Phasen bzw. Entwicklungstätigkeiten hergestellt, was als horizontale Verfolgbarkeit bezeichnet wird (vgl.

Abschnitt 2.1). Im Rahmen dieser Arbeit bezieht sich horizontale Verfolgbarkeit folglich auf die Beziehungen, die zwischen Annotationen im Ausgangsdokument und den Elementen im PRM sowie den Elementen im PRM und den Elementen im CIM existieren. Die Beziehung hat die Bedeutung, dass die referenzierten Elemente innerhalb einer Transformation auf einander abgebildet wurden. Sie ist unabhängig von einer Richtung und ermöglicht dadurch sowohl vorwärtsgerichtete, als auch rückwärtsgerichtete Verfolgbarkeit (vgl. Abschnitt 2.1). Die vorwärtsgerichtete Verfolgbarkeit ergibt sich, wenn die Beziehungen ausgehend von einer Annotation im Ausgangsdokument zu Elementen im PRM oder von einem Element im PRM zu Elementen im CIM ausgewertet werden. In diesem Fall konkretisiert sich die Bedeutung der betrachteten Beziehung. Sie kann als „resultiert in" verstanden werden. Beziehungen, die rückwärtsgerichtet betrachtet werden, d.h. ausgehend von einem Element im CIM zu Elementen im PRM oder von einem Element im PRM zu einer Annotation im Ausgangsdokument bzw. einer Klasse der Ontologie, haben die Bedeutung „ist hervorgegangen aus".

Wenn Rechtsvorschriften als Ausgangsdokument verwendet werden, so ermöglicht die vorwärtsgerichtete, horizontale Verfolgbarkeit entsprechend der aufgezeichneten Beziehungen die Navigation von einer annotierten Textpassage des Vorschriftentextes zu dem korrespondierenden Element im PRM des Verwaltungshandelns. Anschließend ermöglicht sie die Navigation vom Element im PRM des Verwaltungshandelns zu den aus ihm resultierenden Elementen in den Prozessmodellen, Anwendungsfall- und Klassendiagrammen des CIM, die die Anforderungsspezifikation einer E-Government-Anwendung bilden. Umgekehrt kann ausgehend von einem Element in den Prozessmodellen, Anwendungsfall- oder Klassendiagrammen die rückwärtsgerichtete, horizontale Verfolgbarkeit genutzt werden, um über die Elemente im PRM, aus denen das Element hervorgegangen ist, zu den annotierten Textpassagen der Rechtsvorschrift zu navigieren. Horizontale Verfolgbarkeit wird durch die aufgezeichneten Verfolgbarkeits-informationen folglich umfassend sowohl vorwärts- als auch rückwärts unterstützt.

Vertikale Verfolgbarkeit kann durch die aufgezeichneten Informationen erreicht werden, wenn Beziehungen zwischen Elementen gleicher Phasen bzw. Entwicklungstätigkeiten hergestellt werden können (vgl. Abschnitt 2.1). Sie bezieht sich hier folglich auf Beziehungen der Elemente des CIM untereinander, der Elemente des PRM untereinander, der Annotationen des Ausgangsdokumentes oder der Klassen der Ontologie untereinander. Weil lediglich von den Annotationen bzw. Ontologie-Klassen zum PRM oder vom PRM zum CIM durchgeführte Transformationen Verfolgbarkeitsinformationen aufzeichnen, gibt es keine umfassenden Informationen über vertikale Beziehungen dieser Elemente innerhalb der gleichen Phasen bzw. Entwicklungstätigkeit. Es werden lediglich zum Element die weiteren in einer Abbildungsregel verwendeten Elemente des gleichen Modells aufgezeichnet, so dass man hieraus einen allgemeinen Zusammenhang schlussfolgern kann. Im Folgenden werden im Sinne dieses Zusammenhangs Elemente des gleichen Modells, die gemeinsam in einer Abbildungsregel verwendet werden, als Geschwisterelemente bezeichnet.

Geschwisterelemente in einem PRM drücken aus, dass es eine vertikale Verfolgbarkeitsbeziehung zwischen diesen Elementen gibt, die sich aus der Semantik der Abbildungsregel ergibt. Beispielsweise kann zwischen zwei Elementen im CIM, jedes von ihnen aus einem anderen Element im PRM hervorgegangen, durch eine Abbildungsregel eine Assoziation angelegt werden, die aus der Abbildung der beiden PRM-Elemente resultiert. Dann sind diese beiden PRM-Elemente so genannte Geschwisterelemente, was bedeutet, dass es eine inhaltliche Beziehung zwischen ihnen gibt, die unabhängig davon existiert, ob diese Beziehung bei der Annotation des Ausgangsdokumentes explizit angelegt wurde. Die Beziehung zwischen Geschwisterelementen eines PRM des Verwaltungshandelns kann deshalb beispielsweise bedeuten, dass in der annotierten Rechtsvorschrift ein Zusammenhang zwischen Textpassagen verschiedener Paragraphen existiert, der zum Zeitpunkt der Annotation noch nicht ersichtlich war.

Weitergehende vertikale Verfolgbarkeit im Rahmen der modellgetriebenen Entwicklung kann prinzipiell nicht ausschließlich durch Aufzeichnung von Informationen über durchgeführte Transformationen erreicht werden. Dies würde voraussetzen, dass Modelländerungen ausschließlich durch Transformationen erfolgen dürften. Zwar kennt die MDA auch Transformationen, bei denen Ausgangs- und Zielmodell identisch und vom gleichen Typ sind. Es ist aus theoretischer Perspektive allerdings fraglich und in der Praxis unrealistisch, auch die routinemäßige Modellierung, die eine anspruchsvolle intellektuelle und höchst kreative Aufgabe sein kann, ausschließlich durch (vordefinierte) Transformationen umzusetzen.

Modelle, die sowohl Ziel- als auch Ausgangspunkt einer Transformation sind, nehmen eine Sonderrolle ein. Das PRM ist ein Beispiel für ein solches Modell. Es ist Zielmodell der Transformation von Annotationen des Ausgangsdokuments in das PRM. Weiterhin ist es Ausgangsmodell für die Transformation der OWL-Individuen in Form von UML-Instanzen in die Elemente des CIM. Durch Auswertung der aufgezeichneten Informationen kann der Zustand den das Modell hatte, als es Ziel der Transformation war, mit dem Zustand verglichen werden, den das Modell hatte, als es Ausgangspunkt einer anschließenden Transformation war. Dies ermöglicht in begrenztem Umfang Rückschlüsse auf Beziehungen zwischen den Elementen des jeweiligen Modellzustands. Würde beispielsweise ein OWL-Individuum bei der Modellierung im PRM gelöscht, so würde zwar die Transformation der Annotationen in das PRM, aber nicht die Transformation des PRM in das CIM Informationen über dieses Individuum aufzeichnen. Neu zum PRM hinzugefügte Elemente wären über den umgekehrten Mechanismus ebenfalls identifizierbar. Änderungen an den existierenden Elementen des PRM sind nur identifizierbar, wenn sich nicht nur der Gesamtzustand des Modells, sondern auch der Zustand jedes Modellelements (z.B. bezogen auf dessen Eigenschaften) ermitteln ließe. Aus diesem Grund wurde vorgeschlagen, im Rahmen der Transformation des Ausgangsdokumentes in das PRM für jede Eigenschaft des OWL-Individuums Checksummen auf Basis seiner Werteausprägung zu berechnen und als Verfolgbarkeitsinformation aufzuzeichnen. Die erneute Berechnung der Checksumme kann jederzeit Aufschluss über Werteänderungen von Eigenschaften eines Individuums geben. Allerdings ermöglicht dieser einfache Mechanismus keine Aussage über den Inhalt oder Grund der Änderung.

Um eine umfassende vertikale Verfolgbarkeit im Rahmen der durchgeführten Transformationen zu erreichen, muss die Aufzeichnung von Verfolgbarkeitsbeziehungen durch Transformationen mit weiteren Techniken gekoppelt werden. Denkbar sind hier beispielsweise die Versionierung der Modelle unter Verwendung einer fachlich aussagekräftigen und technisch auswertbaren Änderungshistorie oder die Dokumentation der Änderungen durch Modellierung entsprechender Beziehungen. Entsprechend dieses Beispiels wurden in der PRM-Modellierung Abhängigkeitsbeziehungen mit dem Stereotyp «`refine`» verwendet, die vom ursprünglichen Modellelement zu dessen Verfeinerung führen.

Die im Rahmen dieser Arbeit vorgeschlagenen Checksummen, die Nutzung der «`refine`»-Abhängigkeitsbeziehungen und die Auswertung der Geschwisterelemente zeigen, dass für die vertikale Verfolgbarkeit innerhalb der Transformationen Ansatzpunkte existieren, deren weitere Ausgestaltung jedoch über das Thema dieser Arbeit hinausgehen. Hier liegt der Schwerpunkt auf der Verfolgbarkeit von Anforderungen an E-Government-Anwendungen zu ihren Ursprüngen in Rechtsvorschriften, was primär in den Anwendungsbereich der horizontalen Verfolgbarkeit fällt.

Die Verfolgbarkeitsbeziehungen müssen prinzipiell sowohl für das Verfolgen funktionaler, als auch nicht-funktionaler Anforderungen geeignet sein (vgl. Abschnitt 2.1). Ursprünge funktionaler Anforderungen werden durch Annotationen in Form von OWL-Individuen formalisiert und in anschließenden Transformationsschritten in ein PRM überführt. Für die Oberklasse Verwaltungskonzept und jede ihrer Spezialisierungen kann ein Data Property verwendet werden, um Rahmenbedingungen der Umsetzung oder des Einsatzes zu formalisieren. OWL-Individuen können eine Werteausprägung für dieses Property festlegen,

die durch die Transformation in eine Attributsausprägung der korrespondierenden Instanz im PRM überführt wird. Bei Transformation des PRM in ein CIM werden die Ursprünge funktionaler Anforderungen durch Anwendung der eLoGo-Referenzmodelle für E-Government formalisiert. Innerhalb dieser Modelle werden auch die Ursprünge nicht-funktionaler Anforderungen in Form von Kommentaren als eigenständige Modellelemente eingefügt. Dadurch werden auch Informationen aufgezeichnet, die im Sinne der nicht-funktionalen Verfolgbarkeit genutzt werden können.

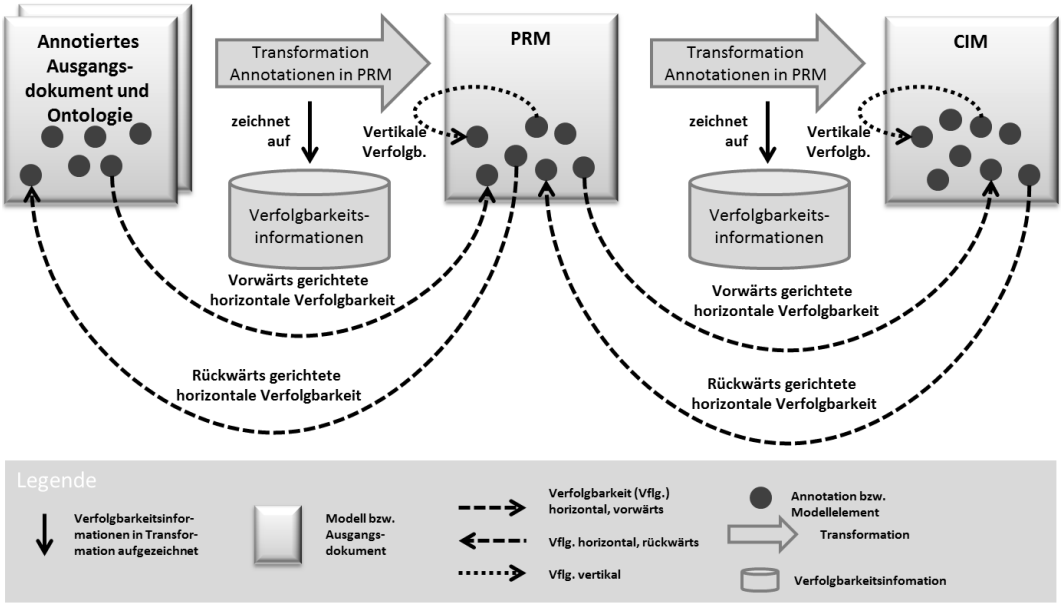


Abbildung 90 – Illustration des Zusammenwirkens der zentralen Bestandteile des Ansatzes (Teil 2)

Zusammengefasst unterstützt der Ansatz dieser Arbeit die horizontale, vorwärts- und rückwärtsgerichtete Verfolgbarkeit und zeigt Ansatzpunkte für die vertikale Verfolgbarkeit von Ursprüngen funktionaler und nicht-funktionaler Anforderungen auf (Abbildung 90). Die Herausforderung besteht in der geeigneten Kombination der getrennt aufgezeichneten Informationen und ihrer Aufbereitung, so dass diese von einem Bearbeiter genutzt werden können.

6.2 Umsetzung und Implementierung

Die Verfolgbarkeitsinformationen müssen aus ihrer vorliegenden Form in eine Form überführt werden, die für die Interpretation und Verwendung durch einen Bearbeiter geeignet ist (vgl. Abschnitt 5.4). Auch die Modelle eignen sich innerhalb der Werkzeugunterstützung nur bedingt dafür, ihre Modellelemente zum Ausgangspunkt oder Ziel einer Verfolgbarkeitsbeziehung in einem anderen Modell bzw. im annotierten Ausgangsdokument zu machen. Deshalb wird hier von den verschiedenen möglichen Darstellungen der Verfolgbarkeitsbeziehungen (vgl. Abschnitt 2.1) die Form des Hypertext-Dokuments gewählt. Prinzipiell sind auch andere Formen möglich, Hypertext-Dokumente haben für den Ansatz dieser Arbeit jedoch den Vorteil, dass sie bereits durch einfache Integration in die Eclipse-Umgebung die verschiedenen Verwendungsmöglichkeiten aufzeigen. Bei der Nutzung anderer Darstellungsformen (z. B. Matrizen oder spezielle Verfolgbarkeitsmodelle) sind weitergehende Werkzeuganpassungen vorzunehmen (z. B. Entwicklung eines Eclipse Plug-Ins für die Matrix-Darstellung und Verlinkung von Elementen oder Erweiterung der Modellierungswerkzeuge für Verlinkung verschiedener Modelle und Notationen), ohne dass für das Thema dieser Arbeit ein zusätzlicher Erkenntnisgewinn zu erwarten wäre.

Der durch die Verfolgbarkeitsdokumentation erweiterte Gesamtansatz wird in Abbildung 91 illustriert und nachfolgend beschrieben. Zu seiner Umsetzung wird aus dem PRM, aus dem

CIM und aus dem Ausgangsdokument jeweils ein Hypertext-Dokument in Form eines HTML-Dokuments generiert. Dieses Dokument wird im Folgenden als Verfolgbarkeits-dokumentation bezeichnet. Sie stellt die im Modell enthaltenen Elemente, die im Ausgangsdokument enthaltenen Annotationen und die Ontologie-Klassen als strukturierten Text dar. Der Text jedes Elements, jeder Annotation oder Klasse wird, basierend auf den aufgezeichneten Verfolgbarkeitsinformationen, um Hyperlinks erweitert, die der Verfolgbarkeitsbeziehung entsprechen. Weiterhin wird eine Verbindung vom Modell zur jeweiligen Verfolgbarkeitsdokumentation hergestellt, so dass zwischen den verschiedenen Darstellungsformen des gleichen Elements navigiert werden kann.

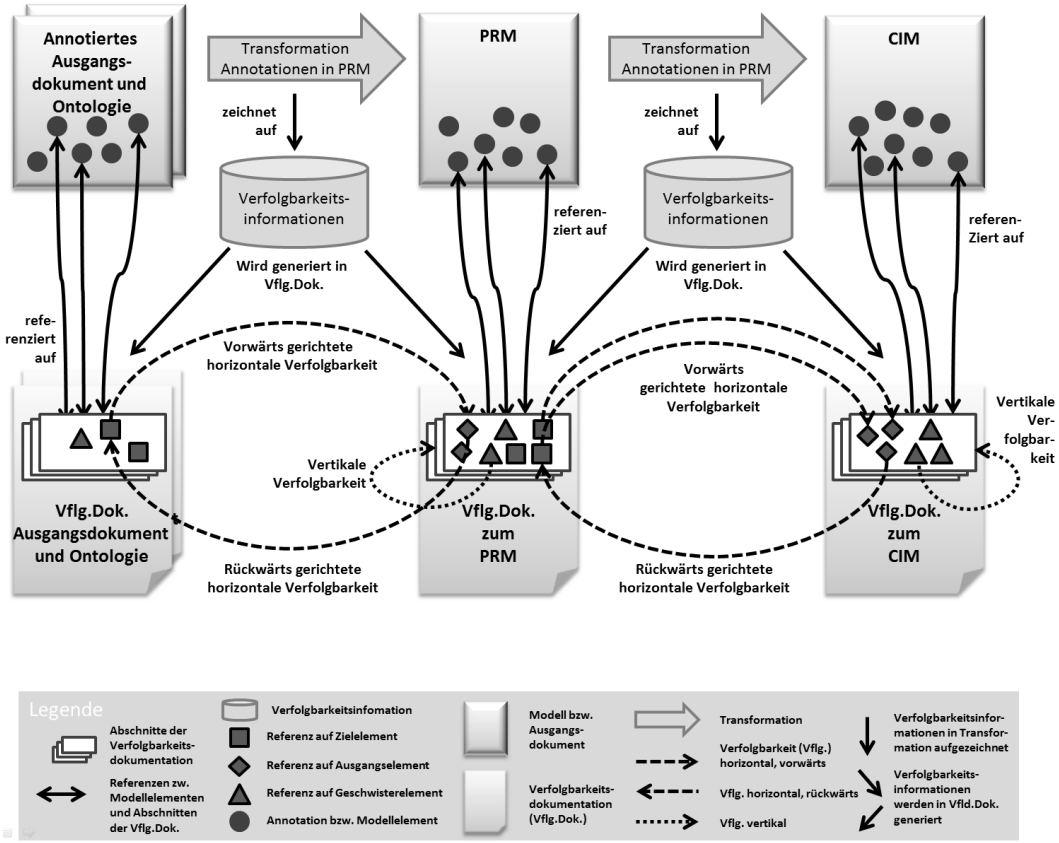


Abbildung 91 – Illustration des Zusammenwirkens der zentralen Bestandteile des Ansatzes (Teil 3)

6.2.1 Verfolgbarkeitsdokumentation des Ausgangsdokuments und der Ontologie

Aus dem im HTML-Format vorliegenden Ausgangsdokument wird eine Verfolgbarkeits-dokumentation generiert, die alle OWL- bzw. RDFa-Annotationen in Form von Individuen und ihre grundlegenden Eigenschaften darstellt. Anhand dieser Eigenschaften kann das Individuum von anderen unterschieden werden. Zu jedem Indviduum werden die Werteausprägungen von Data Properties angezeigt sowie die aus Object Properties resultierenden Beziehungen zu anderen Indviduen. Weiterhin werden alle durch die Transformation erzeugten Elemente im PRM aufgelistet. Sie sind mit einem Hyperlink versehen, der die Navigation zum entsprechenden Abschnitt der Verfolgbarkeits-dokumentation des PRM erlaubt. Die Navigation über diesen Hyperlink im Abschnitt „Zielelemente im PRM“ gewährleistet horizontale, vorwärtsgerichtete Verfolgbarkeit (vgl. Abbildung 92).

Um den Bezug zur Textpassage des Ausgangsdokumentes herzustellen, muss zwischen den OntoMat-Annotationen und den RDFa-Annotationen unterschieden werden. In der Verfolgbarkeitsdokumentation, die aus den OntoMat-Annotationen erzeugt wurde, ist für

die Schaltfläche „Annotierte Textpassage anzeigen“ eine JavaScript-Funktion[389] implementiert. Sie rechnet den XPointer-Verweis des OntoMat (aufgrund der in Abschnitt 3.3.1 beschriebenen Problematik nur näherungsweise) um, navigiert zur entsprechenden Passage im Text des Ausgangsdokuments, hebt diese hervor und erzeugt dynamisch einen Hyperlink, der zum Element in der Verfolgbarkeitsdokumentation zurückführt. Es ist auch möglich, über die Hyperlinks anderer annotierter Textpassagen zu den zugehörigen OWL-Individuen in der Verfolgbarkeitsdokumentation zu navigieren. Ist eine Textpassage mehrfach mit unterschiedlichen Annotationen versehen, so wird dies durch die prototypische Implementierung durch mehrere Stern-Icons dargestellt. Je Stern-Icon ist die Navigation zu genau einem Individuum möglich. Abbildung 92 zeigt Ausschnitte der Verfolgbarkeitsdokumentation und illustriert ausgewählte Hyperlinks durch Pfeile.

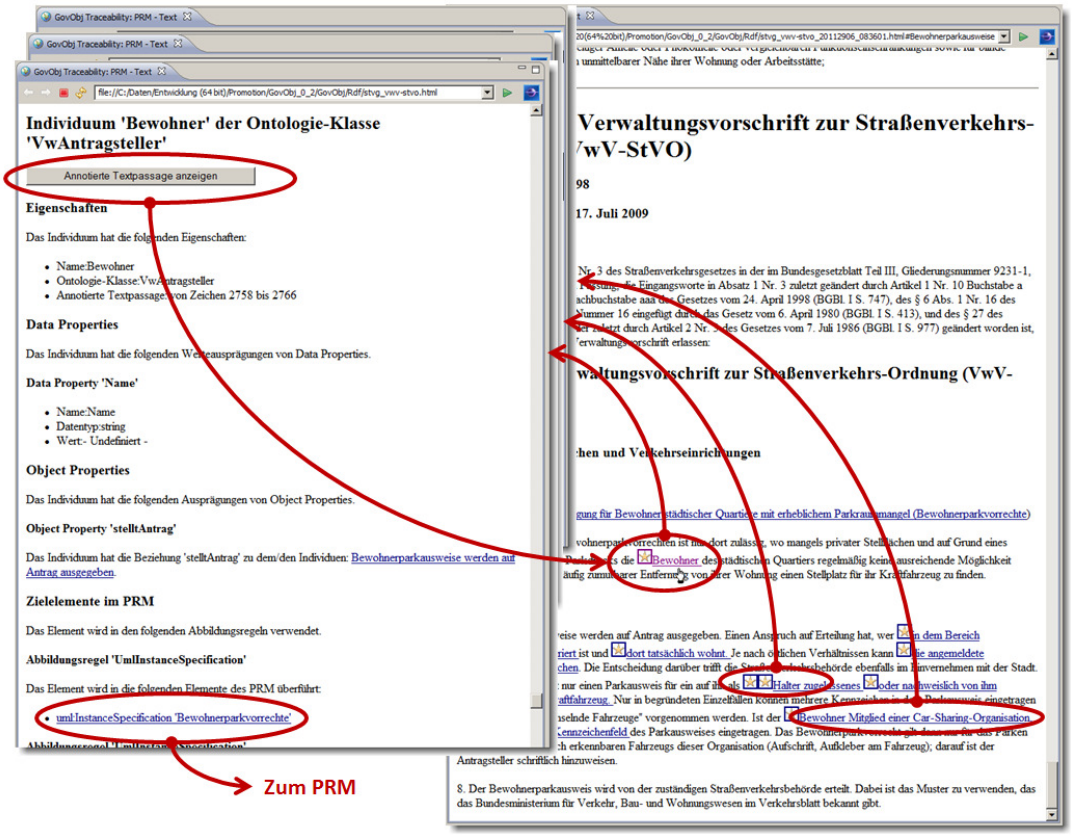


Abbildung 92 – Auszug aus der Verfolgbarkeitsdokumentation des annotierten Ausgangsdokumentes (OntoMat)

Die RDFa-Annotationen haben einen direkten Bezug zur annotierten Textpassage, so dass eine Umrechnung per JavaScript nicht notwendig ist. Stattdessen kann über einen einfachen Hyperlink direkt vom Abschnitt „Annotierte Textpassage“ zum entsprechenden Text navigiert werden. Die Verfolgbarkeitsdokumentation, die auf Basis von RDFa-Annotationen erzeugt wird, ist inhaltlich identisch mit der aus den OntoMat-Annotationen erzeugten Dokumentation (Abbildung 93).

[389] JavaScript ist eine Skriptsprache, die zur Programmierung in Verbindung mit HTML populär ist. Wesentliche Teile des Sprachumfangs sind in Form der „ECMAScript language specification“ als ISO-Norm ISO/IEC 16262:2002 standardisiert.

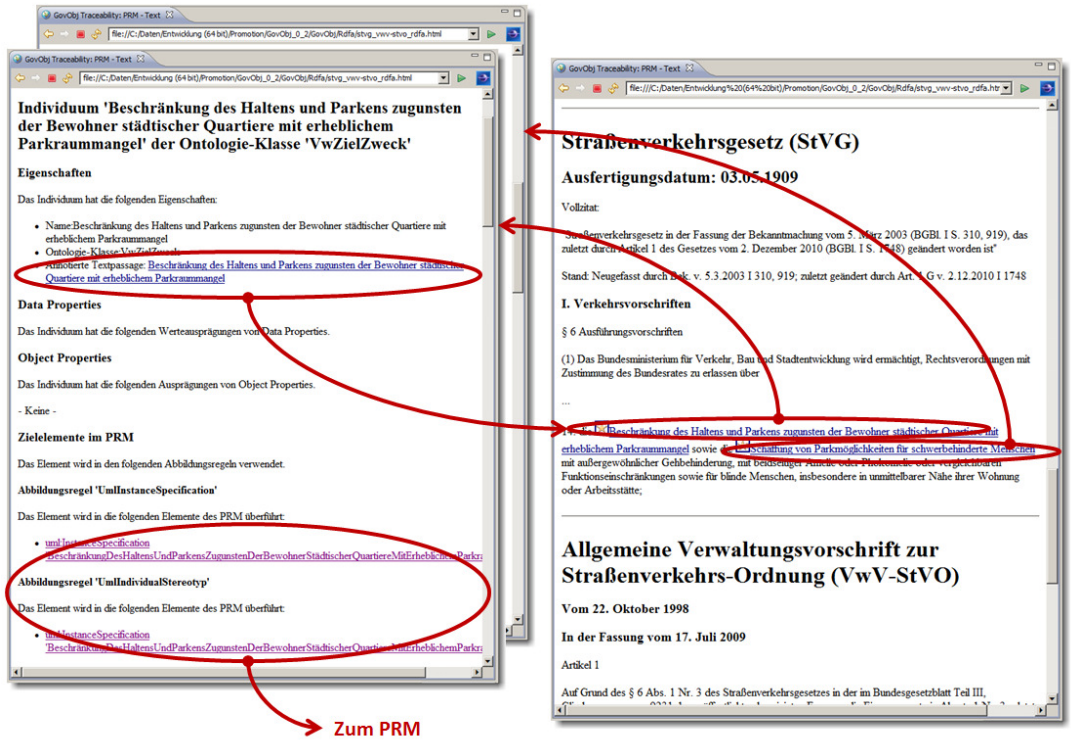


Abbildung 93 – Auszug aus der Verfolgbarkeitsdokumentation des RDFa-annotierten Ausgangsdokumentes

Die prototypische Implementierung der Verfolgbarkeitsdokumentation erfolgte im Rahmen dieser Arbeit als XSL-Stylesheets. Die Stylesheets transformieren die annotierten OntoMat-/RDFa-Ausgangsdokumente unter Verwendung der während der Transformation in das PRM aufgezeichneten Informationen in die Verfolgbarkeitsdokumentation. Dabei wird auch der vollständige Text des Ausgangsdokuments inkl. seiner HTML-Formatierung übernommen. Das integrierte Dokument erlaubt auf diese Weise die Navigation zwischen den Annotationen und ihren zugehörigen OWL-Individuen. Das vollständige Listing des XSL-Stylesheets zur Transformation des OntoMat-Ausgangsdokuments ist in Anhang G.1 und zur Transformation des RDFa-annotierten Ausgangsdokuments in Anhang G.2 enthalten.

Die Ontologie des Verwaltungshandelns wurde in Klassen eines auf dem ODM basierenden UML-Klassendiagramms transformiert. Es bildet zusammen mit den Instanzen, die aus den Annotationen erzeugt wurden, das PRM. Aus diesem Grund muss auch für die Ontologie eine Verfolgbarkeitsdokumentation erzeugt werden. Das vollständige Listing zur Generierung einer Verfolgbarkeitsdokumentation der Ontologie ist in Anhang G.3 dargestellt. Die Abbildung 94 zeigt Ausschnitte aus dieser Dokumentation und illustriert ausgewählte Hyperlinks durch Pfeile.

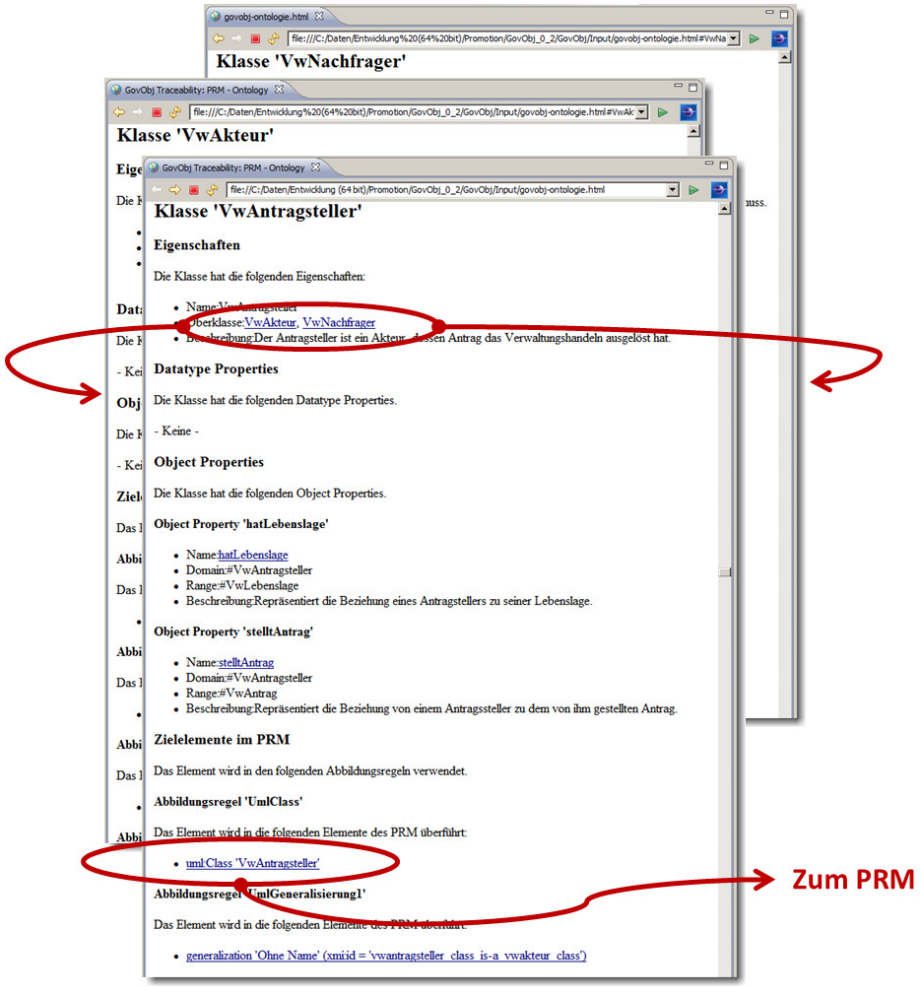


Abbildung 94 – Auszug aus der Verfolgbarkeitsdokumentation der Ontologie

6.2.2 Verfolgbarkeitsdokumentation des PRM und des CIM

Die Verfolgbarkeitsdokumentation für das PRM umfasst für jedes im Modell enthaltene Element grundlegende Informationen zu diesem Element, anhand derer es von anderen Elementen unterschieden werden kann (z.B. Name, Typ, ID, Beschreibung). Die Dokumentation führt auch die Attributsausprägungen und die Beziehungen in Form von Instance Specification Links zu anderen Elementen des Modells auf. Für jedes Modellelement ist weiterhin das OWL-Individuum des Ausgangsdokumentes angegeben, aus dem das Element hervorgegangen ist. Über einen Hyperlink kann zu diesem Element in der Verfolgbarkeitsdokumentation des Ausgangsdokumentes navigiert werden. Dies entspricht der Nutzung einer horizontalen, rückwärtsgerichteten Verfolgbarkeitsbeziehung „ist hervorgegangen aus“. Vorwärtsgerichtete Verfolgbarkeit wird durch Navigation zu den aus dem PRM-Element resultierenden Zielelementen im CIM unterstützt. Die Zielelemente im CIM sind je durchgeführter Transformation aufgeführt, da das PRM-Element potenziell in mehreren Transformationen verwendet werden kann. Durch den Hyperlink der aufgeführten Zielelemente kann zu ihrer Beschreibung in der Verfolgbarkeitsdokumentation des CIM navigiert werden. Dies entspricht der Nutzung der „resultiert in“-Verfolgbarkeitsbeziehung. Wurden innerhalb der gleichen Abbildungsregel weitere PRM-Elemente verwendet, so handelt es sich bei diesen um Geschwisterelemente. Die Geschwisterelemente werden in der Verfolgbarkeitsdokumentation ebenfalls aufgelistet und verlinkt. Über ihren Hyperlink kann zur Beschreibung des Elements in der Verfolgbarkeitsdokumentation des PRM navigiert werden, was vertikalem Verfolgen entspricht. Hinweise auf die Rolle oder Funktion der Elemente in einer Abbildungsregel können die Variablenbezeichnungen geben, die bei der

Implementierung der Abbildungsregel möglichst aussagekräftig bezeichnet werden sollten. Abbildung 95 zeigt Ausschnitte aus der Verfolgbarkeitsdokumentation des PRM zum Bewohnerparken und illustriert ausgewählte Hyperlinks durch Pfeile.

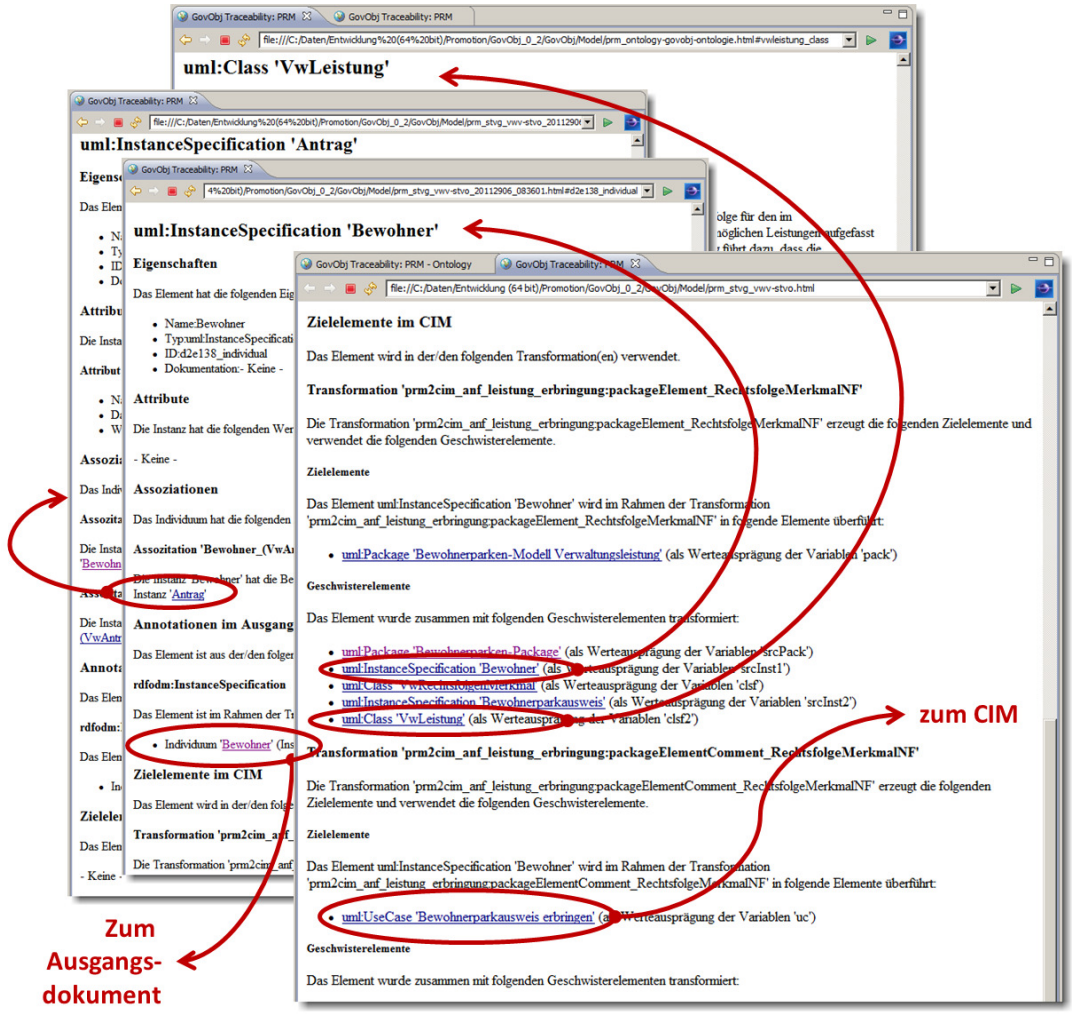


Abbildung 95 – Ausschnitt aus der Verfolgbarkeitsdokumentation zum PRM des Bewohnerparkens

Die für das CIM generierte Verfolgbarkeitsdokumentation stellt das Modellelement mit seinen grundlegenden Eigenschaften, Attributen und Assoziationen zu anderen Modellelementen im CIM dar. Es ist möglich, dass mehrere Abbildungsregeln einer QVT-Transformation dazu geführt haben, dass ein Modellelement im CIM angelegt wurde. Diese Abbildungsregeln sind aufgelistet, jeweils mit den Ausgangselementen im PRM. Über einen Hyperlink kann zu der Beschreibung des Ausgangselements in der Verfolgbarkeitsdokumentation des PRM navigiert werden. Zu der Abbildungsregel sind weiterhin die Geschwisterelemente im CIM aufgelistet, deren Hyperlink die Navigation zu ihrer Beschreibung in der Verfolgbarkeitsdokumentation des CIM ermöglicht. Dadurch werden rückwärtsgerichtete horizontale und vertikale Verfolgbarkeit unterstützt. In der Dokumentation fehlen die weiterführenden Hyperlinks im Sinne einer vorwärtsgerichteten horizontalen Verfolgbarkeit. Prinzipiell kann das hier beschriebene Prinzip auch für PIM- und PSM-Modelle fortgesetzt werden, die potenzielle Zielelemente enthalten würden. Diese Verfolgbarkeitsbeziehungen sind aber nicht mehr Gegenstand der vorliegenden Arbeit, da sie das Vorfeld der Anforderungsspezifikation verlassen. Ausgehend vom CIM wird deshalb keine vorwärtsgerichtete horizontale Verfolgbarkeit angeboten.

Das CIM-Modell besteht aus einem Prozessmodell und einem Anforderungsmodell. Deshalb muss die Verfolgbarkeitsdokumentation für beide Modelle erzeugt werden. Die Abbildung 96 zeigt einen Ausschnitt aus der Verfolgbarkeitsdokumentation zum Anforderungsmodell.

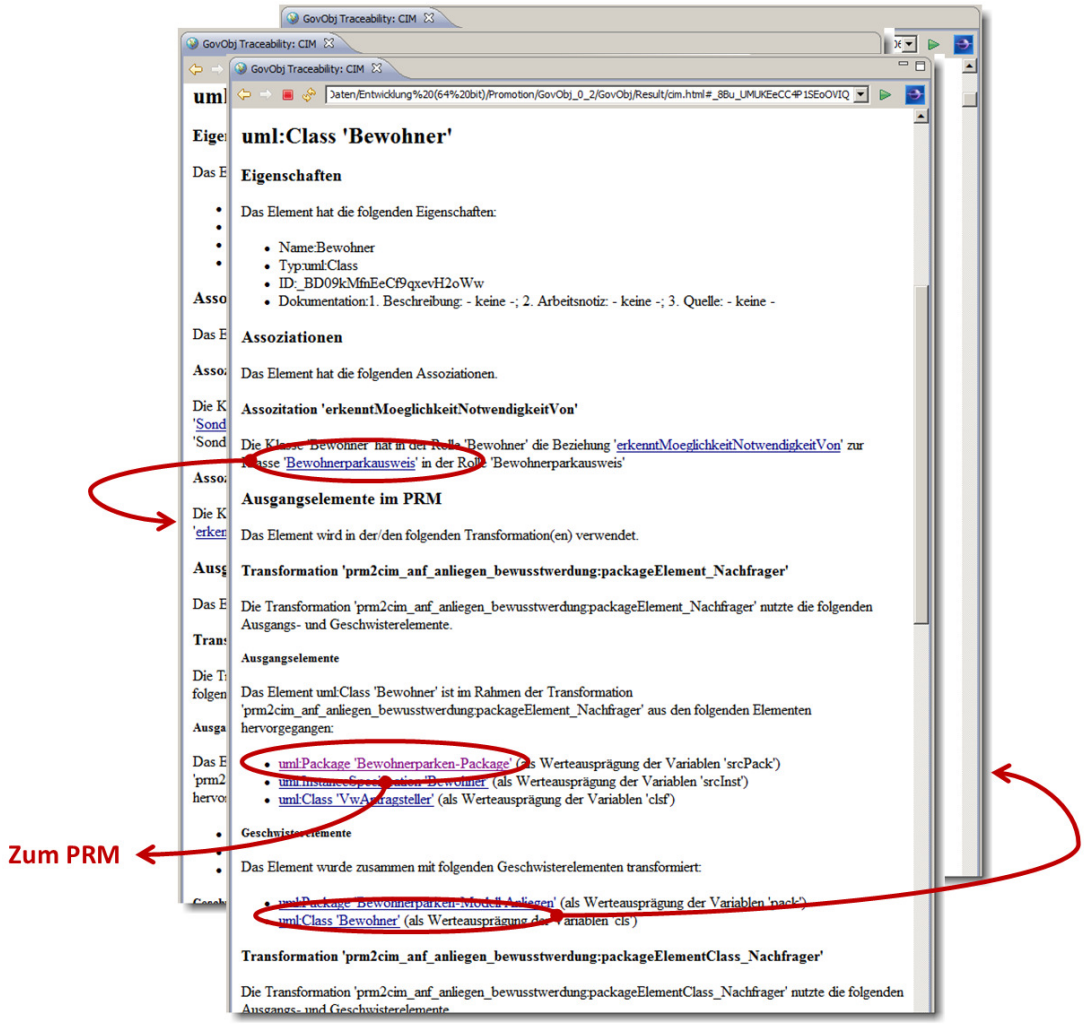


Abbildung 96 – Ausschnitt aus der Verfolgbarkeitsdokumentation zum CIM des Bewohnerparkens

Das Prozessmodell besteht aus BPMN-Diagrammen, die jeweils in eigenen Dateien vorliegen. Entsprechend des hier gewählten Ansatzes müsste für jedes BPMN-Diagramm eine Verfolgbarkeitsdokumentation erzeugt werden. Sie würde analog zu den UML-Modellen über die Informationen zum Element und über Hyperlinks auf in Beziehung stehende Elemente, auf Geschwisterelemente und auf Ausgangselemente im PRM. Auf die prototypische Implementierung dieser Dokumentation wurde deshalb im Rahmen dieser Arbeit verzichtet.

Für die Generierung der Verfolgbarkeitsdokumentation des PRM und der Anwendungsfall- und Klassendiagramme des CIM wurden zwei XSL-Stylesheets verwendet, die im Rahmen dieser Arbeit prototypisch implementiert wurden. Das XSL-Stylesheet zur Generierung der Verfolgbarkeitsdokumentation für das PRM ist in Anhang G.4 dargestellt. Der Anhang G.5 stellt das Stylesheet zur Generierung der Verfolgbarkeitsdokumentation des CIM dar.

6.2.3 Verbindung mit der Modellierung

Bei der Verfolgbarkeitsdokumentation handelt es sich jeweils um ein zusätzlich zum eigentlichen Modell generiertes HTML-Dokument. Die eigentliche Modellierung findet aber in der graphischen Darstellung des Modells und seiner Elemente in einem Diagramm statt. Um die Verbindung zwischen einem graphischen Modellelement und seiner Beschreibung in

der Verfolgbarkeitsdokumentation herzustellen, muss das Modell geeignet erweitert werden. Die Erweiterung kann beispielsweise durch einen Hyperlink erfolgen, der die Navigation vom Modellelement zum korrespondierenden Abschnitt der Verfolgbarkeitsdokumentation ermöglicht. Die Umsetzung des Hyperlinks ist von den Möglichkeiten des eingesetzten Modellierungswerkzeugs abhängig, da die Verbindung in der Diagrammdarstellung der Modelle sichtbar und nutzbar sein muss. Für das PRM und die Anwendungsfall- und Klassendiagramme des CIM müssen im Rahmen dieser Arbeit die Möglichkeiten des Werkzeugs TOPCASED, für die Prozessmodelle die Möglichkeiten des BPMN Modeler verwendet werden.

Das eingesetzte Werkzeug TOPCASED unterstützt die Verlinkung von Elementen im PRM- bzw. CIM-Modell mit der Verfolgbarkeitsdokumentation in Form von Ressourcen, die Modellelementen zugewiesen werden können. Abbildung 97 zeigt dies am Beispiel eines Akteurs im CIM des Bewohnerparkens. Im eingeblendeten Fenster erkennt man in der TOPCASED-Dokumentation auf der Registerkarte „Resources“ den Hyperlink zur Verfolgbarkeitsdokumentation in Form eines HTML-Dokuments, gefolgt von der Angabe eines Anchors (nach dem #-Zeichen) zum Sprung auf den entsprechenden Abschnitt im HTML-Dokument.[390] Diese Verknüpfung ist in Listing 44 im XMI-Format dargestellt. Die Zeilen 8 bis 10 enthalten eine `eAnnotation`, die in Zeile 9 den Verweis auf das entsprechende HTML-Dokument der Verfolgbarkeitsdokumentation enthält.

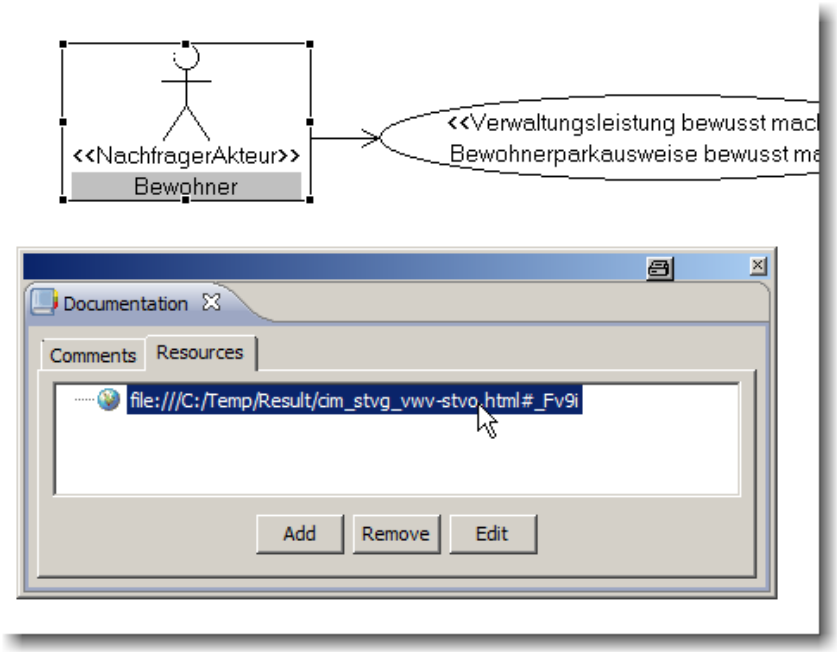


Abbildung 97 – Verknüpfung des graphischen UML-Modellelements mit der Verfolgbarkeitsdokumentation

```
(1)   <xmi:XMI ... >
(2)     <uml:Model xmi:id="_FyX9kKIaEeC9-e_u7Odn_Q" name="Bewohnerparken">
(3)     <!-- ... -->
(4)       <packagedElement xmi:type="uml:Actor" xmi:id="_Fv9i"
            name="Bewohner">
(5)           <eAnnotations xmi:id="_Fv-w4KIaEeC9-e_u7Odn_Q"
              source="http://www.topcased.org/documentation">
(6)             <details xmi:id="_Fv1X" key="documentation" value="..."/>
(7)           </eAnnotations>
```

[390] Die umgekehrte Navigationsrichtung aus dem HTML-Dokument zur grafischen Repräsentation des Elements erscheint ebenfalls möglich. Auf ihre praktische Implementierung wurde im Rahmen dieser Arbeit verzichtet. Dem damit verbundenen Implementierungsaufwand zur Erweiterung des TOPCASED-Werkzeugs steht kein vergleichbarer Erkenntnisgewinn gegenüber. Die Möglichkeiten der bidirektionalen Navigation können anhand der Ausgangsdokumente und ihrer Verfolgbarkeits-dokumentation untersucht werden.

```
(8)          <eAnnotations xmi:id="d2e97_govobj"
     source="http://www.topcased.org/resources">
(9)             <details xmi:id="d2e97_dtls" key="RR0" value="file:///C:/Temp/
     Result/cim_stvg_vwv-stvo.html#_Fv9i"/>
(10)         </eAnnotations>
(11)      </packagedElement>
(12)    <!-- ... -->
(13)    </uml:Model>
(14)    <!-- ... -->
(15)  </xmi:XMI>
```

Listing 44 – Verknüpfung des graphischen UML-Modellelements mit der Verfolgbarkeitsdokumentation in XMI

Der BPMN Modeler unterstützt Hyperlinks über einen mit TOPCASED vergleichbaren Mechanismus. In der graphischen Darstellung des Modellelements wird das verlinkte Dokument mit einem dem Dateityp entsprechenden Icon dargestellt und durch einen Tooltipp erläutert. Über das Kontext-Menü kann zum entsprechenden Dokument navigiert werden. Der BPMN-Modeller unterstützt allerdings keine Anchor-Angabe in Verbindung mit HTML-Dokumenten. Deshalb müsste die Verfolgbarkeitsdokumentation der Prozessmodelle in Form mehrerer untereinander verlinkter Einzeldokumente generiert werden. Pro Einzeldokument ist dann genau ein Modellelement beschrieben, das Verweise auf die Einzeldokumente der anderen BPMN-Modellelemente enthält und nicht auf Abschnitte im gleichen Dokument, wie bei der Verfolgbarkeitsdokumentation der Anwendungsfall- und Klassendiagramme.

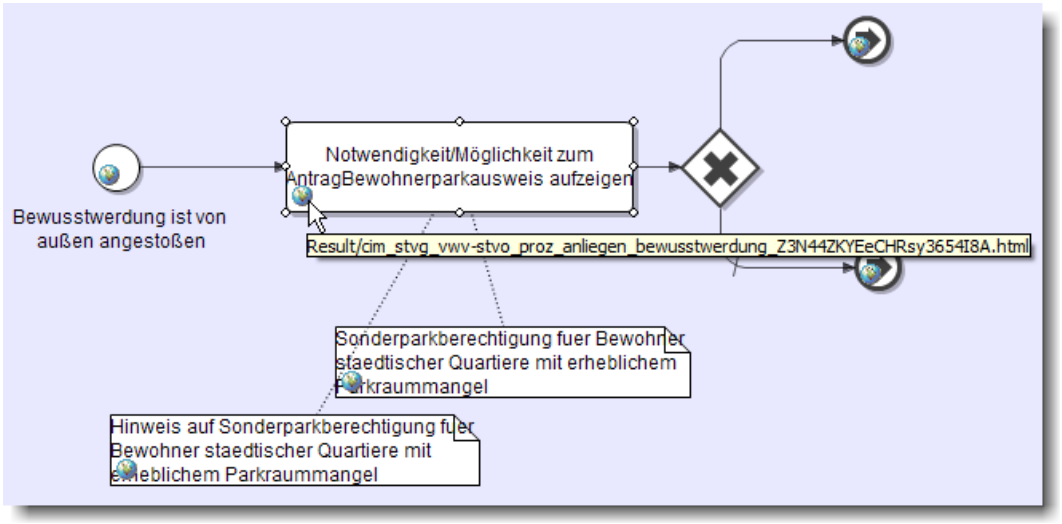


Abbildung 98 – Verknüpfung des graphischen BPMN-Modellelements mit der Verfolgbarkeitsdokumentation

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <bpmn:BpmnDiagram xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
      xmlns:bpmn="http://stp.eclipse.org/bpmn"
      xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore" xmi:id="_Z2x" iD="_Z2x"
      name="BewohnerparkenModel">
(3)      <!-- ... -->
(4)       <vertices xmi:type="bpmn:Activity" xmi:id="_Z3N44ZKYEeCHRsy3654I8A"
      iD="_rU" associations="_Z3K _Z3M" outgoingEdges="_Z3N" incomingEdges="_Z3O"
      documentation="..." name="Notwendigkeit/Möglichkeit
      zum&#xD;&#xA;AntragBewohnerparkausweis aufzeigen"
      ncname="BewusstwerdungAufzeigen" lanes="_Z3Q">
(5)         <eAnnotations xmi:type="ecore:EAnnotation" xmi:id="_g2r"
      source="genericFile">
(6)           <details xmi:type="ecore:EStringToStringMapEntry" xmi:id="_g2q"
                key="projectRelativePath" value="Result/cim_stvg_vwv-
      stvo_proz_anliegen_
                bewusstwerdung_Z3N44ZKYEeCHRsy3654I8A.html"/>
(7)         </eAnnotations>
(8)       </vertices>
(9)      <!-- ... -->
(10)  </bpmn:BpmnDiagram>
```

Listing 45 – Verknüpfung des graphischen Modellelements mit der Verfolgbarkeitsdokumentation im BPMN-Format

Um zu den Modellelementen des PRM und Elementen der Anwendungsfall- und Klassendiagramme des CIM die Referenzen auf die Abschnitte der jeweiligen Verfolgbarkeitsdokumentation hinzuzufügen, wurde ein XSL-Stylesheet prototypisch implementiert. Listing 46 zeigt dieses Stylesheet in Auszügen, der Anhang G.6 zeigt es vollständig. Es wird auf das XMI-Format des Modells angewendet und fügt die zusätzlichen eAnnotations-Elemente ein, die vom Werkzeug TOPCASED als Referenz interpretiert werden (Listing 46, Zeilen 19-31). Die dafür notwendige Referenzierungsangabe wird anhand der übergebenen Parameter und der `xmi:id` des aktuell bearbeiteten Elements berechnet und gesetzt (Listing 46, Zeile 29). Auf die prototypische Implementierung eines analogen Stylesheet für die BPMN-Modelle wurde im Rahmen dieser Arbeit verzichtet, da der Erkenntnisfortschritt durch die Betrachtung der UML-Modellierung ebenfalls erzielt werden kann.

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0" ... >
(3)
(4)   <!-- ... -->
(5)
(6)     <xsl:template match="/">
(7)       <xsl:apply-templates/>
(8)     </xsl:template>
(9)
(10)    <xsl:template match="@*|node()">
(11)      <xsl:copy>
(12)      <xsl:apply-templates select="@*|node()"/>
(13)      </xsl:copy>
(14)    </xsl:template>
(15)
(16)    <xsl:template match="packagedElement">
(17)      <xsl:copy>
(18)      <xsl:apply-templates select="@*|node()"/>
(19)      <xsl:element name="eAnnotations">
(20)        <xsl:attribute name="xmi:id" select="concat(generate-id(.),
      '_govobj')"/>
(21)        <xsl:attribute name="source">
(22)          <xsl:text>http://www.topcased.org/resources</xsl:text>
(23)        </xsl:attribute>
(24)        <xsl:element name="details">
(25)          <xsl:attribute name="xmi:id" select="concat(generate-
      id(.),'_dtls')"/>
(26)          <xsl:attribute name="key">
(27)            <xsl:text>RR0</xsl:text>
(28)          </xsl:attribute>
(29)          <xsl:attribute name="value" select="concat($root-dir, $trace-file ,
      '#',
                  @xmi:id)/>
(30)        </xsl:element>
(31)      </xsl:element>
(32)      </xsl:copy>
(33)    </xsl:template>
(34)
(35)  </xsl:stylesheet>
```

Listing 46 – XSL-Stylesheet zur Ergänzung des PRM und CIM um Referenzen zur Verfolgbarkeitsdokumentation

6.3 Verwendungsmöglichkeiten

Das selektive, das interaktive und das ungeführte Nachvollziehen wurden als prinzipielle Verwendungsmöglichkeiten von Verfolgbarkeitsinformationen in Abschnitt 2.1 eingeführt. Für den Gesamtansatz dieser Arbeit muss deshalb gezeigt werden, ob und in welcher Form sie unterstützt werden. Anhand konkreter Beispiele können aus diesen prinzipiellen Verwendungsmöglichkeiten spezielle Einsatzszenarien für die Nutzung des Zusammenhangs zwischen Rechtsvorschriften und den Anforderungen an E-Government-Anwendungen abgeleitet werden.

6.3.1 Prinzipielle Verwendungsmöglichkeiten

Selektives Nachvollziehen ermöglicht die Eingrenzung der vorliegenden Informationen (z.B. anhand des Typs von Ausgangselementen, Beziehungen oder Zielelementen), so dass verschiedene Sichtweisen auf die Gesamtmenge der Verfolgbarkeitsinformationen konstruiert werden können. Selektives Nachvollziehen im Rahmen des Ansatzes dieser Arbeit würde entweder an der Verfolgbarkeitsdokumentation oder direkt an den Modellen bzw. dem annotierten Ausgangsdokument ansetzen können. Die Erfassung von Selektionskriterien und deren Anwendung auf die Inhalte der Verfolgbarkeitsdokumentation bzw. der Modelle bzw. des annotierten Ausgangsdokumentes sind hierfür Voraussetzung. Das Modellierungswerkzeug TOPCASED unterstützt die Selektion von Modellelementen lediglich durch einen einfachen Filter, der Modellelemente anhand ihres Namens selektiert (Abbildung 99). Der eingesetzte BPMN-Modeller unterstützt keine Selektion.

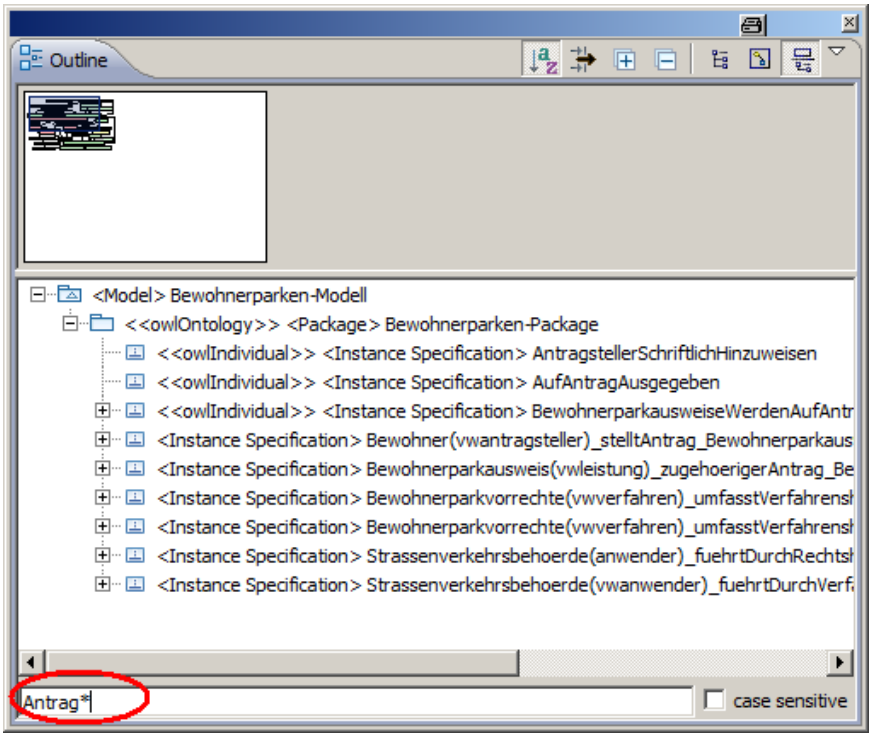


Abbildung 99 – Selektives Nachvollziehen unter Nutzung der Filterfunktion in TOPCASED und ausgehend vom PRM

Um das selektive Nachvollziehen über diese einfache Form hinausgehend im Rahmen dieser Arbeit zu ermöglichen, wurde für die Verfolgbarkeitsdokumentation eine Filterfunktion prototypisch implementiert. Diese Filterfunktion ermöglicht die Erfassung von Selektionskriterien für die Elemente im PRM und im CIM. Per JavaScript werden die Inhalte des Dokuments dynamisch gefiltert. Damit ist es möglich, die Verfolgbarkeitsinformationen in Form benutzerspezifischer Sichten einzugrenzen.

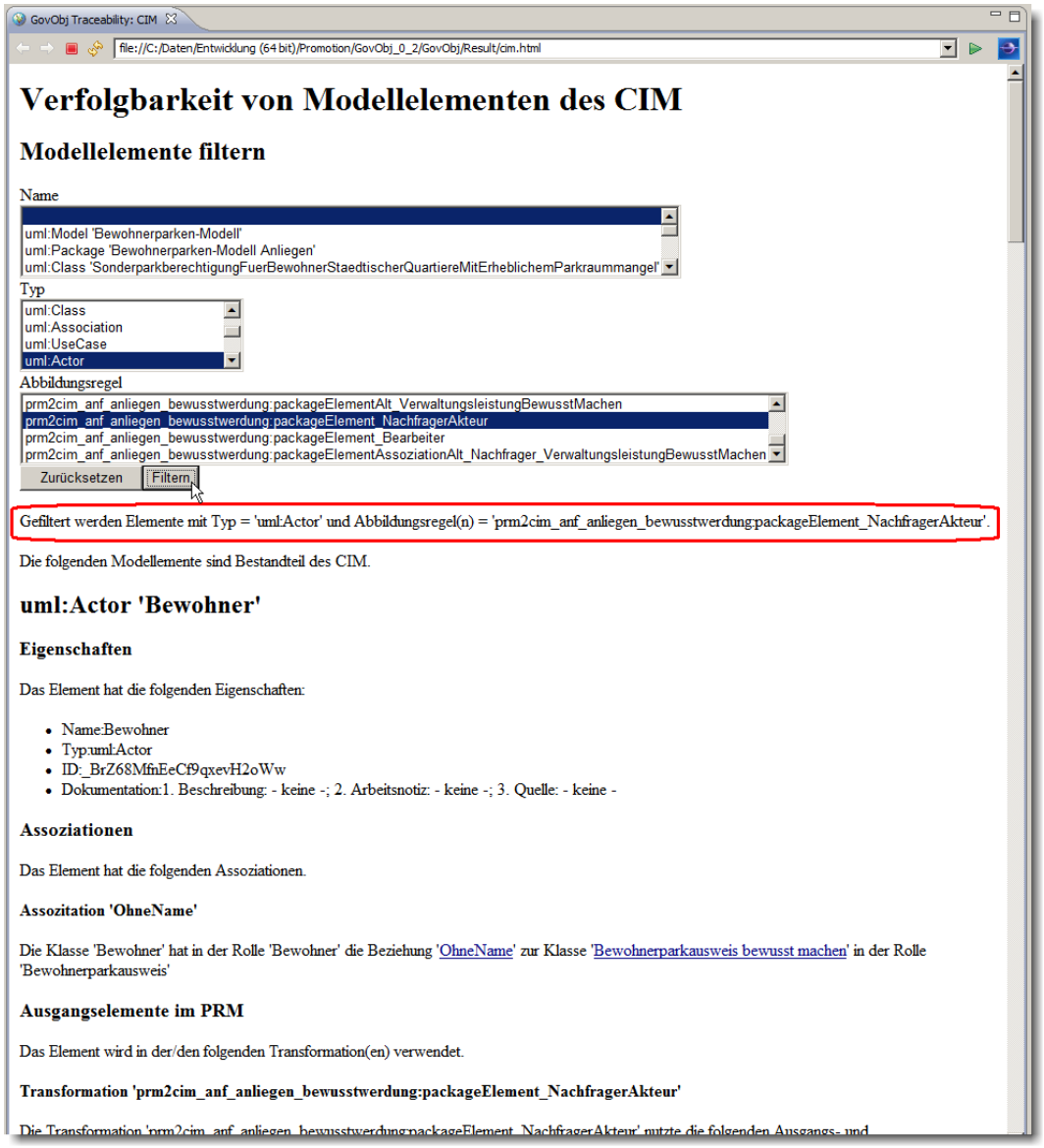


Abbildung 100 – Selektives Nachvollziehen unter Nutzung der Filterfunktion in der Verfolgbarkeitsdokumentation des PRM

Interaktives Nachvollziehen wird durch die bereitgestellten Verfolgbarkeitsinformationen ermöglicht, indem entlang des Beziehungspfades, der die Elemente verbindet, horizontal vorwärts und rückwärts bzw. vertikal navigiert wird. Im Rahmen dieser Arbeit wird die Navigation in der Verfolgbarkeitsdokumentation durch die untereinander verlinkten Ausgangs-, Geschwister- und Zielelemente ermöglicht. Es ist weiterhin möglich, zwischen den Annotationen und der annotierten Textpassage sowie zwischen den Modellelementen und den zu ihnen korrespondierenden Abschnitten der Verfolgbarkeitsdokumentation zu navigieren. Das ungeführte Nachvollziehen wird dadurch unterstützt, dass die Verfolgbarkeitsdokumentation es ermöglicht, eine Annotation im Ausgangsdokument oder ein Element in einem Modell auszuwählen, um dessen Kontext, Beziehungen und Details innerhalb der Dokumentation einzusehen.

Die prinzipiellen Verwendungsmöglichkeiten von Verfolgbarkeitsinformationen werden durch den Gesamtansatz zur Verfolgbarkeit somit unterstützt. Diese Unterstützung ist sowohl für funktionale, als auch für nicht-funktionale Anforderungen gegeben, die hinsichtlich ihrer Verfolgbarkeit im Gesamtansatz keine Unterschiede aufweisen.

6.3.2 Exemplarische Verwendungsmöglichkeiten

Aus den prinzipiellen Verwendungsmöglichkeiten können anhand von Beispielen konkrete Einsatzszenarien abgeleitet werden, die die Verwendung des Gesamtansatzes in seinem Anwendungskontext deutlich machen. Da die vertikale Verfolgbarkeit innerhalb des Gesamtansatzes eine untergeordnete Rolle spielt, stellen die folgenden Beispiele die Nutzung der horizontalen Verfolgbarkeit vor.

6.3.2.1 Beispiel 1: Priorisierung von Anforderungen

Bei der professionellen Softwareentwicklung kann im Projektverlauf eine Situation eintreten, in der Kosten- oder Zeitrestriktionen es erfordern, Anforderungen und deren Umsetzung für das aktuelle Release und folgende Releases zu priorisieren. In einer solchen Situation muss u. a. zwischen dem Entwicklungsaufwand und den umgesetzten Anforderungen ein Kompromiss gefunden werden. Wright weist in diesem Zusammenhang darauf hin, dass Verfolgbarkeitsbeziehungen vor „Vergoldungen" der Systemkomponenten schützen können, indem sichergestellt wird, dass zwischen notwendigen Anforderungen und teuren, unnötigen Leistungsmerkmalen unterschieden wird.[391] Durch den Gesamtansatz dieser Arbeit wird es möglich, Anforderungen auf ihre Ursprünge in den Texten von Rechtsvorschriften zurückzuführen. Dadurch wird für die Beteiligten transparent, welche Anforderung einen Anteil an der Erfüllung des gesetzlichen Auftrags hat. Diese Information kann für die Priorisierung hilfreich sein, wenn beispielsweise durch das aktuelle Release der gesetzliche Auftrag erfüllt werden muss, die Umsetzung weitergehender Anforderungen aber auf nachfolgende Releases verschoben werden könnte. In diesem Fall würde das selektive, rückwärtsgerichtete Nachvollziehen eingesetzt werden können, indem alle Elemente des CIM selektiert werden, die keine Ausgangselemente im PRM des Verwaltungshandelns haben und auch selbst keine Verfeinerung eines Elements mit solchen Ausgangselementen sind. Weil diese Elemente keine Verfolgbarkeitsbeziehungen zu Ursprüngen in Rechtsvorschriften haben, können sie im Rahmen der Priorisierungsaktivitäten gezielt geprüft werden.

6.3.2.2 Beispiel 2: Auswirkungsanalyse bei Gesetzesänderungen

Im Falle einer Gesetzesänderung kann der Ansatz dieser Arbeit dazu beitragen, dass die von den geänderten Textpassagen betroffenen Anforderungen identifiziert werden können. Dazu sind die annotierten Textpassagen der Rechtsvorschrift mit dem geänderten Text zu vergleichen. Für geänderte Passagen, die mit einer Annotation versehen sind, kann durch vorwärtsgerichtetes Verfolgen zum PRM des Verwaltungshandelns und weiter zu Elementen in Prozessmodellen oder Anforderungsmodellen des CIM navigiert werden. Diese Elemente im CIM können dann gezielt geprüft werden, ob und wie sie von der Änderung betroffen sind. Durch Nutzung der sich an das CIM anschließenden Verfolgbarkeitsbeziehungen kann man bis zu den Systemteilen navigieren, die von der Änderung konkret betroffen sind.

Der Ansatz dieser Arbeit kann auch in Vorbereitung einer Gesetzesänderung nützlich sein, wenn (noch) Spielraum bei der Ausgestaltung der Änderung besteht. In einem solchen Fall können für jede zur Auswahl stehende Variante die Auswirkungen der Änderung ermittelt werden. Durch den Vergleich der Auswirkungen kann die Variante bestimmt werden, die beispielsweise den geringsten Änderungsumfang verursacht oder eine schnelle Umsetzung ermöglicht. Diese Informationen können anschließend im Sinne des Erfüllungsaufwandes nach NKRG §2 bei der Gesetzesänderung berücksichtigt werden.

6.3.2.3 Beispiel 3: Verfahrensverzeichnis der Sozialversicherung

Jedermann hat das Recht, eine in der Praxis als Verfahrensverzeichnis bezeichnete Aufstellung der in einer Behörde eingesetzten automatisierten Verfahren zur Erhebung, Verarbeitung und Nutzung personenbezogener Daten einzusehen. Dieses Verzeichnis muss u.a. die Zweckbestimmungen der Datenerhebung, -verarbeitung oder -nutzung sowie eine

[391] Vgl. [Wright, 1991], zitiert nach [Rahmesh&Jarke, 1999], S. 5.

Beschreibung der betroffenen Personengruppen und der diesbezüglichen Daten oder Datenkategorien enthalten.[392] Für große E-Government-Anwendungen, beispielsweise solche der gesetzlichen Unfallversicherung, ist die Erstellung des Verzeichnisses nach eigenen Beobachtungen mit erheblichem Aufwand verbunden. In der Praxis wird beispielsweise von der vorliegenden Anforderungsspezifikation und der Beschreibung der Prozessmodelle ausgegangen und für jedes Element geprüft, ob und welche personenbezogenen Daten es betrifft.[393]

Für die personenbezogenen Daten muss im Verfahrensverzeichnis aufgeführt werden, zu welchem Zweck ihre Erhebung, Verarbeitung oder Nutzung erfolgt. Bei Nutzung des Ansatzes dieser Arbeit kann von den betroffenen Elementen zur annotierten Textpassage navigiert werden, aus der das Element hervorgegangen ist. Durch diesen Bezug zur Rechtsvorschrift kann der Zweck anhand des gesetzlichen Auftrags formuliert werden. Eine weitere Nutzungsmöglichkeit bietet der Ansatz dieser Arbeit, wenn Annotationen ausgewertet werden, die auf Instanzen der Ontologie-Klassen `VwAntragsteller`, `VwBetroffenerDritter` usw. basieren. Werden diese Annotationen vorwärtsgerichtet bis zu den Elementen der Anforderungsspezifikation verfolgt, bieten sie Ansatzpunkte, um für diese Personengruppen die erhobenen, verarbeiteten und genutzten personenbezogenen Daten festzustellen. Somit ergeben sich aufgrund der aufgezeichneten Verfolgbarkeitsbeziehungen vorwärts- und rückwärtsgerichtete Navigationsmöglichkeiten, die zur Bereitstellung eines aktuellen und vollständigen Verfahrensverzeichnisses zur Einsicht durch jedermann genutzt werden können.

6.4 Zusammenfassung

Die während der durchgeführten Transformationen vom Ausgangsdokument in das PRM und vom PRM in das CIM aufgezeichneten Verfolgbarkeitsinformationen ermöglichen umfassend die horizontale Verfolgbarkeit. Es wird sowohl die vorwärts- als auch die rückwärtsgerichtete Navigation entlang der horizontalen Verfolgbarkeitsbeziehungen unterstützt. Für die vertikale Verfolgbarkeit bietet die aufgezeichnete Information Ansatzpunkte, die zusammen mit weiteren Informationen oder anderen Ansätzen sinnvoll kombiniert werden können, um eine umfassende vertikale Verfolgbarkeitsunterstützung zu erhalten.[394] Sowohl für funktionale und nicht-funktionale Anforderungen ist die Verfolgbarkeitsinformation nutzbar.

Für die Aufbereitung dieser Informationen gibt es verschiedene technische Lösungsansätze. Für die Demonstration der prinzipiellen Verwendungsmöglichkeiten ist im Rahmen dieser Arbeit die Generierung von verlinkten HTML-Dokumenten mit eingebettetem JavaScript durch eine XSL-Transformation eine grundsätzlich ausreichende technische Lösung. Um darüber hinausgehende Auswertungsmöglichkeiten zu bieten, könnten die aufgezeichneten Verfolgbarkeitsinformationen in einer Datenbank (z.B. Graphdatenbank oder XML-Datenbank) oder in einem Repository hinterlegt werden. Dann könnten mit einer geeigneten Abfragesprache dynamisch Auswertungen erzeugt werden.

Ein solches Repository ist beispielsweise das auf Basis der Eclipse-Plattform basierende *EMFStore*[395], das zum Zeitpunkt der Fertigstellung dieser Arbeit im Rahmen des Projektes *EMFTrace*[396] an der TU-Illmenau zu einem Repository für die Speicherung und Auswertung von Traceability-Links weiterentwickelt wird. Das Repository ist flexibel gehalten und bietet

[392] Vgl. BDSG §4, §4e und §4f Satz 1, Absätze 4 und 5.
[393] Freundliche telefonische Auskunft von Anita Fischer (Abteilungsleiterin, Referentin der Geschäftsführung und Datenschutzbeauftragte bei der Berufsgenossenschaft der Bauwirtschaft (BG BAU) von 1992 bis 2005, heute Fachkoordinatorin eKommunikation/Führungsinformation BG BAU) am 01.07.2011.
[394] Ein solchen diese Arbeit ergänzenden Ansatz für die Eclipse-Plattform stellt beispielsweise [Wenzel, 2007] vor.
[395] Vgl. [Marcus&Maletic, 2003].
[396] Vgl. [Bode et al., 2011].

Verbindungsmöglichkeiten für verschiedene Formate. Die Übernahme von den durch die MDA-konformen QVT-Transformation aufgezeichneten Verfolgbarkeitsklassen und -instanzen erscheint prinzipiell möglich.

Die verlinkten HTML-Dokumente der im Rahmen dieser Arbeit erzeugten Verfolgbarkeitsdokumentation bieten Möglichkeiten zum selektiven, zum interaktiven und zum ungeführten Nachvollziehen. Dadurch können für unterschiedliche Rollen im Softwareentwicklungsprozess unterschiedliche Fragestellungen beantwortet werden. Dies wurde an ausgewählten Beispielen vorgestellt.

Durch die Kombination der Verfolgbarkeitsinformationen mit den in den Abschnitten 3 bis 5 entwickelten Bestandteilen in einem Gesamtansatz, durch seine prototypische Implementierung und die exemplarisch dargestellten Verwendungsmöglichkeiten eröffnen sich Nutzenpotenziale, die zur Zusammenfassung dieser Arbeit überleiten (Abschnitt 7).

7 Ergebnis und Abschluss

Aufgabe der vorliegenden Arbeit ist es, einen Beitrag zur Verbesserung der Verfolgbarkeit im Vorfeld der Softwareentwicklung von E-Government-Anwendungen zu leisten. In diesem Abschnitt werden die bei der Bearbeitung der Aufgabenstellung erarbeiteten Ergebnisse zusammengefasst. Es wird von der Fragestellung dieser Arbeit ausgegangen und die zugrunde liegende Arbeitshypothese betrachtet (Abschnitt 7.1). Durch Belegen der Arbeitshypothese mit den in dieser Arbeit erzielten Ergebnissen wird gezeigt, dass eine Antwort auf die Fragestellung gefunden werden kann (Abschnitt 7.2). Darüber hinaus wird der durch die Arbeit erzielte Erkenntnisfortschritt zusammengefasst (Abschnitt 7.3). Abschließend werden eine Einschätzung zu Rahmenbedingungen des Einsatzes gegeben sowie weitere Entwicklungen aufgezeigt (Abschnitt 7.4).

7.1 Hintergrund und Arbeitshypothese

Im Rahmen des E-Governments steht die Durchführung der Verwaltungsprozesse unter intensiver Nutzung der Informationstechnik im Vordergrund. Für diese Arbeit ist die Anwendungssoftware relevant, die einen wichtigen Teil dieser Informationstechnik ausmacht und zur Erledigung spezieller fachlicher Aufgaben dient. Weiterhin beschränkt sich diese Arbeit auf Verwaltungsprozesse, in deren Rahmen Verwaltungsleistungen mit unmittelbarem Bezug zum Einzelfall eines Bürgers (so genannte Bürgerdienste) erbracht werden. Sie untersucht daher den Zusammenhang zwischen Rechtsvorschriften, die die Erstellung dieser Verwaltungsleistungen regeln und Anforderungen an die eingesetzte E-Government-Anwendung. Aus Sicht der Softwaretechnik handelt es sich dabei um eine spezielle Form der Verfolgbarkeit von Anforderungen, die so genannte Verfolgbarkeit im Vorfeld der Anforderungsspezifikation (Pre-RS Traceability), da sie Aspekte betrifft, die relevant sind, bevor die Anforderungen in eine Spezifikation eingeflossen sind (Ursprünge von Anforderungen). Diese Form der Verfolgbarkeit wird in der Praxis noch nicht ausreichend beherrscht und es zeigt sich noch Forschungsbedarf. Im Gegensatz dazu gilt die Verfolgbarkeit im Nachgang der Anforderungsspezifikation (Post-RS Traceability) als umfassend erforscht und wird in der Praxis gut beherrscht. Da der Zusammenhang zwischen Rechtsvorschriften und Anforderungen an E-Government-Anwendungen das Vorfeld der Anforderungsspezifikation betrifft, hat dieses Thema grundsätzlich Relevanz, so dass hier ein Beitrag zur Verbesserung der Verfolgbarkeit geleistet werden kann. Daraus ergibt sich die Fragestellung, wie im Vorfeld der Softwareentwicklung die Verfolgbarkeit der Ursprünge von Anforderungen in Rechtsvorschriften an E-Government-Anwendungen verbessert werden kann (vgl. Abschnitt 1.3 auf Seite 7).

Aktuelle Entwicklungen im Bereich der modellgetriebenen Architektur, die Konvergenz von Modellierungsstandards mit Ontologiesprachen, die Techniken und Standards des Semantic Web sowie die Entwicklungen im Rahmen des E-Governments wurden zu Beginn dieser Arbeit analysiert, um Ansatzpunkte für Verbesserungsmöglichkeiten zu finden. Mit den Techniken des Semantic Web ist es möglich, ein Vokabular in Form einer Ontologie zu definieren und, darauf basierend, Inhalte im World Wide Web des Internets (WWW) mit einer definierten Semantik zu versehen. Für Ontologien ist durch das Ontology Definition Metamodel (ODM) eine prinzipielle Möglichkeit geschaffen worden, ihre Konzepte auf Konzepte aus dem Bereich der Modellierung abzubilden. Da die Bedeutung der Modellierung in der Softwaretechnik in den vergangenen Jahren stark zugenommen hat, ist heute die modellgetriebene Softwareentwicklung mit der MDA (Model Driven Architecture) als ihrem wichtigsten Vertreter ein zentrales Paradigma der Softwaretechnik. Auf diesen Entwicklungen gründete die Arbeitshypothese, die davon ausging, dass durch geeignete Kombination ausgewählter Elemente dieser Entwicklungen ein neuartiger Ansatz entstehen könnte. Als Arbeitshypothese dieser Arbeit wurde einleitend formuliert (vgl. Abschnitt 1.3 auf Seite 8):

Werden die Techniken des Semantic Web auf die Texte von Rechtsvorschriften angewendet, können Ursprünge von Anforderungen, basierend auf einer geeigneten Ontologie, mit einer definierten Semantik versehen werden, so dass unter Berücksichtigung verwaltungsspezifischer Aspekte und unter Nutzung der Abbildungsregeln des ODM-Standards der Anschluss an die MDA-basierte, modellgetriebene Softwareentwicklung von E-Government-Anwendungen gefunden wird, der die Aufzeichnung von Verfolgbarkeitsinformationen und deren Nutzung im Sinne der Fragestellung, basierend auf offenen Standards, ermöglicht.

Die Arbeitshypothese umfasst die folgenden Bestandteile, deren Gültigkeit durch die erzielten Arbeitsergebnisse zu belegen ist, um auf die Gültigkeit der gesamten Hypothese schließen zu können:

(a) Die Techniken des Semantic Web können auf Texte von Rechtsvorschriften angewendet werden, um Ursprünge von Anforderungen, basierend auf einer geeigneten Ontologie, mit einer definierten Semantik zu versehen.

(b) Es erfolgt die Nutzung verwaltungsspezifischer Aspekte.

(c) Es erfolgt die Berücksichtigung des ODM-Standards.

(d) Es wird Anschluss an die MDA-basierte, modellgetriebene Softwareentwicklung von E-Government-Anwendungen gefunden.

(e) Die aufgezeichneten Verfolgbarkeitsinformationen können genutzt werden, um die Verfolgbarkeit der Ursprünge von Anforderungen in Rechtsvorschriften an E-Government-Anwendungen zu verbessern.

(f) Eine Implementierung mittels offener Standards ist möglich.

Der Vorgehensweise zur Bearbeitung der Fragestellung folgend, wird im Folgenden Bezug auf diese Bestandteile der Arbeitshypothese genommen.

7.2 Bearbeitung der Fragestellung und ihre Ergebnisse

Bestandteil (a) der Arbeitshypothese war die begründete Vermutung, dass die Techniken des Semantic Web auf die Texte von Rechtsvorschriften angewendet werden könnten, um Ursprünge von Anforderungen mit einer definierten Semantik zu versehen. Generell ermöglichen Standards, wie RDF (Resource Description Framework), RDFS (Resource Description Framework Schema), OWL (Web Ontology Language) und RDFa (Resource Description Framework in attributes) Inhalte des WWW zu annotieren, die beispielsweise als HTML- oder XHTML-Dokumente vorliegen. Gesetzes- und Vorschriftensammlungen im Internet zeigen, dass Rechtsvorschriften als HTML- oder XHTML-Dokumente vorliegen oder in diese Form gebracht werden können.[397] Eine generelle Voraussetzung für Annotationen ist, dass die Semantik der Annotationen in Form von Konzepten definiert und in einer geeigneten Sprache als Ontologie formalisiert ist.

Die im Rahmen dieser Arbeit durchgeführte Analyse vorhandener Ontologien der Rechtsanwendung und vorhandener Ontologien aus dem Bereich der öffentlichen Verwaltung zeigte einerseits aus Sicht der Rechtsanwendung Ansatzpunkte für Konzepte, deren Existenz in Rechtsvorschriften vorausgesetzt werden konnte. Andererseits zeigten Ontologien und ontologische Ansätze aus Sicht der Verwaltung die Besonderheiten ihres Handelns auf. Im Ergebnis ergab sich dadurch die Notwendigkeit einer Ontologie, die zum einen Konzepte umfasst, mit denen Ursprünge von Anforderungen aus dem

[397] Beispiele sind zu finden unter: http://www.juris.de, http://www.gesetze-im-internet.de, http://www.bravors.brandenburg.de (letzter Aufruf: 21.08.2011).

Verwaltungshandeln annotiert werden können und die zum anderen auf existierende Konzepte in Rechtsvorschriften zurück geht. Eine solche Ontologie existierte bisher nicht, so dass sie im Rahmen dieser Arbeit entwickelt wurde. Diese Ontologie des Verwaltungshandelns formalisiert Handlungsgegenstände, Handlungen und Akteure sowie Beziehungen, die diese Konzepte untereinander haben. Die Konzepte basieren auf entsprechenden Konzepten der Rechtsanwendung und diese auf Elementen in Rechtssätzen, die einen unmittelbaren Bezug zu Textelementen (z.B. Wörtern, Wortgruppen, Sätzen, Absätzen, Paragraphen) haben. Aus konzeptioneller Sicht kann deshalb davon ausgegangen werden, dass die Konzepte des Verwaltungshandelns zur Annotation der Texte von Rechtsvorschriften geeignet sind. Weiterhin umfasst die Ontologie auch exemplarische Erweiterungen, die etablierte Konzepte des Verwaltungshandelns formalisieren und nicht (transitiv) auf Elemente von Rechtssätzen zurückgeführt werden können. Ein Beispiel hierfür ist das Konzept der Lebenslage, das als wichtiger Eckpfeiler für Bürgerdienste im Kontext des E-Government etabliert ist. Die Ontologie des Verwaltungshandelns wurde mit dem Werkzeug Protegé erstellt und im OWL-Format (Web Ontology Language) bereitgestellt. Da OWL eine Sprache für die Annotation von WWW-Inhalten ist, kann die Ontologie des Verwaltungshandelns in dieser Form direkt zur Annotation von Rechtsvorschriften, die als HTML- oder XHTML-Dokumente vorliegen, verwendet werden. Mit der frei verfügbaren Werkzeugunterstützung OntoMat-Annotizer, die am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB) der Universität Karlsruhe entwickelt wurde, wurde in dieser Arbeit praktisch demonstriert, wie in Rechtsvorschriften Textpassagen, die Ursprünge von Anforderungen sind, mit Annotationen, basierend auf der Ontologie des Verwaltungshandelns versehen werden können. Das Werkzeug OntoMat-Annotizer vermittelt zudem einen Eindruck davon, wie Fachmitarbeiter in einem konkreten Softwareentwicklungsprojekt den Ansatz nutzen und Annotationen erstellen können. Allerdings ist die Weiterentwicklung dieses Werkzeugs eingestellt worden, so dass es nicht den aktuellen Standard RDFa zur Annotation von XHTML-Dokumenten unterstützt. Ein vergleichbares Werkzeug für die RDFa-Annotation stand bis zur Fertigstellung dieser Arbeit nicht zur Verfügung. Deshalb wurden die RDFa-Annotationen mit einem einfachen XML-Editor manuell zu XHTML-Dokumenten hinzugefügt. Im Ergebnis waren Ursprünge von Anforderungen in den Texten von Rechtsvorschriften einerseits durch OntoMat-Annotationen und andererseits durch RDFa-Annotationen mit einer definierten Semantik versehen, die sich durch Instanziierung von Konzepten der Ontologie des Verwaltungshandelns ergab.

Annotationen sind einerseits Instanzen der Ontologie-Konzepte (Individuen). Andererseits verfügen sie definitionsgemäß über einen Annotationskontext, der sie mit der annotierten Textpassage verbindet. Aus Sicht der Verfolgbarkeit sind die Individuen aufgrund des Annotationskontextes der korrespondierenden Annotation zu den durch sie annotierten (Text-)Elementen stets verfolgbar. Mit diesem Ergebnis kann der Bestandteil (a) der Arbeitshypothese belegt werden, demzufolge in der Anwendung der Techniken des Semantic Web auf die Texte von Rechtsvorschriften eine Möglichkeit existieren soll, die Ursprünge von Anforderungen, basierend auf einer geeigneten Ontologie mit einer definierten Semantik zu versehen.

Die so annotierten Rechtsvorschriften sind für die modellgetriebene Softwareentwicklung nicht direkt nutzbar, geht die MDA doch immer von der Existenz eines initialen Modells aus. Für die Überführung von Ontologien, die in der Sprache RDFS und der darauf aufbauenden OWL-Sprache vorliegen, in die modellgetriebene Architektur hat die Object Management Group (OMG) das ODM von 2003 bis 2009 erarbeitet. Damit besteht seit Mai 2009 die Möglichkeit, basierend auf einem offenen Standard die Ontologie des Verwaltungshandeln in ein Modell zu überführen, das im Rahmen der MDA genutzt werden kann. Allerdings beschränkt sich die MDA auf die Softwareentwicklung im engeren Sinne und vernachlässigt in ihrer Darstellung für die professionelle Softwareentwicklung relevante Phasen, die sich an die Programmierung anschließen (z.B. Test, Einführung, Schulung) oder der Anforderungsanalyse vorausgehen. Die vorliegende Arbeit untersucht jedoch das Vorfeld

der Anforderungsanalyse. Deshalb wurde hier eine Erweiterung der MDA entwickelt, die ihren Einsatz auch im Vorfeld der Anforderungsanalyse ermöglicht. Zentrales Element dieser Erweiterung ist das Pre-Requirements Model (PRM, dt. Modell im Vorfeld der Anforderungsspezifikation), das die vorhandenen Modelltypen Computation Independent Model (CIM, dt. informationstechnikunabhängiges Modell), Platform Independent Model (PIM, dt. plattformunabhängiges Modell) und Platform Specific Model (PSM, dt. plattformspezifisches Modell) erweitert. Das PRM formalisiert als MDA-konformes Modell Elemente und deren Beziehungen, die ein System von Aspekten vereinfacht beschreiben, die im Vorfeld einer Anforderungsspezifikation relevant sind und Ursprünge von Anforderungen darstellen können. Das CIM repräsentiert einen weiteren für diese Arbeit relevanten Modelltyp. Modelle dieses Typs stellen die Umgebung des Systems und die Anforderungen an das System dar und repräsentieren deshalb die Anforderungsspezifikation im Kontext der MDA. Eine E-Government-Anforderungsspezifikation muss Anforderungen an die Verwaltungsprozesse und an die diese Prozesse unterstützende Anwendungssoftware umfassen. Dazu werden im Rahmen dieser Arbeit Prozessmodelle in BPMN (bis Version 1.2: Business Process Modelling Notation bzw. ab Version 2: Business Process Model and Notation) und Anwendungsfall- und Klassendiagramme der UML (Unified Modeling Language) verwendet, deren Elemente über eine fachliche Beschreibung verfügen. Es ist möglich, diese Diagramme um (weitere) Verhaltensdiagramme zu ergänzen, die verhaltensorientierte Aspekte nicht nur in Form von Text beschreiben sondern auch formal im Modell ausdrücken (z.B. mittels Aktivitäten-, Zustands- oder Sequenzdiagrammen der UML).

Mit dem PRM wird es möglich, die durch Annotationen gewonnenen OWL-Individuen für die MDA in Form eines Modells nutzbar zu machen. Dazu werden im Rahmen dieser Arbeit aus den Annotationen die Individuen extrahiert und diese unter Verwendung der aus dem ODM-Standard abgeleiteten Abbildungsregeln in Modellelemente des PRM transformiert. Die Erweiterung der MDA um das PRM schafft die Voraussetzung dafür, das Anschluss an die MDA-basierte, modellgetriebene Softwareentwicklung gefunden werden kann.

Die prototypische Implementierung nutzt im Rahmen dieser Arbeit entwickelte XSL-Stylesheets, eine kurze in Java programmierte Konvertierungsfunktion und ein Eclipse-Plugin, mit dem das im ODM beschriebene UML Profil für RDF und OWL für die Modellierung genutzt werden kann. Mit einem XSL-Stylesheet werden aus dem XHTML-Dokument die mit dem Werkzeug OntoMat erzeugten Annotationen in Form von OWL-Individuen extrahiert. Auf HTML-Dokumente wird zu diesem Zweck die Java-Konvertierfunktion angewandt, da HTML-Dokumente mit dem Stylesheet nicht verarbeitet werden können. Anschließend wird ein weiteres XSL-Stylesheet verwendet, um die ODM-Abbildungsregeln auf die extrahierten OWL-Individuen anzuwenden. Der frei verfügbare XSL-Prozessor Saxon überführt anhand des XSL-Stylesheets das XML-Dokument mit den OWL-Individuen in ein XML-Dokument des PRM. Für die Transformation RDFa-annotierter XHTML-Dokumente wurde ein anderer Weg implementiert, da eine einleitende Extraktion der Annotationen nicht notwendig war. Ein XSL-Stylesheet, das die ODM-Abbildungsregeln implementiert, wird direkt auf das XHTML-Dokument mit den RDFa-Annotationen angewendet und überführt sie in das PRM. Das XML-Dokument des PRM liegt im von der OMG standardisierten XMI-Format (XML Metadata Interchange) vor, das zum Austausch von Modellen zwischen Modellierungs- und Softwareentwicklungswerkzeugen verwendet wird. Dadurch kann es in dem für diese Arbeit verwendeten frei verfügbaren Modellierungswerkzeug TOPCASED weiterbearbeitet werden.

Durch diese prototypische Umsetzung können die als HTML- bzw. XHTML-Dokumente vorliegenden annotierten Rechtvorschriften in ein PRM des Verwaltungshandeln überführt werden, das Aspekte der Rechtsvorschrift vereinfacht beschreibt, die im Vorfeld einer Anforderungsspezifikation relevant sind und Ursprünge von Anforderungen darstellen. Das initiale PRM des Verwaltungshandeln kann durch weitergehende Modellierung um verwaltungsspezifische Aspekte erweitert werden, die sich nicht unmittelbar aus dem Text der Rechtsvorschrift ergeben. Im Rahmen dieser Arbeit wurde anhand von Beispielen

gezeigt, wie mit dem Werkzeug TOPCASED diese Aspekte im PRM berücksichtigt werden können. Dazu werden insbesondere solche Modellelemente verwendet, die in der Ontologie des Verwaltungshandeln formalisiert wurden, ohne dass sie (transitiv) auf Elemente von Rechtssätzen zurückgeführt werden konnten. Der Bestandteil (b) der Arbeitshypothese wird hierdurch erfüllt, da die Berücksichtigung verwaltungsspezifischer Aspekte möglich ist, die sich nicht oder nicht vollständig aus den Texten von Rechtsvorschriften ableiten lassen.

Die Aufzeichnung von Informationen über die durchgeführte Abbildung von Annotationen in Form von OWL-Individuen bzw. von RDFa-Annotationen auf Modellelemente im PRM ist Voraussetzung für die Verfolgbarkeit der durchgeführten Transformation. Weder das ODM noch XSLT sehen einen standardisierten Mechanismus hierfür vor. Deshalb wurde, ausgehend von den bekannten Möglichkeiten der XSL-Programmiersprache, eine eigene Umsetzung für die Aufzeichnung der Verfolgbarkeitsinformationen implementiert. Sie zeichnet Informationen in einer Form auf, die sich am QVT-Standard (siehe unten) der MDA-Transformationen orientiert. Der Bestandteil (c) der Arbeitshypothese, der in der Nutzung von Abbildungsregeln des ODM-Standards einen Beitrag zur Verfolgbarkeit sieht, ist dadurch belegt.

Durch das PRM wird die MDA auf das Vorfeld der Anforderungsspezifikation ausgeweitet, allerdings ist allein hierdurch noch kein Anschluss an die MDA-basierte, modellgetriebene Softwareentwicklung basierend auf den vorhandenen Modelltypen gefunden. Deshalb wurde im Rahmen dieser Arbeit nach Möglichkeiten gesucht, wie mit MDA-konformen Transformationen die Abbildung der Elemente eines PRM auf Elemente in einem CIM erfolgen kann, das im Rahmen der MDA die Rolle einer Anforderungsspezifikation einnimmt. Eine Herausforderung ergab sich auch aus der semantischen Lücke zwischen den beiden Modelltypen. Um diese Lücke zu schließen, wurde ein Transformationsansatz gewählt, der ein spezielles Hilfsmodell verwendet. Dieses Hilfsmodell nimmt eine vergleichbare Rolle ein, wie das Plattformmodell bei der Transformation von PIM in PSM, die bei der Darstellung der MDA häufig in den Vordergrund gerückt wird und deshalb auch gut dokumentiert ist. Als Hilfsmodell wurden die im Rahmen des Projektes Local Electronic Government (eLoGo) am Kommunalwissenschaftlichen Institut der Universität Potsdam entwickelten Referenzmodelle für E-Governement verwendet. Da Referenzmodelle bei der Erstellung anderer Modelle nützlich sein sollen, können sie auch prinzipiell in einem MDA-Transformationsansatz verwendet werden. Dazu mussten die ursprünglichen eLoGo-Projektergebnisse im Rahmen dieser Arbeit in eine Form überführt werden, die die Erstellung einer MDA-konformen Anforderungsspezifikation als CIM ermöglicht. Die dazu durchgeführte Analyse verschiedener Bereitstellungsformen berücksichtigte die Möglichkeiten, Beschränkungen und Perspektiven der verfügbaren Sprachen und Werkzeuge zur Modellierung von Verwaltungsprozessen und zur Modellierung von funktionalen und nicht-funktionalen Anforderungen an die Anwendungssoftware. Im Ergebnis der Analyse wurde das eLoGo-Referenzprozessmodell in BPMN-Diagramme migriert, da die Verwaltungsprozesse im CIM mittels BPMN zu modellieren waren. Das eLoGo-Referenzanforderungsmodell wurde in ein UML-Profil überführt, um es für die Modellierung von Anwendungsfall- und Klassendiagrammen zur Spezifikation der funktionalen und nicht-funktionalen Anforderungen nutzen zu können. Im Rahmen dieser Arbeit wurden dann Abbildungsregeln definiert, mit denen die Elemente eines PRM des Verwaltungshandelns mit den Elementen der eLoGo-Referenzmodelle zusammengeführt und auf die Elemente eines CIM abgebildet werden können. Diese Abbildungsregeln wurden anschließend für einen Ausschnitt der Modelle prototypisch implementiert. Dazu wurde der QVT-Standard verwendet (Query View Transformation, Teil der Meta Object Facilities), den die OMG von 2002 bis 2008 entwickelte und dessen zum Zeitpunkt der Erstellung dieser Arbeit aktuelle Version 1.1 im Januar 2011 verabschiedet wurde. Mit dem Werkzeug medini QVT wurden die implementierten QVT-Transformationen ausgeführt. Es wurde praktisch gezeigt, wie aus dem PRM des Verwaltungshandelns BPMN-Diagramme eines Verwaltungsprozesses sowie Anwendungsfall- und Klassendiagramme in UML erzeugt werden können. Im Ergebnis entsteht durch die Transformation ein CIM, das eine initiale

Anforderungsspezifikation eines E-Government-Anwendungssystems ist. Es umfasst zum einen Prozessmodelle, die Verwaltungsprozesse mittels BPMN darstellen und mit Fließtext beschreiben. Zum anderen sind Anwendungsfall- und Klassendiagramme Bestandteil des CIM. Ihre Modellelemente sind mit strukturiertem Fließtext beschrieben. In der prototypischen Implementierung der Transformation wurden diese Elemente hinsichtlich ihrer verhaltensorientierten Aspekte nicht weiter durch formale Modelle ergänzt, was durch Anpassung der eLoGo-Referenzmodelle möglich wäre. Diese initiale Spezifikation ist in nachfolgenden Aktivitäten des Softwareentwicklungsprozesses zu vervollständigen und weiterzuentwickeln. Für diese Aktivitäten stehen etablierte Techniken und Werkzeuge bereit, die insbesondere auch das Problem der Verfolgbarkeit im Nachgang der Anforderungsspezifikation (Post-RS Traceability) lösen. Die Weiterentwicklung der Spezifikation verlässt somit den Betrachtungsbereich dieser Arbeit. Sie belegt allerdings, dass im Sinne des Bestandteils (d) der Arbeitshypothese ein nahtloser Anschluss an den standardisierten modellgetriebenen Softwareentwicklungsprozess einer E-Government-Anwendung hergestellt wurde und auch Anschluss an die weitere Verfolgbarkeit gefunden wurde.

Während der Durchführung der PRM-CIM-Transformation werden entsprechend des QVT-Standards durch die Ausführungsumgebung Informationen in Form von Verfolgbarkeitsklassen und Verfolgbarkeitsinstanzen aufgezeichnet, die für die inkrementelle Modelltransformation ausgelegt sind. Da diese Informationen Aufschluss darüber geben, welches Modellelement des Ausgangsmodells in welches Modellelement des Zielmodells aufgrund welcher Abbildungsregel transformiert wurde, können sie prinzipiell durch Benutzer verwendet werden, um Zusammenhänge zwischen Elementen zu verfolgen.

Für einen Gesamtansatz ist es notwendig, die aufgezeichneten Verfolgbarkeitsinformationen aus der Transformation der Annotationen von Textelementen einer Rechtsvorschrift in das PRM des Verwaltungshandelns mit den Verfolgbarkeitsinformationen der anschließenden Transformation dieses PRM in das CIM einer initialen Anforderungsspezifikation eines E-Government-Anwendungssystems zu kombinieren. Im Rahmen dieser Arbeit wurde ein solcher Gesamtansatz entwickelt. Er basiert auf einer so genannten Verfolgbarkeitsdokumentation in Form verlinkter Hypertextdokumente, die jeweils für das annotierte Ausgangsdokument, für die Ontologie, für das PRM und für das CIM erzeugt werden. In der Verfolgbarkeitsdokumentation ist jede Annotation bzw. jedes Modellelemente mit seinen zentralen Eigenschaften (z. B. Name, Typ) und Beziehungen aufgeführt. Weiterhin ist dargestellt, in welchen Abbildungsregeln es verwendet wurde, aus welcher Annotation bzw. aus welchem Element es dabei hervorgegangen ist (Ausgangselement) bzw. in welchen Elementen es resultierte (Zielelemente). In der gleichen Abbildungsregel weiterhin verwendete Annotationen und Modellelemente des gleichen Modells sind ebenfalls dargestellt (Geschwisterelemente). Durch Hyperlinks sind die jeweiligen Elemente verbunden, so dass zwischen den Verfolgbarkeitsdokumenten entlang der Hyperlinks navigiert werden kann.

Um die Beziehung zwischen einem Element in der graphischen Darstellung des Diagramms (z. B. Anwendungsfall-, Klassen-, BPMN-Diagramm) zur Beschreibung des Elements in der Verfolgbarkeitsdokumentation herzustellen, wurden die Modelle um Hyperlinks erweitert. Ebenso wird mit der Beziehung einer Annotation im Ausgangsdokument zur Beschreibung des Elements in der Verfolgbarkeitsdokumentation verfahren. Die konkrete Ausgestaltung des Hyperlinks erfolgt in Abhängigkeit vom eingesetzten Werkzeug. Zur prototypischen Implementierung wurden im Rahmen dieser Arbeit mehrere XSL-Stylesheets entwickelt, die aus dem annotierten Ausgangsdokument, aus der XMI-Repräsentation des PRM und des CIM eine Verfolgbarkeitsdokumentation in Form von HTML-Dokumenten generieren, die untereinander verlinkt sind.

Die Verfolgbarkeitsdokumentation des Ausgangsdokumentes umfasst in der prototypischen Implementierung neben den Annotationen auch das ursprüngliche Ausgangsdokument. Durch JavaScript wird im Dokument der Annotationskontext dynamisch ausgewertet und

dadurch die Navigation von einer annotierten Textpassage zu ihrer Annotation unterstützt. Ausgehend von der Annotation kann dann beispielsweise weiter zu den Modellelementen im PRM und CIM navigiert werden. Von jeder Annotation kann über eine analoge JavaScript-Funktion auch zu ihrem Kontext, d.h. der durch sie annotierten Textpassage im Ausgangsdokument navigiert werden.

Ein weiteres XSL-Stylesheet dient dazu, in der XMI-Repräsentation des PRM und des CIM für jedes Element einen werkzeugspezifischen Verweis auf das korrespondierende Element in der Verfolgbarkeitsdokumentation hinzuzufügen. Dadurch kann aus der graphischen Darstellung eines Modellelements zum Element in der Verfolgbarkeitsdokumentation navigiert werden. Auf die prototypische Implementierung der umgekehrten Navigationsrichtung, die prinzipiell technisch möglich erscheint, wurde aufgrund der Abwägung zwischen Aufwand und Erkenntnisgewinn verzichtet.

Für den Gesamtansatz wurde im Rahmen dieser Arbeit gezeigt, dass das selektive, das interaktive und das ungeführte Verfolgen unterstützt werden. Die horizontale Verfolgbarkeit zwischen Elementen unterschiedlicher Modelle ist dabei umfassend vorwärts- und rückwärtsgerichtet möglich. Für die vertikale Verfolgbarkeit, die Elemente des gleichen Modells und verschiedener Modellversionen in Beziehung setzt, existieren Ansatzpunkte (z. B. Verfolgbarkeit zu Geschwisterelementen). Dieser Unterschied in der Unterstützung horizontaler und vertikaler Verfolgbarkeit hat seine Ursache im horizontalen Charakter der Verfolgbarkeitsinformation, die während der Durchführung von Transformationen unterschiedlicher Modelle (z. B. von Ausgangsdokument in das PRM bzw. vom PRM in das CIM) aufgezeichnet wird. Durch den entwickelten Gesamtansatz ist der Bestandteil (e) der Arbeitshypothese belegt, der in der Nutzung der Verfolgbarkeitsinformationen die Möglichkeit zur Beantwortung der Fragestellung sieht.

Für die prototypische Implementierung des Ansatzes dieser Arbeit wurden Standards des W3C (z.B. HTML, XHTML, RDFS, OWL, XSL und XSLT) und der OMG (z.B. UML, BPMN, MOF, QVT) zugrunde gelegt, die im Wesentlichen als Standards und Architekturen für E-Government-Anwendungen (SAGA) vom Beauftragten der Bundesregierung für Informationstechnik anerkannt sind. Darüber hinaus sind die eingesetzten Werkzeuge frei als Open Source verfügbar. Eine Einschränkung gibt es bei der Implementierung der Verfolgbarkeitsbeziehung von der graphischen Darstellung des Modellelements zu dessen Element in der Verfolgbarkeitsdokumentation. Für diese Implementierung gibt es keinen Standard, weshalb hier eine werkzeugspezifische Umsetzung (z. B. innerhalb des ansonsten standardisierten XMI-Formats) erfolgen musste. Trotz dieser geringfügigen Einschränkungen kann der Bestandteil (f) der Arbeitshypothese, der den Einsatz von offenen Standards und die Unabhängigkeit von proprietären Implementierungen fordert, als im Wesentlichen belegt angesehen werden.

Im Ergebnis dieser Arbeit konnten die Bestandteile (a) bis (f) der Arbeitshypothese belegt werden. Deshalb kann die Arbeitshypothese durch den konzipierten und prototypisch implementierten Ansatz auch als insgesamt belegt angesehen werden. Auf die Fragestellung, wie im Vorfeld der Softwareentwicklung die Verfolgbarkeit zu den Ursprüngen von Anforderungen in Rechtsvorschriften an E-Government-Anwendungen verbessert werden kann, wird deshalb folgende Antwort formuliert:

Mit den Techniken des Semantic Web ist es möglich, in Rechtsvorschriften die Ursprünge von Anforderungen, basierend auf der Ontologie des Verwaltungshandelns, mit einer definierten Semantik zu versehen, so dass sie durch Nutzung der Abbildungsregeln des ODM-Standards in ein PRM des Verwaltungshandeln transformiert werden können, das die Berücksichtigung verwaltungsspezifischer Aspekte ermöglicht und zusammen mit den eLoGo-Referenzmodellen für E-Government per QVT in ein CIM als Anforderungsspezifikation eines E-Government-Anwendungssystems überführt werden kann, das Anschluss an die MDA-basierte, modellgetriebene Softwareentwicklung und durch Nutzung der aufgezeichneten Verfolgbarkeitsinformationen die umfassende horizontale und in Ansätzen auch die vertikale Verfolgbarkeit, basierend auf offenen Standards, ermöglicht.

7.3 Erkenntnisfortschritt und Nutzen

Der durch diese Arbeit erzielte Erkenntnisfortschritt liegt zum einen in der Kombination der Techniken, Standards und Ergebnisse aus den Bereichen E-Government, Semantic Web, Ontologien und modellgetriebene Softwareentwicklung zur Konstruktion eines Ansatzes, der einen Beitrag zur bisher nur unzureichend unterstützten Verfolgbarkeit im Vorfeld der Anforderungsspezifikation liefert. Zum anderen wurden in den einzelnen Themengebieten weitere spezielle Erkenntnisse erzielt: Durch die im Rahmen dieser Arbeit entwickelte Ontologie des Verwaltungshandelns wurde es möglich, Texte von Rechtsvorschriften mit einer Annotation zu versehen, die Ursprünge von Anforderungen mit einer maschinenlesbaren Semantik versieht. Um die Ursprünge von Anforderungen in der MDA überhaupt berücksichtigen zu können, wurde im Rahmen dieser Arbeit eine Erweiterung der MDA zur Modellierung des Vorfelds der Anforderungsspezifikation entwickelt, das PRM. Für die Implementierung der Abbildungsregeln des ODM, um OWL-Individuen in das PRM transformieren zu können, war es anschließend notwendig, einen Ansatz zur Aufzeichnung der Verfolgbarkeitsinformationen zu entwickeln. Für die MDA-konforme Transformation des PRM in das CIM wurde im Rahmen dieser Arbeit weiterhin analog zum Plattformmodell einer PIM-PSM-Transformation mit den eLoGo-Referenzmodellen für E-Government ein Hilfsmodell eingeführt. Das Hilfsmodell dient auf neuartige Weise zur Überbrückung der semantischen Lücke zwischen beiden Modellen. Die Kombination der isoliert entwickelten Elemente in einem Gesamtansatz zur Verfolgbarkeit trägt abschließend dazu bei, dass im Vorfeld der Anforderungsspezifikation die horizontale vorwärts- und rückwärtsgerichtete Verfolgbarkeit umfassend und die vertikale Verfolgbarkeit in Ansätzen möglich wird.

Die so erzielten Erkenntnisse erlauben bei der Anwendung des Ansatzes dieser Arbeit, Ursprünge von Anforderungen in Rechtsvorschriften durchgängig bis zu Elementen in einer Anforderungsspezifikation zu verfolgen. Daran können sich etablierte Methoden, Werkzeuge und Techniken der Verfolgbarkeit im Nachgang der Anforderungsspezifikation anschließen. Für die Praxis wird es möglich, Änderungen an annotierten Passagen einer Rechtsvorschrift bis in die Softwareentwicklung zu verfolgen. Dadurch können Auswirkungen von Gesetzesänderungen analysiert werden. Nach Verabschiedung einer solchen Änderung können gezielt die betroffenen Anforderungen und die aus der Anforderung resultierenden Bestandteile der E-Government-Anwendung identifiziert werden. Aber auch für den Prozess der Gesetzgebung kann die Analyse relevant sein. Stehen beispielsweise mehrere gleichwertige Gestaltungsoptionen zur Auswahl, können die Auswirkungen jeder Option analysiert und der Aufwand ihrer Umsetzung in E-Government-Anwendungen als Auswahlkriterium berücksichtigt werden. Die Softwareentwicklung profitiert weiterhin, wenn beispielsweise Anforderungen auf ihre Ursprünge in Rechtsvorschriften zurückgeführt werden müssen. Dies kann der Fall sein, wenn Anforderungen zu priorisieren sind oder wenn die Rechtmäßigkeit des durch die E-Government-Anwendung unterstützten Verwaltungshandelns nachzuweisen ist.

7.4 Rahmenbedingungen und Perspektiven

Der im Rahmen dieser Arbeit entwickelte Ansatz basierte nicht nur auf der Kombination der Elemente unterschiedlicher Anwendungsbereiche. In seinem Rahmen wurden auch auf einander aufbauende Arbeitsergebnisse mit den Standards und Techniken des jeweiligen Anwendungsbereichs erstellt. Zwischen diesen Arbeitsergebnissen ergeben sich daher inhaltliche Abhängigkeiten. Beispielsweise muss die Ontologie des Verwaltungshandelns genau solche Konzepte umfassen, die nach ihrer Ergänzung und Verfeinerung im Rahmen der PRM-Modellierung mit den Elementen der zur Transformation in das CIM eingesetzten Referenzmodelle durch eine geeignete Implementierung der QVT-Transformation zusammengefügt werden können. Für die Umsetzung des Ansatzes dieser Arbeit in einem anderen oder erweiterten Kontext müssen diese Abhängigkeiten zwischen Ontologie, PRM-Modellierung, Referenzmodellen und QVT-Implementierung ebenfalls berücksichtigt werden. Dabei ist insbesondere auf eine geeignete Strategie und Umsetzung der QVT-Transformation zu achten, um aussagekräftige Verfolgbarkeitsinformationen bei Durchführung der QVT-Transformation aufzeichnen zu können.

Für die Zukunft ist zu erwarten, dass in der MDA-basierten Softwareentwicklung die Nutzung der während QVT-Transformationen aufgezeichneten Informationen auch für die Verbesserung der Verfolgbarkeit im Nachgang der Anforderungsspezifikation eingesetzt wird. Werden dabei ähnliche Erfahrungen mit Beschränkungen des durch die QVT-Spezifikation definierten Verfahrens gesammelt, wie im Rahmen dieser Arbeit (z.B. fehlende Möglichkeit, Abhängigkeiten in Transformationsdiagrammen darzustellen, fehlende Zuordnung der Attribute einer Verfolgbarkeitsklasse zum Quell- oder Zielmodell, Überschreiben der zuletzt aufgezeichneten Verfolgbarkeitsinformation bei Transformation des nächsten Modells), sind Änderungen des QVT-Standards oder seiner Implementierung in konkreten Werkzeugen zu erwarten. Grundsätzlich ist die im Rahmen dieser Arbeit gewählte Werkzeugunterstützung stabil und ermöglicht eine produktive Softwareentwicklung. Werden die Werkzeuge allerdings außerhalb des Bereichs eingesetzt, in dem ihre Anwendung üblicherweise erfolgt, zeigen sich Instabilitäten und Beschränkungen (z.B. bei Objektdiagrammen in TOPCASED, durch Importprobleme modularer QVT-Transformationen in medini QVT oder durch ungenügende Unterstützung der UML Profil-Application in Ecore). Aufgrund des Open Source-Charakters dieser Werkzeuge erscheint es prinzipiell möglich, diese Probleme selbst zu lösen. Eine aktive Community oder ein Produktsupport der Herstellerfirma ist bei der Lösung dieser Probleme hilfreich, setzt aber eigenständige Prioritäten. Die zunehmende Stabilität der Werkzeuge ist für die Zukunft aber grundsätzlich auch in Randbereichen ihres Einsatzspektrums zu erwarten.

Durch diese Arbeit wurde für das Vorfeld der Softwareentwicklung ein Ansatz konzipiert und prototypisch implementiert, mit dem die Verfolgbarkeit der Ursprünge von Anforderungen in Rechtsvorschriften an E-Government-Anwendungen verbessert werden kann. Für die öffentliche Verwaltung stellen sich hierdurch neue Möglichkeiten aber auch neue Grenzen der Softwareentwicklung für E-Government-Anwendungen dar. Heute, mehr als 10 Jahre nach dem wegweisenden E-Government-Memorandum, wird weiterhin an der „umfassende[n] Gestaltung der Prozesse und Ressourcen der Verwaltungsarbeit im Sinne eines Verwaltungs-Engineering unter weitestgehender Nutzung der Informationstechnik“[398] gearbeitet.[399] Durch die bisherige Bearbeitung wurde erkannt, dass einerseits die systematische Gestaltung im Verwaltungs-Engineering in besonderem Maße „Wissen über die Praxis und die theoretischen Grundlagen des arbeitenden Staates [...], ja vom gesamten öffentlichen Handeln“[400] erfordert. Andererseits bedarf es geeigneter Vorgehensmodelle und Verfahren für die systematische Systementwicklung[401], die auch verwaltungsspezifische Nutzungsmöglichkeiten, wie beispielsweise eine den Gesetzgebungsprozess „von Beginn an

[398] [GI&VDE, 2000], S. 6.
[399] Vgl. [Lenk, 2011a], S. 17.
[400] [Lenk, 2011a], S. 17.
[401] Vgl. [Lenk, 2011a], S. 17.

begleitende Informationstechnik-Folgenabschätzung und Validierung von Normen“[402] ermöglichen. Der im Rahmen dieser Arbeit entwickelte Ansatz basiert auf der Formalisierung von Wissen über das Verwaltungshandeln, berücksichtigt verwaltungsspezifische Aspekte und ermöglicht die vorausschauende Analyse der Auswirkungen von Gesetzesänderungen. Er ist daher prinzipiell geeignet, zur Beantwortung unmittelbar anschließender und weiterführender Fragestellungen beizutragen, die das „technisch ermöglichte sozio-technische Handlungssystem der Verwaltung“[403] ins Blickfeld rücken und der „Transformation des öffentlichen Sektors und dem Entstehen der technisch-organisatorischen Infrastruktur des Verwaltungshandelns Schwung zu verleihen.“[404]

[402] [Wulff, 2010], S. 43.
[403] [Lenk, 2010], S. 7.
[404] [Lenk, 2011a], S. 20.

Anhang

A Ontologie des Verwaltungshandelns in OWL

Die Dokumentation der Ontologie der Rechtsanwendung im Verwaltungshandeln erfolgt in diesem Abschnitt vollständig im OWL-Format, das aus dem Werkzeug protegé mit der zugehörigen OWL-API[405] in Version 2.2.1.1138 exportiert wurde (Listing 47).

```
(1)   <?xml version="1.0"?>
(2)
(3)
(4)   <!DOCTYPE rdf:RDF [
(5)       <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
(6)       <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
(7)       <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
(8)       <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
(9)       <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
(10)      <!ENTITY govobjects-ontology
      "http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#" >
(11)  ]>
(12)
(13)
(14)  <rdf:RDF xmlns="http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#"
(15)       xml:base="http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl"
(16)       xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(17)       xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(18)       xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(19)       xmlns:owl="http://www.w3.org/2002/07/owl#"
(20)       xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(21)       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
(22)      <owl:Ontology rdf:about="">
(23)          <rdfs:label rdf:datatype="&xsd;string"
(24)              >Verwaltungshandeln</rdfs:label>
(25)          <rdfs:comment rdf:datatype="&xsd;string"
(26)              >Ontologie des Verwaltungshandelns f&#252;r die Verwendung im
      GovObjects-Ansatz.</rdfs:comment>
(27)          <owl:imports rdf:resource=""/>
(28)      </owl:Ontology>
(29)
(30)
(31)
(32)      <!--
(33)      ///////////////////////////////////////////////////////////////////////
(34)      //
(35)      // Object Properties
(36)      //
(37)     ///////////////////////////////////////////////////////////////////////
(38)       -->
(39)
(40)
(41)
(42)
(43)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#abstrahiertLebenssachverhalt -->
(44)
(45)      <owl:ObjectProperty rdf:about="#abstrahiertLebenssachverhalt">
(46)          <rdfs:label rdf:datatype="&xsd;string"
(47)              >abtrahiertLebenssachverhalt</rdfs:label>
(48)          <rdfs:comment rdf:datatype="&xsd;string"
(49)              >Repr&#228;sentiert die Beziehung, die ein Sachverhalt zum
      Lebenssachverhalt hat, aus dem er durch Abstraktion der
      tatbestandsrelevanten Eigenschaften hervorgegangen ist.</rdfs:comment>
(50)          <rdfs:range rdf:resource="#Lebenssachverhalt"/>
(51)          <rdfs:domain rdf:resource="#Sachverhaltsermittlung"/>
(52)      </owl:ObjectProperty>
```

[405] Die Homepage des Projektes „OWL-API“ ist erreichbar unter: http://owlapi.sourceforge.net, letzter Aufruf: 21.08.2011.

```
(53)
(54)
(55)
(56)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#ausgangsVerwaltungsprodukt -->
(57)
(58)      <owl:ObjectProperty rdf:about="#ausgangsVerwaltungsprodukt">
(59)          <rdfs:label
(60)              >erbrachteVerwaltungsleistung</rdfs:label>
(61)          <rdfs:comment rdf:datatype="&xsd;string"
(62)              >Repr&#228;sentiert den im Kontext eines Leistungszusammenhangs
      erbrachten Typ von Verwaltungsleitung (VwRechtsfolge).</rdfs:comment>
(63)          <rdfs:domain rdf:resource="#VwLeistungszusammenhang"/>
(64)          <rdfs:range rdf:resource="#VwRechtsfolge"/>
(65)      </owl:ObjectProperty>
(66)
(67)
(68)
(69)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#beantragteLeistung -->
(70)
(71)      <owl:ObjectProperty rdf:about="#beantragteLeistung">
(72)          <rdfs:label
(73)              >beantragteLeistung</rdfs:label>
(74)          <rdfs:comment rdf:datatype="&xsd;string"
(75)              >Repr&#228;sentiert die Beziehung eines Antrag zu der durch den
      Antrag zu erbringenden/erbrachten Verwaltungsleistung.</rdfs:comment>
(76)          <rdfs:range rdf:resource="#VwAntrag"/>
(77)          <rdfs:domain rdf:resource="#VwLeistung"/>
(78)          <owl:inverseOf rdf:resource="#zugehoerigerAntrag"/>
(79)      </owl:ObjectProperty>
(80)
(81)
(82)
(83)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#beschreibtRechtsfolge -->
(84)
(85)      <owl:ObjectProperty rdf:about="#beschreibtRechtsfolge">
(86)          <rdfs:label rdf:datatype="&xsd;string"
(87)              >beschreibtRechtsfolge</rdfs:label>
(88)          <rdfs:comment rdf:datatype="&xsd;string"
(89)              >Repr&#228;sentiert den Zusammenhang zwischen einem
      vollst&#228;ndigen Rechtssatz und der durch ihn beschriebenen
      Rechtsfolge.</rdfs:comment>
(90)          <rdfs:range rdf:resource="#Rechtsfolge"/>
(91)          <rdfs:domain rdf:resource="#VollstaendigerRechtssatz"/>
(92)      </owl:ObjectProperty>
(93)
(94)
(95)
(96)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#beschreibtRechtsfolgeUnvollstaendig -->
(97)
(98)      <owl:ObjectProperty rdf:about="#beschreibtRechtsfolgeUnvollstaendig">
(99)          <rdfs:label rdf:datatype="&xsd;string"
(100)             >beschreibtRechtsfolgeUnvollstaendig</rdfs:label>
(101)         <rdfs:comment rdf:datatype="&xsd;string"
(102)             >Repr&#228;sentiert den Zusammenhang zwischen einem
      vollst&#228;ndigen Rechtssatz und der durch ihn unvollst&#228;ndig
      beschriebenen Rechtsfolge.</rdfs:comment>
(103)         <rdfs:range rdf:resource="#Rechtsfolge"/>
(104)         <rdfs:domain rdf:resource="#UnvollstaendigerRechtssatz"/>
(105)     </owl:ObjectProperty>
(106)
(107)
(108)
(109)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#beschreibtTatbestand -->
(110)
(111)     <owl:ObjectProperty rdf:about="#beschreibtTatbestand">
(112)         <rdfs:label rdf:datatype="&xsd;string"
(113)             >beschreibtTatbestand</rdfs:label>
(114)         <rdfs:comment rdf:datatype="&xsd;string"
(115)             >Repr&#228;sentiert den Zusammenhang zwischen einem Rechtssatz
      und dem durch ihn beschriebenen Tatbestand.</rdfs:comment>
```

```
(116)         <rdfs:range rdf:resource="#Tatbestand"/>
(117)         <rdfs:domain rdf:resource="#VollstaendigerRechtssatz"/>
(118)     </owl:ObjectProperty>
(119)
(120)
(121)
(122)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#beschreibtTatbestandUnvollstaendig -->
(123)
(124)     <owl:ObjectProperty rdf:about="#beschreibtTatbestandUnvollstaendig">
(125)         <rdfs:label rdf:datatype="&xsd;string"
(126)             >repr&#228;sentiertTatbestandUnvollst&#228;ndig</rdfs:label>
(127)         <rdfs:comment rdf:datatype="&xsd;string"
(128)             >Repr&#228;sentiert den Zusammenhang zwischen einem Rechtssatz
      und dem durch ihn unvollst&#228;ndig beschriebenen
      Tatbestand.</rdfs:comment>
(129)         <rdfs:range rdf:resource="#Tatbestand"/>
(130)         <rdfs:domain rdf:resource="#UnvollstaendigerRechtssatz"/>
(131)     </owl:ObjectProperty>
(132)
(133)
(134)
(135)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#bestehtAusRechtsnormbestandteil -->
(136)
(137)     <owl:ObjectProperty rdf:about="#bestehtAusRechtsnormbestandteil">
(138)         <rdfs:label rdf:datatype="&xsd;string"
(139)             >bestehtAusRechtsNormbestandteil</rdfs:label>
(140)         <rdfs:comment rdf:datatype="&xsd;string"
(141)             >Repr&#228;sentiert die Beziehung einer Rechtsnorm zu ihren
      Bestandteilen.</rdfs:comment>
(142)         <rdfs:domain rdf:resource="#RechtsNorm"/>
(143)         <rdfs:range rdf:resource="#RechtsNormbestandteil"/>
(144)     </owl:ObjectProperty>
(145)
(146)
(147)
(148)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#bestehtAusTextbestandteil -->
(149)
(150)     <owl:ObjectProperty rdf:about="#bestehtAusTextbestandteil">
(151)         <rdfs:label rdf:datatype="&xsd;string"
(152)             >bestehtAusTextbestandteil</rdfs:label>
(153)         <rdfs:comment rdf:datatype="&xsd;string"
(154)             >Beziehung eines Textes zu seinen allgemeinen
      Textbestandteilen</rdfs:comment>
(155)         <rdfs:range rdf:resource="#AllgemeinerTextbestandteil"/>
(156)         <rdfs:domain rdf:resource="#Text"/>
(157)     </owl:ObjectProperty>
(158)
(159)
(160)
(161)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#bestimmtRechtsfolge -->
(162)
(163)     <owl:ObjectProperty rdf:about="#bestimmtRechtsfolge">
(164)         <rdfs:label rdf:datatype="&xsd;string"
(165)             >bestimmtRechtsfolge</rdfs:label>
(166)         <rdfs:comment rdf:datatype="&xsd;string"
(167)             >Repr&#228;sentiert die Beziehung der Rechtsfolgenbestimmung zur
      Rechtsfolge.</rdfs:comment>
(168)         <rdfs:range rdf:resource="#Rechtsfolge"/>
(169)         <rdfs:domain rdf:resource="#Rechtsfolgenbestimmung"/>
(170)     </owl:ObjectProperty>
(171)
(172)
(173)
(174)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#beziehtSichAufLeistung -->
(175)
(176)     <owl:ObjectProperty rdf:about="#beziehtSichAufLeistung">
(177)         <rdfs:label
(178)             >beziehtSichAufLeistung</rdfs:label>
(179)         <rdfs:comment rdf:datatype="&xsd;string"
```

```
(180)             >Repr&#228;sentiert eine Beziehung zwischen einem Hinweis/einer
      Information und einer Verwaltungsleistung. Sie dr&#252;ckt f&#252;r einen
      Hinweis/eine Information aus, dass die Verwaltungsleistung hilfreich,
      n&#252;tzlich oder erforderlich ist.</rdfs:comment>
(181)         <rdfs:domain rdf:resource="#VwHinweisInformation"/>
(182)         <rdfs:range rdf:resource="#VwLeistung"/>
(183)     </owl:ObjectProperty>
(184)
(185)
(186)
(187)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#erbringtLeistung -->
(188)
(189)     <owl:ObjectProperty rdf:about="#erbringtLeistung">
(190)         <rdfs:label rdf:datatype="&xsd;string"
(191)             >erbringtLeistung</rdfs:label>
(192)         <rdfs:comment rdf:datatype="&xsd;string"
(193)             >Pr&#228;sentiert die Beziehung des Konzepts Leistungserbringung
      zum Konzepts Verwaltungsleistung, das die zu erbringende Leistung
      repr&#228;sentiert.</rdfs:comment>
(194)         <rdfs:range rdf:resource="#VwLeistung"/>
(195)         <rdfs:domain rdf:resource="#VwLeistungserbringung"/>
(196)     </owl:ObjectProperty>
(197)
(198)
(199)
(200)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#erlaeutertRechtssatz -->
(201)
(202)     <owl:ObjectProperty rdf:about="#erlaeutertRechtssatz">
(203)         <rdfs:label
(204)             >erlaeutertRechtssatz</rdfs:label>
(205)         <rdfs:comment rdf:datatype="&xsd;string"
(206)             >Repr&#228;sentiert die Beziehung eines unvollst&#228;ndigen
      Rechtssatzes zu dem durch ihn erl&#228;uterten Rechtssatz.</rdfs:comment>
(207)         <rdfs:range rdf:resource="#Rechtssatz"/>
(208)         <rdfs:domain rdf:resource="#UnvollstaendigerRechtssatz"/>
(209)     </owl:ObjectProperty>
(210)
(211)
(212)
(213)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#ermitteltSachverhalt -->
(214)
(215)     <owl:ObjectProperty rdf:about="#ermitteltSachverhalt">
(216)         <rdfs:label rdf:datatype="&xsd;string"
(217)             >ermitteltSachverhalt</rdfs:label>
(218)         <rdfs:comment rdf:datatype="&xsd;string"
(219)             >Repr&#228;sentiert die Beziehung der Sachverhaltsermittlung zu
      dem in ihrem Rahmen ermittelten Sachverhalt.</rdfs:comment>
(220)         <rdfs:range rdf:resource="#Sachverhalt"/>
(221)         <rdfs:domain rdf:resource="#Sachverhaltsermittlung"/>
(222)     </owl:ObjectProperty>
(223)
(224)
(225)
(226)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#fuehrtAus -->
(227)
(228)     <owl:ObjectProperty rdf:about="#fuehrtAus">
(229)         <rdfs:label>fuehrtAus</rdfs:label>
(230)         <rdfs:comment rdf:datatype="&xsd;string"
(231)             >Repr&#228;sentiert die Beziehung eines Ausf&#252;hrenden zur
      Leistungserbringung.</rdfs:comment>
(232)         <rdfs:domain rdf:resource="#VwAusfuehrender"/>
(233)         <rdfs:range rdf:resource="#VwLeistungserbringung"/>
(234)     </owl:ObjectProperty>
(235)
(236)
(237)
(238)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#fuehrtDurch -->
(239)
(240)     <owl:ObjectProperty rdf:about="#fuehrtDurch">
(241)         <rdfs:label rdf:datatype="&xsd;string">fuehrtDurch</rdfs:label>
```

```
(242)         <rdfs:comment rdf:datatype="&xsd;string"
(243)             >Repr&#228;sentiert die Beziehung eines Nachfragers zur
      Informationsbeschaffung und Vorbereitung.</rdfs:comment>
(244)         <rdfs:domain rdf:resource="#VwNachfrager"/>
(245)         <rdfs:range rdf:resource="#VwInformationsbeschaffung"/>
(246)         <rdfs:range rdf:resource="#VwVorbereitung"/>
(247)     </owl:ObjectProperty>
(248)
(249)
(250)
(251)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#fuehrtDurchRechtshandlung -->
(252)
(253)     <owl:ObjectProperty rdf:about="#fuehrtDurchRechtshandlung">
(254)         <rdfs:label>wendetAn</rdfs:label>
(255)         <rdfs:comment rdf:datatype="&xsd;string"
(256)             >Repr&#228;sentiert die Beziehung eines Rechtsanwenders zu den
      von ihm durchgef&#252;hrten Rechtshandlungen.</rdfs:comment>
(257)         <rdfs:domain rdf:resource="#Anwender"/>
(258)         <rdfs:range rdf:resource="#Rechtsfindung"/>
(259)         <rdfs:range rdf:resource="#Subsumtion"/>
(260)         <rdfs:range rdf:resource="#Sachverhaltsermittlung"/>
(261)         <rdfs:range rdf:resource="#Rechtsfolgenbestimmung"/>
(262)         <rdfs:range rdf:resource="#Mitteilung"/>
(263)         <rdfs:range rdf:resource="#Einzelfallfestsetzung"/>
(264)     </owl:ObjectProperty>
(265)
(266)
(267)
(268)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#fuehrtDurchVerwaltungshandlung -->
(269)
(270)     <owl:ObjectProperty rdf:about="#fuehrtDurchVerwaltungshandlung">
(271)         <rdfs:label
(272)             >fuehrtDurchVerwaltungshandlung</rdfs:label>
(273)         <rdfs:comment rdf:datatype="&xsd;string"
(274)             >Repr&#228;sentiert die Beziehung des Anwenders im
      Verwaltungshandeln zu den Handlungen, die f&#252;r das Verwaltungshandeln
      spezifisch sind.</rdfs:comment>
(275)         <rdfs:domain rdf:resource="#VwAnwender"/>
(276)         <rdfs:range rdf:resource="#VwVerfahrenseinleitung"/>
(277)         <rdfs:range rdf:resource="#VwNachbearbeitung"/>
(278)     </owl:ObjectProperty>
(279)
(280)
(281)
(282)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#gehoertZuAntragsteller -->
(283)
(284)     <owl:ObjectProperty rdf:about="#gehoertZuAntragsteller">
(285)         <rdfs:label
(286)             >gehoertZuAntragsteller</rdfs:label>
(287)         <rdfs:comment rdf:datatype="&xsd;string"
(288)             >Repr&#228;sentiert den Zusammenhang zwischen einer Lebenlage
      und dem Antragsteller.</rdfs:comment>
(289)         <rdfs:range rdf:resource="#VwAntragsteller"/>
(290)         <rdfs:domain rdf:resource="#VwLebenslage"/>
(291)         <owl:inverseOf rdf:resource="#hatLebenslage"/>
(292)     </owl:ObjectProperty>
(293)
(294)
(295)
(296)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#gehoertZuMassnahme -->
(297)
(298)     <owl:ObjectProperty rdf:about="#gehoertZuMassnahme">
(299)         <rdfs:label
(300)             >gehoertZuMassnahme</rdfs:label>
(301)         <rdfs:comment rdf:datatype="&xsd;string"
(302)             >Repr&#228;sentiert den Zusammenhang zwischen einer
      Ma&#223;nahmenEigenschaft und der Ma&#223;nahme, zu der die Eigenschaft
      ge&#246;hrt.</rdfs:comment>
(303)         <rdfs:domain rdf:resource="#MassnahmenEigenschaft"/>
(304)         <rdfs:range rdf:resource="#Massnahme"/>
(305)     </owl:ObjectProperty>
```

```
(306)
(307)
(308)
(309)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#hatAntragsteller -->
(310)
(311)     <owl:ObjectProperty rdf:about="#hatAntragsteller">
(312)         <rdfs:label rdf:datatype="&xsd;string"
(313)             >hatAntragsterller</rdfs:label>
(314)         <rdfs:comment rdf:datatype="&xsd;string"
(315)             >Repr&#228;sentiert die Beziehung eines Antrags zu seinem
      Antragsteller.</rdfs:comment>
(316)         <rdfs:domain rdf:resource="#VwAntrag"/>
(317)         <rdfs:range rdf:resource="#VwAntragsteller"/>
(318)         <owl:inverseOf rdf:resource="#stelltAntrag"/>
(319)     </owl:ObjectProperty>
(320)
(321)
(322)
(323)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#hatLebenslage -->
(324)
(325)     <owl:ObjectProperty rdf:about="#hatLebenslage">
(326)         <rdfs:label>hatLebenslage</rdfs:label>
(327)         <rdfs:comment rdf:datatype="&xsd;string"
(328)             >Repr&#228;sentiert die Beziehung eines Antragstellers zu seiner
      Lebenslage.</rdfs:comment>
(329)         <rdfs:domain rdf:resource="#VwAntragsteller"/>
(330)         <rdfs:range rdf:resource="#VwLebenslage"/>
(331)     </owl:ObjectProperty>
(332)
(333)
(334)
(335)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#hatMassnahmenEigenschaft -->
(336)
(337)     <owl:ObjectProperty rdf:about="#hatMassnahmenEigenschaft">
(338)         <rdfs:label rdf:datatype="&xsd;string"
(339)             >hatMassnahmenEigenschaft</rdfs:label>
(340)         <rdfs:comment rdf:datatype="&xsd;string"
(341)             >Repr&#228;sentiert die Beziehung einer Ma&#223;nahme zu den
      zugeh&#246;rigen Ma&#223;nahmenEigenschaften.</rdfs:comment>
(342)         <rdfs:domain rdf:resource="#Massnahme"/>
(343)         <rdfs:range rdf:resource="#MassnahmenEigenschaft"/>
(344)         <owl:inverseOf rdf:resource="#gehoertZuMassnahme"/>
(345)     </owl:ObjectProperty>
(346)
(347)
(348)
(349)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#hatRechtsfolgenMerkmal -->
(350)
(351)     <owl:ObjectProperty rdf:about="#hatRechtsfolgenMerkmal">
(352)         <rdfs:label rdf:datatype="&xsd;string"
(353)             >hatRechtsfolgenMerkmal</rdfs:label>
(354)         <rdfs:comment rdf:datatype="&xsd;string"
(355)             >Repr&#228;sentiert den Zusammenhang zwischen einer Rechtsfolge
      und ihren Merkmalen.</rdfs:comment>
(356)         <rdfs:domain rdf:resource="#Rechtsfolge"/>
(357)         <rdfs:range rdf:resource="#RechtsfolgenMerkmal"/>
(358)         <owl:inverseOf rdf:resource="#istMerkmalVonRechtsfolge"/>
(359)     </owl:ObjectProperty>
(360)
(361)
(362)
(363)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#hatSachverhaltsEigenschaft -->
(364)
(365)     <owl:ObjectProperty rdf:about="#hatSachverhaltsEigenschaft">
(366)         <rdfs:label rdf:datatype="&xsd;string"
(367)             >hatSachverhaltsEigenschaft</rdfs:label>
(368)         <rdfs:comment rdf:datatype="&xsd;string"
(369)             >Repr&#228;sentiert den Zusammenhang zwischen einem Sachverhalt
      und seinen Eigenschaften.</rdfs:comment>
(370)         <rdfs:domain rdf:resource="#Sachverhalt"/>
```

```
(371)         <rdfs:range rdf:resource="#SachverhaltsEigenschaft"/>
(372)         <owl:inverseOf rdf:resource="#istEigenschaftVonSachverhalt"/>
(373)     </owl:ObjectProperty>
(374)
(375)
(376)
(377)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#hatTatbestandsBedingung -->
(378)
(379)     <owl:ObjectProperty rdf:about="#hatTatbestandsBedingung">
(380)         <rdfs:label rdf:datatype="&xsd;string"
(381)             >hatTatbestandsBedingung</rdfs:label>
(382)         <rdfs:comment rdf:datatype="&xsd;string"
(383)             >Repr&#228;sentiert die Beziehung zwischen einem Tatbestand und
      den durch ihn formulierten Bedingungen.</rdfs:comment>
(384)         <rdfs:domain rdf:resource="#Tatbestand"/>
(385)         <rdfs:range rdf:resource="#Tatbestandsbedingung"/>
(386)         <owl:inverseOf rdf:resource="#istBedingungVonTatbestand"/>
(387)     </owl:ObjectProperty>
(388)
(389)
(390)
(391)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#impliziertRechtsfolge -->
(392)
(393)     <owl:ObjectProperty rdf:about="#impliziertRechtsfolge">
(394)         <rdfs:label rdf:datatype="&xsd;string"
(395)             >impliziertRechtsfolge</rdfs:label>
(396)         <rdfs:comment rdf:datatype="&xsd;string"
(397)             >Repr&#228;sentiert die Beziehung bei Vorliegen eines
      Tatbestandes zur Rechtsfolge.</rdfs:comment>
(398)         <rdfs:range rdf:resource="#Rechtsfolge"/>
(399)         <rdfs:domain rdf:resource="#Tatbestand"/>
(400)     </owl:ObjectProperty>
(401)
(402)
(403)
(404)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istAbhaengigVonBeurteilungsspielraum -->
(405)
(406)     <owl:ObjectProperty rdf:about="#istAbhaengigVonBeurteilungsspielraum">
(407)         <rdfs:label
(408)             >istAbhaengigVonBeurteilungsspielraum</rdfs:label>
(409)         <rdfs:comment rdf:datatype="&xsd;string"
(410)             >Repr&#228;sentiert die Beziehung einer Voraussetzungsbedingung
      zu dem aus ihr resultierenden Beurteilungsspielraum.</rdfs:comment>
(411)         <rdfs:range rdf:resource="#VwBeurteilungsspielraum"/>
(412)         <rdfs:domain rdf:resource="#VwVoraussetzungsBedingung"/>
(413)     </owl:ObjectProperty>
(414)
(415)
(416)
(417)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istAbhaengigVonErmessen -->
(418)
(419)     <owl:ObjectProperty rdf:about="#istAbhaengigVonErmessen">
(420)         <rdfs:label
(421)             >istAbhaengigVonErmessen</rdfs:label>
(422)         <rdfs:comment rdf:datatype="&xsd;string"
(423)             >Repr&#228;sentiert die Beziehung eines RechtsfolgenMerkmals zu
      dem aus ihm resultierenden Ermessen.</rdfs:comment>
(424)         <rdfs:range rdf:resource="#VwErmessen"/>
(425)         <rdfs:domain rdf:resource="#VwRechtsfolgenMerkmal"/>
(426)     </owl:ObjectProperty>
(427)
(428)
(429)
(430)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istAdressatVon -->
(431)
(432)     <owl:ObjectProperty rdf:about="#istAdressatVon">
(433)         <rdfs:label>istAdressatVon</rdfs:label>
(434)         <rdfs:comment rdf:datatype="&xsd;string"
(435)             >Repr&#228;sentiert die Beziehung eines Adressaten zur
      Rechtsnorm, die sich an ihn richtet.</rdfs:comment>
```

```
(436)         <rdfs:domain rdf:resource="#Adressat"/>
(437)         <rdfs:range rdf:resource="#RechtsNorm"/>
(438)     </owl:ObjectProperty>
(439)
(440)
(441)
(442)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istBedingungVonTatbestand -->
(443)
(444)     <owl:ObjectProperty rdf:about="#istBedingungVonTatbestand">
(445)         <rdfs:label
(446)             >istBedingungVonTatbestand</rdfs:label>
(447)         <rdfs:comment rdf:datatype="&xsd;string"
(448)             >Repr&#228;sentiert die Beziehung einer Bedingung zu dem
      Tatbestand, zu dem sie geh&#246;rt.</rdfs:comment>
(449)         <rdfs:range rdf:resource="#Tatbestand"/>
(450)         <rdfs:domain rdf:resource="#Tatbestandsbedingung"/>
(451)     </owl:ObjectProperty>
(452)
(453)
(454)
(455)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istBeteiligtAn -->
(456)
(457)     <owl:ObjectProperty rdf:about="#istBeteiligtAn">
(458)         <rdfs:label>istBeteiligtAn</rdfs:label>
(459)         <rdfs:comment rdf:datatype="&xsd;string"
(460)             >Repr&#228;sentiert die Beteiligung eines Akteurs am
      Verwaltungsverfahren.</rdfs:comment>
(461)         <rdfs:domain rdf:resource="#VwBeteiligterBetroffener"/>
(462)         <rdfs:range rdf:resource="#VwVerfahren"/>
(463)     </owl:ObjectProperty>
(464)
(465)
(466)
(467)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istEigenschaftVonSachverhalt -->
(468)
(469)     <owl:ObjectProperty rdf:about="#istEigenschaftVonSachverhalt">
(470)         <rdfs:label
(471)             >istEigenschaftVonSachverhalt</rdfs:label>
(472)         <rdfs:comment rdf:datatype="&xsd;string"
(473)             >Repr&#228;sentiert die Beziehung einer SachverhaltsEigenschaft
      zu dem Sachverhalt, zu dem sie geh&#246;rt.</rdfs:comment>
(474)         <rdfs:range rdf:resource="#Sachverhalt"/>
(475)         <rdfs:domain rdf:resource="#SachverhaltsEigenschaft"/>
(476)     </owl:ObjectProperty>
(477)
(478)
(479)
(480)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istExstrahiertAus -->
(481)
(482)     <owl:ObjectProperty rdf:about="#istExstrahiertAus">
(483)         <rdfs:label
(484)             >istExstrahiertAus</rdfs:label>
(485)         <rdfs:comment rdf:datatype="&xsd;string"
(486)             >Repr&#228;sentiert die Beziehung eines Sachverhalts zu dem
      Lebenssachverhalt, aus dem sie durch Abstaktion der tatbestandsrelevanten
      Eigenschaften hervorgegangen ist.</rdfs:comment>
(487)         <rdfs:range rdf:resource="#Lebenssachverhalt"/>
(488)         <rdfs:domain rdf:resource="#Sachverhalt"/>
(489)         <owl:inverseOf rdf:resource="#wirdUeberfuehrtIn"/>
(490)     </owl:ObjectProperty>
(491)
(492)
(493)
(494)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istMerkmalVonRechtsfolge -->
(495)
(496)     <owl:ObjectProperty rdf:about="#istMerkmalVonRechtsfolge">
(497)         <rdfs:label
(498)             >istMerkmalVonRechtsfolge</rdfs:label>
(499)         <rdfs:comment rdf:datatype="&xsd;string"
```

```
(500)             >Repr&#228;sentiert den Zusammenhang eines RechtsfolgenMerkmals
      zur Rechtsfolge, zu der es geh&#246;rt.</rdfs:comment>
(501)         <rdfs:range rdf:resource="#Rechtsfolge"/>
(502)         <rdfs:domain rdf:resource="#RechtsfolgenMerkmal"/>
(503)     </owl:ObjectProperty>
(504)
(505)
(506)
(507)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istVomTyp -->
(508)
(509)     <owl:ObjectProperty rdf:about="#istVomTyp">
(510)         <rdfs:label>istVomTyp</rdfs:label>
(511)         <rdfs:comment rdf:datatype="&xsd;string"
(512)             >Repr&#228;sentiert die Beziehung, die eine im konkreten
      Einzelfall erbrachte Verwaltungsleistung zur Rechtsfolge hat.</rdfs:comment>
(513)         <rdfs:domain rdf:resource="#VwLeistung"/>
(514)         <rdfs:range rdf:resource="#VwRechtsfolge"/>
(515)     </owl:ObjectProperty>
(516)
(517)
(518)
(519)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#konkretisiertRechtsKonzept -->
(520)
(521)     <owl:ObjectProperty rdf:about="#konkretisiertRechtsKonzept">
(522)         <rdfs:label
(523)             >konkretisiertRechtsKonzept</rdfs:label>
(524)         <rdfs:comment rdf:datatype="&xsd;string"
(525)             >Repr&#228;sentiert die Beziehung, die ein Verwaltungskonzept zu
      einem Rechtskonzeptes hat, das es bezogen auf das Verwaltungshandeln
      konkretisiert.</rdfs:comment>
(526)         <rdfs:range rdf:resource="#RechtsKonzept"/>
(527)         <rdfs:domain rdf:resource="#VerwaltungsKonzept"/>
(528)     </owl:ObjectProperty>
(529)
(530)
(531)
(532)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#konkretisiertTextKonzept -->
(533)
(534)     <owl:ObjectProperty rdf:about="#konkretisiertTextKonzept">
(535)         <rdfs:label
(536)             >konkretisiertTextKonzept</rdfs:label>
(537)         <rdfs:comment rdf:datatype="&xsd;string"
(538)             >Repr&#228;sentiert die Beziehung, die ein Rechtskonzept zu
      einem Textkonzept hat, das es bezogen auf die Rechtsanwendung
      konkretisiert.</rdfs:comment>
(539)         <rdfs:domain rdf:resource="#RechtsKonzept"/>
(540)         <rdfs:range rdf:resource="#TextKonzept"/>
(541)     </owl:ObjectProperty>
(542)
(543)
(544)
(545)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#konkretisiertZiel -->
(546)
(547)     <owl:ObjectProperty rdf:about="#konkretisiertZiel">
(548)         <rdfs:label
(549)             >konkretisiertZiel</rdfs:label>
(550)         <rdfs:comment rdf:datatype="&xsd;string"
(551)             >Repr&#228;sentiert die Beziehung, die ein Ziel zu einem anderen
      Ziel hat, wenn es dieses weitergehend konkretisiert.</rdfs:comment>
(552)         <rdfs:range rdf:resource="#VwZielZweck"/>
(553)         <rdfs:domain rdf:resource="#VwZielZweck"/>
(554)     </owl:ObjectProperty>
(555)
(556)
(557)
(558)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#liefertBeitragZuZielZweck -->
(559)
(560)     <owl:ObjectProperty rdf:about="#liefertBeitragZuZielZweck">
(561)         <rdfs:label rdf:datatype="&xsd;string"
(562)             >liefertBeitragZuZielZweck</rdfs:label>
```

```
(563)         <rdfs:comment rdf:datatype="&xsd;string"
(564)             >Repr&#228;sentiert die Beziehung einer Rechtsfolge und der aus
      ihr abgeleiteten konkreten Ma&#223;nahme zu Zielen und Zwecken des
      Verwaltungshandeln, um auszudr&#252;cken, dass von mehreren zul&#228;ssigen
      Rechtsfolgen  die Verwaltungs stets diejenige w&#228;hlt, die den
      gr&#246;&#223;ten Beitrag zur Zielerreichung leistet.</rdfs:comment>
(565)         <rdfs:domain rdf:resource="#VwRechtsfolge"/>
(566)         <rdfs:range rdf:resource="#VwZielZweck"/>
(567)         <owl:inverseOf rdf:resource="#wirdErreichtDuchRechtsfolge"/>
(568)     </owl:ObjectProperty>
(569)
(570)
(571)
(572)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#moeglicheRechtsfolge -->
(573)
(574)     <owl:ObjectProperty rdf:about="#moeglicheRechtsfolge">
(575)         <rdfs:label
(576)             >moeglicheRechtsfolge</rdfs:label>
(577)         <rdfs:comment rdf:datatype="&xsd;string"
(578)             >Repr&#228;sentiert eine Beziehung zwischen einem Hinweis/einer
      Information und einer Rechtsfolge. Sie dr&#252;ckt f&#252;r einen
      Hinweis/eine Information aus, dass die Rechtsfolge m&#246;glicherweise als
      Verwaltungsleistung erbracht werden kann.</rdfs:comment>
(579)         <rdfs:domain rdf:resource="#VwHinweisInformation"/>
(580)         <rdfs:range rdf:resource="#VwRechtsfolge"/>
(581)     </owl:ObjectProperty>
(582)
(583)
(584)
(585)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#notwendigeRechtsfolge -->
(586)
(587)     <owl:ObjectProperty rdf:about="#notwendigeRechtsfolge">
(588)         <rdfs:label
(589)             >notwendigeRechtsfolge</rdfs:label>
(590)         <rdfs:comment rdf:datatype="&xsd;string"
(591)             >Repr&#228;sentiert eine Beziehung zwischen einem Hinweis/einer
      Information und einer Rechtsfolge. Sie dr&#252;ckt f&#252;r einen
      Hinweis/eine Information aus, dass die Rechtsfolge als Verwaltungsleistung
      erbracht werden muss.</rdfs:comment>
(592)         <rdfs:domain rdf:resource="#VwHinweisInformation"/>
(593)         <rdfs:range rdf:resource="#VwRechtsfolge"/>
(594)     </owl:ObjectProperty>
(595)
(596)
(597)
(598)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#nutztBeurteilungsspielraum -->
(599)
(600)     <owl:ObjectProperty rdf:about="#nutztBeurteilungsspielraum">
(601)         <rdfs:label
(602)             >nutztBeurteilungsspielraum</rdfs:label>
(603)         <rdfs:comment rdf:datatype="&xsd;string"
(604)             >Repr&#228;sentiert die Beziehung der Verwaltungshandlung zur
      Nutzung von Beurteilungsspielraum zu einem
      Beurteilungsspielraum.</rdfs:comment>
(605)         <rdfs:range rdf:resource="#VwBeurteilungsspielraum"/>
(606)         <rdfs:domain rdf:resource="#VwNutzungVonBeurteilungsspielraum"/>
(607)     </owl:ObjectProperty>
(608)
(609)
(610)
(611)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#nutztErmessen -->
(612)
(613)     <owl:ObjectProperty rdf:about="#nutztErmessen">
(614)         <rdfs:label>nutztErmessen</rdfs:label>
(615)         <rdfs:comment rdf:datatype="&xsd;string"
(616)             >Repr&#228;sentiert die Beziehung der Verwaltungshandlung zur
      Nutzung des Ermessens zu einem Ermessen.</rdfs:comment>
(617)         <rdfs:range rdf:resource="#VwErmessen"/>
(618)         <rdfs:domain rdf:resource="#VwErmessensausuebung"/>
(619)     </owl:ObjectProperty>
(620)
```

```
(621)
(622)
(623)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#orientiertSichAnZielZweck -->
(624)
(625)     <owl:ObjectProperty rdf:about="#orientiertSichAnZielZweck">
(626)         <rdfs:label rdf:datatype="&xsd;string"
(627)             >orientiertSichAnZielZweck</rdfs:label>
(628)         <rdfs:comment rdf:datatype="&xsd;string"
(629)             >Repr&#228;sentiert die Beziehung, die sich ergibt, wenn im
      Rahmen der Mittelwahl von den abstrakten Rechtsfolgen diejenige gew&#228;hlt
      werden soll, die den gr&#246;&#223;ten Beitrag zur Zielerreichung
      liefert.</rdfs:comment>
(630)         <rdfs:domain rdf:resource="#VwMittelwahl"/>
(631)         <rdfs:range rdf:resource="#VwZielZweck"/>
(632)     </owl:ObjectProperty>
(633)
(634)
(635)
(636)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#prueftVorliegenTatbestand -->
(637)
(638)     <owl:ObjectProperty rdf:about="#prueftVorliegenTatbestand">
(639)         <rdfs:label rdf:datatype="&xsd;string"
(640)             >prueftVorliegenTatbestand</rdfs:label>
(641)         <rdfs:comment rdf:datatype="&xsd;string"
(642)             >Repr&#228;sentiert die Verbindung, die im Rahmen der
      Durchf&#252;hrung einer Subsumtion zum anzuwendenden Tatbestand
      exisitert.</rdfs:comment>
(643)         <rdfs:domain rdf:resource="#Subsumtion"/>
(644)         <rdfs:range rdf:resource="#Tatbestand"/>
(645)     </owl:ObjectProperty>
(646)
(647)
(648)
(649)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#resultiertInMassnahme -->
(650)
(651)     <owl:ObjectProperty rdf:about="#resultiertInMassnahme">
(652)         <rdfs:label rdf:datatype="&xsd;string"
(653)             >resultiertInMassnahme</rdfs:label>
(654)         <rdfs:comment rdf:datatype="&xsd;string"
(655)             >Repr&#228;sentiert die Beziehung einer Rechtfolge zu der
      Ma&#223;nahme, die durch Konkretisierung bezogen auf den vorliegenden
      Einzelfall entsteht.</rdfs:comment>
(656)         <rdfs:range rdf:resource="#Massnahme"/>
(657)         <rdfs:domain rdf:resource="#Rechtsfolge"/>
(658)     </owl:ObjectProperty>
(659)
(660)
(661)
(662)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#schaenktEinRechtssatz -->
(663)
(664)     <owl:ObjectProperty rdf:about="#schaenktEinRechtssatz">
(665)         <rdfs:label
(666)             >schaenktEinRechtssatz</rdfs:label>
(667)         <rdfs:comment rdf:datatype="&xsd;string"
(668)             >Repr&#228;sentiert die Beziehung eines unvollst&#228;ndigen
      Rechtssatzes zu einem Rechtssatz, auf den er einschr&#228;nkend
      wirkt.</rdfs:comment>
(669)         <rdfs:range rdf:resource="#Rechtssatz"/>
(670)         <rdfs:domain rdf:resource="#UnvollstaendigerRechtssatz"/>
(671)     </owl:ObjectProperty>
(672)
(673)
(674)
(675)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#setztfestMassnahme -->
(676)
(677)     <owl:ObjectProperty rdf:about="#setztfestMassnahme">
(678)         <rdfs:label rdf:datatype="&xsd;string"
(679)             >setztfestMa&#223;nahme</rdfs:label>
(680)         <rdfs:comment rdf:datatype="&xsd;string"
```

```
(681)             >Repr&#228;sentiert die Beziehung der Rechtshandlung zur
      Einzelfallfestsetzung zu der durch sie im Einzelfall festgesetzten
      Ma&#223;nahme.</rdfs:comment>
(682)         <rdfs:domain rdf:resource="#Einzelfallfestsetzung"/>
(683)         <rdfs:range rdf:resource="#Massnahme"/>
(684)     </owl:ObjectProperty>
(685)
(686)
(687)
(688)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#stelltAntrag -->
(689)
(690)     <owl:ObjectProperty rdf:about="#stelltAntrag">
(691)         <rdfs:label rdf:datatype="&xsd;string">stelltAntrag</rdfs:label>
(692)         <rdfs:comment rdf:datatype="&xsd;string"
(693)             >Repr&#228;sentiert die Beziehung von einem Antragssteller zu
      dem von ihm gestellten Antrag.</rdfs:comment>
(694)         <rdfs:range rdf:resource="#VwAntrag"/>
(695)         <rdfs:domain rdf:resource="#VwAntragsteller"/>
(696)     </owl:ObjectProperty>
(697)
(698)
(699)
(700)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#stoestAnBewusstwedung -->
(701)
(702)     <owl:ObjectProperty rdf:about="#stoestAnBewusstwedung">
(703)         <rdfs:label rdf:datatype="&xsd;string"
(704)             >stoestAnBewusstwedung</rdfs:label>
(705)         <rdfs:comment rdf:datatype="&xsd;string"
(706)             >Repr&#228;sentiert die Beziehung eines Bewusstmachers zur
      Bewusstwedung als Verwaltungshandlungserweiterung.</rdfs:comment>
(707)         <rdfs:domain rdf:resource="#VwBewusstmacher"/>
(708)         <rdfs:range rdf:resource="#VwBewusstwerdung"/>
(709)     </owl:ObjectProperty>
(710)
(711)
(712)
(713)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#strebtAnRechtsfolge -->
(714)
(715)     <owl:ObjectProperty rdf:about="#strebtAnRechtsfolge">
(716)         <rdfs:label rdf:datatype="&xsd;string"
(717)             >strebtAnRechtsfolge</rdfs:label>
(718)         <rdfs:comment rdf:datatype="&xsd;string"
(719)             >Repr&#228;sentiert die Beziehung eines Antrags zu der
      Rechtsfolge, die durch den Antrag angestrebt wird.</rdfs:comment>
(720)         <rdfs:domain rdf:resource="#VwAntrag"/>
(721)         <rdfs:range rdf:resource="#VwRechtsfolge"/>
(722)     </owl:ObjectProperty>
(723)
(724)
(725)
(726)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#umfassstNutzungBeurteilungsspielraum -->
(727)
(728)     <owl:ObjectProperty rdf:about="#umfassstNutzungBeurteilungsspielraum">
(729)         <rdfs:label rdf:datatype="&xsd;string"
(730)             >umfasstNutzungBeurteilungsspielraum</rdfs:label>
(731)         <rdfs:comment rdf:datatype="&xsd;string"
(732)             >Repr&#228;sentiert die Beziehung der Voraussetzungspr&#252;fung
      zur Nutzung von Beurteilungsspielraum, die im Rahmen der
      Voraussetzungspr&#252;fung erfolgen kann.</rdfs:comment>
(733)         <rdfs:range rdf:resource="#VwNutzungVonBeurteilungsspielraum"/>
(734)         <rdfs:domain rdf:resource="#VwVoraussetzungspruefung"/>
(735)     </owl:ObjectProperty>
(736)
(737)
(738)
(739)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#umfasstErmessensAusuebung -->
(740)
(741)     <owl:ObjectProperty rdf:about="#umfasstErmessensAusuebung">
(742)         <rdfs:label rdf:datatype="&xsd;string"
(743)             >umfasstErmessensAusuebung</rdfs:label>
```

```
(744)         <rdfs:comment rdf:datatype="&xsd;string"
(745)             >Die Mittelwahl kann im Fall von ungebundemen Handeln die
      Aus&#252;bung von Ermessen erfordern, was durch diese Beziehung
      repr&#228;sentiert wird.</rdfs:comment>
(746)         <rdfs:range rdf:resource="#VwErmessensausuebung"/>
(747)         <rdfs:domain rdf:resource="#VwMittelwahl"/>
(748)     </owl:ObjectProperty>
(749)
(750)
(751)
(752)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#umfasstVerfahrenshandlung -->
(753)
(754)     <owl:ObjectProperty rdf:about="#umfasstVerfahrenshandlung">
(755)         <rdfs:label
(756)             >umfasstVerfahrenshandlung</rdfs:label>
(757)         <rdfs:comment rdf:datatype="&xsd;string"
(758)             >Repr&#228;sentiert den Zusammenhang zwischen einem
      Verwaltungsverfahren und die in Zusammenhang mit dem Verfahren
      durchgef&#252;hrten Handlungen.</rdfs:comment>
(759)         <rdfs:domain rdf:resource="#VwVerfahren"/>
(760)         <rdfs:range rdf:resource="#VwSachverhaltsermittlung"/>
(761)         <rdfs:range rdf:resource="#VwVoraussetzungspruefung"/>
(762)         <rdfs:range rdf:resource="#VwNutzungVonBeurteilungsspielraum"/>
(763)         <rdfs:range rdf:resource="#VwMittelwahl"/>
(764)         <rdfs:range rdf:resource="#VwErmessensausuebung"/>
(765)         <rdfs:range rdf:resource="#VwVerfahrenseinleitung"/>
(766)         <rdfs:range rdf:resource="#VwMassnahmenfestlegung"/>
(767)         <rdfs:range rdf:resource="#VwLeistungserbringung"/>
(768)         <rdfs:range rdf:resource="#VwErgebnismitteilung"/>
(769)     </owl:ObjectProperty>
(770)
(771)
(772)
(773)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#verweistAufRechtssatz -->
(774)
(775)     <owl:ObjectProperty rdf:about="#verweistAufRechtssatz">
(776)         <rdfs:label
(777)             >verweistAufRechtssatz</rdfs:label>
(778)         <rdfs:comment rdf:datatype="&xsd;string"
(779)             >Repr&#228;sentiert die Beziehung eines unvollst&#228;ndigen
      Rechtssatzes zu einem Rechtssatz, auf den er verweist.</rdfs:comment>
(780)         <rdfs:range rdf:resource="#Rechtssatz"/>
(781)         <rdfs:domain rdf:resource="#UnvollstaendigerRechtssatz"/>
(782)     </owl:ObjectProperty>
(783)
(784)
(785)
(786)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#wirdAngewandtAufRechtsfolge -->
(787)
(788)     <owl:ObjectProperty rdf:about="#wirdAngewandtAufRechtsfolge">
(789)         <rdfs:label rdf:datatype="&xsd;string"
(790)             >wirdAngewandtAufRechtsfolge</rdfs:label>
(791)         <rdfs:comment rdf:datatype="&xsd;string"
(792)             >Repr&#228;sentiert die Beziehung einer Einzelfallfestsetzung zu
      der in ihrem Rahmen festgesetzten Rechtsfolge.</rdfs:comment>
(793)         <rdfs:domain rdf:resource="#Einzelfallfestsetzung"/>
(794)         <rdfs:range rdf:resource="#Rechtsfolge"/>
(795)     </owl:ObjectProperty>
(796)
(797)
(798)
(799)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#wirdAngewandtAufSachverhalt -->
(800)
(801)     <owl:ObjectProperty rdf:about="#wirdAngewandtAufSachverhalt">
(802)         <rdfs:label rdf:datatype="&xsd;string"
(803)             >wirdAngewandtAufSachverhalt</rdfs:label>
(804)         <rdfs:comment rdf:datatype="&xsd;string"
(805)             >Repr&#228;sentiert die Beziehung der Subsumtion zu dem in ihrem
      Rahmen betrachteten Sachverhalt.</rdfs:comment>
(806)         <rdfs:range rdf:resource="#Sachverhalt"/>
(807)         <rdfs:domain rdf:resource="#Subsumtion"/>
```

```
(808)     </owl:ObjectProperty>
(809)
(810)
(811)
(812)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#wirdAngewandtAufTatbestand -->
(813)
(814)     <owl:ObjectProperty rdf:about="#wirdAngewandtAufTatbestand">
(815)         <rdfs:label rdf:datatype="&xsd;string"
(816)             >wirdAngewandtAufTatbestand</rdfs:label>
(817)         <rdfs:comment rdf:datatype="&xsd;string"
(818)             >Repr&#228;sentiert die Beziehung der Subsumtion zu dem in ihrem
      Rahmen betrachteten Tatbestand.</rdfs:comment>
(819)         <rdfs:domain rdf:resource="#Rechtsfolgenbestimmung"/>
(820)         <rdfs:range rdf:resource="#Tatbestand"/>
(821)     </owl:ObjectProperty>
(822)
(823)
(824)
(825)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#wirdErreichtDuchRechtsfolge -->
(826)
(827)     <owl:ObjectProperty rdf:about="#wirdErreichtDuchRechtsfolge">
(828)         <rdfs:label rdf:datatype="&xsd;string"
(829)             >wirdErreichtDurchRechtsfolge</rdfs:label>
(830)         <rdfs:comment rdf:datatype="&xsd;string"
(831)             >Repr&#228;sentiert die Beziehung eines Ziels/Zecks zu der
      Rechtsfolge, durch die es/er erreicht werden soll.</rdfs:comment>
(832)         <rdfs:range rdf:resource="#Rechtsfolge"/>
(833)         <rdfs:domain rdf:resource="#VwZielZweck"/>
(834)     </owl:ObjectProperty>
(835)
(836)
(837)
(838)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#wirdUeberfuehrtIn -->
(839)
(840)     <owl:ObjectProperty rdf:about="#wirdUeberfuehrtIn">
(841)         <rdfs:label rdf:datatype="&xsd;string"
(842)             >wirdUeberfuehrtIn</rdfs:label>
(843)         <rdfs:comment rdf:datatype="&xsd;string"
(844)             >Repr&#228;sentiert die Beziehung eines Lebenssachverhalts zu
      dem Sachverhalt in den er durch Abstraktion der tatbestandsrelevanten
      Eigenschaften &#252;berf&#252;hrt wird.</rdfs:comment>
(845)         <rdfs:domain rdf:resource="#Lebenssachverhalt"/>
(846)         <rdfs:range rdf:resource="#Sachverhalt"/>
(847)     </owl:ObjectProperty>
(848)
(849)
(850)
(851)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#zugehoerigeLebenslage -->
(852)
(853)     <owl:ObjectProperty rdf:about="#zugehoerigeLebenslage">
(854)         <rdfs:label
(855)             >zugehoerigeLebenslage</rdfs:label>
(856)         <rdfs:comment rdf:datatype="&xsd;string"
(857)             >Repr&#228;sentiert die Beziehung, die ein Hinweis eine
      Information zu einer Lebenlage hat, in der er relevant ist.</rdfs:comment>
(858)         <rdfs:domain rdf:resource="#VwHinweisInformation"/>
(859)         <rdfs:range rdf:resource="#VwLebenslage"/>
(860)         <owl:inverseOf rdf:resource="#zugehoerigerHinweis"/>
(861)     </owl:ObjectProperty>
(862)
(863)
(864)
(865)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#zugehoerigerAntrag -->
(866)
(867)     <owl:ObjectProperty rdf:about="#zugehoerigerAntrag">
(868)         <rdfs:label
(869)             >zugehoerigerAntrag</rdfs:label>
(870)         <rdfs:comment rdf:datatype="&xsd;string"
(871)             >Repr&#228;sentiert die Beziehung, die eine Verwaltungsleistung
      zu dem Antrag hat, aufgrund dessen sie erbracht wird.</rdfs:comment>
```

```
(872)         <rdfs:range rdf:resource="#VwAntrag"/>
(873)         <rdfs:domain rdf:resource="#VwLeistung"/>
(874)     </owl:ObjectProperty>
(875)
(876)
(877)
(878)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#zugehoerigerHinweis -->
(879)
(880)     <owl:ObjectProperty rdf:about="#zugehoerigerHinweis">
(881)         <rdfs:label
(882)             >zugehoerigerHinweis</rdfs:label>
(883)         <rdfs:comment rdf:datatype="&xsd;string"
(884)             >Repr&#228;sentiert die Beziehung, die eine Lebenslage zu einem
      Hinweis/einer Information hat, der f&#252;r sie relevant ist.</rdfs:comment>
(885)         <rdfs:range rdf:resource="#VwHinweisInformation"/>
(886)         <rdfs:domain rdf:resource="#VwLebenslage"/>
(887)     </owl:ObjectProperty>
(888)
(889)
(890)
(891)     <!--
(892)     ////////////////////////////////////////////////////////////////////////
(893)     //
(894)     // Data properties
(895)     //
(896)     ////////////////////////////////////////////////////////////////////////
(897)      -->
(898)
(899)
(900)
(901)
(902)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Arbeitsnotiz -->
(903)
(904)     <owl:DatatypeProperty rdf:about="#Arbeitsnotiz">
(905)         <rdfs:label>Arbeitsnotiz</rdfs:label>
(906)         <rdfs:comment
(907)             >Bemerkungen, die w&#228;hrend der weiteren Umsetzung hilfreich
      sein k&#246;nnen.</rdfs:comment>
(908)         <rdfs:domain rdf:resource="#VerwaltungsKonzept"/>
(909)         <rdfs:range rdf:resource="&xsd;string"/>
(910)     </owl:DatatypeProperty>
(911)
(912)
(913)
(914)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Beschreibung -->
(915)
(916)     <owl:DatatypeProperty rdf:about="#Beschreibung">
(917)         <rdfs:label>Beschreibung</rdfs:label>
(918)         <rdfs:comment
(919)             >Inhaltliche Beschreibung des durch die Annotation
      repr&#228;sentierten Individuums.</rdfs:comment>
(920)         <rdfs:domain rdf:resource="#VerwaltungsKonzept"/>
(921)         <rdfs:range rdf:resource="&xsd;string"/>
(922)     </owl:DatatypeProperty>
(923)
(924)
(925)
(926)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#HandlungNachher -->
(927)
(928)     <owl:DatatypeProperty rdf:about="#HandlungNachher">
(929)         <rdfs:label>HandlungNachher</rdfs:label>
(930)         <rdfs:comment
(931)             >Beschreibung, was durch die Handlung erreicht wurde (z.B. der
      erreichte Zustand).</rdfs:comment>
(932)         <rdfs:domain rdf:resource="#VwHandlung"/>
(933)         <rdfs:domain rdf:resource="#VwHandlungsErweiterung"/>
(934)         <rdfs:range rdf:resource="&xsd;string"/>
(935)     </owl:DatatypeProperty>
(936)
(937)
(938)
```

```
(939)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#HandlungVorher -->
(940)
(941)     <owl:DatatypeProperty rdf:about="#HandlungVorher">
(942)         <rdfs:label>HandlungVorher</rdfs:label>
(943)         <rdfs:comment
(944)             >Beschreibung, was vor Durchf&#252;hrung der Handlungen bereits
      erfolgt sein muss.</rdfs:comment>
(945)         <rdfs:domain rdf:resource="#VwHandlung"/>
(946)         <rdfs:domain rdf:resource="#VwHandlungsErweiterung"/>
(947)         <rdfs:range rdf:resource="&xsd;string"/>
(948)     </owl:DatatypeProperty>
(949)
(950)
(951)
(952)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#HandlungsZiel -->
(953)
(954)     <owl:DatatypeProperty rdf:about="#HandlungsZiel">
(955)         <rdfs:label>HandlungsZiel</rdfs:label>
(956)         <rdfs:comment
(957)             >Beschreibung des mit der Handlung verfolgten
      Ziels.</rdfs:comment>
(958)         <rdfs:domain rdf:resource="#VwHandlung"/>
(959)         <rdfs:domain rdf:resource="#VwHandlungsErweiterung"/>
(960)         <rdfs:range rdf:resource="&xsd;string"/>
(961)     </owl:DatatypeProperty>
(962)
(963)
(964)
(965)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Name -->
(966)
(967)     <owl:DatatypeProperty rdf:about="#Name">
(968)         <rdfs:label>Name</rdfs:label>
(969)         <rdfs:comment
(970)             >Bietet die M&#246;glichkeit zur Angabe eines alternativen
      Names, der sich nicht direkt aus dem Text einer annotierten Textpassage
      ergibt.</rdfs:comment>
(971)         <rdfs:domain rdf:resource="#VerwaltungsKonzept"/>
(972)         <rdfs:range rdf:resource="&xsd;string"/>
(973)     </owl:DatatypeProperty>
(974)
(975)
(976)
(977)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#NormativerSinn -->
(978)
(979)     <owl:DatatypeProperty rdf:about="#NormativerSinn">
(980)         <rdfs:label>NormativerSinn</rdfs:label>
(981)         <rdfs:comment rdf:datatype="&xsd;string"
(982)             >Beschreibung des normativen Sinns eines
      Rechtssatzes.</rdfs:comment>
(983)         <rdfs:domain rdf:resource="#Rechtssatz"/>
(984)         <rdfs:range rdf:resource="&xsd;string"/>
(985)     </owl:DatatypeProperty>
(986)
(987)
(988)
(989)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Quelle -->
(990)
(991)     <owl:DatatypeProperty rdf:about="#Quelle">
(992)         <rdfs:label>Quelle</rdfs:label>
(993)         <rdfs:comment rdf:datatype="&xsd;string"
(994)             >Erm&#246;glicht eine Angabe zur Quelle des
      Rechtsnormbestandteils. (Bei der Annotation von Rechtsvorschriften mit
      Instanzen der Ontologie ergibt sich dieser Bezug durch den
      Annotationskontext.)</rdfs:comment>
(995)         <rdfs:domain rdf:resource="#RechtsNormbestandteil"/>
(996)         <rdfs:range rdf:resource="&xsd;string"/>
(997)     </owl:DatatypeProperty>
(998)
(999)
(1000)
```

```
(1001)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Rahmenbedingung -->
(1002)
(1003)      <owl:DatatypeProperty rdf:about="#Rahmenbedingung">
(1004)          <rdfs:label>Rahmenbedingung</rdfs:label>
(1005)          <rdfs:comment
(1006)              >Beschreibung der besonderen Bedingungen f&#252;r die Umsetzung
      oder den Einsatz des Konzepts.</rdfs:comment>
(1007)          <rdfs:domain rdf:resource="#VerwaltungsKonzept"/>
(1008)          <rdfs:range rdf:resource="&xsd;string"/>
(1009)      </owl:DatatypeProperty>
(1010)
(1011)
(1012)
(1013)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istKostenpflichtig -->
(1014)
(1015)      <owl:DatatypeProperty rdf:about="#istKostenpflichtig">
(1016)          <rdfs:label
(1017)              >istKostenpflichtig</rdfs:label>
(1018)          <rdfs:comment rdf:datatype="&xsd;string"
(1019)              >Kennzeichnet, ob eine Rechtsfolge in Form einer
      kostenpflichtigen Verwaltungsleistung erbracht wird.</rdfs:comment>
(1020)          <rdfs:domain rdf:resource="#VwRechtsfolge"/>
(1021)          <rdfs:range rdf:resource="&xsd;boolean"/>
(1022)      </owl:DatatypeProperty>
(1023)
(1024)
(1025)
(1026)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istRahmenbedingungEinsatz -->
(1027)
(1028)      <owl:DatatypeProperty rdf:about="#istRahmenbedingungEinsatz">
(1029)          <rdfs:label
(1030)              >istRahmenbedingungEinsatz</rdfs:label>
(1031)          <rdfs:comment
(1032)              >Kennzeichen gibt an, ob die formulierte Rahmenbedingung zum
      Zeitpunkt des (sp&#228;teren) Einsatzes f&#252;r das Konzept relevant
      ist.</rdfs:comment>
(1033)          <rdfs:domain rdf:resource="#VerwaltungsKonzept"/>
(1034)          <rdfs:range rdf:resource="&xsd;boolean"/>
(1035)      </owl:DatatypeProperty>
(1036)
(1037)
(1038)
(1039)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#istRahmenbedingungUmsetzung -->
(1040)
(1041)      <owl:DatatypeProperty rdf:about="#istRahmenbedingungUmsetzung">
(1042)          <rdfs:label
(1043)              >istRahmenbedingungUmsetzung</rdfs:label>
(1044)          <rdfs:comment
(1045)              >Kennzeichen gibt an, ob die formulierte Rahmenbedingung bei
      der Umsetzung des Konzepts zu beachten ist.</rdfs:comment>
(1046)          <rdfs:domain rdf:resource="#VerwaltungsKonzept"/>
(1047)          <rdfs:range rdf:resource="&xsd;boolean"/>
(1048)      </owl:DatatypeProperty>
(1049)
(1050)
(1051)
(1052)      <!--
(1053)      ////////////////////////////////////////////////////////////////////////
(1054)      //
(1055)      // Classes
(1056)      //
(1057)      ////////////////////////////////////////////////////////////////////////
(1058)       -->
(1059)
(1060)
(1061)
(1062)
(1063)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Adressat -->
(1064)
(1065)      <owl:Class rdf:about="#Adressat">
```

```
(1066)          <rdfs:label rdf:datatype="&xsd;string">Adressat</rdfs:label>
(1067)          <rdfs:subClassOf rdf:resource="#RechtsAkteur"/>
(1068)          <rdfs:subClassOf>
(1069)              <owl:Restriction>
(1070)                  <owl:onProperty rdf:resource="#istAdressatVon"/>
(1071)                  <owl:someValuesFrom rdf:resource="#RechtsNorm"/>
(1072)              </owl:Restriction>
(1073)          </rdfs:subClassOf>
(1074)          <owl:disjointWith rdf:resource="#Anwender"/>
(1075)          <rdfs:comment rdf:datatype="&xsd;string"
(1076)              >Adressat der Rechtsvorschrift, die ihm Tun oder Unterlassen
      vorschreibt oder ihm ein bestimmtes Recht einr&#228;umt oder
      entzieht.</rdfs:comment>
(1077)      </owl:Class>
(1078)
(1079)
(1080)
(1081)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#AllgemeinerTextbestandteil -->
(1082)
(1083)      <owl:Class rdf:about="#AllgemeinerTextbestandteil">
(1084)          <rdfs:label rdf:datatype="&xsd;string"
(1085)              >AllgemeinerTextbestandteil</rdfs:label>
(1086)          <rdfs:subClassOf rdf:resource="#Textbestandteil"/>
(1087)          <rdfs:comment rdf:datatype="&xsd;string"
(1088)              >AllgemeinerTextbestandteil, repr&#228;sentiert sprachliche
      Elemente, die Bestandteil von Texten aus sprachlicher Sicht sind (z.B.
      W&#246;rter, Wortgruppen, S&#228;tze, Abs&#228;tze, Abschnitte, Paragraphen,
      Kapitel).
(1089)
(1090)  Hinweis: Die Regeln ihres Aufbaus m&#252;ssen f&#252;r die beabsichtigte
      Verwendung der Ontologie zur Annotation vorhandener Rechtsvorschriften nicht
      weitergehend formalisiert werden, da vorhandene Text mit Annotationen
      versehen werden und ihre sprachliche Struktur als Bezugspunkt nicht
      erforderlich ist. Das Konzept des AllgemeinenTextbestandteils dient in der
      Ontologie vor allem der Abgrenzung der weiteren Spezialisierung des
      Textbestandteils</rdfs:comment>
(1091)      </owl:Class>
(1092)
(1093)
(1094)
(1095)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Anwender -->
(1096)
(1097)      <owl:Class rdf:about="#Anwender">
(1098)          <rdfs:label rdf:datatype="&xsd;string">Anwender</rdfs:label>
(1099)          <rdfs:subClassOf rdf:resource="#RechtsAkteur"/>
(1100)          <rdfs:subClassOf>
(1101)              <owl:Restriction>
(1102)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1103)                  <owl:someValuesFrom
      rdf:resource="#Sachverhaltsermittlung"/>
(1104)              </owl:Restriction>
(1105)          </rdfs:subClassOf>
(1106)          <rdfs:subClassOf>
(1107)              <owl:Restriction>
(1108)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1109)                  <owl:someValuesFrom
      rdf:resource="#Rechtsfolgenbestimmung"/>
(1110)              </owl:Restriction>
(1111)          </rdfs:subClassOf>
(1112)          <rdfs:subClassOf>
(1113)              <owl:Restriction>
(1114)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1115)                  <owl:someValuesFrom rdf:resource="#Mitteilung"/>
(1116)              </owl:Restriction>
(1117)          </rdfs:subClassOf>
(1118)          <rdfs:subClassOf>
(1119)              <owl:Restriction>
(1120)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1121)                  <owl:someValuesFrom rdf:resource="#Einzelfallfestsetzung"/>
(1122)              </owl:Restriction>
(1123)          </rdfs:subClassOf>
(1124)          <rdfs:subClassOf>
(1125)              <owl:Restriction>
```

```
(1126)                 <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1127)                 <owl:someValuesFrom rdf:resource="#Rechtsfindung"/>
(1128)             </owl:Restriction>
(1129)         </rdfs:subClassOf>
(1130)         <rdfs:comment rdf:datatype="&xsd;string"
(1131)             >Person oder Organisation, die Recht anwendet, d.h. aus
      Rechtsnormen zun&#228;chst die Rechtss&#228;tze mit ihren Bestandteilen
      ableitet, die daraus resultierenden Rechtshandlungen durchf&#252;hrt und
      dabei die rechtlich relevanten Handlungsgegenst&#228;nde
      bearbeitet.</rdfs:comment>
(1132)     </owl:Class>
(1133)
(1134)
(1135)
(1136)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#DomainConcept -->
(1137)
(1138)     <owl:Class rdf:about="#DomainConcept">
(1139)         <rdfs:label>DomainConcept</rdfs:label>
(1140)         <rdfs:comment
(1141)             >Im Rahmen dieser Arbeit f&#252;r die Konzepte der Ontologie
      verwendete abstrakte Oberklasse.</rdfs:comment>
(1142)     </owl:Class>
(1143)
(1144)
(1145)
(1146)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Einzelfallfestsetzung -->
(1147)
(1148)     <owl:Class rdf:about="#Einzelfallfestsetzung">
(1149)         <rdfs:label rdf:datatype="&xsd;string"
(1150)             >Einzelfallfestsetzung</rdfs:label>
(1151)         <rdfs:subClassOf rdf:resource="#RechtsHandlung"/>
(1152)         <rdfs:subClassOf>
(1153)             <owl:Restriction>
(1154)                 <owl:onProperty rdf:resource="#setztfestMassnahme"/>
(1155)                 <owl:someValuesFrom rdf:resource="#Massnahme"/>
(1156)             </owl:Restriction>
(1157)         </rdfs:subClassOf>
(1158)         <rdfs:subClassOf>
(1159)             <owl:Restriction>
(1160)                 <owl:onProperty
      rdf:resource="#wirdAngewandtAufRechtsfolge"/>
(1161)                 <owl:someValuesFrom rdf:resource="#Rechtsfolge"/>
(1162)             </owl:Restriction>
(1163)         </rdfs:subClassOf>
(1164)         <rdfs:comment rdf:datatype="&xsd;string"
(1165)             >Ein Schritt der Rechtsanwendung, in dessen Rahmen aus einer
      abstrakten Rechtfolge durch Konkretisierung vor dem Hintergrund des
      vorliegenden Einzelfalls eine konkrete Ma&#223;nahme abgeleitet und
      festgesetzt wird.</rdfs:comment>
(1166)     </owl:Class>
(1167)
(1168)
(1169)
(1170)     <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Lebenssachverhalt -->
(1171)
(1172)     <owl:Class rdf:about="#Lebenssachverhalt">
(1173)         <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
(1174)         <rdfs:subClassOf>
(1175)             <owl:Restriction>
(1176)                 <owl:onProperty rdf:resource="#wirdUeberfuehrtIn"/>
(1177)                 <owl:someValuesFrom rdf:resource="#Sachverhalt"/>
(1178)             </owl:Restriction>
(1179)         </rdfs:subClassOf>
(1180)         <rdfs:comment rdf:datatype="&xsd;string"
(1181)             >Der Lebenssacherhalt ist ein tats&#228;chliches Ereignis oder
      eine tats&#228;chlich vorliege Situation, die durch eine beliebige Menge von
      Eigenschaften gekennzeichnet ist, von denen f&#252;r die Rechtsanwendung
      jedoch nur eine durch den Sachverhalt definierte Teilmenge relevant  ist. Da
      Lebenssachverhalte durch eine Beliebigkeit der tats&#228;chlichen Situation
      und der sie charakterisieren Merkmale gekennzeichnet sind, k&#246;nnen
      Normtexte per Definition keinen direkten Bezug zu ihnen enthalten.
(1182)  (syn. Einzelfall)</rdfs:comment>
```

```
(1183)      </owl:Class>
(1184)
(1185)
(1186)
(1187)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Massnahme -->
(1188)
(1189)      <owl:Class rdf:about="#Massnahme">
(1190)          <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
(1191)          <rdfs:subClassOf>
(1192)              <owl:Restriction>
(1193)                  <owl:onProperty rdf:resource="#hatMassnahmenEigenschaft"/>
(1194)                  <owl:onClass rdf:resource="#MassnahmenEigenschaft"/>
(1195)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">1</owl:minQualifiedCardinality>
(1196)              </owl:Restriction>
(1197)          </rdfs:subClassOf>
(1198)          <rdfs:subClassOf>
(1199)              <owl:Restriction>
(1200)                  <owl:onProperty rdf:resource="#hatMassnahmenEigenschaft"/>
(1201)                  <owl:someValuesFrom rdf:resource="#MassnahmenEigenschaft"/>
(1202)              </owl:Restriction>
(1203)          </rdfs:subClassOf>
(1204)          <rdfs:comment rdf:datatype="&xsd;string"
(1205)              >Konkretisierung einer abstrakten Rechtsfolge, in dem alle
      abstrakten Merkmale der Rechtsfolge f&#252;r den vorliegenden Einzelfall
      konkretisiert werden. Die Ma&#223;nahme ist Element der Menge von
      m&#246;glichen Ma&#223;nahmen, die durch die Rechtsfolge repr&#228;sentiert
      werden.</rdfs:comment>
(1206)      </owl:Class>
(1207)
(1208)
(1209)
(1210)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#MassnahmenEigenschaft -->
(1211)
(1212)      <owl:Class rdf:about="#MassnahmenEigenschaft">
(1213)          <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
(1214)          <rdfs:subClassOf>
(1215)              <owl:Restriction>
(1216)                  <owl:onProperty rdf:resource="#gehoertZuMassnahme"/>
(1217)                  <owl:someValuesFrom rdf:resource="#Massnahme"/>
(1218)              </owl:Restriction>
(1219)          </rdfs:subClassOf>
(1220)          <rdfs:comment rdf:datatype="&xsd;string"
(1221)              >Eine konkretisierte Eigenschaft einer Ma&#223;nahme wird durch
      eine Instanz des Konzepts Ma&#223;nahmeneigenschaft ausgedr&#252;ckt. Es ist
      immer eine Konkretisierung eines abstrakten Merkmales einer
      Rechtsfolge.</rdfs:comment>
(1222)      </owl:Class>
(1223)
(1224)
(1225)
(1226)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Mitteilung -->
(1227)
(1228)      <owl:Class rdf:about="#Mitteilung">
(1229)          <rdfs:label rdf:datatype="&xsd;string">Mitteilung</rdfs:label>
(1230)          <rdfs:subClassOf rdf:resource="#RechtsHandlung"/>
(1231)          <rdfs:comment rdf:datatype="&xsd;string"
(1232)              >Ein Schritt der Rechtsanwendung, in dessen Rahmen die
      vorgenommene Einzelfallfestsetzung gegen&#252;ber dem/den Adressaten
      verbindlich kommuniziert wird, u.a. damit die Einzelfallfestsetzung ihre
      Rechtswirkung entfalten kann.</rdfs:comment>
(1233)      </owl:Class>
(1234)
(1235)
(1236)
(1237)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#RechtsAkteur -->
(1238)
(1239)      <owl:Class rdf:about="#RechtsAkteur">
(1240)          <rdfs:label rdf:datatype="&xsd;string">RechtsAkteur</rdfs:label>
(1241)          <rdfs:subClassOf rdf:resource="#RechtsKonzept"/>
(1242)          <rdfs:subClassOf>
```

```
(1243)              <owl:Restriction>
(1244)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1245)                  <owl:someValuesFrom rdf:resource="#Einzelfallfestsetzung"/>
(1246)              </owl:Restriction>
(1247)          </rdfs:subClassOf>
(1248)          <rdfs:subClassOf>
(1249)              <owl:Restriction>
(1250)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1251)                  <owl:someValuesFrom
      rdf:resource="#Sachverhaltsermittlung"/>
(1252)              </owl:Restriction>
(1253)          </rdfs:subClassOf>
(1254)          <rdfs:subClassOf>
(1255)              <owl:Restriction>
(1256)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1257)                  <owl:someValuesFrom rdf:resource="#Mitteilung"/>
(1258)              </owl:Restriction>
(1259)          </rdfs:subClassOf>
(1260)          <rdfs:subClassOf>
(1261)              <owl:Restriction>
(1262)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1263)                  <owl:someValuesFrom rdf:resource="#Subsumtion"/>
(1264)              </owl:Restriction>
(1265)          </rdfs:subClassOf>
(1266)          <rdfs:subClassOf>
(1267)              <owl:Restriction>
(1268)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1269)                  <owl:someValuesFrom rdf:resource="#Rechtsfindung"/>
(1270)              </owl:Restriction>
(1271)          </rdfs:subClassOf>
(1272)          <rdfs:subClassOf>
(1273)              <owl:Restriction>
(1274)                  <owl:onProperty rdf:resource="#fuehrtDurchRechtshandlung"/>
(1275)                  <owl:someValuesFrom
      rdf:resource="#Rechtsfolgenbestimmung"/>
(1276)              </owl:Restriction>
(1277)          </rdfs:subClassOf>
(1278)          <rdfs:comment rdf:datatype="&xsd;string"
(1279)              >RechtsAkteur ist die Abstraktion f&#252;r Personen oder
      Organisationen, die an der Rechtsanwendung beteiligt sind.</rdfs:comment>
(1280)      </owl:Class>
(1281)
(1282)
(1283)
(1284)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#RechtsHandlung -->
(1285)
(1286)      <owl:Class rdf:about="#RechtsHandlung">
(1287)          <rdfs:subClassOf rdf:resource="#RechtsKonzept"/>
(1288)          <rdfs:comment rdf:datatype="&xsd;string"
(1289)              >Eine Rechtshandlung repr&#228;sentiert Schritte der
      Rechtsanwendung, die durch etablierte Methoden der Rechtswissenschaft
      vorgegeben sind und dazu dienen, Recht und insbesondere Rechtss&#228;tze auf
      einen Einzelfall anzuwenden.</rdfs:comment>
(1290)      </owl:Class>
(1291)
(1292)
(1293)
(1294)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#RechtsHandlungsgegenstand -->
(1295)
(1296)      <owl:Class rdf:about="#RechtsHandlungsgegenstand">
(1297)          <rdfs:subClassOf rdf:resource="#RechtsKonzept"/>
(1298)          <rdfs:comment rdf:datatype="&xsd;string"
(1299)              >Der Rechtshandlungsgegenstand repr&#228;sentiert abstrakte
      Konzepte und konkrete Dinge, die im Rahmen von Rechtshandlungen erzeugt,
      verwendet und/oder manipuliert werden.</rdfs:comment>
(1300)      </owl:Class>
(1301)
(1302)
(1303)
(1304)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#RechtsKonzept -->
(1305)
(1306)      <owl:Class rdf:about="#RechtsKonzept">
```

```
(1307)          <rdfs:label>RechtsKonzept</rdfs:label>
(1308)          <rdfs:subClassOf rdf:resource="#DomainConcept"/>
(1309)          <rdfs:subClassOf>
(1310)              <owl:Restriction>
(1311)                  <owl:onProperty rdf:resource="#konkretisiertTextKonzept"/>
(1312)                  <owl:allValuesFrom rdf:resource="#TextKonzept"/>
(1313)              </owl:Restriction>
(1314)          </rdfs:subClassOf>
(1315)          <rdfs:comment rdf:datatype="&xsd;string"
(1316)              >Rechtskonzept ist die Abstraktion f&#252;r die an der
      Rechtsanwendung Beteiligten, die durchgef&#252;hrten Rechtshandlungen und
      die dabei bearbeiteten Handlungsgegenst&#228;nde, sowie f&#252;r den
      Normtext und die Normtextbestandteile.</rdfs:comment>
(1317)      </owl:Class>
(1318)
(1319)
(1320)
(1321)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#RechtsNorm -->
(1322)
(1323)      <owl:Class rdf:about="#RechtsNorm">
(1324)          <rdfs:label rdf:datatype="&xsd;string">RechtsNorm</rdfs:label>
(1325)          <rdfs:subClassOf rdf:resource="#Text"/>
(1326)          <rdfs:subClassOf>
(1327)              <owl:Restriction>
(1328)                  <owl:onProperty
      rdf:resource="#bestehtAusRechtsnormbestandteil"/>
(1329)                  <owl:someValuesFrom rdf:resource="#RechtsNormbestandteil"/>
(1330)              </owl:Restriction>
(1331)          </rdfs:subClassOf>
(1332)          <rdfs:comment rdf:datatype="&xsd;string"
(1333)              >Eine Rechtsnorm ist ein spezieller Text, der sich in der Regel
      in mehrere inhaltlich zusammengeh&#246;rige Abschnitte gliedert, die
      ihrerseits aus W&#246;rtern, Wortgruppen, S&#228;tzen und Satzteilen
      bestehen und sich in weitere Unterabschnitte gliedern k&#246;nnen. Die
      Rechtsnorm beschreibt die Regelung eines zusammengeh&#246;rigen Themas bzw.
      Rechtsgebietes. Per Definition besteht jede Rechtsnorm aus mindestens einem
      Rechtssatz, in der Regel aber aus mehreren Rechtss&#228;tzen.</rdfs:comment>
(1334)      </owl:Class>
(1335)
(1336)
(1337)
(1338)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#RechtsNormbestandteil -->
(1339)
(1340)      <owl:Class rdf:about="#RechtsNormbestandteil">
(1341)          <rdfs:label rdf:datatype="&xsd;string"
(1342)              >RechtsNormbestandteil</rdfs:label>
(1343)          <rdfs:subClassOf rdf:resource="#Textbestandteil"/>
(1344)          <rdfs:comment rdf:datatype="&xsd;string"
(1345)              >Abstrakte Oberklasse f&#252;r Textbestandteile von
      Rechtsnormen.</rdfs:comment>
(1346)      </owl:Class>
(1347)
(1348)
(1349)
(1350)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Rechtsfindung -->
(1351)
(1352)      <owl:Class rdf:about="#Rechtsfindung">
(1353)          <rdfs:label rdf:datatype="&xsd;string">Rechtsfindung</rdfs:label>
(1354)          <rdfs:subClassOf rdf:resource="#RechtsHandlung"/>
(1355)          <rdfs:comment rdf:datatype="&xsd;string"
(1356)              >Rechtsfindung repr&#228;sentiert einen Schritt der
      Rechtsanwendung, in dessen Rahmen die auf den vorliegenden Einzelfall
      anzuwendenden Rechtsnormen identifiziert werden.</rdfs:comment>
(1357)      </owl:Class>
(1358)
(1359)
(1360)
(1361)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Rechtsfolge -->
(1362)
(1363)      <owl:Class rdf:about="#Rechtsfolge">
(1364)          <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
```

```
(1365)          <rdfs:subClassOf rdf:resource="#Rechtssatzbestandteil"/>
(1366)          <rdfs:subClassOf>
(1367)              <owl:Restriction>
(1368)                  <owl:onProperty rdf:resource="#hatRechtsfolgenMerkmal"/>
(1369)                  <owl:allValuesFrom rdf:resource="#RechtsfolgenMerkmal"/>
(1370)              </owl:Restriction>
(1371)          </rdfs:subClassOf>
(1372)          <rdfs:subClassOf>
(1373)              <owl:Restriction>
(1374)                  <owl:onProperty rdf:resource="#resultiertInMassnahme"/>
(1375)                  <owl:someValuesFrom rdf:resource="#Massnahme"/>
(1376)              </owl:Restriction>
(1377)          </rdfs:subClassOf>
(1378)          <rdfs:subClassOf>
(1379)              <owl:Restriction>
(1380)                  <owl:onProperty rdf:resource="#hatRechtsfolgenMerkmal"/>
(1381)                  <owl:onClass rdf:resource="#RechtsfolgenMerkmal"/>
(1382)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">1</owl:minQualifiedCardinality>
(1383)              </owl:Restriction>
(1384)          </rdfs:subClassOf>
(1385)          <rdfs:comment rdf:datatype="&xsd;string"
(1386)              >Die Rechtsfolge ist die in einen Rechtssatz f&#252;r einen
      Tatbestand vorgesehene abstrakte Beschreibung der rechtlichen Konsequenz und
      kann als eine Menge von konkreten Ma&#223;nahmen aufgefasst werden, zu der
      alle Ma&#223;nahmen geh&#246;ren, die der abstrakten Beschreibung der
      Rechtsfolge gen&#252;gen.</rdfs:comment>
(1387)      </owl:Class>
(1388)
(1389)
(1390)
(1391)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#RechtsfolgenMerkmal -->
(1392)
(1393)      <owl:Class rdf:about="#RechtsfolgenMerkmal">
(1394)          <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
(1395)          <rdfs:subClassOf rdf:resource="#Rechtssatzbestandteil"/>
(1396)          <rdfs:subClassOf>
(1397)              <owl:Restriction>
(1398)                  <owl:onProperty rdf:resource="#istMerkmalVonRechtsfolge"/>
(1399)                  <owl:someValuesFrom rdf:resource="#Rechtsfolge"/>
(1400)              </owl:Restriction>
(1401)          </rdfs:subClassOf>
(1402)          <rdfs:comment rdf:datatype="&xsd;string"
(1403)              >Ein Rechtsfolgenmerkmal abstrakte Beschreibung eines Merkmals
      der Rechtsfolge, ist die in einen Rechtssatz f&#252;r einen Tatbestand
      vorgesehen ist. Dieses Merkmal muss f&#252;r die Anwendung auf einen
      vorliegenden Einzelfall konkretisiert werden.</rdfs:comment>
(1404)      </owl:Class>
(1405)
(1406)
(1407)
(1408)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Rechtsfolgenbestimmung -->
(1409)
(1410)      <owl:Class rdf:about="#Rechtsfolgenbestimmung">
(1411)          <rdfs:label rdf:datatype="&xsd;string"
(1412)              >Rechtsfolgenbestimmung</rdfs:label>
(1413)          <rdfs:subClassOf rdf:resource="#RechtsHandlung"/>
(1414)          <rdfs:subClassOf>
(1415)              <owl:Restriction>
(1416)                  <owl:onProperty rdf:resource="#bestimmtRechtsfolge"/>
(1417)                  <owl:someValuesFrom rdf:resource="#Rechtsfolge"/>
(1418)              </owl:Restriction>
(1419)          </rdfs:subClassOf>
(1420)          <rdfs:subClassOf>
(1421)              <owl:Restriction>
(1422)                  <owl:onProperty
      rdf:resource="#wirdAngewandtAufTatbestand"/>
(1423)                  <owl:someValuesFrom rdf:resource="#Tatbestand"/>
(1424)              </owl:Restriction>
(1425)          </rdfs:subClassOf>
(1426)          <rdfs:comment rdf:datatype="&xsd;string"
```

```
(1427)              >Ein Schritt der Rechtsanwendung, in dessen Rahmen eine oder
      mehrere abstrakte Rechtsfolgen, die von einer Rechtsnorm f&#252;r einen
      Tatbestand vorgesehen und rechtlich zul&#228;ssig sind, bestimmt
      werden.</rdfs:comment>
(1428)      </owl:Class>
(1429)
(1430)
(1431)
(1432)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Rechtssatz -->
(1433)
(1434)      <owl:Class rdf:about="#Rechtssatz">
(1435)          <rdfs:label rdf:datatype="&xsd;string">Rechtssatz</rdfs:label>
(1436)          <rdfs:subClassOf rdf:resource="#RechtsNormbestandteil"/>
(1437)          <rdfs:subClassOf>
(1438)              <owl:Restriction>
(1439)                  <owl:onProperty rdf:resource="#NormativerSinn"/>
(1440)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">1</owl:minQualifiedCardinality>
(1441)                  <owl:onDataRange rdf:resource="&xsd;string"/>
(1442)              </owl:Restriction>
(1443)          </rdfs:subClassOf>
(1444)          <rdfs:comment rdf:datatype="&xsd;string"
(1445)              >Ein Rechtssatz ist die sprachliche Form zum Ausdruck von
      Bestandteilen und Zusammenh&#228;ngen einer Rechtsnorm, die in einem Text
      dokumentiert ist. Der Rechtssatz verf&#252;gt immer &#252;ber einen
      normativen Sinn indem eine Geltungsanordnung getroffen wird, durch die eine
      bestimmte Regelung erfolgt bzw. erfolgen soll. Es k&#246;nnen
      vollst&#228;ndige, unvollst&#228;ndige Rechtss&#228;tze hand ihrer
      Bestandteile unterschieden werden. Da eine Rechtsnorm keine blo&#223;e
      Abfolge von Rechtss&#228;tze ist, setzt sich der Rechtssatz aus den
      sprachlichen Bestandteilen des konkret vorliegenden Textes zusammen.
      Rechtss&#228;tze beschreiben als Rechtskonzept den Tatbestand mit dessen
      abstrakten Merkmalen und Bedingungen, sowie die vorgesehene rechtliche
      Konsequenz mit deren abstrakten Merkmalen in sprachlicher Form.
      Rechtss&#228;tze k&#246;nnen aus Normtexten gewonnen werden, in dem
      W&#246;rter, Wortgruppen, S&#228;tze/Teils&#228;tze durch einen
      Rechtsanwender interpretiert und in ihnen Rechtss&#228;tze mit ihren
      Bestandteilen identifiziert und zu einander in Beziehung gesetzt werden. Es
      k&#246;nnen vollst&#228;ndige und unvollst&#228;ndige Rechtss&#228;tze
      anhand ihrer Bestandteile unterschieden werden.</rdfs:comment>
(1446)      </owl:Class>
(1447)
(1448)
(1449)
(1450)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Rechtssatzbestandteil -->
(1451)
(1452)      <owl:Class rdf:about="#Rechtssatzbestandteil">
(1453)          <rdfs:subClassOf rdf:resource="#RechtsNormbestandteil"/>
(1454)          <rdfs:comment rdf:datatype="&xsd;string"
(1455)              >Konzept f&#252;r die Oberklasse der in einem Rechtssatz
      enthaltenen Bestandteile aus denen sich der Regelungscharakter
      ergibt.</rdfs:comment>
(1456)      </owl:Class>
(1457)
(1458)
(1459)
(1460)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Rechtsvorschrift -->
(1461)
(1462)      <owl:Class rdf:about="#Rechtsvorschrift">
(1463)          <rdfs:label rdf:datatype="&xsd;string"
(1464)              >Rechtsvorschrift</rdfs:label>
(1465)          <rdfs:subClassOf rdf:resource="#RechtsNorm"/>
(1466)          <rdfs:comment rdf:datatype="&xsd;string"
(1467)              >Bezeichnet eine schriftlich fixierte Rechtsnorm, der sich an
      Adressaten au&#223;erhalb und innerhalb der Verwaltung richtet und ihnen
      Handlungsvorgaben macht.</rdfs:comment>
(1468)      </owl:Class>
(1469)
(1470)
(1471)
(1472)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Sachverhalt -->
```

```
(1473)
(1474)      <owl:Class rdf:about="#Sachverhalt">
(1475)          <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
(1476)          <rdfs:subClassOf>
(1477)              <owl:Restriction>
(1478)                  <owl:onProperty
      rdf:resource="#hatSachverhaltsEigenschaft"/>
(1479)                  <owl:someValuesFrom
      rdf:resource="#SachverhaltsEigenschaft"/>
(1480)              </owl:Restriction>
(1481)          </rdfs:subClassOf>
(1482)          <rdfs:subClassOf>
(1483)              <owl:Restriction>
(1484)                  <owl:onProperty rdf:resource="#istExstrahiertAus"/>
(1485)                  <owl:someValuesFrom rdf:resource="#Lebenssachverhalt"/>
(1486)              </owl:Restriction>
(1487)          </rdfs:subClassOf>
(1488)          <rdfs:comment rdf:datatype="&xsd;string"
(1489)              >Ergebnis der Extraktion und Abstraktion tatbestandsrelevanter
      Eigenschaften eines vorliegenden Lebenssachverhalts in einer
      subsumtionsf&#228;higen Form. Die tatbestandsrelevanten Eigenschaften
      ergeben sich aus den Eigenschaften, die der Tatbestand aufweist, unter den
      der Sachverhalt subsummiert werden soll. Ein Sachverhalt hat folglich eine
      verfolgbare Beziehung zu den Worten, Wortgruppen, Teils&#228;tzen,
      S&#228;tzen und/oder Gliederungselementen eines Normtextes, durch die der
      Tatbestand beschrieben wird.</rdfs:comment>
(1490)      </owl:Class>
(1491)
(1492)
(1493)
(1494)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#SachverhaltsEigenschaft -->
(1495)
(1496)      <owl:Class rdf:about="#SachverhaltsEigenschaft">
(1497)          <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
(1498)          <rdfs:subClassOf>
(1499)              <owl:Restriction>
(1500)                  <owl:onProperty
      rdf:resource="#istEigenschaftVonSachverhalt"/>
(1501)                  <owl:someValuesFrom rdf:resource="#Sachverhalt"/>
(1502)              </owl:Restriction>
(1503)          </rdfs:subClassOf>
(1504)          <rdfs:comment rdf:datatype="&xsd;string"
(1505)              >Jeweils eine tatbestandrelevante Eigenschaft, die durch den
      vorliegenden Lebenssachverhalt belegt wird, wird als eine Instanz dieses
      Konzepte repr&#228;sentiert.</rdfs:comment>
(1506)      </owl:Class>
(1507)
(1508)
(1509)
(1510)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Sachverhaltsermittlung -->
(1511)
(1512)      <owl:Class rdf:about="#Sachverhaltsermittlung">
(1513)          <rdfs:label rdf:datatype="&xsd;string"
(1514)              >Sachverhaltsermittlung</rdfs:label>
(1515)          <rdfs:subClassOf rdf:resource="#RechtsHandlung"/>
(1516)          <rdfs:subClassOf>
(1517)              <owl:Restriction>
(1518)                  <owl:onProperty rdf:resource="#ermitteltSachverhalt"/>
(1519)                  <owl:allValuesFrom rdf:resource="#Sachverhalt"/>
(1520)              </owl:Restriction>
(1521)          </rdfs:subClassOf>
(1522)          <rdfs:subClassOf>
(1523)              <owl:Restriction>
(1524)                  <owl:onProperty
      rdf:resource="#abstrahiertLebenssachverhalt"/>
(1525)                  <owl:someValuesFrom rdf:resource="#Lebenssachverhalt"/>
(1526)              </owl:Restriction>
(1527)          </rdfs:subClassOf>
(1528)          <rdfs:comment rdf:datatype="&xsd;string"
(1529)              >Ein Schritt der Rechtsanwendung, in dessen Rahmen aus dem
      tats&#228;chlich vorliegenden Lebenssachverhalt alle tatbestandsrelevanten
      Eigenschaften abstrahiert und in die subsumtionsf&#228;hige Form des
      rechtlich relevanten Sachverhalts gebracht werden.</rdfs:comment>
```

```
(1530)      </owl:Class>
(1531)
(1532)
(1533)
(1534)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Subsumtion -->
(1535)
(1536)      <owl:Class rdf:about="#Subsumtion">
(1537)          <rdfs:label rdf:datatype="&xsd;string">Subsumtion</rdfs:label>
(1538)          <rdfs:subClassOf rdf:resource="#RechtsHandlung"/>
(1539)          <rdfs:subClassOf>
(1540)              <owl:Restriction>
(1541)                  <owl:onProperty rdf:resource="#prueftVorliegenTatbestand"/>
(1542)                  <owl:someValuesFrom rdf:resource="#Tatbestand"/>
(1543)              </owl:Restriction>
(1544)          </rdfs:subClassOf>
(1545)          <rdfs:subClassOf>
(1546)              <owl:Restriction>
(1547)                  <owl:onProperty
      rdf:resource="#wirdAngewandtAufSachverhalt"/>
(1548)                  <owl:someValuesFrom rdf:resource="#Sachverhalt"/>
(1549)              </owl:Restriction>
(1550)          </rdfs:subClassOf>
(1551)          <rdfs:comment rdf:datatype="&xsd;string"
(1552)              >Ein Schritt der Rechtsanwendung, in dessen Rahmen versucht
      wird, einen vorliegenden Sachverhalt unter den Tatbestand zu subsumieren.
      Vollzieht sich f&#252;r jedes Tatbestandsmerkmal, indem jeweils die zu einem
      Merkmal/einer Bedingung des Tatbestandes korrespondierende Eigenschaft des
      Sachverhaltes gegen&#252;bergestellt und gepr&#252;ft wird, ob die
      Sachverhaltseigenschaft den Bedingungen gen&#252;gt, die durch das
      Tatbestandsmerkmal formuliert sind. Der Schritt endet mit der Entscheidung,
      ob alle Merkmale des Tatbestandes durch den Sachverhalt erf&#252;llt sind
      und damit die Subsumtion gelungen ist, oder ob dies nicht der Fall ist, weil
      ein Tatbestandsmerkmal keine Entsprechung hat bzw. die Pr&#252;fung ergibt,
      dass ein Merkmal durch den Sachverhalt nicht erf&#252;llt
      ist.</rdfs:comment>
(1553)      </owl:Class>
(1554)
(1555)
(1556)
(1557)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Tatbestand -->
(1558)
(1559)      <owl:Class rdf:about="#Tatbestand">
(1560)          <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
(1561)          <rdfs:subClassOf rdf:resource="#Rechtssatzbestandteil"/>
(1562)          <rdfs:subClassOf>
(1563)              <owl:Restriction>
(1564)                  <owl:onProperty rdf:resource="#impliziertRechtsfolge"/>
(1565)                  <owl:someValuesFrom rdf:resource="#Rechtsfolge"/>
(1566)              </owl:Restriction>
(1567)          </rdfs:subClassOf>
(1568)          <rdfs:subClassOf>
(1569)              <owl:Restriction>
(1570)                  <owl:onProperty rdf:resource="#hatTatbestandsBedingung"/>
(1571)                  <owl:someValuesFrom rdf:resource="#Tatbestandsbedingung"/>
(1572)              </owl:Restriction>
(1573)          </rdfs:subClassOf>
(1574)          <rdfs:comment rdf:datatype="&xsd;string"
(1575)              >Beschreibung bestimmter abstrakter Merkmale und
      Voraussetzungen, die ein Rechtssatz f&#252;r das Eintreten einer Rechtsfolge
      vorsieht. Der Tatbestand kann als eine Menge von Sachverhalten aufgefasst
      werden, zu der alle Sachverhalte geh&#246;ren, deren Eigenschaften den
      Bedingungen der Tatbestandsmerkmale gen&#252;gen.</rdfs:comment>
(1576)      </owl:Class>
(1577)
(1578)
(1579)
(1580)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Tatbestandsbedingung -->
(1581)
(1582)      <owl:Class rdf:about="#Tatbestandsbedingung">
(1583)          <rdfs:subClassOf rdf:resource="#RechtsHandlungsgegenstand"/>
(1584)          <rdfs:subClassOf rdf:resource="#Rechtssatzbestandteil"/>
(1585)          <rdfs:subClassOf>
```

```
(1586)              <owl:Restriction>
(1587)                  <owl:onProperty rdf:resource="#istBedingungVonTatbestand"/>
(1588)                  <owl:someValuesFrom rdf:resource="#Tatbestand"/>
(1589)              </owl:Restriction>
(1590)          </rdfs:subClassOf>
(1591)          <rdfs:comment rdf:datatype="&xsd;string"
(1592)              >Genau ein abstraktes Merkmal bzw. eine Voraussetzung, die
      f&#252;r das Vorliegen eines Tatbestandes erf&#252;llt sein m&#252;ssen,
      werden durch eine Instanz dieses Konzepts ausgedr&#252;ckt.</rdfs:comment>
(1593)      </owl:Class>
(1594)
(1595)
(1596)
(1597)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Text -->
(1598)
(1599)      <owl:Class rdf:about="#Text">
(1600)          <rdfs:label rdf:datatype="&xsd;string">Text</rdfs:label>
(1601)          <rdfs:subClassOf rdf:resource="#TextKonzept"/>
(1602)          <rdfs:subClassOf>
(1603)              <owl:Restriction>
(1604)                  <owl:onProperty rdf:resource="#bestehtAusTextbestandteil"/>
(1605)                  <owl:someValuesFrom
      rdf:resource="#AllgemeinerTextbestandteil"/>
(1606)              </owl:Restriction>
(1607)          </rdfs:subClassOf>
(1608)          <rdfs:comment rdf:datatype="&xsd;string"
(1609)              >Eine abgegrenzte, kommunikative Einheit, die aus einer
      zusammenh&#228;ngenden Folge von sprachlichen Elementen (Worten,
      S&#228;tzen, Abs&#228;tzen, Abschnitten) besteht und ein Thema
      behandelt.</rdfs:comment>
(1610)      </owl:Class>
(1611)
(1612)
(1613)
(1614)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#TextKonzept -->
(1615)
(1616)      <owl:Class rdf:about="#TextKonzept">
(1617)          <rdfs:label>TextKonzept</rdfs:label>
(1618)          <rdfs:subClassOf rdf:resource="#DomainConcept"/>
(1619)          <rdfs:comment rdf:datatype="&xsd;string"
(1620)              >Textkonzept ist die abstrakte Oberklasse f&#252;r Konzepte,
      die Normtexte und ihre sprachlichen Bestandteile, sowie die in ihnen
      enthaltenen Rechtss&#228;tze und deren Bestandteile
      repr&#228;sentieren.</rdfs:comment>
(1621)      </owl:Class>
(1622)
(1623)
(1624)
(1625)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#Textbestandteil -->
(1626)
(1627)      <owl:Class rdf:about="#Textbestandteil">
(1628)          <rdfs:label rdf:datatype="&xsd;string">Textbestandteil</rdfs:label>
(1629)          <rdfs:subClassOf rdf:resource="#TextKonzept"/>
(1630)          <rdfs:comment rdf:datatype="&xsd;string"
(1631)              >Abstrakte Oberklasse f&#252;r die Bestandteile eines
      Textes.</rdfs:comment>
(1632)      </owl:Class>
(1633)
(1634)
(1635)
(1636)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#UnvollstaendigerRechtssatz -->
(1637)
(1638)      <owl:Class rdf:about="#UnvollstaendigerRechtssatz">
(1639)          <rdfs:label rdf:datatype="&xsd;string"
(1640)              >UnvollstaendigerRechtssatz</rdfs:label>
(1641)          <rdfs:subClassOf rdf:resource="#Rechtssatz"/>
(1642)          <rdfs:subClassOf>
(1643)              <owl:Restriction>
(1644)                  <owl:onProperty rdf:resource="#schaenktEinRechtssatz"/>
(1645)                  <owl:allValuesFrom rdf:resource="#Rechtssatz"/>
(1646)              </owl:Restriction>
```

```
(1647)          </rdfs:subClassOf>
(1648)          <rdfs:subClassOf>
(1649)              <owl:Restriction>
(1650)                  <owl:onProperty
      rdf:resource="#beschreibtTatbestandUnvollstaendig"/>
(1651)                  <owl:someValuesFrom rdf:resource="#Tatbestand"/>
(1652)              </owl:Restriction>
(1653)          </rdfs:subClassOf>
(1654)          <rdfs:subClassOf>
(1655)              <owl:Restriction>
(1656)                  <owl:onProperty
      rdf:resource="#beschreibtRechtsfolgeUnvollstaendig"/>
(1657)                  <owl:someValuesFrom rdf:resource="#Rechtsfolge"/>
(1658)              </owl:Restriction>
(1659)          </rdfs:subClassOf>
(1660)          <rdfs:subClassOf>
(1661)              <owl:Restriction>
(1662)                  <owl:onProperty rdf:resource="#erlaeutertRechtssatz"/>
(1663)                  <owl:allValuesFrom rdf:resource="#Rechtssatz"/>
(1664)              </owl:Restriction>
(1665)          </rdfs:subClassOf>
(1666)          <rdfs:subClassOf>
(1667)              <owl:Restriction>
(1668)                  <owl:onProperty rdf:resource="#erlaeutertRechtssatz"/>
(1669)                  <owl:onClass rdf:resource="#Rechtssatz"/>
(1670)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
(1671)              </owl:Restriction>
(1672)          </rdfs:subClassOf>
(1673)          <rdfs:subClassOf>
(1674)              <owl:Restriction>
(1675)                  <owl:onProperty rdf:resource="#verweistAufRechtssatz"/>
(1676)                  <owl:allValuesFrom rdf:resource="#Rechtssatz"/>
(1677)              </owl:Restriction>
(1678)          </rdfs:subClassOf>
(1679)          <rdfs:subClassOf>
(1680)              <owl:Restriction>
(1681)                  <owl:onProperty rdf:resource="#schaenktEinRechtssatz"/>
(1682)                  <owl:onClass rdf:resource="#Rechtssatz"/>
(1683)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
(1684)              </owl:Restriction>
(1685)          </rdfs:subClassOf>
(1686)          <rdfs:subClassOf>
(1687)              <owl:Restriction>
(1688)                  <owl:onProperty rdf:resource="#verweistAufRechtssatz"/>
(1689)                  <owl:onClass rdf:resource="#Rechtssatz"/>
(1690)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
(1691)              </owl:Restriction>
(1692)          </rdfs:subClassOf>
(1693)          <owl:disjointWith rdf:resource="#VollstaendigerRechtssatz"/>
(1694)          <rdfs:comment rdf:datatype="&xsd;string"
(1695)              >Ein unvollst&#228;ndiger Rechtssatz ist ein sprachlich
      vollst&#228;ndiger Satz, der &#252;ber einen normativen Sinn verf&#252;gt.
      Er weist im Unterschied zu vollst&#228;ndigen Rechtss&#228;tzen nicht alle
      Elemente eines Rechtssatzes auf und wirkt stattdessen auf die Bestandteile
      eines anderen Rechtssatzes erl&#228;uternd, einschr&#228;nkend oder
      verweisend.</rdfs:comment>
(1696)      </owl:Class>
(1697)
(1698)
(1699)
(1700)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VerwaltungsKonzept -->
(1701)
(1702)      <owl:Class rdf:about="#VerwaltungsKonzept">
(1703)          <rdfs:label rdf:datatype="&xsd;string"
(1704)              >Verwaltungskonzept</rdfs:label>
(1705)          <rdfs:subClassOf rdf:resource="#DomainConcept"/>
(1706)          <rdfs:subClassOf>
(1707)              <owl:Restriction>
(1708)                  <owl:onProperty rdf:resource="#Beschreibung"/>
(1709)                  <owl:someValuesFrom rdf:resource="&xsd;string"/>
(1710)              </owl:Restriction>
```

```
(1711)          </rdfs:subClassOf>
(1712)          <rdfs:subClassOf>
(1713)              <owl:Restriction>
(1714)                  <owl:onProperty
      rdf:resource="#konkretisiertRechtsKonzept"/>
(1715)                  <owl:allValuesFrom rdf:resource="#RechtsKonzept"/>
(1716)              </owl:Restriction>
(1717)          </rdfs:subClassOf>
(1718)          <rdfs:subClassOf>
(1719)              <owl:Restriction>
(1720)                  <owl:onProperty rdf:resource="#Quelle"/>
(1721)                  <owl:someValuesFrom rdf:resource="&xsd;string"/>
(1722)              </owl:Restriction>
(1723)          </rdfs:subClassOf>
(1724)          <rdfs:subClassOf>
(1725)              <owl:Restriction>
(1726)                  <owl:onProperty rdf:resource="#Arbeitsnotiz"/>
(1727)                  <owl:someValuesFrom rdf:resource="&xsd;string"/>
(1728)              </owl:Restriction>
(1729)          </rdfs:subClassOf>
(1730)          <rdfs:comment rdf:datatype="&xsd;string"
(1731)              >Verwaltungskonzept ist die Abstraktion und Formalisierung der
      Konzepte des Verwaltungshandelns</rdfs:comment>
(1732)      </owl:Class>
(1733)
(1734)
(1735)
(1736)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VollstaendigerRechtssatz -->
(1737)
(1738)      <owl:Class rdf:about="#VollstaendigerRechtssatz">
(1739)          <rdfs:label rdf:datatype="&xsd;string"
(1740)              >VollstaendigerRechtssatz</rdfs:label>
(1741)          <rdfs:subClassOf rdf:resource="#Rechtssatz"/>
(1742)          <rdfs:subClassOf>
(1743)              <owl:Restriction>
(1744)                  <owl:onProperty rdf:resource="#beschreibtRechtsfolge"/>
(1745)                  <owl:allValuesFrom rdf:resource="#Rechtsfolge"/>
(1746)              </owl:Restriction>
(1747)          </rdfs:subClassOf>
(1748)          <rdfs:subClassOf>
(1749)              <owl:Restriction>
(1750)                  <owl:onProperty rdf:resource="#beschreibtTatbestand"/>
(1751)                  <owl:allValuesFrom rdf:resource="#Tatbestand"/>
(1752)              </owl:Restriction>
(1753)          </rdfs:subClassOf>
(1754)          <rdfs:comment rdf:datatype="&xsd;string"
(1755)              >Ein vollst&#228;ndiger Rechtssatz weist alle Elemente eines
      Rechtssatzes auf. Er verkn&#252;pft einen generell und abstrakt
      beschriebenen Tatbestand, der auf eine Vielzahl konkreter Sachverhalte
      zutreffen kann, mit einer oder mehreren ebenso abstrakt beschriebenen
      Rechtsfolgen.</rdfs:comment>
(1756)      </owl:Class>
(1757)
(1758)
(1759)
(1760)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAdressat -->
(1761)
(1762)      <owl:Class rdf:about="#VwAdressat">
(1763)          <rdfs:label rdf:datatype="&xsd;string">VwAdressat</rdfs:label>
(1764)          <rdfs:subClassOf rdf:resource="#Adressat"/>
(1765)          <rdfs:subClassOf rdf:resource="#VwAkteur"/>
(1766)          <rdfs:comment rdf:datatype="&xsd;string"
(1767)              >Adressat der Rechtsvorschrift, die dem Verwaltungshandeln
      zugrunde liegt, und ihm ihm Tun oder Unterlassen vorschreibt oder ihm ein
      bestimmtes Recht einr&#228;umt oder entzieht.</rdfs:comment>
(1768)      </owl:Class>
(1769)
(1770)
(1771)
(1772)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAkteur -->
(1773)
(1774)      <owl:Class rdf:about="#VwAkteur">
```

```
(1775)          <rdfs:subClassOf rdf:resource="#VerwaltungsKonzept"/>
(1776)          <rdfs:comment
(1777)              >Repr&#228;sentiert eine Person oder Organisation, die
      Tr&#228;ger &#246;ffentlicher Aufgaben ist und an der Rechtsanwendung
      beteiligt ist und/oder diese durch- bzw. ausf&#252;hrt.</rdfs:comment>
(1778)      </owl:Class>
(1779)
(1780)
(1781)
(1782)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAkteursErweiterung -->
(1783)
(1784)      <owl:Class rdf:about="#VwAkteursErweiterung">
(1785)          <rdfs:label rdf:datatype="&xsd;string"
(1786)              >VwAkteurErweiterung</rdfs:label>
(1787)          <rdfs:subClassOf rdf:resource="#VerwaltungsKonzept"/>
(1788)          <rdfs:comment rdf:datatype="&xsd;string"
(1789)              >Repr&#228;sentiert eine Person oder Organisation, die
      Tr&#228;ger &#246;ffentlicher Aufgaben ist und an der Erbringung von
      B&#252;rgerdiensten bei intensiver Unterst&#252;tzung durch
      Informationstechnik beteiligt ist.</rdfs:comment>
(1790)      </owl:Class>
(1791)
(1792)
(1793)
(1794)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAnliegen -->
(1795)
(1796)      <owl:Class rdf:about="#VwAnliegen">
(1797)          <rdfs:label>Anliegen</rdfs:label>
(1798)          <rdfs:subClassOf
      rdf:resource="#VwHandlungsgegenstandsErweiterung"/>
(1799)          <rdfs:comment rdf:datatype="&xsd;string"
(1800)              >Aus einer Lebenslage ggf. unter Verwendung von Hinweisen und
      Informationen konkretisierter Bedarf eines B&#252;rgers an einer oder
      mehreren Verwaltungsleistungen.</rdfs:comment>
(1801)      </owl:Class>
(1802)
(1803)
(1804)
(1805)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAntrag -->
(1806)
(1807)      <owl:Class rdf:about="#VwAntrag">
(1808)          <rdfs:subClassOf rdf:resource="#VwHandlungsausloeser"/>
(1809)          <rdfs:subClassOf>
(1810)              <owl:Restriction>
(1811)                  <owl:onProperty rdf:resource="#beantragteLeistung"/>
(1812)                  <owl:allValuesFrom rdf:resource="#VwLeistung"/>
(1813)              </owl:Restriction>
(1814)          </rdfs:subClassOf>
(1815)          <rdfs:subClassOf>
(1816)              <owl:Restriction>
(1817)                  <owl:onProperty rdf:resource="#strebtAnRechtsfolge"/>
(1818)                  <owl:someValuesFrom rdf:resource="#VwRechtsfolge"/>
(1819)              </owl:Restriction>
(1820)          </rdfs:subClassOf>
(1821)          <rdfs:subClassOf>
(1822)              <owl:Restriction>
(1823)                  <owl:onProperty rdf:resource="#hatAntragsteller"/>
(1824)                  <owl:allValuesFrom rdf:resource="#VwAntragsteller"/>
(1825)              </owl:Restriction>
(1826)          </rdfs:subClassOf>
(1827)          <rdfs:comment rdf:datatype="&xsd;string"
(1828)              >Der Antrag pr&#228;sentiert eine spezielle Form des
      Handlungsausl&#246;ser von Verwaltungshandeln.</rdfs:comment>
(1829)      </owl:Class>
(1830)
(1831)
(1832)
(1833)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAntragsteller -->
(1834)
(1835)      <owl:Class rdf:about="#VwAntragsteller">
(1836)          <rdfs:label>VwAntragsteller</rdfs:label>
```

```
(1837)          <rdfs:subClassOf rdf:resource="#VwAkteur"/>
(1838)          <rdfs:subClassOf rdf:resource="#VwNachfrager"/>
(1839)          <rdfs:subClassOf>
(1840)              <owl:Restriction>
(1841)                  <owl:onProperty rdf:resource="#hatLebenslage"/>
(1842)                  <owl:allValuesFrom rdf:resource="#VwLebenslage"/>
(1843)              </owl:Restriction>
(1844)          </rdfs:subClassOf>
(1845)          <rdfs:subClassOf>
(1846)              <owl:Restriction>
(1847)                  <owl:onProperty rdf:resource="#stelltAntrag"/>
(1848)                  <owl:allValuesFrom rdf:resource="#VwAntrag"/>
(1849)              </owl:Restriction>
(1850)          </rdfs:subClassOf>
(1851)          <rdfs:comment
(1852)              >Der Antragsteller ist ein Akteur, dessen Antrag das
      Verwaltungshandeln ausgel&#246;st hat.</rdfs:comment>
(1853)      </owl:Class>
(1854)
(1855)
(1856)
(1857)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAnwender -->
(1858)
(1859)      <owl:Class rdf:about="#VwAnwender">
(1860)          <rdfs:subClassOf rdf:resource="#Anwender"/>
(1861)          <rdfs:subClassOf rdf:resource="#VwAkteur"/>
(1862)          <rdfs:subClassOf>
(1863)              <owl:Restriction>
(1864)                  <owl:onProperty
      rdf:resource="#fuehrtDurchVerwaltungshandlung"/>
(1865)                  <owl:someValuesFrom
      rdf:resource="#VwVerfahrenseinleitung"/>
(1866)              </owl:Restriction>
(1867)          </rdfs:subClassOf>
(1868)          <rdfs:subClassOf>
(1869)              <owl:Restriction>
(1870)                  <owl:onProperty
      rdf:resource="#fuehrtDurchVerwaltungshandlung"/>
(1871)                  <owl:someValuesFrom rdf:resource="#VwNachbearbeitung"/>
(1872)              </owl:Restriction>
(1873)          </rdfs:subClassOf>
(1874)          <rdfs:comment rdf:datatype="&xsd;string"
(1875)              >Akteur in einer Verwaltungsorganisation oder
      Verwaltungsorganisation die Recht anwendet, die dazu notwendigen
      Verwaltungshandlungen durchf&#252;hrt und dabei die relevanten
      Handlungsgegenst&#228;nde bearbeitet.</rdfs:comment>
(1876)      </owl:Class>
(1877)
(1878)
(1879)
(1880)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAnwenderBackOffice -->
(1881)
(1882)      <owl:Class rdf:about="#VwAnwenderBackOffice">
(1883)          <rdfs:label
(1884)              >VwAnwenderBackOffice</rdfs:label>
(1885)          <rdfs:subClassOf rdf:resource="#VwAkteursErweiterung"/>
(1886)          <rdfs:subClassOf rdf:resource="#VwAnwender"/>
(1887)      </owl:Class>
(1888)
(1889)
(1890)
(1891)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAnwenderFrontOffice -->
(1892)
(1893)      <owl:Class rdf:about="#VwAnwenderFrontOffice">
(1894)          <rdfs:label
(1895)              >VwAnwenderFrontOffice</rdfs:label>
(1896)          <rdfs:subClassOf rdf:resource="#VwAkteursErweiterung"/>
(1897)          <rdfs:subClassOf rdf:resource="#VwAnwender"/>
(1898)          <rdfs:comment rdf:datatype="&xsd;string"
(1899)              >Ein spezieller Anwender, der Handlungen durchf&#252;hrt, die
      zur Erstellung einer Leistung durchgef&#252;hrt werden und einen direkten
      Kontakt mit dem Nachfrager erfordern.</rdfs:comment>
```

```
(1900)      </owl:Class>
(1901)
(1902)
(1903)
(1904)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAnzuhoerender -->
(1905)
(1906)      <owl:Class rdf:about="#VwAnzuhoerender">
(1907)          <rdfs:subClassOf rdf:resource="#VwBeteiligterBetroffener"/>
(1908)          <rdfs:comment
(1909)              >Repr&#228;sentiert einen Beteiligten am Verwaltungshandeln,
      der in diesem Rahmen anzuh&#246;ren ist. (Sachverst&#228;ndige, die ein
      Gutachten oder Zeugen, die eine Aussage in das Verwaltungshandeln
      einbringen, k&#246;nnen beispielsweise Anzuh&#246;rende
      sein.)</rdfs:comment>
(1910)      </owl:Class>
(1911)
(1912)
(1913)
(1914)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAusfuehrender -->
(1915)
(1916)      <owl:Class rdf:about="#VwAusfuehrender">
(1917)          <rdfs:subClassOf rdf:resource="#VwAkteur"/>
(1918)          <rdfs:subClassOf>
(1919)              <owl:Restriction>
(1920)                  <owl:onProperty rdf:resource="#fuehrtAus"/>
(1921)                  <owl:someValuesFrom rdf:resource="#VwLeistungserbringung"/>
(1922)              </owl:Restriction>
(1923)          </rdfs:subClassOf>
(1924)          <rdfs:comment
(1925)              >Umfassen die Verwaltungshandlungen auch die tats&#228;chliche
      Ausf&#252;hrung der getroffenen Entscheidung, so wird diese von einer Person
      oder Organisation ausgef&#252;hrt, f&#252;r die im Rahmen dieser Arbeit das
      Konzept VwAusf&#252;hrender eingef&#252;hrt wird.</rdfs:comment>
(1926)      </owl:Class>
(1927)
(1928)
(1929)
(1930)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAusfuehrenderBackOffice -->
(1931)
(1932)      <owl:Class rdf:about="#VwAusfuehrenderBackOffice">
(1933)          <rdfs:label
(1934)              >VwAusfuehrenderBackOffice</rdfs:label>
(1935)          <rdfs:subClassOf rdf:resource="#VwAkteursErweiterung"/>
(1936)          <rdfs:subClassOf rdf:resource="#VwAusfuehrender"/>
(1937)          <rdfs:comment rdf:datatype="&xsd;string"
(1938)              >Ein spezieller Anwender, der Handlungen zur tats&#228;chlichen
      Umsetzung einer Leistung ohne direkten Nachfragerkontakt
      ausf&#252;hrt.</rdfs:comment>
(1939)      </owl:Class>
(1940)
(1941)
(1942)
(1943)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAusfuehrenderFrontOffice -->
(1944)
(1945)      <owl:Class rdf:about="#VwAusfuehrenderFrontOffice">
(1946)          <rdfs:label
(1947)              >VwAusfuehrenderFrontOffice</rdfs:label>
(1948)          <rdfs:subClassOf rdf:resource="#VwAkteursErweiterung"/>
(1949)          <rdfs:subClassOf rdf:resource="#VwAusfuehrender"/>
(1950)          <rdfs:comment rdf:datatype="&xsd;string"
(1951)              >Ein spezieller Anwender, der Handlungen zur tats&#228;chlichen
      Umsetzung einer Leistung in direktem Kontakt mit dem Nachfrager
      ausf&#252;hrt.</rdfs:comment>
(1952)      </owl:Class>
(1953)
(1954)
(1955)
(1956)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwAuswahlkriterium -->
(1957)
(1958)      <owl:Class rdf:about="#VwAuswahlkriterium">
```

```
(1959)          <rdfs:label rdf:datatype="&xsd;string"
(1960)              >VwAuswahlkriterien</rdfs:label>
(1961)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(1962)          <rdfs:comment rdf:datatype="&xsd;string"
(1963)              >Das Auswahlkriterium im Verwaltungshandeln repr&#228;sentiert
      Kriterien, die in Rechtsvorschriften dokumentiert sind und bei Vorliegen der
      Voraussetzungen f&#252;r die Auswahl einer von mehreren m&#246;glichen
      Rechtsfolgen herangezogen werden. Dieses Konzept hat keine Entsprechung in
      den  Konzepten der Rechtsanwendung.</rdfs:comment>
(1964)      </owl:Class>
(1965)
(1966)
(1967)
(1968)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwBeteiligterBetroffener -->
(1969)
(1970)      <owl:Class rdf:about="#VwBeteiligterBetroffener">
(1971)          <rdfs:subClassOf rdf:resource="#VwAkteur"/>
(1972)          <rdfs:subClassOf>
(1973)              <owl:Restriction>
(1974)                  <owl:onProperty rdf:resource="#istBeteiligtAn"/>
(1975)                  <owl:someValuesFrom rdf:resource="#VwVerfahren"/>
(1976)              </owl:Restriction>
(1977)          </rdfs:subClassOf>
(1978)          <rdfs:comment
(1979)              >Repr&#228;sentiert einen VerwaltungsAkteur der am oder vom
      Verwaltungshandeln beteiligt bzw. betroffen ist.</rdfs:comment>
(1980)      </owl:Class>
(1981)
(1982)
(1983)
(1984)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwBeteiligungMitwirkung -->
(1985)
(1986)      <owl:Class rdf:about="#VwBeteiligungMitwirkung">
(1987)          <rdfs:label
(1988)              >BeteiligungMitwirkung</rdfs:label>
(1989)          <rdfs:subClassOf
      rdf:resource="#VwHandlungsgegenstandsErweiterung"/>
(1990)          <rdfs:comment rdf:datatype="&xsd;string"
(1991)              >Fasst Angaben zu den zu beteiligenden bzw. mitzuwirkenden
      Akteure das daraus resultierende Beteiligung- bzw. Mitwirkungsergebnis
      zusammen.</rdfs:comment>
(1992)      </owl:Class>
(1993)
(1994)
(1995)
(1996)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwBetroffenerDritter -->
(1997)
(1998)      <owl:Class rdf:about="#VwBetroffenerDritter">
(1999)          <rdfs:subClassOf rdf:resource="#VwBeteiligterBetroffener"/>
(2000)          <rdfs:comment
(2001)              >Der Betroffene Dritte wird absehbar durch das
      Verwaltungshandeln in seinen Rechten betroffen (z.B. eingeschr&#228;nkt)
      werden..</rdfs:comment>
(2002)      </owl:Class>
(2003)
(2004)
(2005)
(2006)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwBeurteilungsspielraum -->
(2007)
(2008)      <owl:Class rdf:about="#VwBeurteilungsspielraum">
(2009)          <rdfs:label rdf:datatype="&xsd;string"
(2010)              >VwBeurteilungsspielraum</rdfs:label>
(2011)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2012)          <rdfs:comment rdf:datatype="&xsd;string"
(2013)              >Entscheidungsspielraum der sich w&#228;hrend der Beurteilung
      von Sachverhaltseigenschaften vor dem Hintergrund nicht vollst&#228;ndig
      oder hinreichend definierter Voraussetzungsbedingungen
      ergibt.</rdfs:comment>
(2014)      </owl:Class>
(2015)
(2016)
```

```
(2017)
(2018)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwBewusstmacher -->
(2019)
(2020)      <owl:Class rdf:about="#VwBewusstmacher">
(2021)          <rdfs:label>VwBewusstmacher</rdfs:label>
(2022)          <rdfs:subClassOf rdf:resource="#VwAkteursErweiterung"/>
(2023)          <rdfs:subClassOf>
(2024)              <owl:Restriction>
(2025)                  <owl:onProperty rdf:resource="#stoestAnBewusstwedung"/>
(2026)                  <owl:someValuesFrom rdf:resource="#VwBewusstwerdung"/>
(2027)              </owl:Restriction>
(2028)          </rdfs:subClassOf>
(2029)          <rdfs:comment rdf:datatype="&xsd;string"
(2030)              >Ein Aktuer, der personalisierte Hinweise und Informationen
      abgibt, die einen Akteur dazu anregen, ein konkretes Anliegen zu
      entwickeln.</rdfs:comment>
(2031)      </owl:Class>
(2032)
(2033)
(2034)
(2035)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwBewusstwerdung -->
(2036)
(2037)      <owl:Class rdf:about="#VwBewusstwerdung">
(2038)          <rdfs:label>Bewusstwerdung</rdfs:label>
(2039)          <rdfs:subClassOf rdf:resource="#VwHandlungsErweiterung"/>
(2040)          <rdfs:comment rdf:datatype="&xsd;string"
(2041)              >Handlungen die dazu f&#252;hren, dass von au&#223;en angeregt
      oder aufgrund eigener Erkenntnis der Nachfrager ein Anliegen
      entwickelt.</rdfs:comment>
(2042)      </owl:Class>
(2043)
(2044)
(2045)
(2046)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwErgebnismitteilung -->
(2047)
(2048)      <owl:Class rdf:about="#VwErgebnismitteilung">
(2049)          <rdfs:label
(2050)              >VwErgebnismitteilung</rdfs:label>
(2051)          <rdfs:subClassOf rdf:resource="#Mitteilung"/>
(2052)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2053)      </owl:Class>
(2054)
(2055)
(2056)
(2057)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwErmessen -->
(2058)
(2059)      <owl:Class rdf:about="#VwErmessen">
(2060)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2061)          <rdfs:comment rdf:datatype="&xsd;string"
(2062)              >Entscheidungsspielraum, der sich bei der Wahl einer Rechtfolge
      und/oder der Ausgestaltung der resultierenden
      Verwaltungsleistungseigenschaft ergibt.</rdfs:comment>
(2063)          <rdfs:comment rdf:datatype="&xsd;string">VwErmessen</rdfs:comment>
(2064)      </owl:Class>
(2065)
(2066)
(2067)
(2068)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwErmessensausuebung -->
(2069)
(2070)      <owl:Class rdf:about="#VwErmessensausuebung">
(2071)          <rdfs:subClassOf rdf:resource="#Rechtsfolgenbestimmung"/>
(2072)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2073)          <rdfs:subClassOf>
(2074)              <owl:Restriction>
(2075)                  <owl:onProperty rdf:resource="#nutztErmessen"/>
(2076)                  <owl:someValuesFrom rdf:resource="#VwErmessen"/>
(2077)              </owl:Restriction>
(2078)          </rdfs:subClassOf>
(2079)          <rdfs:comment rdf:datatype="&xsd;string"
```

```
(2080)              >Eine Handlung die von der Verwaltung in Falle von ungebundenem
      Verwaltungshandeln durchgef&#252;hrt wird, um (entsprechend dem Zweck der
      Erm&#228;chtigung und in den gesetzlichen Grenzen) zu entscheiden, ob
      und/oder wie gehandelt werden soll.Das Konzept ist eine Spezialisierung des
      Konzepts der Rechtsfolgenbestimmung.</rdfs:comment>
(2081)      </owl:Class>
(2082)
(2083)
(2084)
(2085)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwHandlung -->
(2086)
(2087)      <owl:Class rdf:about="#VwHandlung">
(2088)          <rdfs:subClassOf rdf:resource="#VerwaltungsKonzept"/>
(2089)          <rdfs:subClassOf>
(2090)              <owl:Restriction>
(2091)                  <owl:onProperty rdf:resource="#HandlungNachher"/>
(2092)                  <owl:someValuesFrom rdf:resource="&xsd;string"/>
(2093)              </owl:Restriction>
(2094)          </rdfs:subClassOf>
(2095)          <rdfs:subClassOf>
(2096)              <owl:Restriction>
(2097)                  <owl:onProperty rdf:resource="#HandlungVorher"/>
(2098)                  <owl:someValuesFrom rdf:resource="&xsd;string"/>
(2099)              </owl:Restriction>
(2100)          </rdfs:subClassOf>
(2101)          <rdfs:subClassOf>
(2102)              <owl:Restriction>
(2103)                  <owl:onProperty rdf:resource="#HandlungsZiel"/>
(2104)                  <owl:someValuesFrom rdf:resource="&xsd;string"/>
(2105)              </owl:Restriction>
(2106)          </rdfs:subClassOf>
(2107)          <rdfs:comment rdf:datatype="&xsd;string"
(2108)              >Abstraktion aller Handlungen der &#246;ffentlichen Verwaltung
      zur Erbringung von Verwaltungslesitungen durch Anwendung von Recht auf den
      konkret vorliegenden Einzelfall.</rdfs:comment>
(2109)      </owl:Class>
(2110)
(2111)
(2112)
(2113)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwHandlungsErweiterung -->
(2114)
(2115)      <owl:Class rdf:about="#VwHandlungsErweiterung">
(2116)          <rdfs:label rdf:datatype="&xsd;string"
(2117)              >VwHandlungErweiterung</rdfs:label>
(2118)          <rdfs:subClassOf rdf:resource="#VerwaltungsKonzept"/>
(2119)          <rdfs:comment rdf:datatype="&xsd;string"
(2120)              >Abstraktion aller Handlungen der &#246;ffentlichen Verwaltung
      die zur der Erbringung von B&#252;rgerdiensten bei intensiver
      Unterst&#252;tzung durch Informationstechnik im konkret vorliegenden
      Einzelfall.</rdfs:comment>
(2121)      </owl:Class>
(2122)
(2123)
(2124)
(2125)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwHandlungsausloeser -->
(2126)
(2127)      <owl:Class rdf:about="#VwHandlungsausloeser">
(2128)          <rdfs:label rdf:datatype="&xsd;string"
(2129)              >VwHandlungsausl&#246;ser</rdfs:label>
(2130)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2131)          <rdfs:comment rdf:datatype="&xsd;string"
(2132)              >Der Handlungsausl&#246;ser repr&#228;sentiert den
      Ausl&#246;ser des Verwaltungshandelns, d.h. das Auftreten eines rechtlich
      relevanten Ereignisses, das Verwaltungshandeln ausl&#246;st (z.B. ein Antrag
      oder T&#228;tigwerden der Verwaltung von Amtswegen).</rdfs:comment>
(2133)      </owl:Class>
(2134)
(2135)
(2136)
(2137)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwHandlungsgegenstand -->
(2138)
```

```
(2139)      <owl:Class rdf:about="#VwHandlungsgegenstand">
(2140)          <rdfs:label rdf:datatype="&xsd;string"
(2141)              >VwHandlungsgegenstand</rdfs:label>
(2142)          <rdfs:subClassOf rdf:resource="#VerwaltungsKonzept"/>
(2143)          <rdfs:comment
(2144)              >Der VerwaltungsHandlungsgegenstand repr&#228;sentiert
      abstrakte Konzepte und konkrete Dinge, die im Rahmen von Rechtshandlungen
      der &#246;ffentlichen Verwaltung erzeugt, verwendet und/oder manipuliert
      werden.</rdfs:comment>
(2145)      </owl:Class>
(2146)
(2147)
(2148)
(2149)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwHandlungsgegenstandsErweiterung -->
(2150)
(2151)      <owl:Class rdf:about="#VwHandlungsgegenstandsErweiterung">
(2152)          <rdfs:label rdf:datatype="&xsd;string"
(2153)              >VwHandlungsgegenstandErweiterung</rdfs:label>
(2154)          <rdfs:subClassOf rdf:resource="#VerwaltungsKonzept"/>
(2155)          <rdfs:comment rdf:datatype="&xsd;string"
(2156)              >Das Konzept der VerwaltungsHandlungsgegenstandsErweiterung
      repr&#228;sentiert abstrakte Konzepte und konkrete Dinge, die zur der
      Erbringung von B&#252;rgerdiensten bei intensiver Unterst&#252;tzung durch
      Informationstechnik von einem Tr&#228;ger von &#246;ffentlicher Aufgaben
      (d.h. einer &#246;ffentlichen Verwaltung) erzeugt, verwendet und/oder
      manipuliert werden.</rdfs:comment>
(2157)      </owl:Class>
(2158)
(2159)
(2160)
(2161)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwHinweisInformation -->
(2162)
(2163)      <owl:Class rdf:about="#VwHinweisInformation">
(2164)          <rdfs:subClassOf
      rdf:resource="#VwHandlungsgegenstandsErweiterung"/>
(2165)          <rdfs:subClassOf>
(2166)              <owl:Restriction>
(2167)                  <owl:onProperty rdf:resource="#zugehoerigeLebenslage"/>
(2168)                  <owl:someValuesFrom rdf:resource="#VwLebenslage"/>
(2169)              </owl:Restriction>
(2170)          </rdfs:subClassOf>
(2171)          <rdfs:subClassOf>
(2172)              <owl:Restriction>
(2173)                  <owl:onProperty rdf:resource="#notwendigeRechtsfolge"/>
(2174)                  <owl:someValuesFrom rdf:resource="#VwRechtsfolge"/>
(2175)              </owl:Restriction>
(2176)          </rdfs:subClassOf>
(2177)          <rdfs:subClassOf>
(2178)              <owl:Restriction>
(2179)                  <owl:onProperty rdf:resource="#moeglicheRechtsfolge"/>
(2180)                  <owl:someValuesFrom rdf:resource="#VwRechtsfolge"/>
(2181)              </owl:Restriction>
(2182)          </rdfs:subClassOf>
(2183)          <rdfs:subClassOf>
(2184)              <owl:Restriction>
(2185)                  <owl:onProperty rdf:resource="#beziehtSichAufLeistung"/>
(2186)                  <owl:allValuesFrom rdf:resource="#VwLeistung"/>
(2187)              </owl:Restriction>
(2188)          </rdfs:subClassOf>
(2189)          <rdfs:comment rdf:datatype="&xsd;string"
(2190)              >Eine Information, die vom Erkennen der Notwendigkeit oder
      Eignung einer Verwaltungsleistung bis zur Beantragung der Leistung in einer
      bestimmten Lebenslage hilfreich sein kann.</rdfs:comment>
(2191)      </owl:Class>
(2192)
(2193)
(2194)
(2195)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwInformationsbeschaffung -->
(2196)
(2197)      <owl:Class rdf:about="#VwInformationsbeschaffung">
(2198)          <rdfs:label
(2199)              >Informationsbeschaffung</rdfs:label>
```

```
(2200)          <rdfs:subClassOf rdf:resource="#VwHandlungsErweiterung"/>
(2201)          <rdfs:comment rdf:datatype="&xsd;string"
(2202)              >Handlungen, die dazu dienen die notwendigen Informationen
      f&#252;r Vorbereitung eines Verwaltungskontaktes zu
      beschaffen.</rdfs:comment>
(2203)      </owl:Class>
(2204)
(2205)
(2206)
(2207)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwLebenslage -->
(2208)
(2209)      <owl:Class rdf:about="#VwLebenslage">
(2210)          <rdfs:subClassOf
      rdf:resource="#VwHandlungsgegenstandsErweiterung"/>
(2211)          <rdfs:subClassOf>
(2212)              <owl:Restriction>
(2213)                  <owl:onProperty rdf:resource="#zugehoerigerHinweis"/>
(2214)                  <owl:someValuesFrom rdf:resource="#VwHinweisInformation"/>
(2215)              </owl:Restriction>
(2216)          </rdfs:subClassOf>
(2217)          <rdfs:subClassOf>
(2218)              <owl:Restriction>
(2219)                  <owl:onProperty rdf:resource="#gehoertZuAntragsteller"/>
(2220)                  <owl:allValuesFrom rdf:resource="#VwAntragsteller"/>
(2221)              </owl:Restriction>
(2222)          </rdfs:subClassOf>
(2223)          <rdfs:comment rdf:datatype="&xsd;string"
(2224)              >Situation bzw. Lebensabschnitt eines B&#252;rgers, in denen
      Verwaltungsleistungen m&#246;glich oder notwendig sind.</rdfs:comment>
(2225)      </owl:Class>
(2226)
(2227)
(2228)
(2229)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwLebenssachverhalt -->
(2230)
(2231)      <owl:Class rdf:about="#VwLebenssachverhalt">
(2232)          <rdfs:label rdf:datatype="&xsd;string"
(2233)              >VwLebenssachverhalt</rdfs:label>
(2234)          <rdfs:subClassOf rdf:resource="#Lebenssachverhalt"/>
(2235)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2236)          <rdfs:comment rdf:datatype="&xsd;string"
(2237)              >Der Lebenssacherhalt im Verwaltungshandeln repr&#228;sentiert
      ein tats&#228;chliches Ereignis oder eine tats&#228;chlich vorliege
      Situation, die durch eine beliebige Menge von Eigenschaften gekennzeichnet
      ist, von denen f&#252;r die Rechtsanwendung in der Verwaltung jedoch nur
      eine definierte Teilmenge relevant ist.</rdfs:comment>
(2238)      </owl:Class>
(2239)
(2240)
(2241)
(2242)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwLeistung -->
(2243)
(2244)      <owl:Class rdf:about="#VwLeistung">
(2245)          <rdfs:label rdf:datatype="&xsd;string">VwLeistung</rdfs:label>
(2246)          <rdfs:subClassOf rdf:resource="#Massnahme"/>
(2247)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2248)          <rdfs:subClassOf>
(2249)              <owl:Restriction>
(2250)                  <owl:onProperty rdf:resource="#zugehoerigerAntrag"/>
(2251)                  <owl:someValuesFrom rdf:resource="#VwAntrag"/>
(2252)              </owl:Restriction>
(2253)          </rdfs:subClassOf>
(2254)          <rdfs:subClassOf>
(2255)              <owl:Restriction>
(2256)                  <owl:onProperty rdf:resource="#istVomTyp"/>
(2257)                  <owl:allValuesFrom rdf:resource="#VwRechtsfolge"/>
(2258)              </owl:Restriction>
(2259)          </rdfs:subClassOf>
(2260)          <rdfs:comment rdf:datatype="&xsd;string"
```

```
(2261)              >Die Verwaltungsleistung repr&#228;sentiert die Konkretisierung
      einer abstrakten Rechtsfolge f&#252;r den im Verwaltungshandeln vorliegenden
      Einzelfall. Sie kann als Element einer Menge von ggf. mehreren
      m&#246;glichen Leistungen aufgefasst werden, die durch die Rechtsfolge
      repr&#228;sentiert werden. Das streben nach optimaler Zielerreichung
      f&#252;hrt dazu, dass die Konkretisierung anhand der Ziele des
      Verwaltungshandelns durchgef&#252;hrt wird.Ma&#223;nahme: Das Konzept
      &#252;bernimmt nicht die allgemeine Bezeichnung Ma&#223;nahme in das
      Verwaltungshandeln, um deutlich zu machen, dass in der Leistungsverwaltung
      Ma&#223;nahmen in Form von Leistungen erbracht werden. Semantisch sonst
      identisch.</rdfs:comment>
(2262)      </owl:Class>
(2263)
(2264)
(2265)
(2266)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwLeistungsEigenschaft -->
(2267)
(2268)      <owl:Class rdf:about="#VwLeistungsEigenschaft">
(2269)          <rdfs:label rdf:datatype="&xsd;string"
(2270)              >VwLeistungsEigenschaft</rdfs:label>
(2271)          <rdfs:subClassOf rdf:resource="#MassnahmenEigenschaft"/>
(2272)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2273)          <rdfs:comment rdf:datatype="&xsd;string"
(2274)              >Repr&#228;sentiert genau eine Eigenschaft der
      Verwaltungsleistung, die bezogen auf den Einzelfall unterschiedliche
      ausgepr&#228;gt sein kann.</rdfs:comment>
(2275)      </owl:Class>
(2276)
(2277)
(2278)
(2279)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwLeistungserbringung -->
(2280)
(2281)      <owl:Class rdf:about="#VwLeistungserbringung">
(2282)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2283)          <rdfs:subClassOf>
(2284)              <owl:Restriction>
(2285)                  <owl:onProperty rdf:resource="#erbringtLeistung"/>
(2286)                  <owl:allValuesFrom rdf:resource="#VwLeistung"/>
(2287)              </owl:Restriction>
(2288)          </rdfs:subClassOf>
(2289)          <rdfs:subClassOf>
(2290)              <owl:Restriction>
(2291)                  <owl:onProperty rdf:resource="#erbringtLeistung"/>
(2292)                  <owl:onClass rdf:resource="#VwLeistung"/>
(2293)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">1</owl:minQualifiedCardinality>
(2294)              </owl:Restriction>
(2295)          </rdfs:subClassOf>
(2296)          <rdfs:comment rdf:datatype="&xsd;string"
(2297)              >Repr&#228;sentiert alle Handlungen, die zur Kommunikation der
      getroffenen Entscheidung und zur tats&#228;chlichen Erbringung der
      Verwaltungsleistung dienen, sofern sich diese nicht in der Mitteilung der
      getroffenen Entscheidung (z.B. im Falle einer Erlaubnis oder eines Verbotes)
      ersch&#246;pft.Dieses Konzept hat keine Entsprechung in den bisher Konzepten
      der klassischen Rechtsanwendung.</rdfs:comment>
(2298)      </owl:Class>
(2299)
(2300)
(2301)
(2302)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwLeistungszusammenhang -->
(2303)
(2304)      <owl:Class rdf:about="#VwLeistungszusammenhang">
(2305)          <rdfs:label
(2306)              >Leistungszusammenhang</rdfs:label>
(2307)          <rdfs:subClassOf rdf:resource="#VwHinweisInformation"/>
(2308)          <rdfs:subClassOf>
(2309)              <owl:Restriction>
(2310)                  <owl:onProperty
      rdf:resource="#ausgangsVerwaltungsprodukt"/>
(2311)                  <owl:someValuesFrom rdf:resource="#VwRechtsfolge"/>
(2312)              </owl:Restriction>
(2313)          </rdfs:subClassOf>
```

```
(2314)          <rdfs:comment rdf:datatype="&xsd;string"
(2315)              >Informationen, die bezogen auf die aktuell auf Basis eines
      Verwaltungsproduktes (Ausgangsprodukt) erbrachte Verwaltungsleistung und
      eine bekannte Lebenslage Hinweise auf m&#246;gliche oder notwendige
      Folgeleistungen aus Verwaltungsprodukten geben k&#246;nnen.</rdfs:comment>
(2316)      </owl:Class>
(2317)
(2318)
(2319)
(2320)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwMassnahmenfestlegung -->
(2321)
(2322)      <owl:Class rdf:about="#VwMassnahmenfestlegung">
(2323)          <rdfs:subClassOf rdf:resource="#Einzelfallfestsetzung"/>
(2324)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2325)          <rdfs:subClassOf>
(2326)              <owl:Restriction>
(2327)                  <owl:onProperty rdf:resource="#resultiertInMassnahme"/>
(2328)                  <owl:allValuesFrom rdf:resource="#VwLeistung"/>
(2329)              </owl:Restriction>
(2330)          </rdfs:subClassOf>
(2331)          <rdfs:comment rdf:datatype="&xsd;string"
(2332)              >Die Ma&#223;nahmenfestlegung repr&#228;sentiert die
      Verwaltungshandlung zur Festlegung einer konkreten Ma&#223;nahme als
      Verwaltungsleistung im Einzelfall.Im Rahmen der Ma&#223;nahmenfestlegung
      werden Handlungen durchgef&#252;hrt, die f&#252;r jedes Merkmal der
      festgelegten abstrakten Rechtsfolge eine konkrete Auspr&#228;gung
      entsprechend des vorliegenden Einzelfalls vorsehen und dokumentieren.Die
      Ma&#223;nahmenfestlegung ist die Anwendung der Einzelfallfestsetzung in Form
      einer Verwaltungshandlung.</rdfs:comment>
(2333)      </owl:Class>
(2334)
(2335)
(2336)
(2337)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwMittelwahl -->
(2338)
(2339)      <owl:Class rdf:about="#VwMittelwahl">
(2340)          <rdfs:subClassOf rdf:resource="#Rechtsfolgenbestimmung"/>
(2341)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2342)          <rdfs:subClassOf>
(2343)              <owl:Restriction>
(2344)                  <owl:onProperty rdf:resource="#umfasstErmessensAusuebung"/>
(2345)                  <owl:allValuesFrom rdf:resource="#VwErmessensausuebung"/>
(2346)              </owl:Restriction>
(2347)          </rdfs:subClassOf>
(2348)          <rdfs:subClassOf>
(2349)              <owl:Restriction>
(2350)                  <owl:onProperty rdf:resource="#umfasstErmessensAusuebung"/>
(2351)                  <owl:onClass rdf:resource="#VwErmessensausuebung"/>
(2352)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
(2353)              </owl:Restriction>
(2354)          </rdfs:subClassOf>
(2355)          <rdfs:subClassOf>
(2356)              <owl:Restriction>
(2357)                  <owl:onProperty rdf:resource="#orientiertSichAnZielZweck"/>
(2358)                  <owl:allValuesFrom rdf:resource="#VwZielZweck"/>
(2359)              </owl:Restriction>
(2360)          </rdfs:subClassOf>
(2361)          <rdfs:comment rdf:datatype="&xsd;string"
(2362)              >Die Mittelwahl pr&#228;sentiert die Auswahl einer abstrakten
      Rechtsfolge oder die  Auswahl keiner Rechtsfolge, sofern auch dies eine
      zul&#228;ssige Handlungsoption ist. Im Falle von ungebundenem Handeln kann
      die Mittelwahl die Aus&#252;bung von Ermessen umfassen. Die Mittelwahl
      repr&#228;sentiert die Anwendung der Rechtsfolgenbestimmung im Rahmen eines
      Verwaltungsverfahrens.
(2363)  Ist durch die Rechtsvorschrift oder durch eine andere Vorschrift bei
      erf&#252;lltem Tatbestand genau eine abstrakte Rechtsfolge vorgesehen, so
      ist die durchzuf&#252;hrende Handlung trivial: Die vorgesehene Rechtsfolge
      wird als Mittel gew&#228;hlt. Stehen mehrere Rechtsfolgen zur Auswahl, so
      wird von diesen genau eine Rechtsfolge ausgew&#228;hlt, die den h&#246;chten
      Beitrag zur Zielereichung liefert. Es ist auch m&#246;glich, dass keine
      Rechtsfolge gew&#228;hlt wird, d.h. das sich die Verwaltung entscheidet,
      nicht zu handeln.</rdfs:comment>
```

```
(2364)      </owl:Class>
(2365)
(2366)
(2367)
(2368)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwNachbearbeitung -->
(2369)
(2370)      <owl:Class rdf:about="#VwNachbearbeitung">
(2371)          <rdfs:label
(2372)              >VwNachbearbeitung</rdfs:label>
(2373)          <rdfs:subClassOf rdf:resource="#VwHandlungsErweiterung"/>
(2374)          <rdfs:comment rdf:datatype="&xsd;string"
(2375)              >Handlungen die dazu dienen, die Leistungserstellung intern zu
      dokumentieren oder abzuschlie&#223;en und ggf. Hinweise auf nachfolgend
      relevante Verwaltungsleistungen an den Nachfrager zu geben.</rdfs:comment>
(2376)      </owl:Class>
(2377)
(2378)
(2379)
(2380)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwNachbearbeitungsinformation -->
(2381)
(2382)      <owl:Class rdf:about="#VwNachbearbeitungsinformation">
(2383)          <rdfs:label
(2384)              >Nachbearbeitungsinformation</rdfs:label>
(2385)          <rdfs:subClassOf
      rdf:resource="#VwHandlungsgegenstandsErweiterung"/>
(2386)          <rdfs:comment rdf:datatype="&xsd;string"
(2387)              >Informationen, die f&#252;r den internen Abschluss der
      Verwaltungst&#228;tigkeit notwendig sind.</rdfs:comment>
(2388)      </owl:Class>
(2389)
(2390)
(2391)
(2392)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwNachfrager -->
(2393)
(2394)      <owl:Class rdf:about="#VwNachfrager">
(2395)          <rdfs:label>VwNachfrager</rdfs:label>
(2396)          <rdfs:subClassOf rdf:resource="#VwAkteursErweiterung"/>
(2397)          <rdfs:subClassOf>
(2398)              <owl:Restriction>
(2399)                  <owl:onProperty rdf:resource="#fuehrtDurch"/>
(2400)                  <owl:someValuesFrom
      rdf:resource="#VwInformationsbeschaffung"/>
(2401)              </owl:Restriction>
(2402)          </rdfs:subClassOf>
(2403)          <rdfs:subClassOf>
(2404)              <owl:Restriction>
(2405)                  <owl:onProperty rdf:resource="#fuehrtDurch"/>
(2406)                  <owl:someValuesFrom rdf:resource="#VwVorbereitung"/>
(2407)              </owl:Restriction>
(2408)          </rdfs:subClassOf>
(2409)          <rdfs:comment rdf:datatype="&xsd;string"
(2410)              >Ein Akteur, der ein zu befriedigendes Anliegen entwickelt hat.
      Er unterscheidet sich vom Antragsteller dadurch, dass ein Nachfrager nicht
      notwendigerweise bereits einen Antrag gestellt haben muss.</rdfs:comment>
(2411)      </owl:Class>
(2412)
(2413)
(2414)
(2415)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwNutzungVonBeurteilungsspielraum -->
(2416)
(2417)      <owl:Class rdf:about="#VwNutzungVonBeurteilungsspielraum">
(2418)          <rdfs:subClassOf rdf:resource="#Subsumtion"/>
(2419)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2420)          <rdfs:subClassOf>
(2421)              <owl:Restriction>
(2422)                  <owl:onProperty
      rdf:resource="#nutztBeurteilungsspielraum"/>
(2423)                  <owl:someValuesFrom
      rdf:resource="#VwBeurteilungsspielraum"/>
(2424)              </owl:Restriction>
(2425)          </rdfs:subClassOf>
```

```
(2426)          <rdfs:comment rdf:datatype="&xsd;string"
(2427)              >Repr&#228;sentiert Handlungen, die im Rahmen einer
      Pr&#252;fung von Leistungsvoraussetzungen erfolgen, bei denen die zugrunde
      liegende Rechtsvorschrift und sonstige Vorschriften einen Spielraum in der
      Beurteilung lassen. Im Rahmen dieser Handlungen wird der gelassene Spielraum
      durch Wertung und Beurteilung vor dem Hintergrund eines konkreten
      Sachverhalts genutzt.Die Nutzung von Beurteilungsspielraum ist in der
      Rechtsanwendung Teil der Subsumtion.</rdfs:comment>
(2428)      </owl:Class>
(2429)
(2430)
(2431)
(2432)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwRechtsfolge -->
(2433)
(2434)      <owl:Class rdf:about="#VwRechtsfolge">
(2435)          <rdfs:label rdf:datatype="&xsd;string">VwRechtsfolge</rdfs:label>
(2436)          <rdfs:subClassOf rdf:resource="#Rechtsfolge"/>
(2437)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2438)          <rdfs:subClassOf>
(2439)              <owl:Restriction>
(2440)                  <owl:onProperty rdf:resource="#istKostenpflichtig"/>
(2441)                  <owl:qualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">1</owl:qualifiedCardinality>
(2442)                  <owl:onDataRange rdf:resource="&xsd;boolean"/>
(2443)              </owl:Restriction>
(2444)          </rdfs:subClassOf>
(2445)          <rdfs:subClassOf>
(2446)              <owl:Restriction>
(2447)                  <owl:onProperty rdf:resource="#liefertBeitragZuZielZweck"/>
(2448)                  <owl:allValuesFrom rdf:resource="#VwZielZweck"/>
(2449)              </owl:Restriction>
(2450)          </rdfs:subClassOf>
(2451)          <rdfs:subClassOf>
(2452)              <owl:Restriction>
(2453)                  <owl:onProperty rdf:resource="#istVomTyp"/>
(2454)                  <owl:someValuesFrom rdf:resource="#VwLeistung"/>
(2455)              </owl:Restriction>
(2456)          </rdfs:subClassOf>
(2457)          <rdfs:comment rdf:datatype="&xsd;string"
(2458)              >Die Rechtsfolge im Verwaltungshandeln repr&#228;sentiert die
      in einen Rechtssatz bei Vorliegen der Voraussetzungen vorgesehene abstrakte
      Beschreibung der rechtlichen Konsequenz. Ihre rechtliche Zul&#228;ssigkeit
      wird durch die Methoden der Rechtsanwendung ermittelt, ihre Auswahl aber
      wird auch durch ihren Beitrag zu Zielerreichung des Verwaltungshandelns
      bewertet. Eine Rechtsfolge kann als eine Menge von konkreten Leistungen
      aufgefasst werden, zu der alle Leistungen geh&#246;ren, die der abstrakten
      Beschreibung der Rechtsfolge gen&#252;gen. Die Semantik der Rechtsfolge wird
      an die ver&#228;nderte Bezeichnung, des zum Konzepts Ma&#223;nahme und
      korrespondierenden Konzepts Leistung angepasst. Insbesondere erweitert sich
      der Charakter dadurch, dass sie nicht nur eine rechtliche Zul&#228;ssigkeit
      hat (oder nicht hat), sondern auch an ihrem Beitrag zur Zielerreichung
      gemessen wird.</rdfs:comment>
(2459)      </owl:Class>
(2460)
(2461)
(2462)
(2463)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwRechtsfolgenMerkmal -->
(2464)
(2465)      <owl:Class rdf:about="#VwRechtsfolgenMerkmal">
(2466)          <rdfs:label rdf:datatype="&xsd;string"
(2467)              >VwRechtsfolgenMerkmal</rdfs:label>
(2468)          <rdfs:subClassOf rdf:resource="#RechtsfolgenMerkmal"/>
(2469)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2470)          <rdfs:subClassOf>
(2471)              <owl:Restriction>
(2472)                  <owl:onProperty rdf:resource="#istAbhaengigVonErmessen"/>
(2473)                  <owl:someValuesFrom rdf:resource="#VwErmessen"/>
(2474)              </owl:Restriction>
(2475)          </rdfs:subClassOf>
(2476)          <rdfs:comment rdf:datatype="&xsd;string"
(2477)              >Repr&#228;sentiert genau ein rechtlich relevantes Merkmals
      einer Rechtsfolge im Rahmen des Verwaltungshandelns.</rdfs:comment>
(2478)      </owl:Class>
```

```
(2479)
(2480)
(2481)
(2482)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwSachverhalt -->
(2483)
(2484)      <owl:Class rdf:about="#VwSachverhalt">
(2485)          <rdfs:label rdf:datatype="&xsd;string">VwSachverhalt</rdfs:label>
(2486)          <rdfs:subClassOf rdf:resource="#Sachverhalt"/>
(2487)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2488)          <rdfs:comment rdf:datatype="&xsd;string"
(2489)              >Der Sachverhalt im Verwaltungshandeln repr&#228;sentiert das
      Ergebnis der Extraktion und Abstraktion tatbestandsrelevanter Eigenschaften
      eines real vorliegenden Lebenssachverhalts in einer
      entscheidungsf&#228;higen und damit auch subsumtionsf&#228;higen Form. Ein
      Sachverhalt muss neben der Subsumtionsf&#228;higkeit auch die
      Entscheidbarkeit hinsichtlich der Ziele des Verwaltungshandelns
      aufweisen.</rdfs:comment>
(2490)      </owl:Class>
(2491)
(2492)
(2493)
(2494)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwSachverhaltsEigenschaft -->
(2495)
(2496)      <owl:Class rdf:about="#VwSachverhaltsEigenschaft">
(2497)          <rdfs:label rdf:datatype="&xsd;string"
(2498)              >VwSachverhaltseigenschaft</rdfs:label>
(2499)          <rdfs:subClassOf rdf:resource="#SachverhaltsEigenschaft"/>
(2500)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2501)          <rdfs:comment rdf:datatype="&xsd;string"
(2502)              >Die Sachverhaltseigenschaft im Verwaltungshandeln
      repr&#228;sentiert genau ein rechtlich relevantes Merkmal eines Sachverhalts
      im Rahmen des Verwaltungshandelns.</rdfs:comment>
(2503)      </owl:Class>
(2504)
(2505)
(2506)
(2507)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwSachverhaltsermittlung -->
(2508)
(2509)      <owl:Class rdf:about="#VwSachverhaltsermittlung">
(2510)          <rdfs:subClassOf rdf:resource="#Sachverhaltsermittlung"/>
(2511)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2512)          <rdfs:comment rdf:datatype="&xsd;string"
(2513)              >Repr&#228;sentiert alle Handlungen, ein Tr&#228;ger
      &#246;ffentlicher Aufgaben im Rahmen seiner T&#228;tigkeit (in der Regel
      eines Verwaltungsverfahrens) durchf&#252;hrt, um die tatbestandsrelevanten
      Eigenschaften eines vorliegenden Lebenssachverhalts zu untersuchen,
      herauszuarbeiten, zu abstrahieren und in die subsumtionsf&#228;hige Form
      eines rechtlich relevanten Sacherhaltes zu transformieren, um z.B. dem
      Untersuchungsgrundsatz nach &#167; 24 VwVfG zu gen&#252;gen. Die
      Sachverhaltsermittlung repr&#228;sentiert die Aspekte der Rechtsfindung, die
      der Ableitung des rechtlich relevanten Sachverhalts aus dem vorliegenden
      Lebenssachverhalt dienen. Die Frage der anzuwendenden Rechtsnorm, die in der
      Rechtsanwendung ebenfalls Bestandteil der Rechtsfindung sind, kann
      w&#228;hrend der Ausf&#252;hrung durch die Verwaltung als gekl&#228;rt
      vorausgesetzt werden.</rdfs:comment>
(2514)      </owl:Class>
(2515)
(2516)
(2517)
(2518)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwVerfahren -->
(2519)
(2520)      <owl:Class rdf:about="#VwVerfahren">
(2521)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2522)          <rdfs:subClassOf>
(2523)              <owl:Restriction>
(2524)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2525)                  <owl:someValuesFrom
      rdf:resource="#VwNutzungVonBeurteilungsspielraum"/>
(2526)              </owl:Restriction>
(2527)          </rdfs:subClassOf>
(2528)          <rdfs:subClassOf>
```

```
(2529)              <owl:Restriction>
(2530)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2531)                  <owl:someValuesFrom
      rdf:resource="#VwMassnahmenfestlegung"/>
(2532)              </owl:Restriction>
(2533)          </rdfs:subClassOf>
(2534)          <rdfs:subClassOf>
(2535)              <owl:Restriction>
(2536)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2537)                  <owl:someValuesFrom
      rdf:resource="#VwVerfahrenseinleitung"/>
(2538)              </owl:Restriction>
(2539)          </rdfs:subClassOf>
(2540)          <rdfs:subClassOf>
(2541)              <owl:Restriction>
(2542)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2543)                  <owl:someValuesFrom rdf:resource="#VwErmessensausuebung"/>
(2544)              </owl:Restriction>
(2545)          </rdfs:subClassOf>
(2546)          <rdfs:subClassOf>
(2547)              <owl:Restriction>
(2548)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2549)                  <owl:someValuesFrom
      rdf:resource="#VwSachverhaltsermittlung"/>
(2550)              </owl:Restriction>
(2551)          </rdfs:subClassOf>
(2552)          <rdfs:subClassOf>
(2553)              <owl:Restriction>
(2554)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2555)                  <owl:someValuesFrom rdf:resource="#VwMittelwahl"/>
(2556)              </owl:Restriction>
(2557)          </rdfs:subClassOf>
(2558)          <rdfs:subClassOf>
(2559)              <owl:Restriction>
(2560)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2561)                  <owl:someValuesFrom
      rdf:resource="#VwVoraussetzungspruefung"/>
(2562)              </owl:Restriction>
(2563)          </rdfs:subClassOf>
(2564)          <rdfs:subClassOf>
(2565)              <owl:Restriction>
(2566)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2567)                  <owl:someValuesFrom rdf:resource="#VwLeistungserbringung"/>
(2568)              </owl:Restriction>
(2569)          </rdfs:subClassOf>
(2570)          <rdfs:subClassOf>
(2571)              <owl:Restriction>
(2572)                  <owl:onProperty rdf:resource="#umfasstVerfahrenshandlung"/>
(2573)                  <owl:someValuesFrom rdf:resource="#VwErgebnismitteilung"/>
(2574)              </owl:Restriction>
(2575)          </rdfs:subClassOf>
(2576)          <rdfs:comment rdf:datatype="&xsd;string"
(2577)              >Nach au&#223;en wirkende T&#228;tigkeit eines Tr&#228;gers
      &#246;ffentlicher Aufgaben, die auf die Pr&#252;fung der Voraussetzungen,
      die Vorbereitung und den Erlass eines Verwaltungsaktes gerichtet ist und den
      Erlass des Verwaltungsakte einschlie&#223;t. Spezialisierung des generellen
      Konzepts des Verwaltungshandelns. (Dieses Konzept hat keine Entsprechung in
      den bisher identifizierten Konzepten der Rechtsanwendung.)</rdfs:comment>
(2578)      </owl:Class>
(2579)
(2580)
(2581)
(2582)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwVerfahrenseinleitung -->
(2583)
(2584)      <owl:Class rdf:about="#VwVerfahrenseinleitung">
(2585)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2586)          <rdfs:comment rdf:datatype="&xsd;string"
(2587)              >Die Verfahrenseinleitung repr&#228;sentiert Handlungen der
      Verwaltung die die Leistungserbringung im Rahmen eines Verwaltungsverfahrens
      einleiten und wird durch das Auftreten eines von der Verwaltung
      wahrgenommenen Ereignisses ausgel&#246;st.Dieses Konzept hat keine
      Entsprechung in den bisher identifizierten Konzepten der
      Rechtsanwendung.</rdfs:comment>
(2588)      </owl:Class>
```

```
(2589)
(2590)
(2591)
(2592)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwVoraussetzung -->
(2593)
(2594)      <owl:Class rdf:about="#VwVoraussetzung">
(2595)          <rdfs:label rdf:datatype="&xsd;string">VwVoraussetzung</rdfs:label>
(2596)          <rdfs:subClassOf rdf:resource="#Tatbestand"/>
(2597)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2598)          <rdfs:comment rdf:datatype="&xsd;string"
(2599)              >Die Voraussetzung im Verwaltungshandeln repr&#228;sentiert die
      Zusammenfassung bestimmter abstrakter Merkmale und Voraussetzungen, die ein
      Rechtssatz f&#252;r das Eintreten einer Rechtsfolge vorsieht. Sie kann als
      eine Menge von Sachverhalten aufgefasst werden, zu der alle Sachverhalte
      geh&#246;ren, deren Eigenschaften den Bedingungen der Voraussetzung
      gen&#252;gen. Das Konzept &#252;bernimmt nicht die Bezeichnung Tatbestand in
      das Verwaltungshandeln, um deutlich zu machen, dass hiermit die
      Voraussetzung f&#252;r die Erbringung einer Verwaltungsleistung in der
      Leistungsverwaltung und nicht prim&#228;r ein Straftatbestand in der
      Ordnungsverwaltung oder richterlichen Rechtsanwendung gemeint
      ist.</rdfs:comment>
(2600)      </owl:Class>
(2601)
(2602)
(2603)
(2604)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwVoraussetzungsBedingung -->
(2605)
(2606)      <owl:Class rdf:about="#VwVoraussetzungsBedingung">
(2607)          <rdfs:label rdf:datatype="&xsd;string"
(2608)              >VwVoraussetzungsbedingung</rdfs:label>
(2609)          <rdfs:subClassOf rdf:resource="#Tatbestandsbedingung"/>
(2610)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2611)          <rdfs:subClassOf>
(2612)              <owl:Restriction>
(2613)                  <owl:onProperty
      rdf:resource="#istAbhaengigVonBeurteilungsspielraum"/>
(2614)                  <owl:someValuesFrom
      rdf:resource="#VwBeurteilungsspielraum"/>
(2615)              </owl:Restriction>
(2616)          </rdfs:subClassOf>
(2617)          <rdfs:comment rdf:datatype="&xsd;string"
(2618)              >Die Voraussetzungsbedingung im Verwaltungshandeln
      repr&#228;sentiert genau eine Bedingung der Voraussetzung.</rdfs:comment>
(2619)      </owl:Class>
(2620)
(2621)
(2622)
(2623)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwVoraussetzungspruefung -->
(2624)
(2625)      <owl:Class rdf:about="#VwVoraussetzungspruefung">
(2626)          <rdfs:subClassOf rdf:resource="#Subsumtion"/>
(2627)          <rdfs:subClassOf rdf:resource="#VwHandlung"/>
(2628)          <rdfs:subClassOf>
(2629)              <owl:Restriction>
(2630)                  <owl:onProperty
      rdf:resource="#umfassstNutzungBeurteilungsspielraum"/>
(2631)                  <owl:onClass
      rdf:resource="#VwNutzungVonBeurteilungsspielraum"/>
(2632)                  <owl:minQualifiedCardinality
      rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
(2633)              </owl:Restriction>
(2634)          </rdfs:subClassOf>
(2635)          <rdfs:subClassOf>
(2636)              <owl:Restriction>
(2637)                  <owl:onProperty
      rdf:resource="#umfassstNutzungBeurteilungsspielraum"/>
(2638)                  <owl:allValuesFrom
      rdf:resource="#VwNutzungVonBeurteilungsspielraum"/>
(2639)              </owl:Restriction>
(2640)          </rdfs:subClassOf>
(2641)          <rdfs:comment rdf:datatype="&xsd;string"
```

```
(2642)              >Im Rahmen der Voraussetzungspr&#252;fung wird gepr&#252;ft, ob
      die Auspr&#228;gungen eines im Einzelfall vorliegenden Sachverhalts die
      Voraussetzungen f&#252;r die Erbringung einer Verwaltungsleistung
      erf&#252;llen.Im Rahmen der Voraussetzungspr&#252;fung wird f&#252;r jede
      Tatbestandsbedingung, gepr&#252;ft, ob die korrespondierende Eigenschaft des
      Sacherhaltes den Bedingungen gen&#252;gt. Die Voraussetzungspr&#252;fung
      endet mit der Entscheidung, ob alle Bedingungen des Tatbestandes durch den
      Sachverhalt erf&#252;llt sind und damit die Subsumtion gelungen ist, oder ob
      dies nicht der Fall ist.Die Pr&#252;fung von Voraussetzungen kann auch die
      Nutzung von Berteilungsspielraum umfassen.Die Klasse repr&#228;sentiert die
      Anwendung der Subsumtion im Rahmen eines Verwaltungsverfahrens, d.h. als
      Verwaltungshandlung.</rdfs:comment>
(2643)      </owl:Class>
(2644)
(2645)
(2646)
(2647)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwVorbereitung -->
(2648)
(2649)      <owl:Class rdf:about="#VwVorbereitung">
(2650)          <rdfs:label>VwVorbereitung</rdfs:label>
(2651)          <rdfs:subClassOf rdf:resource="#VwHandlungsErweiterung"/>
(2652)          <rdfs:comment rdf:datatype="&xsd;string"
(2653)              >Handlungen eines Nachfragers, um einen Verwaltungskontakt zu
      planen und vorzubereiten.</rdfs:comment>
(2654)      </owl:Class>
(2655)
(2656)
(2657)
(2658)      <!-- http://govobjects.thomasoff.de/ontology/govobjects-
      ontology.owl#VwZielZweck -->
(2659)
(2660)      <owl:Class rdf:about="#VwZielZweck">
(2661)          <rdfs:subClassOf rdf:resource="#VwHandlungsgegenstand"/>
(2662)          <rdfs:subClassOf>
(2663)              <owl:Restriction>
(2664)                  <owl:onProperty rdf:resource="#konkretisiertZiel"/>
(2665)                  <owl:someValuesFrom rdf:resource="#VwZielZweck"/>
(2666)              </owl:Restriction>
(2667)          </rdfs:subClassOf>
(2668)          <rdfs:subClassOf>
(2669)              <owl:Restriction>
(2670)                  <owl:onProperty
      rdf:resource="#wirdErreichtDuchRechtsfolge"/>
(2671)                  <owl:someValuesFrom rdf:resource="#VwRechtsfolge"/>
(2672)              </owl:Restriction>
(2673)          </rdfs:subClassOf>
(2674)          <rdfs:comment rdf:datatype="&xsd;string"
(2675)              >Die Zwecke und Ziele repr&#228;sentieren Bestandteile von
      Rechtsvorschriften, die auf den Vollzug in der Verwaltung ausgerichtet sind,
      und geben die Motivation f&#252;r die Festlegungen und die beabsichtigte
      Wirkung an. Sie werden von der Verwaltung als Bestandteil eines Zielsystems
      verwendet, an dem sich die &#246;ffentliche Verwaltung bei
      Ma&#223;nahmenwahl und Leistungsfestlegung orientiert.</rdfs:comment>
(2676)      </owl:Class>
(2677)
(2678)
(2679)
(2680)      <!--
(2681)
      ////////////////////////////////////////////////////////////////////////////
      ///////
(2682)      //
(2683)      // General axioms
(2684)      //
(2685)
      ////////////////////////////////////////////////////////////////////////////
      ///////
(2686)       -->
(2687)
(2688)      <rdf:Description>
(2689)          <rdf:type rdf:resource="&owl;AllDisjointClasses"/>
(2690)          <owl:members rdf:parseType="Collection">
(2691)              <rdf:Description rdf:about="#Einzelfallfestsetzung"/>
(2692)              <rdf:Description rdf:about="#Mitteilung"/>
```

```
(2693)              <rdf:Description rdf:about="#Rechtsfindung"/>
(2694)              <rdf:Description rdf:about="#Rechtsfolgenbestimmung"/>
(2695)              <rdf:Description rdf:about="#Sachverhaltsermittlung"/>
(2696)              <rdf:Description rdf:about="#Subsumtion"/>
(2697)          </owl:members>
(2698)      </rdf:Description>
(2699)      <rdf:Description>
(2700)          <rdf:type rdf:resource="&owl;AllDisjointClasses"/>
(2701)          <owl:members rdf:parseType="Collection">
(2702)              <rdf:Description rdf:about="#Rechtsfolge"/>
(2703)              <rdf:Description rdf:about="#RechtsfolgenMerkmal"/>
(2704)              <rdf:Description rdf:about="#Tatbestand"/>
(2705)              <rdf:Description rdf:about="#Tatbestandsbedingung"/>
(2706)          </owl:members>
(2707)      </rdf:Description>
(2708)      <rdf:Description>
(2709)          <rdf:type rdf:resource="&owl;AllDisjointClasses"/>
(2710)          <owl:members rdf:parseType="Collection">
(2711)              <rdf:Description rdf:about="#Lebenssachverhalt"/>
(2712)              <rdf:Description rdf:about="#Massnahme"/>
(2713)              <rdf:Description rdf:about="#MassnahmenEigenschaft"/>
(2714)              <rdf:Description rdf:about="#Rechtsfolge"/>
(2715)              <rdf:Description rdf:about="#RechtsfolgenMerkmal"/>
(2716)              <rdf:Description rdf:about="#Sachverhalt"/>
(2717)              <rdf:Description rdf:about="#SachverhaltsEigenschaft"/>
(2718)              <rdf:Description rdf:about="#Tatbestand"/>
(2719)              <rdf:Description rdf:about="#Tatbestandsbedingung"/>
(2720)          </owl:members>
(2721)      </rdf:Description>
(2722)  </rdf:RDF>
(2723)
(2724)
(2725)
(2726)  <!-- Generated by the OWL API (version 2.2.1.1138)
        http://owlapi.sourceforge.net -->
```

Listing 47 – Ontologie des Verwaltungshandelns im OWL-Format

B Extraktion der OntoMat-Annotationen

Dieser Abschnitt stellt das Java-Programm für die Verarbeitung von OntoMat-Annotationen in HTML-Dokumenten (Anhang B.1) und das XSL-Stylesheet zur Verarbeitung von XHTML-Dokumenten (Anhang B.2) dar.

B.1 Extraktion der Annotationen aus HTML-Dokumenten

```
(1)   package de.thomasoff.govobjects;
(2)
(3)   import java.io.*;
(4)
(5)   public class ontomat2rdf {
(6)
(7)     public static void main(String[] args) {
(8)       String ifilename = null;
(9)       String ofilename = null;
(10)
(11)      if ((args.length > 0) && (args.length < 3)) {
(12)        ifilename = args[0].toLowerCase();
(13)        if ((args.length > 1)) {
(14)          ofilename = args[1].toLowerCase();
(15)        }
(16)      } else {
(17)        help();
(18)        return;
(19)      }
(20)
(21)      try {
(22)
(23)        FileInputStream fstream = new FileInputStream(ifilename);
(24)        DataInputStream in = new DataInputStream(fstream);
(25)        BufferedReader br = new BufferedReader(new InputStreamReader(in));
(26)        String strLine;
(27)
(28)        FileWriter owriter = null;
(29)        if (ofilename!= null) {
(30)          owriter = new FileWriter(ofilename);
(31)        }
(32)
(33)        boolean rdfstarted = false;
(34)        while ((strLine = br.readLine()) != null) {
(35)          if (strLine.toUpperCase().contains("ONTOMAT-ANNOTATION-BEGIN")||
      rdfstarted) {
(36)            if (!rdfstarted) {
(37)              rdfstarted = true;
(38)              int rdfpos = strLine.indexOf("<rdf:RDF");
(39)              strLine = new String(strLine.substring(rdfpos).getBytes(), "UTF-
      8");
(40)              if (owriter == null) {
(41)                System.out.println("<?xml version='1.0' encoding='UTF-8'?>");
(42)              } else {
(43)                owriter.write("<?xml version='1.0' encoding='UTF-8'?>");
(44)              }
(45)            }
(46)
(47)            if (strLine.indexOf("</rdf:RDF") > -1) {
(48)              rdfstarted = false;
(49)            }
(50)
(51)            // Kovertierung des Zeichensatzes der Umlaute vornehmen
(52)            strLine=strLine.replaceAll("„","'");
(53)            strLine=strLine.replaceAll("“","'");
(54)            strLine=strLine.replaceAll("ö","oe");
(55)            strLine=strLine.replaceAll("ä","ae");
(56)            strLine=strLine.replaceAll("ü","ue");
(57)            strLine=strLine.replaceAll("ß","ss");
(58)            strLine=strLine.replaceAll("Ö","Oe");
(59)            strLine=strLine.replaceAll("Ä","Ae");
(60)            strLine=strLine.replaceAll("Ü","Ue");
(61)
(62)            String outLine = new String(strLine.getBytes(), "UTF-8");
(63)
```

```
(64)            if (owriter == null) {
(65)              System.out.println(outLine);
(66)            } else {
(67)              owriter.write(outLine);
(68)            }
(69)
(70)          }
(71)
(72)        }
(73)        in.close();
(74)        if (owriter != null) {
(75)          owriter.close();
(76)        }
(77)      } catch (Exception e) {// Catch exception
(78)        System.err.println("Error: " + e.getMessage());
(79)      }
(80)
(81)    }
(82)
(83)    public static void help() {
(84)      System.err.println("Error: No file found. Nothing to convert.");
(85)      System.out.println("Usage: ontomat2rf <filename>" +
(86)                  "\n        Print RDF-Content to screen.");
(87)      System.out.println("       ontomat2rf <filename> <newfile>" +
(88)                  "\n        Redirect RDF-content to newfile.");
(89)    }
(90)  }
```

Listing 48 – Java-Programm zur Extraktion der OntoMat-Annotationen aus HTML-Dokumenten

B.2 Extraktion der Annotationen aus XHTML-Dokumenten

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(4)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(5)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(6)     xmlns:html="http://www.w3.org/TR/REC-html40"
(7)     xmlns:xhtml="http://www.w3.org/1999/xhtml"
(8)     xmlns="http://www.w3.org/1999/xhtml"
(9)     xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
(10)    xmlns:dc="http://purl.org/dc/elements/1.1/"
(11)    xmlns:ontomat="http://annotation.semanticweb.org/ontologies/cream/ontomat#
      "
(12)    xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(13)    xmlns:rss="http://purl.org/rss/1.0/"
(14)    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
(15)    xmlns:govobjects="http://govobjects.thomasoff.de/bewohnerparken#"
(16)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(17)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(18)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(19)    xmlns:owl="http://www.w3.org/2002/07/owl#"
(20)    xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
(21)    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(22)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(23)    xmlns:rdfodm="urn:rdf2odm.ecore"
(24)    xmlns:xmi="http://www.omg.org/XMI"
(25)    xmlns:xmi1="http://schema.omg.org/spec/XMI/2.1"
(26)    xmlns:exslt="http://exslt.org/common"
(27)    extension-element-prefixes="exslt">
(28)
(29)    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes" use-
      character-maps="umlaute"/>
(30)
(31)    <xsl:character-map name="umlaute">
(32)      <xsl:output-character character="ä" string="ae"></xsl:output-character>
(33)      <xsl:output-character character="ö" string="oe"></xsl:output-character>
(34)      <xsl:output-character character="ü" string="ue"></xsl:output-character>
(35)      <xsl:output-character character="Ä" string="Ae"></xsl:output-character>
(36)      <xsl:output-character character="Ö" string="Oe"></xsl:output-character>
(37)      <xsl:output-character character="Ü" string="Ue"></xsl:output-character>
(38)      <xsl:output-character character="ß" string="ss"></xsl:output-character>
(39)      <xsl:output-character character='"' string=""></xsl:output-character>
(40)    </xsl:character-map>
```

```
(41)
(42)    <xsl:template match="/html:html">
(43)      <xsl:variable name="document-rdf">
(44)        <xsl:value-of select="substring-after(//comment(), 'ONTOMAT-
      ANNOTATION-BEGIN')"/>
(45)      </xsl:variable>
(46)      <xsl:variable name="node-rdf" select="saxon:parse($document-rdf)"
      xmlns:saxon="http://saxon.sf.net/"/>
(47)      <xsl:copy-of select="$node-rdf"/>
(48)    </xsl:template>
(49)
(50)  </xsl:stylesheet>
```

Listing 49 – XSL-Stylesheet zur Extraktion der OntoMat-Annotationen aus XHTML-Dokumenten

C Ontology Definition Metamodel UML-Profile

Die Abbildungen in diesem Abschnitt zeigen das für die Modellierung eingesetzte RDF- und OWL Profil für die UML sowie die eingesetzten Elemente des Standard L1 Profils der UML. Die Generierung des notwendigen ECORE-Formates (ca. 13.500 Codezeilen) erfolgte durch das Werkzeug TOPCASED.

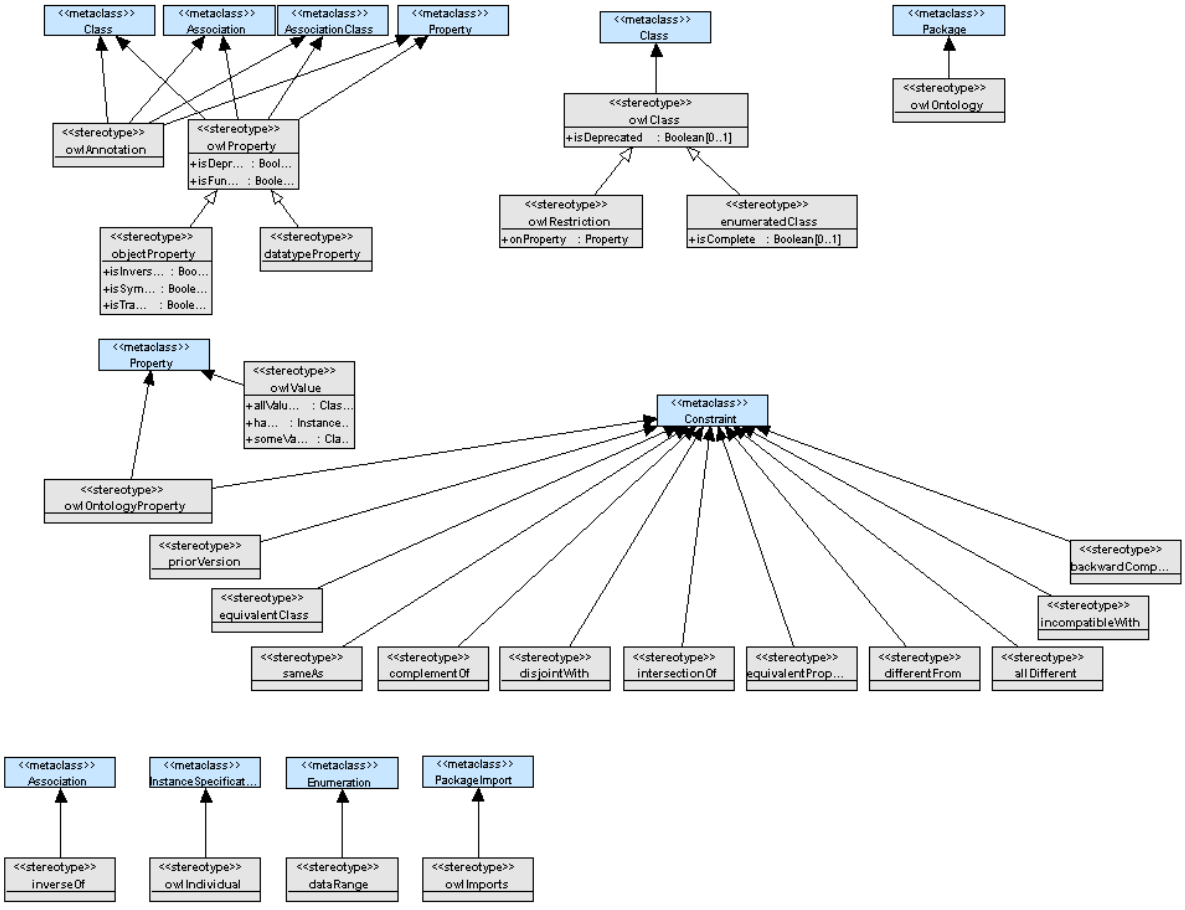


Abbildung 101 – Ausschnitt aus dem OWL UML-Profil als Modell zur Umsetzung des ODM

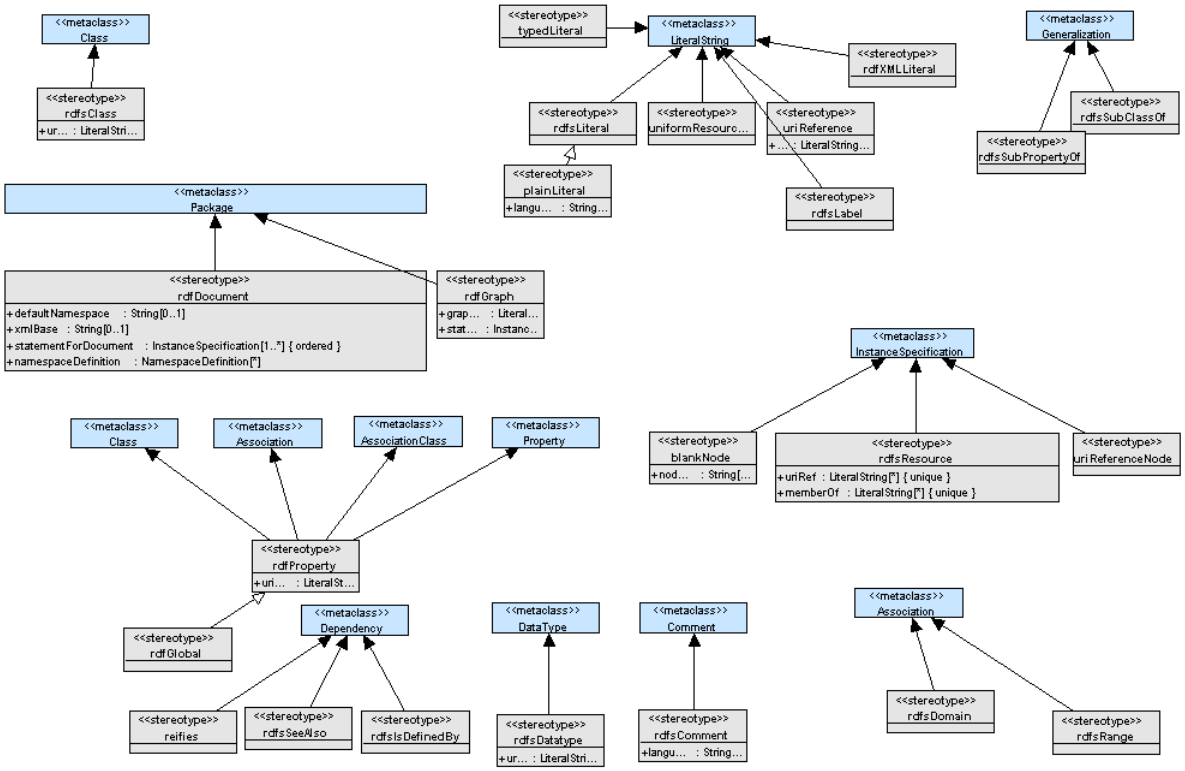


Abbildung 102 – Ausschnitt aus dem RDF UML-Profil als Modell zur Umsetzung des ODM

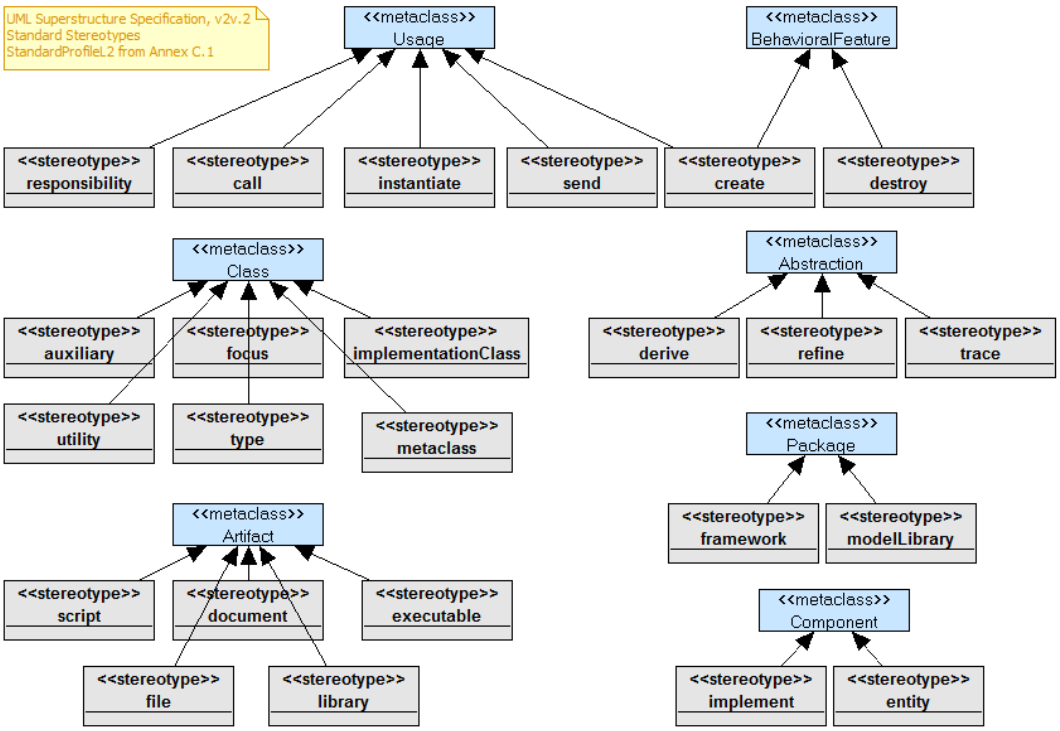


Abbildung 103 – Ausschnitt aus dem UML-Profile

D Initiale PRM-Erzeugung

In diesem Abschnitt werden die zur Erzeugung des initialen PRM eingesetzten XSL-Stylesheets dargestellt. Der Anhang zeigt das Stylesheet zur Transformation der OWL-Ontologie in ein ODM-basiertes UML-Modell als Bestandteil des PRM. Die Stylesheets zur

Transformation der OntoMat-Annotationen und der RDFa-Annotationen in ODM-basierte UML-Modelle, die den zweiten Bestandteil des PRM bilden, sind in Anhang D.2 und D.3 dargestellt. Von diesen Stylesheets werden Funktionen gemeinsam genutzt (Anhang D.4).

D.1 OWL-Ontologie

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <!--
(3)     Konvertiert die im OWL-Format vorliegende Ontologie des
      Verwaltungshandelns
(4)     nach ECORE XMI für die Verwendung als UML-Modell basierend auf dem OUP
(5)   -->
(6)   <xsl:stylesheet version="2.0"
(7)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(8)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(9)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(10)    xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
(11)    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(12)    xmlns:OwlUmlProfile="http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/
      28"
(13)    xmlns:RdfUmlProfile="http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/
      48"
(14)    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
(15)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
(16)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(17)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(18)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(19)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(20)    xmlns:owl="http://www.w3.org/2002/07/owl#"
(21)    xmlns:functx="http://www.functx.com">
(22)
(23)    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
(24)    <!--
(25)    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes" use-
      character-maps="umlaute"/>
(26)
(27)      <xsl:character-map name="umlaute">
(28)          <xsl:output-character character="ä" string="ae"></xsl:output-
      character>
(29)          <xsl:output-character character="ö" string="oe"></xsl:output-
      character>
(30)          <xsl:output-character character="ü" string="ue"></xsl:output-
      character>
(31)          <xsl:output-character character="Ä" string="Ae"></xsl:output-
      character>
(32)          <xsl:output-character character="Ö" string="Oe"></xsl:output-
      character>
(33)          <xsl:output-character character="Ü" string="Ue"></xsl:output-
      character>
(34)          <xsl:output-character character="ß" string="ss"></xsl:output-
      character>
(35)          <xsl:output-character character='"' string=""></xsl:output-
      character>
(36)      </xsl:character-map>
(37)    -->
(38)
(39)    <!--  Dateinamen für Quell und Zielmodell zur Aufzeichnung der
      Verfolgbarkeitsbeziehungen -->
(40)    <xsl:param name="srcMdl" select="'../Input/unknownOwlOntology.rdf'"/>
(41)    <xsl:param name="dstMdl" select="'../Model/unknownUmlModel.uml'"/>
(42)
(43)    <!-- Name der Transformation zur Aufzeichnung der
      Verfolgbarkeitsbeziehungen -->
(44)    <xsl:variable name="transName" select="'owl2prm_ontology'"/>
(45)
(46)    <!-- Import von Hilfsfunktionen -->
(47)    <xsl:include href="xsltrace.xsl"/>
(48)    <xsl:include href="convert-OwlCommon.xsl"/>
(49)
(50)    <!--
(51)      Struktur des Modells mit Wurzel, Modell und Package erzeugen.
      Anschließend
(52)      das Erzeugen der enthaltenen Modellelemente initiieren.
```

```
(53)    -->
(54)    <xsl:template match="/rdf:RDF">
(55)
(56)      <!--  Init Tracing -->
(57)      <xsl:call-template name="initTracing">
(58)        <xsl:with-param name="transName" select="$transName"/>
(59)      </xsl:call-template>
(60)
(61)      <!--  Wurzelelement XMI inkl. aller Namespaces und der SchemaLocation
      anlegen -->
(62)      <xsl:element name="xmi:XMI">
(63)        <xsl:attribute name="xmi:version" select="'2.1'"/>
(64)        <xsl:namespace name="xmi"
      select="'http://schema.omg.org/spec/XMI/2.1'"/>
(65)        <xsl:namespace name="xsi" select="'http://www.w3.org/2001/XMLSchema-
      instance'"/>
(66)        <xsl:namespace name="OwlUmlProfile"
      select="'http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/28'"/>
(67)        <xsl:namespace name="RdfUmlProfile"
      select="'http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/48'"/>
(68)        <xsl:namespace name="ecore"
      select="'http://www.eclipse.org/emf/2002/Ecore'"/>
(69)        <xsl:namespace name="uml"
      select="'http://www.eclipse.org/uml2/3.0.0/UML'"/>
(70)        <xsl:attribute name="xsi:schemaLocation">
(71)          <xsl:text>http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/28
      ../ProfileModels/OwlUmlProfile.uml#_QwFo8CezEeC9_fKmAwBwrw
      http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/48
      ../ProfileModels/RdfUmlProfile.uml#_ZPpPMCezEeC9_fKmAwBwrw</xsl:text>
(72)        </xsl:attribute>
(73)
(74)        <!--  Modell und enthaltene Elemente anlegen -->
(75)        <xsl:call-template name="model"/>
(76)
(77)        <!--  Referenzen auf das RDF- und OWL-Profil anlegen -->
(78)        <xsl:call-template name="ProfileRef"/>
(79)
(80)      </xsl:element>
(81)
(82)      <!--  Finalize Tracing -->
(83)      <xsl:call-template name="finalizeTracing"/>
(84)
(85)    </xsl:template>
(86)
(87)    <!--
(88)      Tranformationsregel
(89)      ID:    XSLT_OWL-ONTOLOGY-PRM-01
(90)      Name:  UmlModell
(91)     -->
(92)    <xsl:template name="model">
(93)      <!-- Variablen belegen -->
(94)      <xsl:variable name="name" select="govobj:name-from-rdfslabel-or-
      about(./owl:Ontology)"/>
(95)      <xsl:variable name="id" select="concat(govobj:id-from-
      about(./owl:Ontology), '_model')"/>
(96)      <xsl:variable name="ref" select="govobj:ref(./owl:Ontology)"/>
(97)
(98)      <!-- Element erzeugen -->
(99)      <xsl:element name="uml:Model">
(100)       <xsl:attribute name="xmi:id" select="$id"/>
(101)       <xsl:attribute name="name">
(102)         <!--  Wenn kein Name als rdfs:label gesetzt ist, dann den Dateinamen
      für das Modell verwenden -->
(103)         <xsl:choose>
(104)           <xsl:when test="string-length(normalize-
      space(owl:Ontology/rdfs:label))=0">
(105)             <xsl:value-of select="functx:substring-after-last(substring-
      before(replace($srcMdl, '\\', '/'), '.owl'), '/')"/>
(106)           </xsl:when>
(107)           <xsl:otherwise>
(108)             <xsl:value-of select="normalize-
      space(owl:Ontology/rdfs:label)"/>
(109)           </xsl:otherwise>
(110)         </xsl:choose>
(111)         <xsl:text>-Model</xsl:text>
```

```
(112)       </xsl:attribute>
(113)
(114)       <!--  Dokumentation zum Modell aus Ontologie übernehmen -->
(115)       <xsl:element name="eAnnotations">
(116)         <xsl:attribute name="xmi:id" select="concat(govobj:id-from-
      about(./owl:Ontology), '_model_annotation')"/>
(117)         <xsl:attribute name="source"
      select="'http://www.topcased.org/documentation'"/>
(118)         <xsl:element name="details">
(119)           <xsl:attribute name="xmi:id" select="concat(govobj:id-from-
      about(./owl:Ontology), '_model_annotation_details')"/>
(120)           <xsl:attribute name="key" select="'documentation'"/>
(121)           <xsl:attribute name="value">
(122)             <xsl:for-each select="owl:Ontology/rdfs:comment">
(123)               <xsl:value-of select="."/>
(124)               <xsl:if test="position()!=last()">
(125)                 <xsl:text> </xsl:text>
(126)               </xsl:if>
(127)             </xsl:for-each>
(128)           </xsl:attribute>
(129)         </xsl:element>
(130)       </xsl:element>
(131)
(132)       <!--  Tracing -->
(133)       <xsl:call-template name="traceTransformation">
(134)         <xsl:with-param name="transName" select="$transName"/>
(135)         <xsl:with-param name="transRule" select="'UmlModel'"/>
(136)         <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
(137)         <xsl:with-param name="dstName" select="'dstUmlModel'"/>
(138)         <xsl:with-param name="srcElement" select="$ref"/>
(139)         <xsl:with-param name="dstElement" select="$id"/>
(140)       </xsl:call-template>
(141)
(142)       <!--  Package und darin enthaltene Elemente unterhalb des Modells
      anlegen -->
(143)       <xsl:call-template name="packagedElementPackage"/>
(144)
(145)       <!--  Stereotypen anlegen -->
(146)       <xsl:call-template name="ProfileApplication"/>
(147)
(148)     </xsl:element>
(149)   </xsl:template>
(150)
(151)   <!--
(152)     Transformationsregel
(153)     ID:    XSLT_OWL-ONTOLOGY-PRM-02
(154)     Name:  UmlPackage
(155)    -->
(156)   <xsl:template name="packagedElementPackage">
(157)     <!-- Variablen belegen -->
(158)     <xsl:variable name="name" select="govobj:name-from-rdfslabel-or-
      about(./owl:Ontology)"/>
(159)     <xsl:variable name="id" select="concat(govobj:id-from-
      about(./owl:Ontology), '_package')"/>
(160)     <xsl:variable name="ref" select="govobj:ref(./owl:Ontology)"/>
(161)
(162)     <!--  Element erzeugen -->
(163)     <xsl:element name="packagedElement">
(164)       <xsl:attribute name="xmi:id" select="$id"/>
(165)       <xsl:attribute name="xmi:type" select="'uml:Package'"/>
(166)       <!--  Wenn kein Name als rdfs:label gesetzt ist, dann den Dateinamen
      für das Package verwenden -->
(167)       <xsl:attribute name="name">
(168)         <xsl:choose>
(169)           <xsl:when test="string-length(normalize-
      space(owl:Ontology/rdfs:label))=0">
(170)             <xsl:value-of select="functx:substring-after-last(substring-
      before(replace($srcMdl, '\\', '/'), '.owl'), '/')"/>
(171)           </xsl:when>
(172)           <xsl:otherwise>
(173)             <xsl:value-of select="normalize-
      space(owl:Ontology/rdfs:label)"/>
(174)           </xsl:otherwise>
(175)         </xsl:choose>
(176)         <xsl:text>-Package</xsl:text>
```

```
(177)       </xsl:attribute>
(178)
(179)       <!--  Tracing -->
(180)       <xsl:call-template name="traceTransformation">
(181)         <xsl:with-param name="transName" select="$transName"/>
(182)         <xsl:with-param name="transRule" select="'UmlPackage'"/>
(183)         <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
(184)         <xsl:with-param name="dstName" select="'dstUmlPackage'"/>
(185)         <xsl:with-param name="srcElement" select="$ref"/>
(186)         <xsl:with-param name="dstElement" select="$id"/>
(187)       </xsl:call-template>
(188)
(189)       <!--  Klassen, ihre Attribute und Generalisierungen -->
(190)       <xsl:call-template name="packagedElementClasses"/>
(191)
(192)       <!--  Assoziationen -->
(193)       <xsl:call-template name="packagedElementAssociations"/>
(194)
(195)       <!--  Contraints -->
(196)       <xsl:call-template name="owendRules"/>
(197)
(198)       <!--  Datentypen  -->
(199)       <xsl:call-template name="primitiveTypes"/>
(200)
(201)     </xsl:element>
(202)   </xsl:template>
(203)
(204)   <!--
(205)     Transformationsregel
(206)     ID:    XSLT_OWL-ONTOLOGY-PRM-06
(207)     Name:  UmlAttribut
(208)    -->
(209)   <xsl:template name="ownedAttributes">
(210)     <xsl:variable name="uri" select="@rdf:about"/>
(211)
(212)     <!-- owl:DataProperties  Attribute -->
(213)     <xsl:for-each
      select="//rdf:RDF/owl:DatatypeProperty/rdfs:domain[@rdf:resource=$uri]">
(214)
(215)       <xsl:variable name="name" select="govobj:name-from-rdfslabel-or-
      about(parent::*)"/>
(216)       <!-- xsl:variable name="id" select="concat(lower-case(substring-
      after($uri,'#')),'_', govobj:id-from-about(parent::*), '_attribute')"/ -->
(217)       <xsl:variable name="id" select="concat(lower-case(govobj:id-from-
      about(parent::*)), '_attribute')"/>
(218)       <xsl:variable name="ref" select="govobj:ref(parent::*)"/>
(219)       <xsl:variable name="type" select="substring-
      after(../rdfs:range/@rdf:resource, '#')"/>
(220)
(221)       <xsl:element name="ownedAttribute">
(222)         <xsl:attribute name="xmi:id" select="$id"/>
(223)         <xsl:attribute name="name" select="$name"/>
(224)         <xsl:attribute name="type">
(225)           <xsl:choose>
(226)             <xsl:when test="$type='boolean'">
(227)               <xsl:text>id-boolean</xsl:text>
(228)             </xsl:when>
(229)             <xsl:when test="$type='string'">
(230)               <xsl:text>id-string</xsl:text>
(231)             </xsl:when>
(232)             <xsl:when test="$type='integer'">
(233)               <xsl:text>id-integer</xsl:text>
(234)             </xsl:when>
(235)             <xsl:otherwise>
(236)               <xsl:text>id-unlimitednatural</xsl:text>
(237)             </xsl:otherwise>
(238)           </xsl:choose>
(239)         </xsl:attribute>
(240)
(241)         <xsl:element name="eAnnotations">
(242)           <xsl:attribute name="xmi:id" select="concat($id,'_annotation')"/>
(243)           <xsl:attribute name="source"
      select="'http://www.topcased.org/documentation'"/>
(244)           <xsl:element name="details">
```

```
(245)             <xsl:attribute name="xmi:id"
      select="concat($id,'_annotation_details')"/>
(246)             <xsl:attribute name="key" select="'documentation'"/>
(247)             <xsl:attribute name="value">
(248)               <xsl:for-each select="../rdfs:comment">
(249)                 <xsl:value-of select="."/>
(250)                 <xsl:if test="position()!=last()">
(251)                   <xsl:text> </xsl:text>
(252)                 </xsl:if>
(253)               </xsl:for-each>
(254)             </xsl:attribute>
(255)
(256)           </xsl:element>
(257)         </xsl:element>
(258)
(259)         <!--  Tracing -->
(260)         <xsl:call-template name="traceTransformation">
(261)           <xsl:with-param name="transName" select="$transName"/>
(262)           <xsl:with-param name="transRule" select="'UmlAttribute'"/>
(263)           <xsl:with-param name="srcName" select="'srcOwlDatatypeProperty'"/>
(264)           <xsl:with-param name="dstName" select="'dstUmlOwnedAttribute'"/>
(265)           <xsl:with-param name="srcElement" select="$ref"/>
(266)           <xsl:with-param name="dstElement" select="$id"/>
(267)         </xsl:call-template>
(268)
(269)       </xsl:element>
(270)     </xsl:for-each>
(271)   </xsl:template>
(272)
(273)   <!--
(274)     Transformationsregel
(275)     ID:    XSLT_OWL-ONTOLOGY-PRM-04a
(276)     Name:  UmlKlasseSingle
(277)     Hinweis:Bestandteil der Transformationsregel XSLT_OWL-ONTOLOGY-PRM-04
(278)   -->
(279)   <xsl:template name="packagedElementClass">
(280)     <xsl:variable name="name" select="govobj:name-from-rdfslabel-or-
      about(.)"/>
(281)     <xsl:variable name="id" select="concat(govobj:id-from-about(.),
      '_class')"/>
(282)     <xsl:variable name="ref" select="govobj:ref(.)"/>
(283)
(284)     <xsl:element name="packagedElement">
(285)       <xsl:attribute name="xmi:id" select="$id"/>
(286)       <xsl:attribute name="xmi:type" select="'uml:Class'"/>
(287)       <xsl:attribute name="name" select="$name"/>
(288)
(289)       <xsl:element name="eAnnotations">
(290)         <xsl:attribute name="xmi:id" select="concat($id,'_annotation')"/>
(291)         <xsl:attribute name="source"
      select="'http://www.topcased.org/documentation'"/>
(292)         <xsl:element name="details">
(293)           <xsl:attribute name="xmi:id"
      select="concat($id,'_annotation_details')"/>
(294)           <xsl:attribute name="key" select="'documentation'"/>
(295)           <xsl:attribute name="value">
(296)             <xsl:for-each select="rdfs:comment">
(297)               <xsl:value-of select="."/>
(298)               <xsl:if test="position()!=last()">
(299)                 <xsl:text> </xsl:text>
(300)               </xsl:if>
(301)             </xsl:for-each>
(302)           </xsl:attribute>
(303)         </xsl:element>
(304)       </xsl:element>
(305)
(306)       <!--  Tracing -->
(307)       <xsl:call-template name="traceTransformation">
(308)         <xsl:with-param name="transName" select="$transName"/>
(309)         <xsl:with-param name="transRule" select="'UmlClass'"/>
(310)         <xsl:with-param name="srcName" select="'srcOwlClass'"/>
(311)         <xsl:with-param name="dstName" select="'dstUmlClass'"/>
(312)         <xsl:with-param name="srcElement" select="$ref"/>
(313)         <xsl:with-param name="dstElement" select="$id"/>
(314)       </xsl:call-template>
```

```
(315)
(316)       <!--  Attribute der Klasse anlegen -->
(317)       <xsl:call-template name="ownedAttributes"/>
(318)
(319)       <!-- Generalisierung -->
(320)       <xsl:call-template name="generalizations">
(321)         <xsl:with-param name="id-cls" select="$id"/>
(322)         <xsl:with-param name="id-ref" select="$ref"/>
(323)       </xsl:call-template>
(324)
(325)     </xsl:element>
(326)   </xsl:template>
(327)
(328)   <!--
(329)     Transformationsregel
(330)     ID:    XSLT_OWL-ONTOLOGY-PRM-04
(331)     Name:  UmlKlasse
(332)   -->
(333)   <xsl:template name="packagedElementClasses">
(334)     <!--  Kommentar zur Orientierung in der XMI-Datei -->
(335)     <xsl:text>&#x0A;&#x9;</xsl:text>
(336)     <xsl:comment>
(337)       <xsl:text>owl:Classes</xsl:text>
(338)     </xsl:comment>
(339)     <xsl:text>&#x0A;&#x9;</xsl:text>
(340)
(341)     <xsl:for-each select="owl:Class">
(342)       <!-- Klasse anlegen -->
(343)       <xsl:call-template name="packagedElementClass"/>
(344)     </xsl:for-each>
(345)   </xsl:template>
(346)
(347)
(348)   <!--
(349)     Transformationsregel
(350)     ID:    XSLT_OWL-ONTOLOGY-PRM-08
(351)     Name:  UmlGeneralisierung
(352)    -->
(353)   <xsl:template name="generalizations">
(354)     <xsl:param name="id-cls"/>
(355)     <xsl:param name="id-ref"/>
(356)
(357)     <!-- Hinweis: Protege drückt Generalisierung in zwei verschiedene
      Varianten aus. -->
(358)     <!-- Varinante 1 -->
(359)     <xsl:for-each select="rdfs:subClassOf[@rdf:resource]">
(360)       <xsl:if test="@rdf:resource">
(361)         <xsl:variable name="ref" select="replace(substring-
      after(@rdf:resource, '#'),':','_')"/>
(362)         <xsl:variable name="gen" select="lower-case(concat(govobj:replace-
      umlaute($ref), '_class'))"/>
(363)         <xsl:variable name="id" select="concat($id-cls, '_is-a_', $gen)"/>
(364)
(365)         <xsl:element name="generalization">
(366)           <xsl:attribute name="xmi:id" select="$id"/>
(367)           <xsl:attribute name="general" select="$gen"/>
(368)         </xsl:element>
(369)
(370)         <!--  Tracing -->
(371)         <xsl:call-template name="traceTransformation">
(372)           <xsl:with-param name="transName" select="$transName"/>
(373)           <xsl:with-param name="transRule" select="'UmlGeneralisierung1'"/>
(374)           <xsl:with-param name="srcName" select="'srcRdfsubClassOf'"/>
(375)           <xsl:with-param name="dstName" select="'dstUmlGeneralization'"/>
(376)           <xsl:with-param name="srcElement" select="$id-ref"/>
(377)           <xsl:with-param name="dstElement" select="$id"/>
(378)         </xsl:call-template>
(379)       </xsl:if>
(380)     </xsl:for-each>
(381)
(382)     <!--  Variante 2 -->
(383)     <xsl:for-each select="rdfs:subClassOf/owl:Class[@rdf:about]">
(384)       <xsl:variable name="ref" select="replace(substring-after(@rdf:about,
      '#'),':','_')"/>
```

```
(385)       <xsl:variable name="gen" select="lower-case(concat(govobj:replace-
      umlaute($ref), '_class'))"/>
(386)       <xsl:variable name="id" select="concat($id-cls, '_is-a_', $gen)"/>
(387)
(388)       <xsl:element name="generalization">
(389)         <xsl:attribute name="xmi:id" select="$id"/>
(390)         <xsl:attribute name="general" select="$gen"/>
(391)       </xsl:element>
(392)
(393)       <!--  Tracing -->
(394)       <xsl:call-template name="traceTransformation">
(395)         <xsl:with-param name="transName" select="$transName"/>
(396)         <xsl:with-param name="transRule" select="'UmlGeneralisierung2'"/>
(397)         <xsl:with-param name="srcName" select="'srcRdfsubClassOf'"/>
(398)         <xsl:with-param name="dstName" select="'dstUmlGeneralization'"/>
(399)         <xsl:with-param name="srcElement" select="$id-ref"/>
(400)         <xsl:with-param name="dstElement" select="$id"/>
(401)       </xsl:call-template>
(402)
(403)     </xsl:for-each>
(404)   </xsl:template>
(405)
(406)   <!--
(407)     Transformationsregel
(408)     ID:  XSLT_OWL-ONTOLOGY-PRM-09
(409)     Name:  UmlConstraint
(410)    -->
(411)   <xsl:template name="owendRules">
(412)     <!--  Kommentar zur Orientierung in der XMI-Datei -->
(413)     <xsl:comment>
(414)       <xsl:text>owl:disjointWiths</xsl:text>
(415)     </xsl:comment>
(416)     <xsl:text>&#x0A;</xsl:text>
(417)
(418)     <xsl:for-each select="owl:Class[child::owl:disjointWith]">
(419)       <xsl:variable name="ref" select="lower-case(replace(substring-
      after(@rdf:about, '#'),':','_'))"/>
(420)       <xsl:variable name="class1" select="concat(govobj:replace-
      umlaute($ref), '_class')"/>
(421)       <xsl:variable name="class2">
(422)         <xsl:for-each select="owl:disjointWith">
(423)           <xsl:value-of select="concat(govobj:replace-umlaute(lower-
      case(replace(substring-after(@rdf:resource, '#'),':','_'))), '_class')"/>
(424)           <xsl:if test="position()!=last()">
(425)             <xsl:text> </xsl:text>
(426)           </xsl:if>
(427)         </xsl:for-each>
(428)       </xsl:variable>
(429)       <xsl:variable name="id" select="concat($class1, '_disjoint-with_',
      $class2, '_ownedRule')"/>
(430)
(431)       <xsl:element name="ownedRule">
(432)         <xsl:attribute name="xmi:id" select="$id"/>
(433)         <xsl:attribute name="name" select="'disjointWith'"/>
(434)         <xsl:attribute name="constrainedElement">
(435)           <xsl:value-of select="$class1"/>
(436)           <xsl:text> </xsl:text>
(437)           <xsl:value-of select="$class2"/>
(438)         </xsl:attribute>
(439)       </xsl:element>
(440)
(441)       <!--  Tracing -->
(442)       <xsl:call-template name="traceTransformation">
(443)         <xsl:with-param name="transName" select="$transName"/>
(444)         <xsl:with-param name="transRule" select="'UmlConstraint'"/>
(445)         <xsl:with-param name="srcName" select="'srcOwlDisjointWith'"/>
(446)         <xsl:with-param name="dstName" select="'dstUmlConstraint'"/>
(447)         <xsl:with-param name="srcElement" select="$ref"/>
(448)         <xsl:with-param name="dstElement" select="$id"/>
(449)       </xsl:call-template>
(450)
(451)     </xsl:for-each>
(452)   </xsl:template>
(453)
(454)   <!--
```

```
(455)     Transformationsregel
(456)     ID:    XSLT_OWL-ONTOLOGY-PRM-11a
(457)     Name:  UmlAssociationSingle
(458)     Hinweis:Bestandteil der Transformation XSLT_OWL-ONTOLOGY-PRM-11
(459)    -->
(460)   <xsl:template name="packagedElementAssociation">
(461)     <xsl:param name="id-domain" select="'unknown-domain-id'"/>
(462)     <xsl:param name="domain" select="'unknown-domain'"/>
(463)     <xsl:param name="name" select="'unknown-name'"/>
(464)     <xsl:param name="property" select="'unknown-property'"/>
(465)     <xsl:param name="id-value" select="'unknown-value'"/>
(466)     <xsl:param name="value" select="'unknown-value'"/>
(467)     <xsl:param name="ref" select="'unknown-ref'"/>
(468)
(469)     <!--  Variablen belegen -->
(470)     <xsl:variable name="id" select="concat(govobj:shorten($id-domain), '_',
      govobj:shorten($property), '_', govobj:shorten($id-value))"/>
(471)     <xsl:variable name="id-end1" select="concat($id, '_',
      govobj:shorten(lower-case($domain)), '_ownedEnd-1')"/>
(472)     <xsl:variable name="id-end2" select="concat($id, '_',
      govobj:shorten(lower-case($value)), '_ownedEnd-2')"/>
(473)
(474)     <!--  Kommentar zur Orientierung in der XMI-Datei -->
(475)     <xsl:text>&#x0A;&#x0A;</xsl:text>
(476)     <xsl:comment>
(477)       <xsl:text>owl:Restriction1 - Domain: </xsl:text>
(478)       <xsl:value-of select="$id-domain"/>
(479)       <xsl:text> - Property: </xsl:text>
(480)       <xsl:value-of select="$property"/>
(481)       <xsl:text> - Value: </xsl:text>
(482)       <xsl:value-of select="$value"/>
(483)     </xsl:comment>
(484)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(485)
(486)     <!-- Assoziation -->
(487)     <xsl:element name="packagedElement">
(488)       <xsl:attribute name="xmi:type">
(489)         <xsl:text>uml:Association</xsl:text>
(490)       </xsl:attribute>
(491)       <xsl:attribute name="xmi:id">
(492)         <xsl:value-of select="$id"/>
(493)       </xsl:attribute>
(494)       <xsl:attribute name="name">
(495)         <xsl:value-of select="$name"/>
(496)       </xsl:attribute>
(497)       <xsl:attribute name="memberEnd">
(498)         <xsl:value-of select="$id-end1"/>
(499)         <xsl:text> </xsl:text>
(500)         <xsl:value-of select="$id-end2"/>
(501)       </xsl:attribute>
(502)       <xsl:attribute name="navigableOwnedEnd">
(503)         <xsl:value-of select="$id-end2"/>
(504)       </xsl:attribute>
(505)
(506)       <!--  Tracing -->
(507)       <xsl:call-template name="traceTransformation">
(508)         <xsl:with-param name="transName" select="$transName"/>
(509)         <xsl:with-param name="transRule" select="'UmlAssociation'"/>
(510)         <xsl:with-param name="srcName" select="'srcOwlObjectProperty'"/>
(511)         <xsl:with-param name="dstName" select="'dstUmlAssociation'"/>
(512)         <xsl:with-param name="srcElement" select="$name"/>
(513)         <xsl:with-param name="dstElement" select="$id"/>
(514)       </xsl:call-template>
(515)
(516)       <!-- Assoziationsende 1 -->
(517)       <xsl:call-template name="ownedEnd">
(518)         <xsl:with-param name="id" select="$id-end1"/>
(519)         <xsl:with-param name="name" select="$domain"/>
(520)         <xsl:with-param name="type" select="concat($id-domain, '_class')"/>
(521)         <xsl:with-param name="association" select="$id"/>
(522)         <xsl:with-param name="upperValue">
(523)           <xsl:choose>
(524)             <xsl:when
      test="parent::*/parent::*/rdfs:subClassOf/owl:Restriction/owl:onProperty[@rd
      f:resource=concat('#', $property)]/parent::*/owl:maxCardinality">
```

```
(525)               <xsl:value-of
      select="parent::*/parent::*/rdfs:subClassOf/owl:Restriction/owl:onProperty[@
      rdf:resource=concat('#', $property)]/parent::*/owl:maxCardinality"/>
(526)             </xsl:when>
(527)             <xsl:when
      test="parent::*/child::*/owl:Restriction/owl:onProperty[@rdf:resource=concat
      ('#', $property)]">
(528)               <xsl:text>2</xsl:text>
(529)             </xsl:when>
(530)             <xsl:otherwise>
(531)               <xsl:text>-1</xsl:text>
(532)             </xsl:otherwise>
(533)           </xsl:choose>
(534)         </xsl:with-param>
(535)         <xsl:with-param name="lowerValue">
(536)           <xsl:choose>
(537)             <xsl:when
      test="parent::*/parent::*/rdfs:subClassOf/owl:Restriction/owl:onProperty[@rd
      f:resource=concat('#', $property)]/parent::*/owl:minCardinality">
(538)               <xsl:value-of
      select="parent::*/parent::*/rdfs:subClassOf/owl:Restriction/owl:onProperty[@
      rdf:resource=concat('#', $property)]/parent::*/owl:minCardinality"/>
(539)             </xsl:when>
(540)             <xsl:otherwise>
(541)               <xsl:text>-1</xsl:text>
(542)             </xsl:otherwise>
(543)           </xsl:choose>
(544)         </xsl:with-param>
(545)       </xsl:call-template>
(546)
(547)       <!--  Tracing -->
(548)       <xsl:call-template name="traceTransformation">
(549)         <xsl:with-param name="transName" select="$transName"/>
(550)         <xsl:with-param name="transRule" select="'UmlAssociationEnd'"/>
(551)         <xsl:with-param name="srcName" select="'srcOwlObjectProperty'"/>
(552)         <xsl:with-param name="dstName" select="'dstUmlAssociationEnd'"/>
(553)         <xsl:with-param name="srcElement" select="$name"/>
(554)         <xsl:with-param name="dstElement" select="$id-end1"/>
(555)       </xsl:call-template>
(556)
(557)       <!-- Assoziationsende 2 -->
(558)       <xsl:call-template name="ownedEnd">
(559)         <xsl:with-param name="id" select="$id-end2"/>
(560)         <xsl:with-param name="name" select="$value"/>
(561)         <xsl:with-param name="type" select="concat($id-value, '_class')"/>
(562)         <xsl:with-param name="association" select="$id"/>
(563)         <xsl:with-param name="upperValue">
(564)           <xsl:choose>
(565)             <xsl:when
      test="parent::*/parent::*/rdfs:subClassOf/owl:Restriction/owl:onProperty[@rd
      f:resource=concat('#', $property)]/parent::*/owl:maxCardinality">
(566)               <xsl:value-of
      select="parent::*/parent::*/rdfs:subClassOf/owl:Restriction/owl:onProperty[@
      rdf:resource=concat('#', $property)]/parent::*/owl:maxCardinality"/>
(567)             </xsl:when>
(568)             <xsl:otherwise>
(569)               <xsl:text>-1</xsl:text>
(570)             </xsl:otherwise>
(571)           </xsl:choose>
(572)         </xsl:with-param>
(573)         <xsl:with-param name="lowerValue">
(574)           <xsl:choose>
(575)             <xsl:when
      test="parent::*/parent::*/rdfs:subClassOf/owl:Restriction/owl:onProperty[@rd
      f:resource=concat('#', $property)]/parent::*/owl:minCardinality">
(576)               <xsl:value-of
      select="parent::*/parent::*/rdfs:subClassOf/owl:Restriction/owl:onProperty[@
      rdf:resource=concat('#', $property)]/parent::*/owl:minCardinality"/>
(577)             </xsl:when>
(578)             <xsl:otherwise>
(579)               <xsl:text>-1</xsl:text>
(580)             </xsl:otherwise>
(581)           </xsl:choose>
(582)         </xsl:with-param>
(583)       </xsl:call-template>
```

```
(584)
(585)       <!--  Tracing -->
(586)       <xsl:call-template name="traceTransformation">
(587)         <xsl:with-param name="transName" select="$transName"/>
(588)         <xsl:with-param name="transRule" select="'UmlAssociationEnd'"/>
(589)         <xsl:with-param name="srcName" select="'srcOwlObjectProperty'"/>
(590)         <xsl:with-param name="dstName" select="'dstUmlAssociationEnd'"/>
(591)         <xsl:with-param name="srcElement" select="$name"/>
(592)         <xsl:with-param name="dstElement" select="$id-end2"/>
(593)       </xsl:call-template>
(594)
(595)     </xsl:element>
(596)
(597)   </xsl:template>
(598)
(599)   <!--
(600)     Transformationsregel
(601)     ID:    XSLT_OWL-ONTOLOGY-PRM-11
(602)     Name:  UmlAssociation
(603)    -->
(604)   <xsl:template name="packagedElementAssociations">
(605)
(606)     <!--  Kommentar zur Orientierung in der XMI-Datei -->
(607)     <xsl:comment>
(608)       <xsl:text>owl:Restrictions</xsl:text>
(609)     </xsl:comment>
(610)     <xsl:text>&#x0A;</xsl:text>
(611)
(612)
(613)     <!--  ObjectProperties anlegen, die nicht als Sub-Klassen definiert sind
      -->
(614)     <xsl:for-each
      select="owl:ObjectProperty[not(rdfs:domain/owl:Class)][rdfs:domain]">
(615)       <!-- Variablen -->
(616)       <xsl:variable name="domain" select="govobj:replace-
      umlaute(replace(substring-after(rdfs:domain[1]/@rdf:resource,
      '#'),':','_'))"/>
(617)       <xsl:variable name="id-domain" select="lower-case($domain)"/>
(618)
(619)       <xsl:variable name="name" select="govobj:replace-
      umlaute(replace(substring-after(@rdf:about, '#'),':','_'))"/>
(620)       <xsl:variable name="property" select="lower-case($name)"/>
(621)
(622)       <xsl:variable name="value" select="govobj:replace-
      umlaute(replace(substring-after(rdfs:range[1]/@rdf:resource,
      '#'),':','_'))"/>
(623)       <xsl:variable name="id-value" select="lower-case($value)"/>
(624)
(625)       <!-- Bereits bearbeitete überspringen -->
(626)       <xsl:if
      test="not(//owl:Class/rdfs:subClassOf/owl:Restriction/owl:onProperty[@rdf:re
      source = concat('#', $property)])">
(627)
(628)         <!-- Assoziation anlegen -->
(629)         <xsl:call-template name="packagedElementAssociation">
(630)           <xsl:with-param name="id-domain" select="$id-domain"/>
(631)           <xsl:with-param name="domain" select="$domain"/>
(632)           <xsl:with-param name="name" select="$name"/>
(633)           <xsl:with-param name="property" select="$property"/>
(634)           <xsl:with-param name="value" select="$value"/>
(635)           <xsl:with-param name="id-value" select="$id-value"/>
(636)         </xsl:call-template>
(637)
(638)       </xsl:if>
(639)
(640)     </xsl:for-each>
(641)
(642)   </xsl:template>
(643)
(644)   <!--
(645)     Transformationsregel
(646)     ID:    XSLT_OWL-ONTOLOGY-PRM-11b
(647)     Name:  UmlAssociationOwnedEnd
(648)     Hinweis:Bestandteil der Abbildungsregel XSLT_OWL-ONTOLOGY-PRM-11
(649)    -->
```

```
(650)   <xsl:template name="ownedEnd">
(651)     <xsl:param name="id" select="'unknown-id'"/>
(652)     <xsl:param name="name" select="'unknown-name'"/>
(653)     <xsl:param name="type" select="'unknown-type'"/>
(654)     <xsl:param name="association" select="'unknown-association'"/>
(655)     <xsl:param name="upperValue" select="-1"/>
(656)     <xsl:param name="lowerValue" select="-1"/>
(657)
(658)     <!-- Assoziationsende -->
(659)     <xsl:element name="ownedEnd">
(660)       <xsl:attribute name="xmi:id" select="$id"/>
(661)       <xsl:attribute name="name" select="concat(lower-case(substring($name,
      1,1)), substring($name, 2))"/>
(662)       <xsl:attribute name="isUnique" select="'false'"/>
(663)       <xsl:attribute name="type" select="$type"/>
(664)       <xsl:attribute name="association" select="$association"/>
(665)
(666)       <!-- Assoziationsende - upperValue -->
(667)       <xsl:element name="upperValue">
(668)         <xsl:attribute name="xmi:id" select="concat($id, '_upper')"/>
(669)         <xsl:attribute name="xmi:type"
      select="'uml:LiteralUnlimitedNatural'"/>
(670)         <xsl:attribute name="value" select="$upperValue"/>
(671)       </xsl:element>
(672)
(673)       <!-- Assoziationsende - lowerValue -->
(674)       <xsl:element name="lowerValue">
(675)         <xsl:attribute name="xmi:id" select="concat($id, '_lower')"/>
(676)         <xsl:attribute name="xmi:type" select="'uml:LiteralInteger'"/>
(677)         <xsl:attribute name="value" select="$lowerValue"/>
(678)       </xsl:element>
(679)
(680)     </xsl:element>
(681)
(682)   </xsl:template>
(683)
(684)
(685)   <!--  Standard UML Datentypen -->
(686)   <xsl:template name="primitiveTypes">
(687)     <!--  Kommentar zur Orientierung in der XMI-Datei -->
(688)     <xsl:comment>
(689)       <xsl:text>uml:PrimitiveTypes</xsl:text>
(690)     </xsl:comment>
(691)     <xsl:text>&#x0A;</xsl:text>
(692)
(693)     <xsl:element name="packagedElement">
(694)       <xsl:attribute name="xmi:type">
(695)         <xsl:text>uml:PrimitiveType</xsl:text>
(696)       </xsl:attribute>
(697)       <xsl:attribute name="xmi:id">
(698)         <xsl:text>id-boolean</xsl:text>
(699)       </xsl:attribute>
(700)       <xsl:attribute name="name">
(701)         <xsl:text>Boolean</xsl:text>
(702)       </xsl:attribute>
(703)     </xsl:element>
(704)     <xsl:element name="packagedElement">
(705)       <xsl:attribute name="xmi:type">
(706)         <xsl:text>uml:PrimitiveType</xsl:text>
(707)       </xsl:attribute>
(708)       <xsl:attribute name="xmi:id">
(709)         <xsl:text>id-integer</xsl:text>
(710)       </xsl:attribute>
(711)       <xsl:attribute name="name">
(712)         <xsl:text>Integer</xsl:text>
(713)       </xsl:attribute>
(714)     </xsl:element>
(715)     <xsl:element name="packagedElement">
(716)       <xsl:attribute name="xmi:type">
(717)         <xsl:text>uml:PrimitiveType</xsl:text>
(718)       </xsl:attribute>
(719)       <xsl:attribute name="xmi:id">
(720)         <xsl:text>id-string</xsl:text>
(721)       </xsl:attribute>
(722)       <xsl:attribute name="name">
```

```
(723)         <xsl:text>String</xsl:text>
(724)       </xsl:attribute>
(725)     </xsl:element>
(726)     <xsl:element name="packagedElement">
(727)       <xsl:attribute name="xmi:type">
(728)         <xsl:text>uml:PrimitiveType</xsl:text>
(729)       </xsl:attribute>
(730)       <xsl:attribute name="xmi:id">
(731)         <xsl:text>id-unlimitednatural</xsl:text>
(732)       </xsl:attribute>
(733)       <xsl:attribute name="name">
(734)         <xsl:text>UnlimitedNatural</xsl:text>
(735)       </xsl:attribute>
(736)     </xsl:element>
(737)   </xsl:template>
(738)
(739)
(740)   <!--
(741)     Zuweisung des RDF- und OWL-Profils
(742)    -->
(743)   <xsl:template name="ProfileApplication">
(744)     <profileApplication xmi:id="id_profileAppl_1">
(745)       <eAnnotations xmi:id="id_profileAppl_1_Annotation"
      source="http://www.eclipse.org/uml2/2.0.0/UML">
(746)       <references xmi:type="ecore:EPackage"
      href="../ProfileModels/OwlUmlProfile.uml#_QwFo8CezEeC9_fKmAwBwrw"/>
(747)       </eAnnotations>
(748)       <appliedProfile href="../ProfileModels/OwlUmlProfile.uml#_XkGiwB07Ed-
      QQ4mYkrb7Gg"/>
(749)     </profileApplication>
(750)     <profileApplication xmi:id="id_profileAppl_2">
(751)       <eAnnotations xmi:id="id_profileAppl_2_Annotation"
      source="http://www.eclipse.org/uml2/2.0.0/UML">
(752)       <references xmi:type="ecore:EPackage"
      href="../ProfileModels/RdfUmlProfile.uml#_ZPpPMCezEeC9_fKmAwBwrw"/>
(753)       </eAnnotations>
(754)       <appliedProfile href="../ProfileModels/RdfUmlProfile.uml#_XkGiwB07Ed-
      QQ4mYkrb7Gg"/>
(755)     </profileApplication>
(756)
(757)   </xsl:template>
(758)
(759)   <!--
(760)     Transformationsregel
(761)     ID:    XSLT_OWL-ONTOLOGY-PRM-05a
(762)     Name:  UmlKlasseStereotypSingle
(763)     Hinweis:Bestandteil der Transformationsregel XSLT_OWL-ONTOLOGY-PRM-05
(764)    -->
(765)   <xsl:template name="owlClass">
(766)     <!-- Variablen belegen -->
(767)     <xsl:variable name="id" select="concat(replace(generate-id(),':','_'),
      '_owlClass')"/>
(768)     <xsl:variable name="name" select="govobj:id-from-about(.)"/>
(769)
(770)     <xsl:element name="OwlUmlProfile:owlClass">
(771)       <xsl:attribute name="xmi:id">
(772)         <xsl:value-of select="$id"/>
(773)       </xsl:attribute>
(774)       <xsl:attribute name="base_Class">
(775)         <xsl:value-of select="concat($name, '_class')"/>
(776)       </xsl:attribute>
(777)     </xsl:element>
(778)
(779)     <!--  Tracing -->
(780)     <xsl:call-template name="traceTransformation">
(781)       <xsl:with-param name="transName" select="$transName"/>
(782)       <xsl:with-param name="transRule" select="'UmlClassStereotype'"/>
(783)       <xsl:with-param name="srcName" select="'srcOwlClass'"/>
(784)       <xsl:with-param name="dstName" select="'dstUmlClassStereotype'"/>
(785)       <xsl:with-param name="srcElement" select="$name"/>
(786)       <xsl:with-param name="dstElement" select="$id"/>
(787)     </xsl:call-template>
(788)
(789)   </xsl:template>
(790)
```

```
(791)   <!--
(792)     Transformationsregel
(793)     ID:    XSLT_OWL-ONTOLOGY-PRM-05
(794)     Name:  UmlKlasseStereotyp
(795)    -->
(796)   <xsl:template name="owlClasses">
(797)     <!-- owl:Class -->
(798)     <xsl:for-each select="owl:Class">
(799)       <xsl:call-template name="owlClass"/>
(800)     </xsl:for-each>
(801)   </xsl:template>
(802)
(803)    <!--
(804)      Transformationsregel
(805)     ID:    XSLT_OWL-ONTOLOGY-PRM-10a
(806)     Name:  UmlConstraintStereotypSingle
(807)     Hinweis:Bestandteil der Transformationsregel XSLT_OWL-ONTOLOGY-PRM-10
(808)     -->
(809)    <xsl:template name="owlDisjointWith">
(810)     <xsl:variable name="id" select="concat(replace(generate-id(),':','_'),
      '_owlDisjointWith')"/>
(811)     <xsl:variable name="name" select="govobj:id-from-about(.)"/>
(812)
(813)     <xsl:variable name="ref" select="substring-after(@rdf:about, '#')"/>
(814)
(815)     <xsl:variable name="class1" select="concat(govobj:replace-umlaute(lower-
      case(replace(substring-after(./@rdf:about, '#'),':','_'))), '_class')"/>
(816)     <xsl:variable name="class2">
(817)       <xsl:for-each select="owl:disjointWith">
(818)         <xsl:value-of select="concat(govobj:replace-umlaute(lower-
      case(replace(substring-after(@rdf:resource, '#'),':','_'))), '_class')"/>
(819)         <xsl:if test="position()!=last()">
(820)           <xsl:text> </xsl:text>
(821)         </xsl:if>
(822)       </xsl:for-each>
(823)     </xsl:variable>
(824)     <xsl:variable name="base" select="concat($class1, '_disjoint-with_',
      $class2, '_ownedRule')"/>
(825)
(826)     <xsl:element name="OwlUmlProfile:disjointWith">
(827)       <xsl:attribute name="xmi:id" select="$id"/>
(828)       <xsl:attribute name="base_Constraint" select="$base"/>
(829)     </xsl:element>
(830)
(831)     <!--  Tracing -->
(832)     <xsl:call-template name="traceTransformation">
(833)       <xsl:with-param name="transName" select="$transName"/>
(834)       <xsl:with-param name="transRule" select="'UmlContraintStereotype'"/>
(835)       <xsl:with-param name="srcName" select="'srcOwlRestriction'"/>
(836)       <xsl:with-param name="dstName" select="'dstUmlContraintStereotype'"/>
(837)       <xsl:with-param name="srcElement" select="$ref"/>
(838)       <xsl:with-param name="dstElement" select="$id"/>
(839)     </xsl:call-template>
(840)    </xsl:template>
(841)
(842)    <!--
(843)      Transformationsregel
(844)     ID:    XSLT_OWL-ONTOLOGY-PRM-10
(845)     Name:  UmlConstraintStereotyp
(846)     -->
(847)    <xsl:template name="owlDisjointWiths">
(848)     <xsl:for-each select="owl:Class[child::owl:disjointWith]">
(849)       <xsl:call-template name="owlDisjointWith"/>
(850)     </xsl:for-each>
(851)    </xsl:template>
(852)
(853)   <!--
(854)     Transformationsregel
(855)     ID:    XSLT_OWL-ONTOLOGY-PRM-12a
(856)     Name:  UmlAssociationStereotypSingle
(857)     Hinweis:Bestandteil der Transformationsregel XSLT_OWL-ONTOLOGY-PRM-12
(858)    -->
(859)   <xsl:template name="owlObjectProperty">
(860)      <xsl:param name="domain" select="'unknown-domain'"/>
(861)      <xsl:param name="property" select="'unknown-property'"/>
```

```
(862)      <xsl:param name="value" select="'unknown-value'"/>
(863)
(864)     <xsl:variable name="id" select="concat(govobj:shorten($domain), '_',
      govobj:shorten($property), '_', govobj:shorten($value))"/>
(865)     <xsl:variable name="ref" select="substring-after(@rdf:about, '#')"/>
(866)
(867)     <xsl:element name="OwlUmlProfile:objectProperty">
(868)       <xsl:attribute name="xmi:id">
(869)         <xsl:value-of select="replace(generate-id(),':','_')"/>
(870)       </xsl:attribute>
(871)       <!-- xsl:attribute name="base_Class">
(872)         <xsl:value-of select="$id"/>
(873)       </xsl:attribute -->
(874)       <xsl:attribute name="base_Association"><xsl:value-of
      select="$id"/></xsl:attribute>
(875)       <!--  xsl:attribute name="base_AssociationClass"><xsl:value-of
      select="$id"/></xsl:attribute -->
(876)     </xsl:element>
(877)
(878)     <!--  Tracing -->
(879)     <xsl:call-template name="traceTransformation">
(880)       <xsl:with-param name="transName" select="$transName"/>
(881)       <xsl:with-param name="transRule" select="'UmlAssociationStereotyp'"/>
(882)       <xsl:with-param name="srcName" select="'srcOwlObjectProperty'"/>
(883)       <xsl:with-param name="dstName"
      select="'dstUmlAssociationStereotype'"/>
(884)       <xsl:with-param name="srcElement" select="$ref"/>
(885)       <xsl:with-param name="dstElement" select="$id"/>
(886)     </xsl:call-template>
(887)
(888)   </xsl:template>
(889)
(890)   <!--
(891)     Transformationsregel
(892)     ID:    XSLT_OWL-ONTOLOGY-PRM-12
(893)     Name:  UmlAssociationStereotyp
(894)    -->
(895)   <xsl:template name="owlObjectProperties">
(896)
(897)     <xsl:for-each
      select="owl:ObjectProperty[not(rdfs:domain/owl:Class)][rdfs:domain]">
(898)
(899)       <xsl:variable name="domain" select="govobj:replace-umlaute(lower-
      case(replace(substring-after(rdfs:domain[1]/@rdf:resource,
      '#'),':','_'))"/>
(900)       <xsl:variable name="property" select="govobj:id-from-about(.)"/>
(901)       <xsl:variable name="value" select="govobj:replace-umlaute(lower-
      case(replace(substring-after(rdfs:range[1]/@rdf:resource, '#'),':','_')))"/>
(902)
(903)       <xsl:call-template name="owlObjectProperty">
(904)         <xsl:with-param name="domain" select="$domain"/>
(905)         <xsl:with-param name="property" select="$property"/>
(906)         <xsl:with-param name="value" select="$value"/>
(907)       </xsl:call-template>
(908)
(909)     </xsl:for-each>
(910)
(911)   </xsl:template>
(912)
(913)   <!--
(914)     Transformationsregel
(915)     ID:    XSLT_OWL-ONTOLOGY-PRM-07a
(916)     Name:  UmlAttributStereotypSingle
(917)     Hinweis:Bestandteil der Transformationsregel XSLT_OWL-ONTOLOGY-PRM-07
(918)    -->
(919)   <xsl:template name="owlDatatypePropterty">
(920)      <!--  xsl:variable name="domain" select="govobj:replace-umlaute(lower-
      case(replace(substring-after(rdfs:domain[1]/@rdf:resource, '#'),':','_')))"/
      -->
(921)      <xsl:variable name="property" select="govobj:id-from-about(.)"/>
(922)      <xsl:variable name="ref" select="substring-after(@rdf:about, '#')"/>
(923)
(924)     <!--  xsl:variable name="id" select="concat($domain, '_', $property,
      '_attribute')"/ -->
(925)     <xsl:variable name="id" select="concat($property, '_attribute')"/>
```

```
(926)
(927)     <xsl:element name="OwlUmlProfile:datatypeProperty">
(928)       <xsl:attribute name="xmi:id" select="replace(generate-id(),':','_')"/>
(929)       <xsl:attribute name="base_Property" select="$id"/>
(930)     </xsl:element>
(931)
(932)     <!--  Tracing -->
(933)     <xsl:call-template name="traceTransformation">
(934)       <xsl:with-param name="transName" select="$transName"/>
(935)       <xsl:with-param name="transRule" select="'UmlAssociationStereotyp'"/>
(936)       <xsl:with-param name="srcName" select="'srcOwlDatatypeProperty'"/>
(937)       <xsl:with-param name="dstName" select="'dstUmlAttributeStereotype'"/>
(938)       <xsl:with-param name="srcElement" select="$ref"/>
(939)       <xsl:with-param name="dstElement" select="$id"/>
(940)     </xsl:call-template>
(941)
(942)   </xsl:template>
(943)
(944)   <!--
(945)     Transformationsregel
(946)     ID:    XSLT_OWL-ONTOLOGY-PRM-07
(947)     Name:  UmlAttributStereotyp
(948)    -->
(949)   <xsl:template name="owlDatatypeProperties">
(950)     <xsl:for-each select="owl:DatatypeProperty">
(951)       <xsl:call-template name="owlDatatypePropterty"/>
(952)     </xsl:for-each>
(953)   </xsl:template>
(954)
(955)   <!--
(956)     Transformationsregel
(957)     ID:    XSLT_OWL-ONTOLOGY-PRM-03
(958)     Name:  UmlPackageStereotyp
(959)    -->
(960)   <xsl:template name="owlOntology">
(961)     <xsl:variable name="id" select="concat(govobj:id-from-
      about(./owl:Ontology), '_package')"/>
(962)     <xsl:variable name="ref" select="govobj:ref(./owl:Ontology)"/>
(963)
(964)     <xsl:element name="OwlUmlProfile:owlOntology">
(965)       <xsl:attribute name="xmi:id" select="concat(replace(generate-
      id(),':','_'), '_package')"/>
(966)       <xsl:attribute name="base_Package" select="$id"/>
(967)     </xsl:element>
(968)
(969)     <!--  Tracing -->
(970)     <xsl:call-template name="traceTransformation">
(971)       <xsl:with-param name="transName" select="$transName"/>
(972)       <xsl:with-param name="transRule" select="'UmlPackageStereotyp'"/>
(973)       <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
(974)       <xsl:with-param name="dstName" select="'dstUmlPackageStereotype'"/>
(975)       <xsl:with-param name="srcElement" select="$ref"/>
(976)       <xsl:with-param name="dstElement" select="$id"/>
(977)     </xsl:call-template>
(978)   </xsl:template>
(979)
(980)   <!--
(981)     ODM-Stereotypen zuweisen
(982)    -->
(983)   <xsl:template name="ProfileRef">
(984)
(985)     <!-- Stereotyp owl:Ontology für das UML-Package -->
(986)     <xsl:call-template name="owlOntology"/>
(987)
(988)     <!-- Stereotyp owl:Class für UML-Klassen -->
(989)     <xsl:call-template name="owlClasses"/>
(990)
(991)     <!-- Stereotyp owl:datatypeProperty für UML-Attribute -->
(992)     <xsl:call-template name="owlDatatypeProperties"/>
(993)
(994)     <!-- Stereotyp owl:disjointWith für UML-Contraints -->
(995)     <xsl:call-template name="owlDisjointWiths"/>
(996)
(997)     <!-- Stereotyp owl:objectProperty für UML-Assoziationen und UML-
      Assoziationsklassen -->
```

```
(998)     <xsl:call-template name="owlObjectProperties"/>
(999)
(1000)   </xsl:template>
(1001)
(1002) </xsl:stylesheet>
```

Listing 50 – Transformation der OWL-Ontologie in ein UML-Modell (convert-OwlOntology2Uml.xsl)

D.2 OntoMat-Annotation

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <!--
(3)     Konvertiert Annotationen, die mit dem OntoMat erzeugt wurden nach
(4)     ECORE XMI für die Verwendung als UML-Modell basierend auf dem OUP.
(5)   -->
(6)   <xsl:stylesheet version="2.0"
(7)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(8)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(9)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(10)    xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
(11)    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(12)
      xmlns:OwlUmlProfile="http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/28
      "
(13)
      xmlns:RdfUmlProfile="http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/48
      "
(14)    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
(15)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
(16)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(17)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(18)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(19)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(20)    xmlns:owl="http://www.w3.org/2002/07/owl#"
(21)    xmlns:functx="http://www.functx.com">
(22)
(23)    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
(24)
(25)    <!-- Parameter -->
(26)    <!-- Dateiname inkl. relativem Pfad zum UML-Modell der Ontologie -->
(27)    <xsl:param name="ontology" select="'../Model/unknownOntologie.uml'"/>
(28)    <!-- Dateiname inkl. relativem Pfad zur Quell-Datei -->
(29)    <xsl:param name="srcMdl" select="'../Rdf/unknownRdfDocument.rdf'"/>
(30)    <!-- Dateiname inkl. relativem Pfad zur Ziel-Datei -->
(31)    <xsl:param name="dstMdl" select="'../Model/unknownUmlModel.uml'"/>
(32)
(33)    <!-- Name der Transformation zur Aufzeichnung der
      Verfolgbarkeitsbeziehungen -->
(34)    <xsl:variable name="transName" select="'owl2prm_annotation'"/>
(35)
(36)    <!-- Import der Hilfsfunktionen -->
(37)    <xsl:include href="xsltrace.xsl"/>
(38)    <xsl:include href="convert-OwlCommon.xsl"/>
(39)
(40)    <!--  Wurzelknoten -->
(41)    <xsl:template match="/rdf:RDF">
(42)
(43)      <xsl:call-template name="initTracing">
(44)        <xsl:with-param name="transName" select="$transName"/>
(45)      </xsl:call-template>
(46)
(47)      <!-- XMI inkl. notwendiger Namespaces und Schema-Verweisen auf OWL- und
      RDF-Profile anlegen -->
(48)      <xsl:element name="xmi:XMI">
(49)        <xsl:attribute name="xmi:version">
(50)        <xsl:text>2.1</xsl:text>
(51)        </xsl:attribute>
(52)        <xsl:namespace name="xmi"
      select="'http://schema.omg.org/spec/XMI/2.1'"/>
(53)        <xsl:namespace name="xsi" select="'http://www.w3.org/2001/XMLSchema-
      instance'"/>
(54)        <xsl:namespace name="OwlUmlProfile"
      select="'http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/28'"/>
```

```
(55)        <xsl:namespace name="RdfUmlProfile"
      select="'http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/48'"/>
(56)        <xsl:namespace name="ecore"
      select="'http://www.eclipse.org/emf/2002/Ecore'"/>
(57)        <xsl:namespace name="uml"
      select="'http://www.eclipse.org/uml2/3.0.0/UML'"/>
(58)        <xsl:attribute name="xsi:schemaLocation">
(59)          <xsl:text>http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/28
      ../ProfileModels/OwlUmlProfile.uml#_QwFo8CezEeC9_fKmAwBwrw
      http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/48
      ../ProfileModels/RdfUmlProfile.uml#_ZPpPMCezEeC9_fKmAwBwrw</xsl:text>
(60)        </xsl:attribute>
(61)
(62)        <!--  Modell und alle darin enthaltenen Elemente anlegen -->
(63)        <xsl:call-template name="model"/>
(64)
(65)        <!--  Stereotypen anlegen -->
(66)        <xsl:call-template name="ProfileRef"/>
(67)      </xsl:element>
(68)
(69)      <xsl:call-template name="finalizeTracing"/>
(70)
(71)    </xsl:template>
(72)
(73)    <!--
(74)      Abbildungsregel
(75)      ID:    XSLT_OWL-ANNOTATION-PRM-01
(76)      Name:  UmlModell
(77)     -->
(78)    <xsl:template name="model">
(79)      <xsl:variable name="id" select="concat(govobj:id-from-
      about(./owl:Ontology), '_model')"/>
(80)      <xsl:variable name="ref" select="govobj:ref(./owl:Ontology)"/>
(81)
(82)      <xsl:element name="uml:Model">
(83)        <xsl:attribute name="xmi:id" select="$id"/>
(84)        <xsl:attribute name="name" select="concat(substring-
      before(functx:substring-after-last(owl:Ontology[1]/@rdf:about, '/'), '#'),
      '-Modell')"/>
(85)
(86)        <!--  Tracing -->
(87)        <xsl:call-template name="traceTransformation">
(88)          <xsl:with-param name="transName" select="$transName"/>
(89)          <xsl:with-param name="transRule" select="'UmlModel'"/>
(90)          <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
(91)          <xsl:with-param name="dstName" select="'dstUmlModel'"/>
(92)          <xsl:with-param name="srcElement" select="$ref"/>
(93)          <xsl:with-param name="dstElement" select="$id"/>
(94)        </xsl:call-template>
(95)
(96)        <!-- Enthaltenes Package anlegen -->
(97)        <xsl:call-template name="packagedElementPackage"/>
(98)
(99)        <!-- OWL- und RDF-Profil auf Modell anwenden -->
(100)       <xsl:call-template name="ProfileApplication"/>
(101)
(102)     </xsl:element>
(103)
(104)   </xsl:template>
(105)
(106)   <!--
(107)     Abbildungsregel
(108)     ID:    XSLT_OWL-ANNOTATION-PRM-02
(109)     Name:  UmlPackage
(110)    -->
(111)   <xsl:template name="packagedElementPackage">
(112)     <xsl:variable name="id" select="concat(govobj:id-from-
      about(./owl:Ontology), '_package')"/>
(113)     <xsl:variable name="ref" select="govobj:ref(./owl:Ontology)"/>
(114)
(115)     <xsl:element name="packagedElement">
(116)       <xsl:attribute name="xmi:id" select="$id"/>
(117)       <xsl:attribute name="xmi:type" select="'uml:Package'"/>
```

```
(118)       <xsl:attribute name="name" select="concat(substring-
      before(functx:substring-after-last(owl:Ontology[1]/@rdf:about, '/'), '#'),
      '-Package')"/>
(119)
(120)       <!--  Tracing -->
(121)       <xsl:call-template name="traceTransformation">
(122)         <xsl:with-param name="transName" select="$transName"/>
(123)         <xsl:with-param name="transRule" select="'UmlModel'"/>
(124)         <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
(125)         <xsl:with-param name="dstName" select="'dstUmlPackage'"/>
(126)         <xsl:with-param name="srcElement" select="$ref"/>
(127)         <xsl:with-param name="dstElement" select="$id"/>
(128)       </xsl:call-template>
(129)
(130)       <!--  Enthaltene Elemente anlegen -->
(131)       <xsl:call-template name="packagedElements"/>
(132)
(133)     </xsl:element>
(134)
(135)   </xsl:template>
(136)
(137)   <!--
(138)     Abbildungsregel
(139)     ID:    XSLT_OWL-ANNOTATION_PRM-04a
(140)     Name:  UmlInstanceSpecificationSingle
(141)     Hinweis:Bestandteil der Abbildungsregel XSLT_OWL-ANNOTATION_PRM-04
(142)    -->
(143)   <xsl:template name="instanceSpecification">
(144)     <xsl:variable name="instance" select="govobj:id-from-about(.)"/>
(145)     <xsl:variable name="instance-ref" select="govobj:ref(.)"/>
(146)     <xsl:variable name="instance-name" select="govobj:name-from-rdfslabel-
      or-about(.)"/>
(147)     <xsl:variable name="class" select="local-name()"/>
(148)     <xsl:variable name="instance-id" select="concat(govobj:generate-id(.),
      '_individual')"/>
(149)
(150)     <!--  Tracing -->
(151)     <xsl:call-template name="traceTransformation">
(152)       <xsl:with-param name="transName" select="$transName"/>
(153)       <xsl:with-param name="transRule" select="'UmlInstanceSpecification'"/>
(154)       <xsl:with-param name="srcName" select="'srcOwlIndividual'"/>
(155)       <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecification'"/>
(156)       <xsl:with-param name="srcElement" select="$instance-ref"/>
(157)       <xsl:with-param name="dstElement" select="$instance-id"/>
(158)     </xsl:call-template>
(159)
(160)     <!-- Kommentar zur Orientierung im erzeugten XMI -->
(161)     <xsl:text>&#x0A;</xsl:text>
(162)     <xsl:comment>
(163)       <xsl:value-of select="$instance"/>
(164)       <xsl:text> (</xsl:text>
(165)       <xsl:value-of select="$class"/>
(166)       <xsl:text>)</xsl:text>
(167)     </xsl:comment>
(168)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(169)
(170)     <!--  uml:InstanceSpecification -->
(171)     <xsl:element name="packagedElement">
(172)       <xsl:attribute name="xmi:id" select="$instance-id"/>
(173)       <xsl:attribute name="xmi:type" select="'uml:InstanceSpecification'"/>
(174)       <xsl:attribute name="name" select="govobj:pascalize-on-
      space($instance-name, ' ')"/>
(175)
(176)       <!-- Referenz auf Classifier -->
(177)       <xsl:element name="classifier">
(178)         <xsl:attribute name="xmi:type" select="'uml:Class'"/>
(179)         <xsl:attribute name="href" select="concat($ontology, '#', lower-
      case($class), '_class')"/>
(180)       </xsl:element>
(181)
(182)       <!-- uml:Slots -->
(183)       <xsl:for-each select="govobjects-ontology:*[@rdf:datatype][string-
      length(normalize-space(.))>0]">
(184)         <!-- Variablen -->
```

```
(185)         <xsl:variable name="slot-id" select="concat(replace(generate-
      id(),':','_'),'_slot',position())"/>
(186)         <xsl:variable name="property" select="lower-case(local-name())"/>
(187)         <xsl:variable name="property-id" select="concat($instance-ref, '/',
      $property)"/>
(188)         <xsl:variable name="value" select="normalize-space(.)"/>
(189)
(190)         <!--  Tracing -->
(191)         <xsl:call-template name="traceTransformation">
(192)           <xsl:with-param name="transName" select="$transName"/>
(193)           <xsl:with-param name="transRule"
      select="'UmlInstanceSpecificationSlot'"/>
(194)           <xsl:with-param name="srcName" select="'srcOwlIndividual'"/>
(195)           <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecificationSlot'"/>
(196)           <xsl:with-param name="srcElement" select="$property-id"/>
(197)           <xsl:with-param name="dstElement" select="$slot-id"/>
(198)           <xsl:with-param name="check-sum" select="round(string-
      length($value)*1000 div functx:word-count($value))"/>
(199)         </xsl:call-template>
(200)
(201)         <!-- Element -->
(202)         <xsl:element name="slot">
(203)           <xsl:attribute name="xmi:id" select="$slot-id"/>
(204)           <xsl:element name="definingFeature">
(205)             <xsl:attribute name="xmi:type"  select="'uml:Property'"/>
(206)             <xsl:attribute name="href" select="concat($ontology, '#',
      $property, '_attribute')"/>
(207)           </xsl:element>
(208)           <xsl:element name="value">
(209)             <xsl:attribute name="xmi:type">
(210)               <!--  Weitere Datentypen analog zu diesem Muster -->
(211)               <xsl:choose>
(212)                 <xsl:when test="ends-with(@rdf:datatype, 'string')">
(213)                   <xsl:text>uml:LiteralString</xsl:text>
(214)                 </xsl:when>
(215)                 <xsl:when test="ends-with(@rdf:datatype, 'boolean')">
(216)                   <xsl:text>uml:LiteralBoolean</xsl:text>
(217)                 </xsl:when>
(218)                 <xsl:otherwise>
(219)                   <xsl:text>uml:LiteralString</xsl:text>
(220)                 </xsl:otherwise>
(221)               </xsl:choose>
(222)             </xsl:attribute>
(223)             <xsl:attribute name="xmi:id" select="concat($slot-
      id,'_value')"/>
(224)             <xsl:attribute name="value" select="$value"/>
(225)           </xsl:element>
(226)         </xsl:element>
(227)       </xsl:for-each>
(228)     </xsl:element>
(229)   </xsl:template>
(230)
(231)   <!--
(232)     Abbildungsregel
(233)     ID:    XSLT_OWL-ANNOTATION_PRM-04
(234)     Name:  UmlInstanceSpecification
(235)    -->
(236)   <xsl:template name="instanceSpecifications">
(237)
(238)     <!-- Kommentar zur Orientierung im erzeugten XMI -->
(239)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(240)     <xsl:comment>InstanceSpecifications</xsl:comment>
(241)     <xsl:text>&#9;&#9;</xsl:text>
(242)
(243)     <!-- Jeweils eine Instanz als Element anlegen -->
(244)     <xsl:for-each select="//govobjects-ontology:*[@rdf:about]">
(245)       <xsl:call-template name="instanceSpecification"/>
(246)     </xsl:for-each>
(247)
(248)   </xsl:template>
(249)
(250)   <!--
(251)     Abbildungsregel
(252)     ID:    XSLT_OWL-PRM-06
```

```
(253)     Name:  UmlInstanceSpecificationLink
(254)    -->
(255)   <xsl:template name="instanceSpecificationLinks">
(256)
(257)     <!-- Kommentar zur Orientierung im erzeugten XMI -->
(258)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(259)     <xsl:comment>InstanceSpecificationsLinks</xsl:comment>
(260)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(261)
(262)     <xsl:for-each select="//govobjects-ontology:*[@rdf:about]">
(263)       <!--  Variablen belegen -->
(264)       <xsl:variable name="instance" select="govobj:name-from-rdfslabel-or-
      about(.)"/>
(265)       <xsl:variable name="instance-ref" select="govobj:id-from-about(.)"/>
(266)       <xsl:variable name="class" select="local-name()"/>
(267)       <xsl:variable name="instance-id" select="concat(replace(generate-
      id(),':','_'), '_individual')"/>
(268)
(269)       <!-- Jeweils eine Instanz als Element anlegen -->
(270)       <xsl:for-each select="child::govobjects-
      ontology:*[not(@rdf:datatype)]">
(271)         <xsl:call-template name="instanceSpecificationLink">
(272)           <xsl:with-param name="instance" select="$instance"/>
(273)           <xsl:with-param name="instance-id" select="$instance-id"/>
(274)           <xsl:with-param name="instance-ref" select="$instance-ref"/>
(275)           <xsl:with-param name="class" select="$class"/>
(276)         </xsl:call-template>
(277)       </xsl:for-each>
(278)     </xsl:for-each>
(279)   </xsl:template>
(280)
(281)   <!--
(282)     Abbildungsregel
(283)     ID:    XSLT_OWL-PRM-06a
(284)     Name:  UmlInstanceSpecificationLinkSingle
(285)     Hinweis:Bestandteil der Abbildungsregel XSLT_OWL-PRM-06
(286)    -->
(287)   <xsl:template name="instanceSpecificationLink">
(288)
(289)     <xsl:param name="instance" select="'unknown-instance'"/>
(290)     <xsl:param name="instance-id" select="'unknown-instance-id'"/>
(291)     <xsl:param name="instance-ref" select="'unknown-instance-ref'"/>
(292)     <xsl:param name="class" select="'unknown-class'"/>
(293)
(294)     <xsl:variable name="property" select="local-name()"/>
(295)     <xsl:variable name="domain" select="substring-
      before(document($ontology)/xmi:XMI/uml:Model//packagedElement[@name=$propert
      y]/ownedEnd[1]/@type, '_class')"/>
(296)     <xsl:variable name="range" select="substring-
      before(document($ontology)/xmi:XMI/uml:Model//packagedElement[@name=$propert
      y]/ownedEnd[2]/@type, '_class')"/>
(297)     <xsl:variable name="resource-ref">
(298)       <xsl:choose>
(299)         <xsl:when test="govobjects-ontology:*">
(300)           <xsl:value-of select="child::*/@rdf:about"/>
(301)         </xsl:when>
(302)         <xsl:otherwise>
(303)           <xsl:value-of select="@rdf:resource"/>
(304)         </xsl:otherwise>
(305)       </xsl:choose>
(306)     </xsl:variable>
(307)     <xsl:variable name="class-ref">
(308)       <xsl:choose>
(309)         <xsl:when test="govobjects-ontology:*">
(310)           <xsl:value-of select="local-name(child::*[1])"/>
(311)         </xsl:when>
(312)         <xsl:otherwise>
(313)           <xsl:value-of select="local-name(//govobjects-
      ontology:*[@rdf:about =$resource-ref])"/>
(314)         </xsl:otherwise>
(315)       </xsl:choose>
(316)     </xsl:variable>
(317)     <xsl:variable name="class-ref-id" select="concat(replace(generate-
      id(//govobjects-ontology:*[@rdf:about =$resource-ref]),':','_'),
      '_individual')"/>
```

```
(318)     <xsl:variable name="property-id" select="concat(replace(generate-
      id(),':','_'), '_instance-specification-link')"/>
(319)
(320)     <!--  Tracing -->
(321)     <xsl:call-template name="traceTransformation">
(322)       <xsl:with-param name="transName" select="$transName"/>
(323)       <xsl:with-param name="transRule"
      select="'UmlInstanceSpecificationLink'"/>
(324)       <xsl:with-param name="srcName" select="'srcOwlIndividualProperty'"/>
(325)       <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecificationLink'"/>
(326)       <xsl:with-param name="srcElement" select="concat($instance-ref, '/',
      $property, '/#', substring-after($resource-ref, '#'))"/>
(327)       <xsl:with-param name="dstElement" select="$property-id"/>
(328)     </xsl:call-template>
(329)
(330)     <!--  Kommentar zur Orientierung im XMI -->
(331)     <xsl:text>&#x0A;&#9;</xsl:text>
(332)     <xsl:comment>
(333)       <xsl:value-of select="concat($instance, ' (', $domain, ') ',
      $property, ' ', substring-after($resource-ref, '#'), ' (', $range, ')')"/>
(334)     </xsl:comment>
(335)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(336)
(337)     <xsl:element name="packagedElement">
(338)       <xsl:attribute name="xmi:type"  select="'uml:InstanceSpecification'"/>
(339)       <xsl:attribute name="xmi:id" select="$property-id"/>
(340)       <xsl:attribute name="name" select="concat(govobj:pascalize-on-
      space($instance, ' '), '(', $domain, ')_', $property, '_',
      govobj:pascalize(substring-after($resource-ref, '#')), '(', $range, ')')"/>
(341)       <xsl:variable name="ass-clsf" select="concat(govobj:shorten(lower-
      case($domain)), '_', govobj:shorten(lower-case($property)), '_',
      govobj:shorten(lower-case($range)))"/>
(342)
(343)       <xsl:element name="classifier">
(344)         <xsl:attribute name="xmi:type" select="'uml:Association'"/>
(345)         <xsl:attribute name="href" select="concat($ontology, '#', $ass-
      clsf)"/>
(346)       </xsl:element>
(347)
(348)       <xsl:call-template name="instanceSpecificationLinkSlot">
(349)         <xsl:with-param name="id" select="concat(replace(generate-
      id(),':','_'), '_slot1')"/>
(350)         <xsl:with-param name="definingFeature" select="concat($ontology,
      '#', $ass-clsf, '_', govobj:shorten($domain) , '_ownedEnd-1')"/>
(351)         <xsl:with-param name="type" select="concat($ontology, '#', lower-
      case($domain), '_class')"/>
(352)         <xsl:with-param name="instance" select="$instance-id"/>
(353)       </xsl:call-template>
(354)
(355)       <xsl:call-template name="instanceSpecificationLinkSlot">
(356)         <xsl:with-param name="id" select="concat(replace(generate-
      id(),':','_'), '_slot2')"/>
(357)         <xsl:with-param name="definingFeature" select="concat($ontology,
      '#', $ass-clsf, '_', govobj:shorten($range) , '_ownedEnd-2')"/>
(358)         <xsl:with-param name="type" select="concat($ontology, '#', lower-
      case($range), '_class')"/>
(359)         <xsl:with-param name="instance" select="$class-ref-id"/>
(360)       </xsl:call-template>
(361)
(362)     </xsl:element>
(363)   </xsl:template>
(364)
(365)   <!--
(366)     Abbildungsregel
(367)     ID:    XSLT_OWL-PRM-06b
(368)     Name:  UmlInstanceSpecificationLinkSlot
(369)     Hinweis:Bestandteil der Abbildungsregel XSLT_OWL-PRM-06
(370)    -->
(371)   <xsl:template name="instanceSpecificationLinkSlot">
(372)     <xsl:param name="id"/>
(373)     <xsl:param name="definingFeature"/>
(374)     <xsl:param name="type"/>
(375)     <xsl:param name="instance"/>
(376)
```

```
(377)     <xsl:element name="slot">
(378)       <xsl:attribute name="xmi:id" select="$id"/>
(379)       <xsl:element name="definingFeature">
(380)         <xsl:attribute name="xmi:type" select="'uml:Property'"/>
(381)         <xsl:attribute name="href" select="$definingFeature"/>
(382)       </xsl:element>
(383)       <xsl:element name="value">
(384)         <xsl:attribute name="xmi:type" select="'uml:InstanceValue'"/>
(385)         <xsl:attribute name="xmi:id" select="concat($id, '_value')"/>
(386)         <xsl:attribute name="instance" select="$instance"/>
(387)         <xsl:element name="type">
(388)           <xsl:attribute name="xmi:type" select="'uml:Class'"/>
(389)           <xsl:attribute name="href" select="$type"/>
(390)         </xsl:element>
(391)       </xsl:element>
(392)     </xsl:element>
(393)
(394)   </xsl:template>
(395)
(396)   <!--  Elemente des Packages anlegen -->
(397)   <xsl:template name="packagedElements">
(398)
(399)     <!-- owl:Individual -> uml:InstanceSpecification -->
(400)     <xsl:call-template name="instanceSpecifications"/>
(401)
(402)     <!-- owl:ObjectProperties -> uml:InstanceLink -->
(403)     <xsl:call-template name="instanceSpecificationLinks"/>
(404)
(405)   </xsl:template>
(406)
(407)   <!--
(408)     Zuweisung des RDF- und OWL-Profils
(409)    -->
(410)   <xsl:template name="ProfileApplication">
(411)     <profileApplication xmi:id="id_profileAppl_1">
(412)       <eAnnotations xmi:id="id_profileAppl_1_Annotation"
      source="http://www.eclipse.org/uml2/2.0.0/UML">
(413)       <references xmi:type="ecore:EPackage"
      href="../ProfileModels/OwlUmlProfile.uml#_QwFo8CezEeC9_fKmAwBwrw"/>
(414)       </eAnnotations>
(415)       <appliedProfile href="../ProfileModels/OwlUmlProfile.uml#_XkGiwB07Ed-
      QQ4mYkrb7Gg"/>
(416)     </profileApplication>
(417)     <profileApplication xmi:id="id_profileAppl_2">
(418)       <eAnnotations xmi:id="id_profileAppl_2_Annotation"
      source="http://www.eclipse.org/uml2/2.0.0/UML">
(419)       <references xmi:type="ecore:EPackage"
      href="../ProfileModels/RdfUmlProfile.uml#_ZPpPMCezEeC9_fKmAwBwrw"/>
(420)       </eAnnotations>
(421)       <appliedProfile href="../ProfileModels/RdfUmlProfile.uml#_XkGiwB07Ed-
      QQ4mYkrb7Gg"/>
(422)     </profileApplication>
(423)   </xsl:template>
(424)
(425)   <!--
(426)     Abbildungsregel
(427)     ID:    XSLT_OWL-ONTOLOGY-PRM-03
(428)     Name:  UmlPackageStereotyp
(429)    -->
(430)   <xsl:template name="owlOntology">
(431)     <xsl:variable name="id" select="concat(govobj:id-from-
      about(./owl:Ontology), '_package')"/>
(432)     <xsl:variable name="ref" select="govobj:ref(./owl:Ontology)"/>
(433)
(434)     <xsl:element name="OwlUmlProfile:owlOntology">
(435)       <xsl:attribute name="xmi:id" select="concat(replace(generate-
      id(),':','_'), '_owl-ontology')"/>
(436)       <xsl:attribute name="base_Package" select="$id"/>
(437)     </xsl:element>
(438)
(439)     <!--  Tracing -->
(440)     <xsl:call-template name="traceTransformation">
(441)       <xsl:with-param name="transName" select="$transName"/>
(442)       <xsl:with-param name="transRule" select="'UmlPackageStereotyp'"/>
(443)       <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
```

```
(444)       <xsl:with-param name="dstName" select="'dstUmlPackageStereotype'"/>
(445)       <xsl:with-param name="srcElement" select="$ref"/>
(446)       <xsl:with-param name="dstElement" select="$id"/>
(447)     </xsl:call-template>
(448)   </xsl:template>
(449)
(450)   <!--
(451)     Abbildungsregel
(452)     ID:    XSLT_OWL-ANNOTATION_PRM-05
(453)     Name:  UmlInstanceSpecificationStereotyp
(454)    -->
(455)   <xsl:template name="owlIndividual">
(456)     <xsl:for-each select="//govobjects-ontology:*[@rdf:about]">
(457)       <xsl:variable name="id" select="concat(replace(generate-id(),':','_'),
      '_individual')"/>
(458)       <xsl:variable name="ref" select="govobj:ref(.)"/>
(459)
(460)       <xsl:element name="OwlUmlProfile:owlIndividual">
(461)         <xsl:attribute name="xmi:id" select="concat(replace(generate-
      id(),':','_'), '_owl-individual')"/>
(462)         <xsl:attribute name="base_InstanceSpecification" select="$id"/>
(463)       </xsl:element>
(464)
(465)       <!--  Tracing -->
(466)       <xsl:call-template name="traceTransformation">
(467)         <xsl:with-param name="transName" select="$transName"/>
(468)         <xsl:with-param name="transRule" select="'UmlIndividualStereotyp'"/>
(469)         <xsl:with-param name="srcName" select="'srcOwlIndividual'"/>
(470)         <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecificationStereotype'"/>
(471)         <xsl:with-param name="srcElement" select="$ref"/>
(472)         <xsl:with-param name="dstElement" select="$id"/>
(473)       </xsl:call-template>
(474)     </xsl:for-each>
(475)   </xsl:template>
(476)
(477)   <!--
(478)     ODM-Stereotypen zuweisen
(479)    -->
(480)   <xsl:template name="ProfileRef">
(481)
(482)     <!-- Stereotyp owl:Ontology für das UML-Package -->
(483)     <xsl:call-template name="owlOntology"/>
(484)
(485)     <!-- Stereotyp owl:Individual für die UML-Instanzen-->
(486)     <xsl:call-template name="owlIndividual"/>
(487)
(488)   </xsl:template>
(489)
(490)   <!--
(491)     The source code for this function 'functx:word-count' is available under
      the GNU LGPL license.
(492)     siehe: http://www.gnu.org/licenses/lgpl.html
(493)     Autor: Priscilla Walmsley, Datypic, pwalmsley@datypic.com,
      http://www.datypic.com
(494)     Quelle: http://www.xsltfunctions.com/xsl/functx_word-count.html
(495)   -->
(496)   <xsl:function name="functx:word-count" as="xs:integer"
      xmlns:functx="http://www.functx.com" >
(497)     <xsl:param name="arg" as="xs:string?"/>
(498)     <xsl:sequence select="count(tokenize($arg, '\W+')[. != ''])"/>
(499)   </xsl:function>
(500)
(501)
(502) </xsl:stylesheet>
```

Listing 51 – Transformation der OntoMat-Annotationen in ein UML-Modell (convert-OwlAnnotation2Uml.xsl)

D.3 RDFa-Annotationen

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <!--
(3)     Konvertiert Annotationen, die mit dem OntoMat erzeugt wurden nach
(4)     ECORE XMI für die Verwendung als UML-Modell basierend auf dem OUP.
(5)   -->
```

```
(6)   <xsl:stylesheet version="2.0"
(7)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(8)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(9)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(10)    xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
(11)    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(12)
      xmlns:OwlUmlProfile="http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/28
      "
(13)
      xmlns:RdfUmlProfile="http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/48
      "
(14)    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
(15)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
(16)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(17)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(18)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(19)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(20)    xmlns:owl="http://www.w3.org/2002/07/owl#"
(21)    xmlns:functx="http://www.functx.com"
(22)    xmlns:html="http://www.w3.org/1999/xhtml">
(23)
(24)    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
(25)
(26)    <!-- Parameter -->
(27)    <!-- Dateiname inkl. relativem Pfad zum UML-Modell der Ontologie -->
(28)    <xsl:param name="ontology" select="'../Model/unknownOntologie.uml'"/>
(29)    <!-- Dateiname inkl. relativem Pfad zur Quell-Datei -->
(30)    <xsl:param name="srcMdl" select="'../Rdf/unknownRdfDocument.rdf'"/>
(31)    <!-- Dateiname inkl. relativem Pfad zur Ziel-Datei -->
(32)    <xsl:param name="dstMdl" select="'../Model/unknownUmlModel.uml'"/>
(33)
(34)    <!-- Name der Transformation zur Aufzeichnung der
      Verfolgbarkeitsbeziehungen -->
(35)    <xsl:variable name="transName" select="'rdfa2prm_annotation'"/>
(36)
(37)    <!-- Import der Hilfsfunktionen -->
(38)    <xsl:include href="xsltrace.xsl"/>
(39)    <xsl:include href="convert-OwlCommon.xsl"/>
(40)
(41)    <!--  Wurzelknoten -->
(42)    <xsl:template match="/html:html">
(43)
(44)      <xsl:call-template name="initTracing">
(45)        <xsl:with-param name="transName" select="$transName"/>
(46)      </xsl:call-template>
(47)
(48)      <!-- XMI inkl. notwendiger Namespaces und Schema-Verweisen auf OWL- und
      RDF-Profile anlegen -->
(49)      <xsl:element name="xmi:XMI">
(50)        <xsl:attribute name="xmi:version">
(51)        <xsl:text>2.1</xsl:text>
(52)        </xsl:attribute>
(53)        <xsl:namespace name="xmi"
      select="'http://schema.omg.org/spec/XMI/2.1'"/>
(54)        <xsl:namespace name="xsi" select="'http://www.w3.org/2001/XMLSchema-
      instance'"/>
(55)        <xsl:namespace name="OwlUmlProfile"
      select="'http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/28'"/>
(56)        <xsl:namespace name="RdfUmlProfile"
      select="'http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/48'"/>
(57)        <xsl:namespace name="ecore"
      select="'http://www.eclipse.org/emf/2002/Ecore'"/>
(58)        <xsl:namespace name="uml"
      select="'http://www.eclipse.org/uml2/3.0.0/UML'"/>
(59)        <xsl:attribute name="xsi:schemaLocation">
(60)          <xsl:text>http://schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/28
      ../ProfileModels/OwlUmlProfile.uml#_QwFo8CezEeC9_fKmAwBwrw
      http://schemas/RdfUmlProfile/_ZPooICezEeC9_fKmAwBwrw/48
      ../ProfileModels/RdfUmlProfile.uml#_ZPpPMCezEeC9_fKmAwBwrw</xsl:text>
(61)        </xsl:attribute>
(62)
(63)        <!--  Modell und alle darin enthaltenen Elemente anlegen -->
(64)        <xsl:call-template name="model"/>
```

```
(65)
(66)        <!--  Stereotypen anlegen -->
(67)        <xsl:call-template name="ProfileRef"/>
(68)      </xsl:element>
(69)
(70)      <xsl:call-template name="finalizeTracing"/>
(71)
(72)    </xsl:template>
(73)
(74)    <!--
(75)      Transformationsregel
(76)      ID:    XSLT_OWL-ANNOTATION-PRM-01
(77)      Name:  UmlModell
(78)     -->
(79)    <xsl:template name="model">
(80)      <xsl:variable name="name"
      select="html:head/html:meta[@property='rdf:about'][1]/@content"/>
(81)      <xsl:variable name="id" select="concat(govobj:generate-id(.),
      '_model')"/>
(82)      <xsl:variable name="ref" select="concat(govobj:generate-id(.),
      '_ontology')"/>
(83)
(84)      <xsl:element name="uml:Model">
(85)        <xsl:attribute name="xmi:id" select="$id"/>
(86)        <xsl:attribute name="name" select="concat($name, '-Modell')"/>
(87)
(88)        <!--  Tracing -->
(89)        <xsl:call-template name="traceTransformation">
(90)          <xsl:with-param name="transName" select="$transName"/>
(91)          <xsl:with-param name="transRule" select="'UmlModel'"/>
(92)          <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
(93)          <xsl:with-param name="dstName" select="'dstUmlModel'"/>
(94)          <xsl:with-param name="srcElement" select="$ref"/>
(95)          <xsl:with-param name="dstElement" select="$id"/>
(96)        </xsl:call-template>
(97)
(98)        <!-- Enthaltenes Package anlegen -->
(99)        <xsl:call-template name="packagedElementPackage"/>
(100)
(101)       <!-- OWL- und RDF-Profil auf Modell anwenden -->
(102)       <xsl:call-template name="ProfileApplication"/>
(103)
(104)     </xsl:element>
(105)
(106)   </xsl:template>
(107)
(108)   <!--
(109)     Transformationsregel
(110)     ID:    XSLT_OWL-ANNOTATION-PRM-02
(111)     Name:  UmlPackage
(112)    -->
(113)   <xsl:template name="packagedElementPackage">
(114)     <xsl:variable name="name"
      select="html:head/html:meta[@property='rdf:about'][1]/@content"/>
(115)     <xsl:variable name="id" select="concat(govobj:generate-id(.),
      '_package')"/>
(116)     <xsl:variable name="ref" select="concat(govobj:generate-id(.),
      '_ontology')"/>
(117)
(118)     <xsl:element name="packagedElement">
(119)       <xsl:attribute name="xmi:id" select="$id"/>
(120)       <xsl:attribute name="xmi:type" select="'uml:Package'"/>
(121)       <xsl:attribute name="name" select="concat($name, '-Package')"/>
(122)
(123)       <!--  Tracing -->
(124)       <xsl:call-template name="traceTransformation">
(125)         <xsl:with-param name="transName" select="$transName"/>
(126)         <xsl:with-param name="transRule" select="'UmlPackage'"/>
(127)         <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
(128)         <xsl:with-param name="dstName" select="'dstUmlModel'"/>
(129)         <xsl:with-param name="srcElement" select="$ref"/>
(130)         <xsl:with-param name="dstElement" select="$id"/>
(131)       </xsl:call-template>
(132)
(133)       <!--  Enthaltene Elemente anlegen -->
```

```
(134)       <xsl:call-template name="packagedElements"/>
(135)
(136)     </xsl:element>
(137)
(138)   </xsl:template>
(139)
(140)   <!--
(141)     Abbildungsregel
(142)     ID:    XSLT_OWL-ANNOTATION_PRM-04a
(143)     Name:  UmlInstanceSpecificationSingle
(144)     Hinweis:Bestandteil der Transformationsregel XSLT_OWL-ANNOTATION_PRM-04
(145)    -->
(146)   <xsl:template name="instanceSpecification">
(147)     <xsl:variable name="instance" select="substring-after(@about, '#')"/>
(148)     <xsl:variable name="instance-ref" select="concat(govobj:generate-id(.),
      '_individual')"/>
(149)     <xsl:variable name="instance-name" select="normalize-space(.)"/>
(150)     <xsl:variable name="class" select="substring-after(@typeof, ':')"/>
(151)     <xsl:variable name="instance-id" select="concat(govobj:generate-id(.),
      '_individual')"/>
(152)
(153)     <!--  Tracing -->
(154)     <xsl:call-template name="traceTransformation">
(155)       <xsl:with-param name="transName" select="$transName"/>
(156)       <xsl:with-param name="transRule" select="'UmlInstanceSpecification'"/>
(157)       <xsl:with-param name="srcName" select="'srcOwlIndividual'"/>
(158)       <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecification'"/>
(159)       <xsl:with-param name="srcElement" select="$instance-ref"/>
(160)       <xsl:with-param name="dstElement" select="$instance-id"/>
(161)     </xsl:call-template>
(162)
(163)     <!-- Kommentar zur Orientierung im erzeugten XMI -->
(164)     <xsl:text>&#x0A;</xsl:text>
(165)     <xsl:comment>
(166)       <xsl:value-of select="$instance"/>
(167)       <xsl:text> (</xsl:text>
(168)       <xsl:value-of select="$class"/>
(169)       <xsl:text>)</xsl:text>
(170)     </xsl:comment>
(171)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(172)
(173)     <!--  uml:InstanceSpecification -->
(174)     <xsl:element name="packagedElement">
(175)       <xsl:attribute name="xmi:id" select="$instance-id"/>
(176)       <xsl:attribute name="xmi:type" select="'uml:InstanceSpecification'"/>
(177)       <xsl:attribute name="name" select="govobj:pascalize-on-
      space($instance-name, ' ')"/>
(178)
(179)       <!-- Referenz auf Classifier -->
(180)       <xsl:element name="classifier">
(181)         <xsl:attribute name="xmi:type" select="'uml:Class'"/>
(182)         <xsl:attribute name="href" select="concat($ontology, '#', lower-
      case($class), '_class')"/>
(183)       </xsl:element>
(184)
(185)       <!-- uml:Slots sind für RDFa-Beispiel nicht implementiert -->
(186)     </xsl:element>
(187)   </xsl:template>
(188)
(189)   <!--
(190)     Abbildungsregel
(191)     ID:    XSLT_OWL-ANNOTATION_PRM-04
(192)     Name:  UmlInstanceSpecification
(193)    -->
(194)   <xsl:template name="instanceSpecifications">
(195)
(196)     <!-- Kommentar zur Orientierung im erzeugten XMI -->
(197)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(198)     <xsl:comment>InstanceSpecifications</xsl:comment>
(199)     <xsl:text>&#9;&#9;</xsl:text>
(200)
(201)     <!-- Jeweils eine Instanz als Element anlegen -->
(202)     <xsl:for-each select="//html:*[@typeof]">
(203)       <xsl:call-template name="instanceSpecification"/>
```

```
(204)     </xsl:for-each>
(205)
(206)   </xsl:template>
(207)
(208)   <!--
(209)     Transformationsregel
(210)     ID:    XSLT_OWL-PRM-06
(211)     Name:  UmlInstanceSpecificationLink
(212)    -->
(213)   <xsl:template name="instanceSpecificationLinks">
(214)
(215)     <!-- Kommentar zur Orientierung im erzeugten XMI -->
(216)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(217)     <xsl:comment>InstanceSpecificationsLinks</xsl:comment>
(218)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(219)
(220)     <xsl:for-each select="//html:*[@rel]">
(221)       <!--  Variablen belegen -->
(222)       <xsl:variable name="about" select="@about"/>
(223)       <xsl:variable name="instance"
      select="//html:*[@about=$about][@typeof]"/>
(224)       <xsl:variable name="instance-ref" select="govobj:generate-
      id(//html:*[@about=$about][@typeof])"/>
(225)       <xsl:variable name="class" select="substring-after(@rel, ':')"/>
(226)       <xsl:variable name="instance-id" select="concat($instance-ref,
      '_individual')"/>
(227)
(228)       <xsl:call-template name="instanceSpecificationLink">
(229)         <xsl:with-param name="instance" select="$instance"/>
(230)         <xsl:with-param name="instance-id" select="$instance-id"/>
(231)         <xsl:with-param name="instance-ref" select="substring-after($about,
      '#')"/>
(232)         <xsl:with-param name="class" select="$class"/>
(233)       </xsl:call-template>
(234)
(235)     </xsl:for-each>
(236)   </xsl:template>
(237)
(238)   <!--
(239)     Transformationsregel
(240)     ID:    XSLT_OWL-PRM-06a
(241)     Name:  UmlInstanceSpecificationLinkSingle
(242)     Hinweis:Bestandteil der Transformationsregel XSLT_OWL-PRM-06
(243)    -->
(244)   <xsl:template name="instanceSpecificationLink">
(245)
(246)     <xsl:param name="instance" select="'unknown-instance'"/>
(247)     <xsl:param name="instance-id" select="'unknown-instance-id'"/>
(248)     <xsl:param name="instance-ref" select="'unknown-instance-ref'"/>
(249)     <xsl:param name="class" select="'unknown-class'"/>
(250)
(251)     <xsl:variable name="property" select="substring-after(@rel, ':')"/>
(252)     <xsl:variable name="domain" select="substring-
      before(document($ontology)/xmi:XMI/uml:Model//packagedElement[@name=$propert
      y]/ownedEnd[1]/@type, '_class')"/>
(253)     <xsl:variable name="range" select="substring-
      before(document($ontology)/xmi:XMI/uml:Model//packagedElement[@name=$propert
      y]/ownedEnd[2]/@type, '_class')"/>
(254)     <xsl:variable name="resource-ref" select="@resource"/>
(255)     <xsl:variable name="resource" select="//html:*[@about=$resource-
      ref][@typeof]"/>
(256)     <xsl:variable name="class-ref" select="//html:*[@about=$resource-
      ref][@typeof]/@typeof"/>
(257)     <xsl:variable name="class-ref-id" select="concat(govobj:generate-
      id(//html:*[@about=$resource-ref][@typeof]), '_individual')"/>
(258)     <xsl:variable name="property-id" select="concat(govobj:generate-id(.),
      '_instance-specification-link')"/>
(259)
(260)     <!--  Tracing -->
(261)     <xsl:call-template name="traceTransformation">
(262)       <xsl:with-param name="transName" select="$transName"/>
(263)       <xsl:with-param name="transRule"
      select="'UmlInstanceSpecificationLink'"/>
(264)       <xsl:with-param name="srcName" select="'srcOwlIndividualProperty'"/>
```

```
(265)       <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecificationLink'"/>
(266)       <xsl:with-param name="srcElement" select="concat($instance-ref, '/',
      $property, '/#', substring-after($resource-ref, '#'))"/>
(267)       <xsl:with-param name="dstElement" select="$property-id"/>
(268)     </xsl:call-template>
(269)
(270)     <!--  Kommentar zur Orientierung im XMI -->
(271)     <xsl:text>&#x0A;&#9;</xsl:text>
(272)     <xsl:comment>
(273)       <xsl:value-of select="concat($instance, ' (', $domain, ') ',
      $property, ' ', $resource, ' (', $range, ')')"/>
(274)     </xsl:comment>
(275)     <xsl:text>&#x0A;&#9;&#9;</xsl:text>
(276)
(277)     <xsl:element name="packagedElement">
(278)       <xsl:attribute name="xmi:type"  select="'uml:InstanceSpecification'"/>
(279)       <xsl:attribute name="xmi:id" select="$property-id"/>
(280)       <xsl:attribute name="name" select="concat(govobj:pascalize-on-
      space($instance, ' '), '(', $domain, ')_', $property, '_',
      govobj:pascalize(substring-after($resource-ref, '#')), '(', $range, ')')"/>
(281)       <xsl:variable name="ass-clsf" select="concat(govobj:shorten(lower-
      case($domain)), '_', govobj:shorten(lower-case($property)), '_',
      govobj:shorten(lower-case($range)))"/>
(282)
(283)       <xsl:element name="classifier">
(284)         <xsl:attribute name="xmi:type" select="'uml:Association'"/>
(285)         <xsl:attribute name="href" select="concat($ontology, '#', $ass-
      clsf)"/>
(286)       </xsl:element>
(287)
(288)       <xsl:call-template name="instanceSpecificationLinkSlot">
(289)         <xsl:with-param name="id" select="concat(replace(generate-
      id(),':','_'), '_slot1')"/>
(290)         <xsl:with-param name="definingFeature" select="concat($ontology,
      '#', $ass-clsf, '_', govobj:shorten($domain) , '_ownedEnd-1')"/>
(291)         <xsl:with-param name="type" select="concat($ontology, '#', lower-
      case($domain), '_class')"/>
(292)         <xsl:with-param name="instance" select="$instance-id"/>
(293)       </xsl:call-template>
(294)
(295)       <xsl:call-template name="instanceSpecificationLinkSlot">
(296)         <xsl:with-param name="id" select="concat(replace(generate-
      id(),':','_'), '_slot2')"/>
(297)         <xsl:with-param name="definingFeature" select="concat($ontology,
      '#', $ass-clsf, '_', govobj:shorten($range) , '_ownedEnd-2')"/>
(298)         <xsl:with-param name="type" select="concat($ontology, '#', lower-
      case($range), '_class')"/>
(299)         <xsl:with-param name="instance" select="$class-ref-id"/>
(300)       </xsl:call-template>
(301)
(302)     </xsl:element>
(303)   </xsl:template>
(304)
(305)   <!--
(306)     Transformationsregel
(307)     ID:    XSLT_OWL-PRM-06b
(308)     Name:  UmlInstanceSpecificationLinkSlot
(309)     Hinweis:Bestandteil der Transformationsregel XSLT_OWL-PRM-06
(310)    -->
(311)   <xsl:template name="instanceSpecificationLinkSlot">
(312)     <xsl:param name="id"/>
(313)     <xsl:param name="definingFeature"/>
(314)     <xsl:param name="type"/>
(315)     <xsl:param name="instance"/>
(316)
(317)     <xsl:element name="slot">
(318)       <xsl:attribute name="xmi:id" select="$id"/>
(319)       <xsl:element name="definingFeature">
(320)         <xsl:attribute name="xmi:type" select="'uml:Property'"/>
(321)         <xsl:attribute name="href" select="$definingFeature"/>
(322)       </xsl:element>
(323)       <xsl:element name="value">
(324)         <xsl:attribute name="xmi:type" select="'uml:InstanceValue'"/>
(325)         <xsl:attribute name="xmi:id" select="concat($id, '_value')"/>
```

```
(326)         <xsl:attribute name="instance" select="$instance"/>
(327)         <xsl:element name="type">
(328)           <xsl:attribute name="xmi:type" select="'uml:Class'"/>
(329)           <xsl:attribute name="href" select="$type"/>
(330)         </xsl:element>
(331)       </xsl:element>
(332)     </xsl:element>
(333)
(334)   </xsl:template>
(335)
(336)   <!--  Elemente des Packages anlegen -->
(337)   <xsl:template name="packagedElements">
(338)
(339)     <!-- owl:Individual -> uml:InstanceSpecification -->
(340)     <xsl:call-template name="instanceSpecifications"/>
(341)
(342)     <!-- owl:ObjectProperties -> uml:InstanceLink -->
(343)     <xsl:call-template name="instanceSpecificationLinks"/>
(344)
(345)   </xsl:template>
(346)
(347)   <!--
(348)     Zuweisung des RDF- und OWL-Profils
(349)    -->
(350)   <xsl:template name="ProfileApplication">
(351)     <profileApplication xmi:id="id_profileAppl_1">
(352)       <eAnnotations xmi:id="id_profileAppl_1_Annotation"
      source="http://www.eclipse.org/uml2/2.0.0/UML">
(353)       <references xmi:type="ecore:EPackage"
      href="../ProfileModels/OwlUmlProfile.uml#_QwFo8CezEeC9_fKmAwBwrw"/>
(354)       </eAnnotations>
(355)       <appliedProfile href="../ProfileModels/OwlUmlProfile.uml#_XkGiwB07Ed-
      QQ4mYkrb7Gg"/>
(356)     </profileApplication>
(357)     <profileApplication xmi:id="id_profileAppl_2">
(358)       <eAnnotations xmi:id="id_profileAppl_2_Annotation"
      source="http://www.eclipse.org/uml2/2.0.0/UML">
(359)       <references xmi:type="ecore:EPackage"
      href="../ProfileModels/RdfUmlProfile.uml#_ZPpPMCezEeC9_fKmAwBwrw"/>
(360)       </eAnnotations>
(361)       <appliedProfile href="../ProfileModels/RdfUmlProfile.uml#_XkGiwB07Ed-
      QQ4mYkrb7Gg"/>
(362)     </profileApplication>
(363)   </xsl:template>
(364)
(365)   <!--
(366)     Transformationsregel
(367)     ID:    XSLT_OWL-ONTOLOGY-PRM-03
(368)     Name:  UmlPackageStereotyp
(369)    -->
(370)   <xsl:template name="owlOntology">
(371)     <xsl:variable name="id" select="concat(govobj:generate-id(.),
      '_package')"/>
(372)     <xsl:variable name="ref" select="concat(govobj:generate-id(.),
      '_ontology')"/>
(373)
(374)     <xsl:element name="OwlUmlProfile:owlOntology">
(375)       <xsl:attribute name="xmi:id" select="concat(govobj:generate-id(.),
      '_owl-ontology')"/>
(376)       <xsl:attribute name="base_Package" select="$id"/>
(377)     </xsl:element>
(378)
(379)     <!--  Tracing -->
(380)     <xsl:call-template name="traceTransformation">
(381)       <xsl:with-param name="transName" select="$transName"/>
(382)       <xsl:with-param name="transRule" select="'UmlPackageStereotyp'"/>
(383)       <xsl:with-param name="srcName" select="'srcOwlOntology'"/>
(384)       <xsl:with-param name="dstName" select="'dstUmlPackageStereotype'"/>
(385)       <xsl:with-param name="srcElement" select="$ref"/>
(386)       <xsl:with-param name="dstElement" select="$id"/>
(387)     </xsl:call-template>
(388)   </xsl:template>
(389)
(390)   <!--
(391)     Transformationsregel
```

```
(392)     ID:    XSLT_OWL-ANNOTATION_PRM-05
(393)     Name:  UmlInstanceSpecificationStereotyp
(394)    -->
(395)   <xsl:template name="owlIndividual">
(396)     <xsl:for-each select="//html:*[@typeof]">
(397)       <xsl:variable name="id" select="concat(govobj:generate-id(.),
      '_individual')"/>
(398)       <xsl:variable name="ref" select="concat(govobj:generate-id(.),
      '_individual')"/>
(399)
(400)       <xsl:element name="OwlUmlProfile:owlIndividual">
(401)         <xsl:attribute name="xmi:id" select="concat(replace(generate-
      id(),':','_'), '_owl-individual')"/>
(402)         <xsl:attribute name="base_InstanceSpecification" select="$id"/>
(403)       </xsl:element>
(404)
(405)       <!--  Tracing -->
(406)       <xsl:call-template name="traceTransformation">
(407)         <xsl:with-param name="transName" select="$transName"/>
(408)         <xsl:with-param name="transRule" select="'UmlIndividualStereotyp'"/>
(409)         <xsl:with-param name="srcName" select="'srcOwlIndividual'"/>
(410)         <xsl:with-param name="dstName"
      select="'dstUmlInstanceSpecificationStereotype'"/>
(411)         <xsl:with-param name="srcElement" select="$ref"/>
(412)         <xsl:with-param name="dstElement" select="$id"/>
(413)       </xsl:call-template>
(414)     </xsl:for-each>
(415)   </xsl:template>
(416)
(417)   <!--
(418)     ODM-Stereotypen zuweisen
(419)    -->
(420)   <xsl:template name="ProfileRef">
(421)
(422)     <!-- Stereotyp owl:Ontology für das UML-Package -->
(423)     <xsl:call-template name="owlOntology"/>
(424)
(425)     <!-- Stereotyp owl:Individual für die UML-Instanzen-->
(426)     <xsl:call-template name="owlIndividual"/>
(427)
(428)   </xsl:template>
(429)
(430)   <!--
(431)     The source code for this function 'functx:word-count' is available under
      the GNU LGPL license.
(432)     siehe: http://www.gnu.org/licenses/lgpl.html
(433)     Autor: Priscilla Walmsley, Datypic, pwalmsley@datypic.com,
      http://www.datypic.com
(434)     Quelle: http://www.xsltfunctions.com/xsl/functx_word-count.html
(435)   -->
(436)   <xsl:function name="functx:word-count" as="xs:integer"
      xmlns:functx="http://www.functx.com" >
(437)     <xsl:param name="arg" as="xs:string?"/>
(438)     <xsl:sequence select="count(tokenize($arg, '\W+')[. != ''])"/>
(439)   </xsl:function>
(440)
(441)
(442) </xsl:stylesheet>
```

Listing 52 – Transformation der OWL-Annotationen in ein UML-Modell (convert-RdfaAnnotation2Uml.xsl)

D.4 Gemeinsam genutzte Funktionen

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns="http://www.w3.org/1999/xhtml"
(4)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(5)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(6)     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(7)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(8)     xmlns:xmi="http://www.omg.org/XMI"
(9)     xmlns:xmi1="http://schema.omg.org/spec/XMI/2.1"
(10)    xmlns:eGovRefAnf="http:///eGovRefAnf.ecore"
(11)    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
(12)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
```

```
(13)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(14)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(15)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(16)    xmlns:exslt="http://exslt.org/common"
(17)      extension-element-prefixes="exslt">
(18)
(19)    <xsl:include href="common.xsl"/>
(20)
(21)    <!--
(22)      Liefert für den Namen einer Klasse die zugehörige Hintergrundfarbe
(23)    -->
(24)    <xsl:function name="govobj:color" as="xs:string">
(25)      <xsl:param name="name"/>
(26)
(27)      <xsl:variable name="green" select="'190,255,190'"/>
(28)      <xsl:variable name="light-green" select="'214,255,214'"/>
(29)      <xsl:variable name="yellow" select="'255,255,179'"/>
(30)      <xsl:variable name="red" select="'255,168,168'"/>
(31)      <xsl:variable name="white" select="'255,255,255'"/>
(32)
(33)      <xsl:variable name="color">
(34)        <xsl:choose>
(35)          <xsl:when test="$name='vwadressat_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(36)          <xsl:when test="$name='vwakteur_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(37)          <xsl:when test="$name='vwakteurserweiterung_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(38)          <xsl:when test="$name='vwanliegen_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(39)          <xsl:when test="$name='vwantrag_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(40)          <xsl:when test="$name='vwantragsteller_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(41)          <xsl:when test="$name='vwanwender_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(42)          <xsl:when test="$name='vwanwenderbackoffice_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(43)          <xsl:when test="$name='vwanwenderfrontoffice_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(44)          <xsl:when test="$name='vwanzuhoerender_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(45)          <xsl:when test="$name='vwausfuehrender_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(46)          <xsl:when test="$name='vwausfuehrenderbackoffice_class'">
      <xsl:value-of select="$yellow"/> </xsl:when>
(47)          <xsl:when test="$name='vwausfuehrenderfrontoffice_class'">
      <xsl:value-of select="$yellow"/> </xsl:when>
(48)          <xsl:when test="$name='vwauswahlkriterium_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(49)          <xsl:when test="$name='vwbeteiligterbetroffener_class'"> <xsl:value-
      of select="$yellow"/> </xsl:when>
(50)          <xsl:when test="$name='vwbeteiligungmitwirkung_class'"> <xsl:value-
      of select="$green"/> </xsl:when>
(51)          <xsl:when test="$name='vwbetroffenerdritter_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(52)          <xsl:when test="$name='vwbeurteilungsspielraum_class'"> <xsl:value-
      of select="$green"/> </xsl:when>
(53)          <xsl:when test="$name='vwbewusstmacher_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(54)          <xsl:when test="$name='vwbewusstwerdung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(55)          <xsl:when test="$name='vwergebnismitteilung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(56)          <xsl:when test="$name='vwermessen_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(57)          <xsl:when test="$name='vwermessensausuebung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(58)          <xsl:when test="$name='vwhandlung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(59)          <xsl:when test="$name='vwhandlungserweiterung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(60)          <xsl:when test="$name='vwhandlungsausloeser_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
```

```
(61)          <xsl:when test="$name='vwhandlungsgegenstand_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(62)          <xsl:when test="$name='vwhandlungsgegenstandserweiterung_class'">
      <xsl:value-of select="$green"/> </xsl:when>
(63)          <xsl:when test="$name='vwhinweisinformation_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(64)          <xsl:when test="$name='vwinformationsbeschaffung_class'">
      <xsl:value-of select="$red"/> </xsl:when>
(65)          <xsl:when test="$name='vwlebenslage_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(66)          <xsl:when test="$name='vwlebenssachverhalt_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(67)          <xsl:when test="$name='vwleistung_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(68)          <xsl:when test="$name='vwleistungseigenschaft_class'"> <xsl:value-of
      select="$light-green"/> </xsl:when>
(69)          <xsl:when test="$name='vwleistungserbringung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(70)          <xsl:when test="$name='vwleistungszusammenhang_class'"> <xsl:value-
      of select="$green"/> </xsl:when>
(71)          <xsl:when test="$name='vwmassnahmenfestlegung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(72)          <xsl:when test="$name='vwmittelwahl_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(73)          <xsl:when test="$name='vwnachbearbeitung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(74)          <xsl:when test="$name='vwnachbearbeitungsinformation_class'">
      <xsl:value-of select="$green"/> </xsl:when>
(75)          <xsl:when test="$name='vwnachfrager_class'"> <xsl:value-of
      select="$yellow"/> </xsl:when>
(76)          <xsl:when test="$name='vwnutzungvonbeurteilungsspielraum_class'">
      <xsl:value-of select="$red"/> </xsl:when>
(77)          <xsl:when test="$name='vwrechtsfolge_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(78)          <xsl:when test="$name='vwrechtsfolgenmerkmal_class'"> <xsl:value-of
      select="$light-green"/> </xsl:when>
(79)          <xsl:when test="$name='vwsachverhalt_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(80)          <xsl:when test="$name='vwsachverhaltseigenschaft_class'">
      <xsl:value-of select="$light-green"/> </xsl:when>
(81)          <xsl:when test="$name='vwsachverhaltsermittlung_class'"> <xsl:value-
      of select="$red"/> </xsl:when>
(82)          <xsl:when test="$name='vwverfahren_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(83)          <xsl:when test="$name='vwverfahrenseinleitung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(84)          <xsl:when test="$name='vwvoraussetzung_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(85)          <xsl:when test="$name='vwvoraussetzungsbedingung_class'">
      <xsl:value-of select="$light-green"/> </xsl:when>
(86)          <xsl:when test="$name='vwvoraussetzungspruefung_class'"> <xsl:value-
      of select="$red"/> </xsl:when>
(87)          <xsl:when test="$name='vwvorbereitung_class'"> <xsl:value-of
      select="$red"/> </xsl:when>
(88)          <xsl:when test="$name='vwzielzweck_class'"> <xsl:value-of
      select="$green"/> </xsl:when>
(89)          <xsl:otherwise>
(90)            <xsl:value-of select="$white"/>
(91)          </xsl:otherwise>
(92)        </xsl:choose>
(93)      </xsl:variable>
(94)      <xsl:sequence select="$color"/>
(95)    </xsl:function>
(96)
(97)    <!--
(98)         Liefert den Teilstring nach # der rdf:about-Eigenschaft. Wenn leer,
      wird der Defaultwert 'unbekannt' verwendet
(99)    -->
(100)   <xsl:function name="govobj:ref" as="xs:string">
(101)     <xsl:param name="elem"/>
(102)
(103)     <xsl:variable name="ref">
(104)       <xsl:choose>
(105)         <xsl:when test="string-length(substring-after($elem/@rdf:about,
      '#')) &gt; 0">
```

```
(106)           <xsl:value-of select="substring-after($elem/@rdf:about, '#')"/>
(107)         </xsl:when>
(108)         <xsl:otherwise>
(109)           <xsl:value-of select="'unbekannt'"/>
(110)         </xsl:otherwise>
(111)       </xsl:choose>
(112)     </xsl:variable>
(113)
(114)     <xsl:sequence select="$ref"/>
(115)   </xsl:function>
(116)
(117)   <!--
(118)       Ermittelt die ID des übergebenen Elementes
(119)
(120)     Verwendet den Wert der die rdfs:label-Eigenschaft. Wenn diese nicht
      gesetzt ist, wird er aus der rdf:about-Eigenschaft
(121)     ermittelt. Ist diese auch nicht gesetzt. Wird eine eindeutige ID erzeugt
      zurückgegeben. Umlaute in der ID werden ersetzt.
(122)   -->
(123)   <xsl:function name="govobj:id-from-about" as="xs:string">
(124)     <xsl:param name="elem"/>
(125)
(126)     <xsl:variable name="id">
(127)       <xsl:choose>
(128)         <xsl:when test="string-length(normalize-space(substring-
      after($elem/@rdf:about, '#'))) &gt; 0">
(129)           <xsl:value-of select="lower-case(normalize-space(substring-
      after($elem/@rdf:about, '#')))"/>
(130)         </xsl:when>
(131)         <xsl:otherwise>
(132)           <xsl:value-of select="replace(generate-id($elem),':','_')"/>
(133)         </xsl:otherwise>
(134)       </xsl:choose>
(135)     </xsl:variable>
(136)     <!--  Umlaute bereinigen -->
(137)     <xsl:sequence select="govobj:replace-umlaute($id)"/>
(138)   </xsl:function>
(139)
(140)
(141)   <!--
(142)       Ermittelt den Namen des übergebenen Elementes
(143)
(144)     Verwendet den Wert der die rdfs:label-Eigenschaft. Wenn diese nicht
      gesetzt ist, wird er aus der rdf:about-Eigenschaft
(145)     ermittelt. Ist diese auch nicht gesetzt. Wird eine eindeutige ID erzeugt
      und als Name zurückgegeben.
(146)   -->
(147)   <xsl:function name="govobj:name-from-rdfslabel-or-about" as="xs:string">
(148)     <xsl:param name="elem"/>
(149)
(150)     <xsl:variable name="name">
(151)       <xsl:choose>
(152)         <xsl:when test="string-length(normalize-space($elem/rdfs:label))
      &gt; 0">
(153)           <xsl:value-of select="normalize-space($elem/rdfs:label)"/>
(154)         </xsl:when>
(155)         <xsl:otherwise>
(156)           <xsl:choose>
(157)             <xsl:when test="string-length(normalize-space(substring-
      after($elem/@rdf:about, '#'))) &gt; 0">
(158)               <xsl:value-of select="replace(normalize-space(substring-
      after($elem/@rdf:about, '#')),':','_')"/>
(159)             </xsl:when>
(160)             <xsl:otherwise>
(161)               <xsl:value-of select="replace(generate-id($elem),':','_')"/>
(162)             </xsl:otherwise>
(163)           </xsl:choose>
(164)         </xsl:otherwise>
(165)       </xsl:choose>
(166)     </xsl:variable>
(167)     <xsl:sequence select="$name"/>
(168)   </xsl:function>
(169)
(170)   <!--
(171)     Konvertiert einen Text in PascalCase anhand des Trennzeichens "_"
```

```
(172)
(173)     basiert auf der Vorlage von:
(174)     Autor: Chris Nava
(175)     Quelle: http://stackoverflow.com/questions/2647327/how-to-format-a-
      string-to-camel-case-in-xslt/2647656#2647656
(176)     Lizenz: licensed under cc-wiki with attribution required
      (http://creativecommons.org/licenses/by-sa/3.0/)
(177)    -->
(178)   <xsl:function name="govobj:pascalize" as="xs:string">
(179)       <xsl:param name="pText" />
(180)
(181) <!--     <xsl:variable name="vLower" select="'abcdefghijklmnopqrstuvwxyz'"/>
(182)     <xsl:variable name="vUpper" select="'ABCDEFGHIJKLMNOPQRSTUVWXYZ'"/>
(183)
(184)       <xsl:variable name="result">
(185)         <xsl:if test="$pText">
(186)             <xsl:value-of select="translate(substring($pText,1,1), $vLower,
      $vUpper)" />
(187)             <xsl:choose>
(188)                 <xsl:when test="contains($pText, '_')">
(189)                     <xsl:value-of select="substring-
      before(substring($pText,2), '_')" />
(190)                 </xsl:when>
(191)                 <xsl:otherwise>
(192)                     <xsl:value-of select="substring($pText,2)" />
(193)                 </xsl:otherwise>
(194)             </xsl:choose>
(195)               <xsl:value-of select="govobj:pascalize(substring-
      after(substring($pText,2), '_'))" />
(196)         </xsl:if>
(197)       </xsl:variable> -->
(198)       <xsl:sequence select="govobj:pascalize-on-space($pText, '_')"/>
(199)   </xsl:function>
(200)
(201)   <xsl:function name="govobj:pascalize-on-space" as="xs:string">
(202)       <xsl:param name="pText"/>
(203)       <xsl:param name="pSpace"/>
(204)
(205)     <xsl:variable name="vLower" select="'abcdefghijklmnopqrstuvwxyz'"/>
(206)     <xsl:variable name="vUpper" select="'ABCDEFGHIJKLMNOPQRSTUVWXYZ'"/>
(207)
(208)       <xsl:variable name="result">
(209)         <xsl:if test="$pText">
(210)             <xsl:value-of select="translate(substring($pText,1,1), $vLower,
      $vUpper)"/>
(211)             <xsl:choose>
(212)                 <xsl:when test="contains($pText, $pSpace)">
(213)                     <xsl:value-of select="substring-
      before(substring($pText,2), $pSpace)"/>
(214)                 </xsl:when>
(215)                 <xsl:otherwise>
(216)                     <xsl:value-of select="substring($pText,2)"/>
(217)                 </xsl:otherwise>
(218)             </xsl:choose>
(219)               <xsl:value-of select="govobj:pascalize-on-space(substring-
      after(substring($pText,2), $pSpace), $pSpace)"/>
(220)         </xsl:if>
(221)       </xsl:variable>
(222)       <xsl:sequence select="$result"/>
(223)   </xsl:function>
(224)
(225)   <xsl:function name="govobj:generate-id" as="xs:string">
(226)     <xsl:param name="node"/>
(227)     <xsl:sequence select="replace(generate-id($node),':','_')"/>
(228)   </xsl:function>
(229)
(230) </xsl:stylesheet>
```

Listing 53 – Gemeinsam genutzte Funktionen (convert-OwlCommon.xsl)

E eLoGo-Referenzmodelle

Die eLoGo-Referenzmodelle wurden im Rahmen dieser Arbeit in unterschiedlicher Form verwendet. Dieser Abschnitt zeigt die BPMN-Darstellung der Referenzprozessmodelle

(Anhang E.1) und die Darstellung der Referenzanforderungsmodelle als UML-Profil (Anhang E.2).

E.1 Referenzprozessmodelle in BPMN

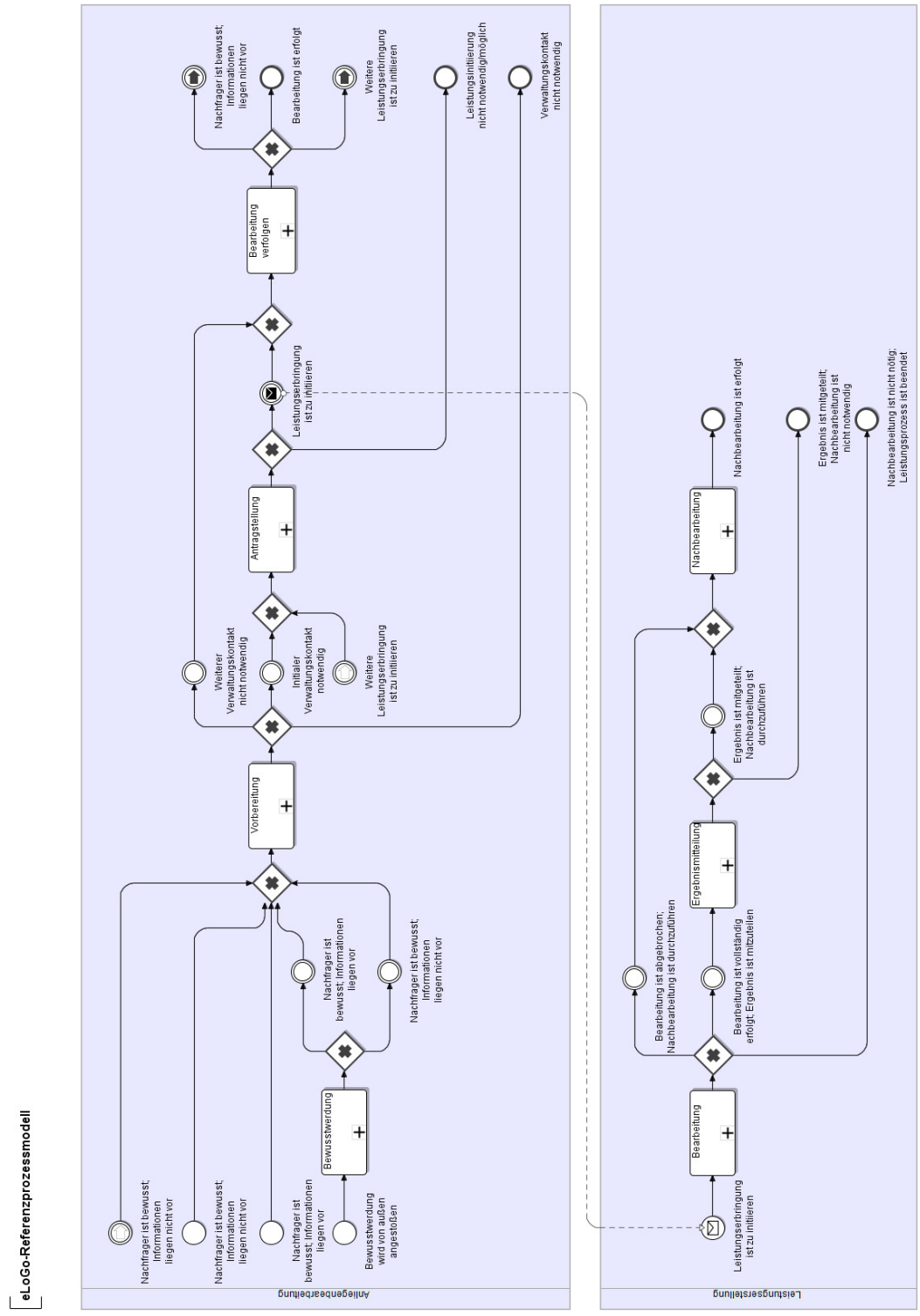


Abbildung 104 – Überblick über die eLoGo-Referenzprozesse in BPMN

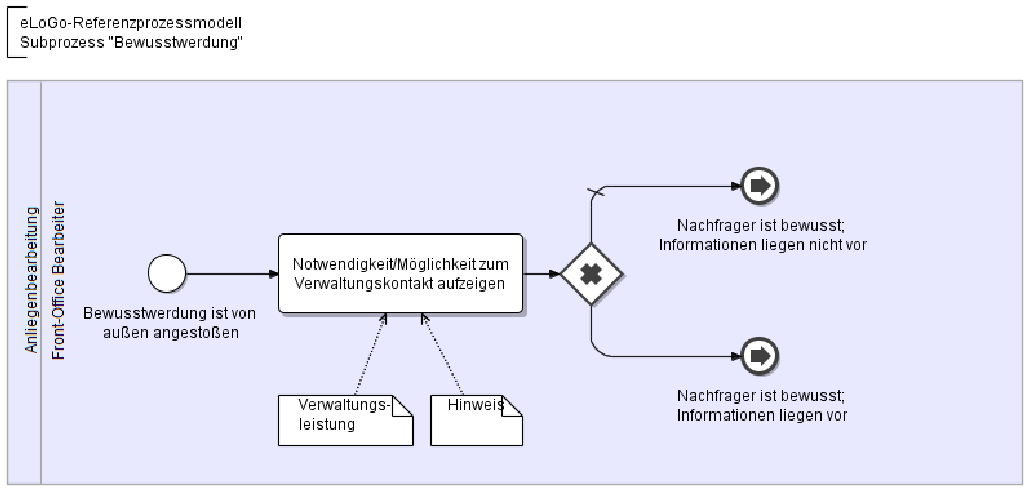


Abbildung 105 – Subprozess „Bewusstwerdung“ in BPMN

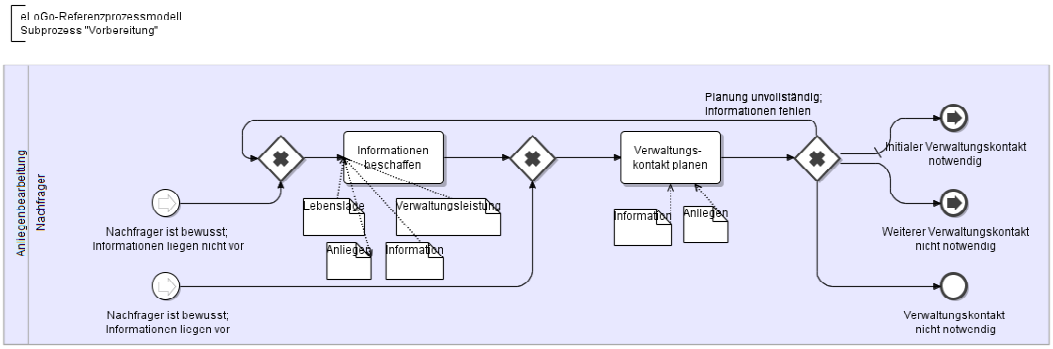


Abbildung 106 – Subprozess „Vorbereitung“ in BPMN

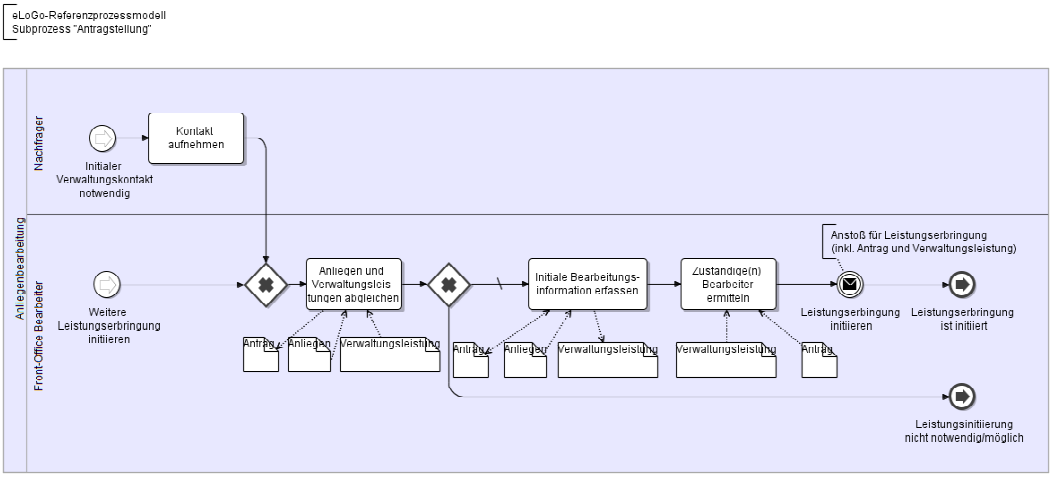


Abbildung 107 – Subprozess „Antragstellung“ in BPMN

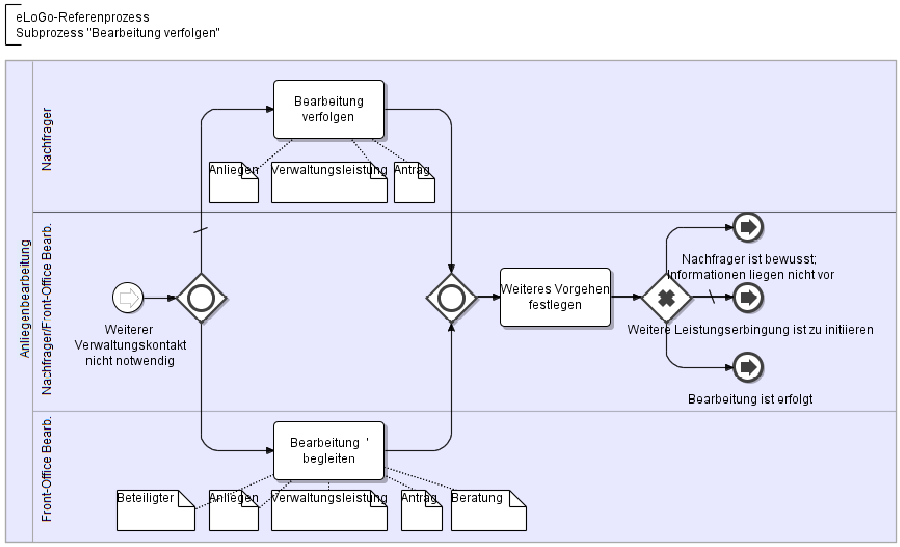


Abbildung 108 – Subprozess „Bearbeitung verfolgen“ in BPMN

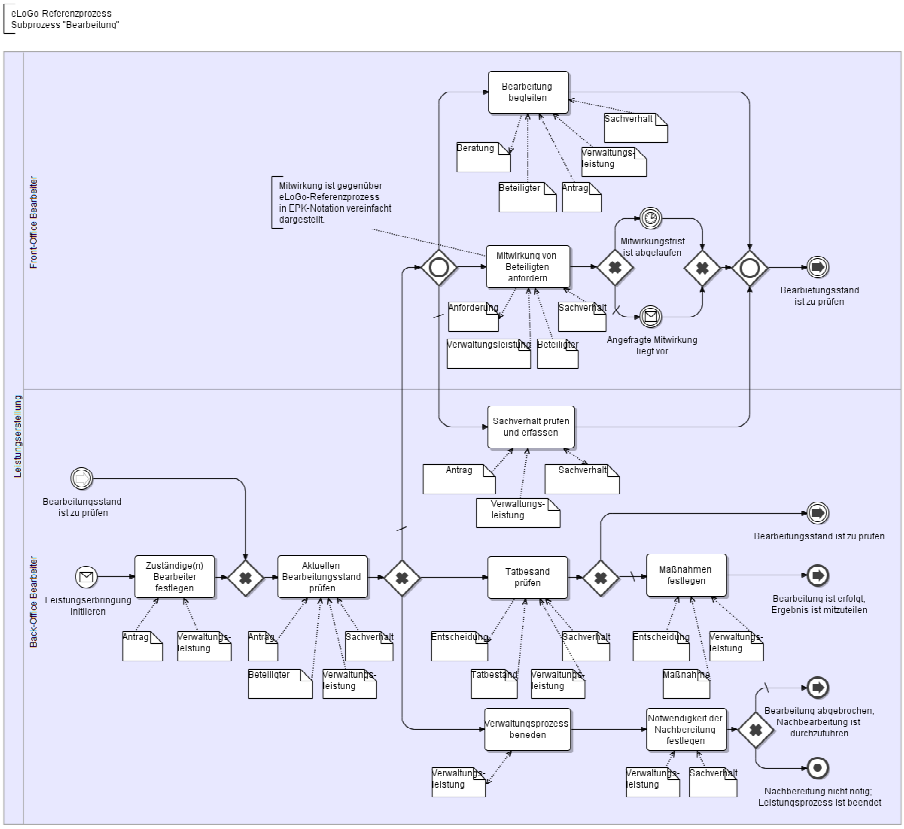


Abbildung 109 – Subprozess „Bearbeitung“ in BPMN

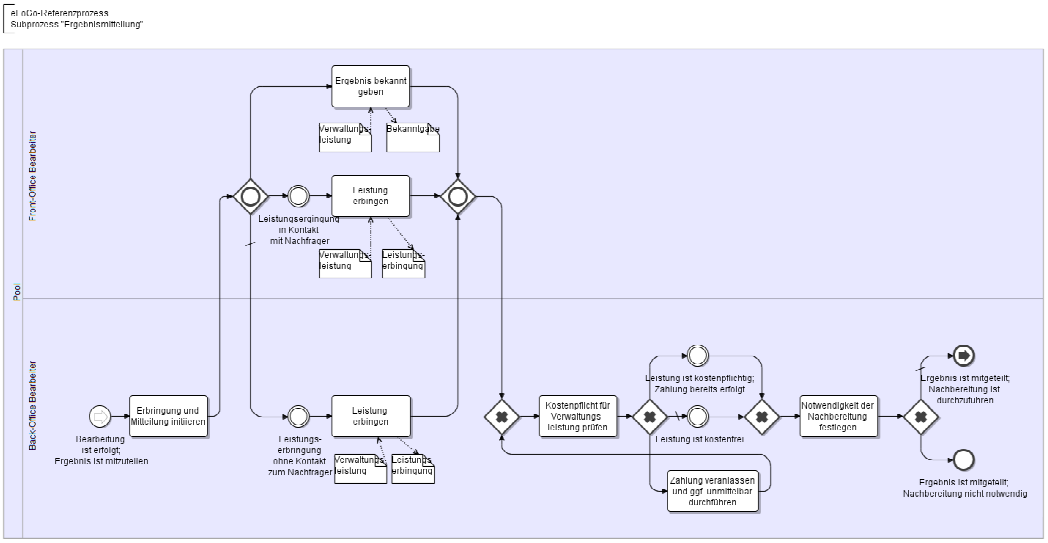


Abbildung 110 – Subprozess „Ergebnismitteilung“ in BPMN[406]

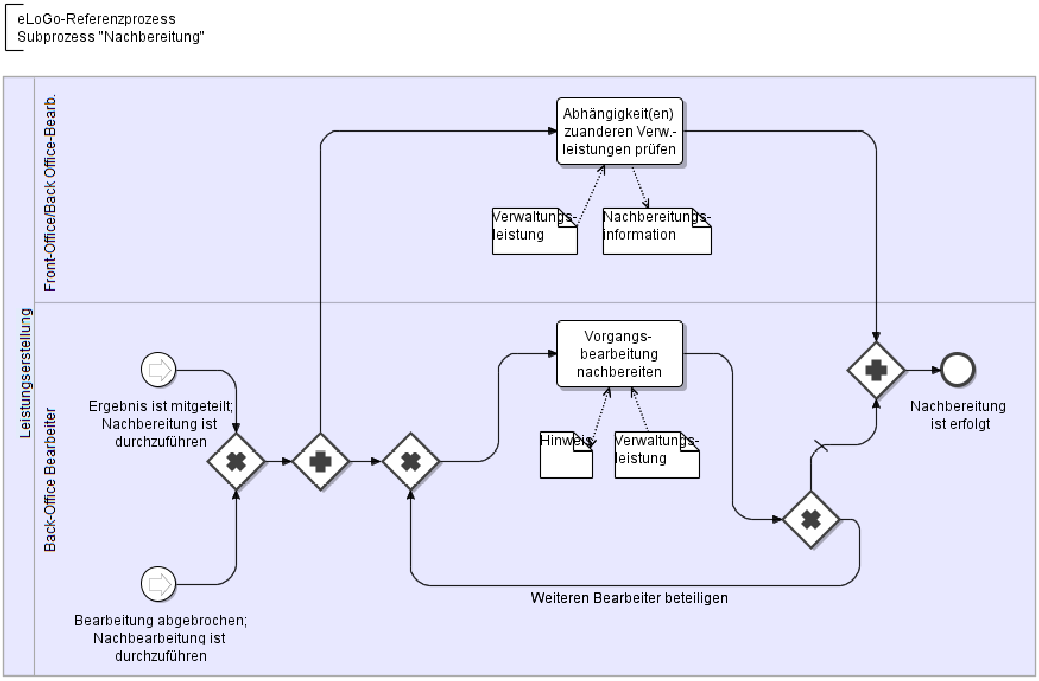


Abbildung 111 – Subprozess „Nachbearbeitung“ in BPMN

E.2 Referenzanforderungsmodelle als UML-Profil

[406] Im Subprozess „Ergebnismitteilung“ wurde auf die detaillierte Beschreibung der Abwicklung kostenpflichtiger Verwaltungsleistungen verzichtet, da diese Zusammenhänge in den BundOnline 2005-Musterprozessen „Zahlungsverkehr/ePayment“ bereits ausführlicher und umfassender dargestellt sind. Auf den Ansatz dieser Arbeit hat diese Einschränkung jedoch keine Auswirkung. Die BundOnline2005 Musterprozesse sind verfügbar unter: http://www.cio.bund.de/DE/Standards/Musterprozesse/musterprozesse_node.html (letzter Aufruf: 21.08.2011).

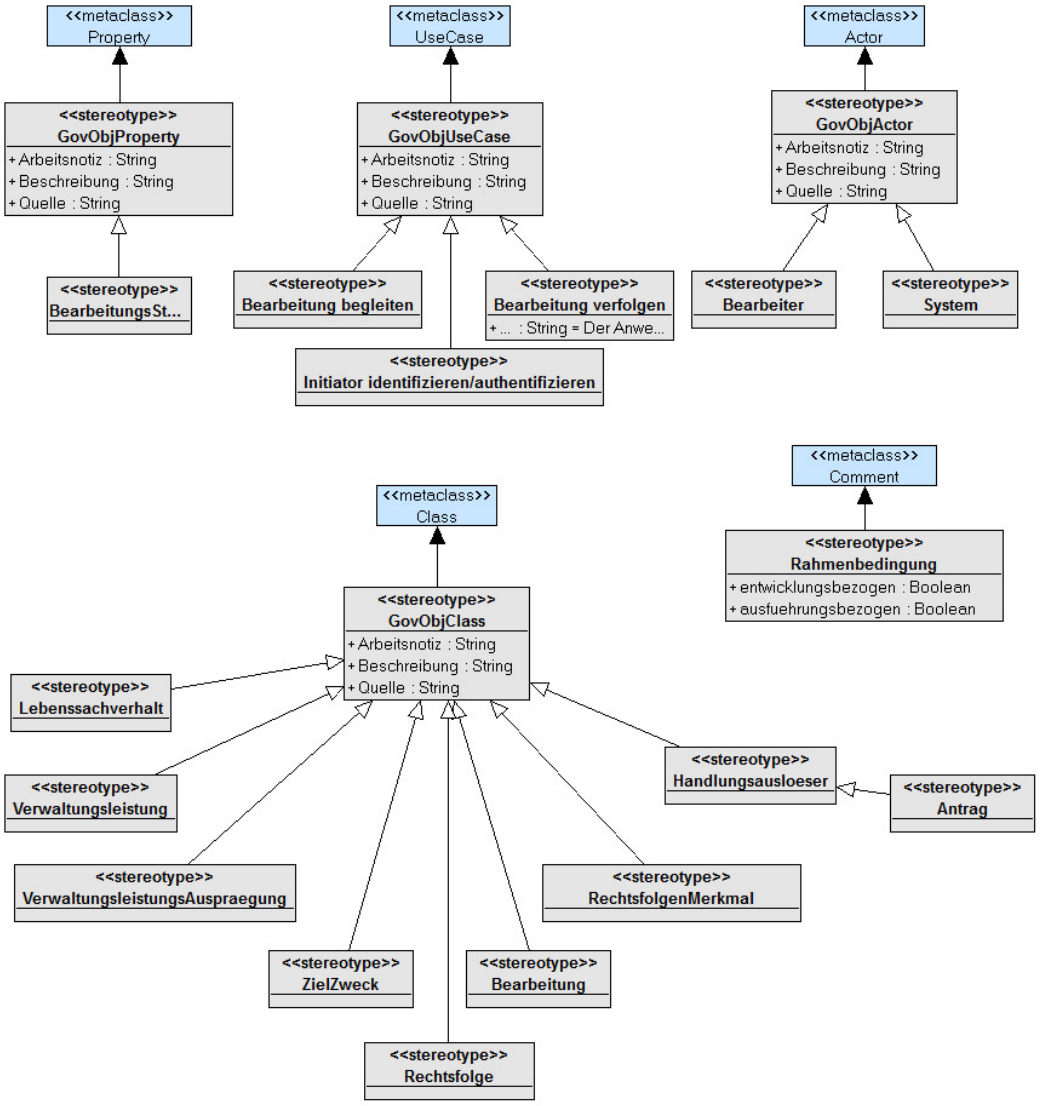


Abbildung 112 – UML-Profildiagramm „eGovRefAnf/Allgemein““

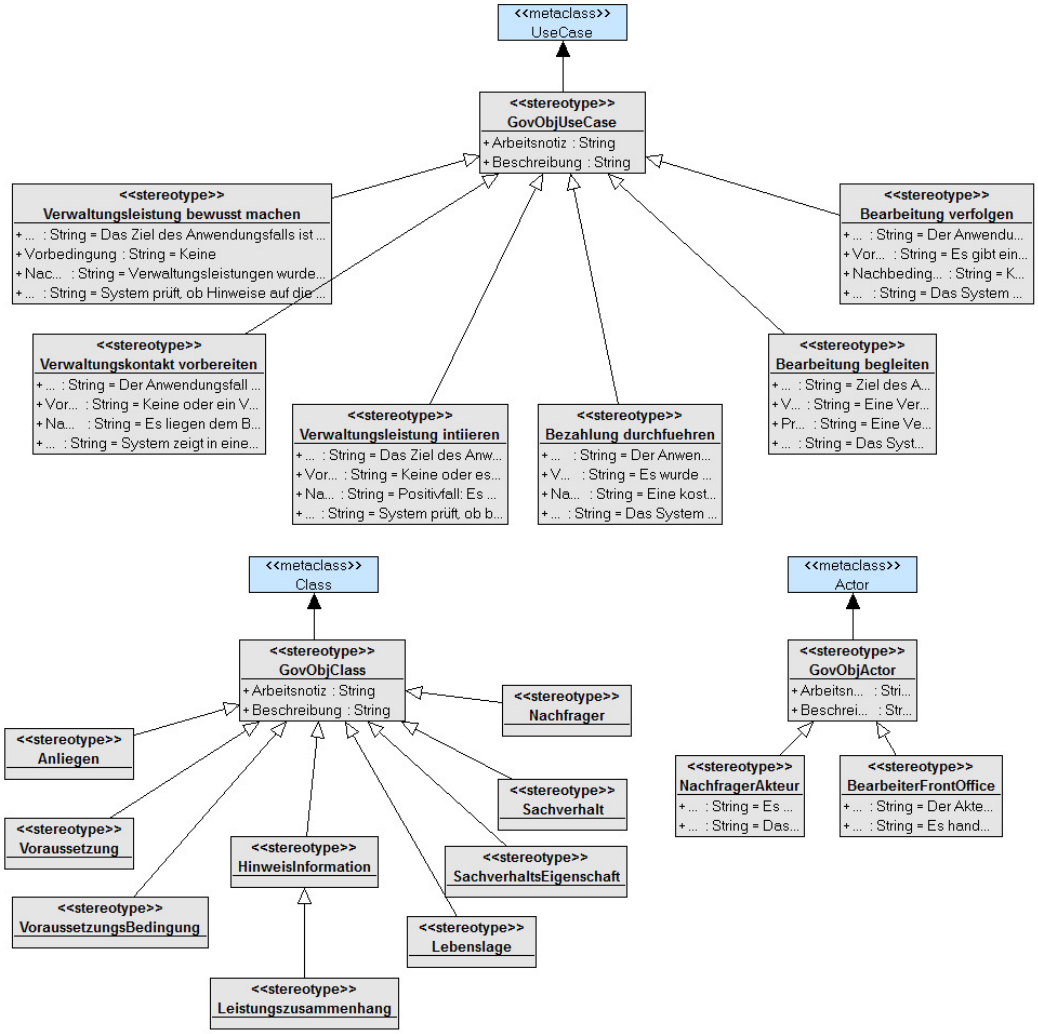


Abbildung 113 – UML-Profildiagramm „eGovRefAnf/Anliegen“

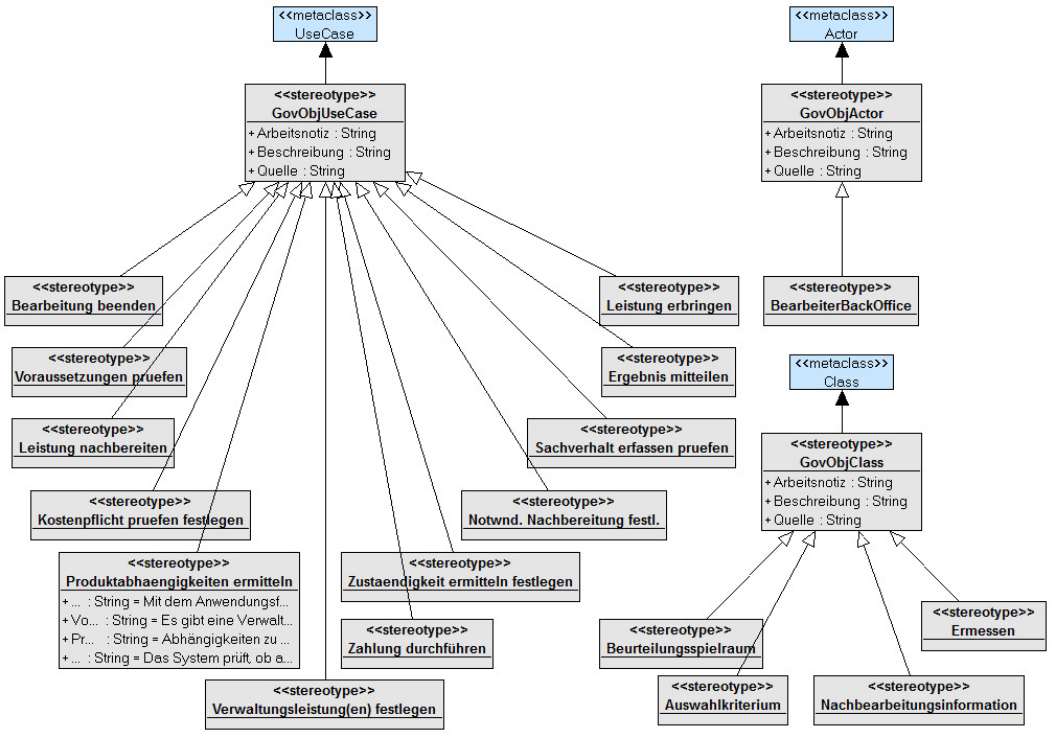


Abbildung 114 – UML-Profildiagramm „eGovRefAnf/Leistung“

F PRM-CIM-Transformation

Dieser Abschnitt stellt die prototypisch implementierten QVT-Transformationen zur Transformation des PRM in Prozessmodelle (Anhang F.1) und Modelle der Anforderungsanalyse (Anhang F.2) dar.

F.1 Prozessmodelle

```
(1)   import prm2cim_proz_anliegen_bewusstwerdung;
(2)   import prm2cim_proz_anliegen_vorbereitung;
(3)   import prm2cim_proz_anliegen_antragstellung;
(4)   import prm2cim_proz_anliegen_bearbeitungverfolgen;
(5)
(6)   import prm2cim_proz_leistung_bearbeitung;
(7)   import prm2cim_proz_leistung_ergebnismitteilung;
(8)   import prm2cim_proz_leistung_nachbereitung;
(9)
(10)  /*
(11)   * Transformation von PRM nach CIM basierend auf dem Referenzmodell
(12)   * für die Prozess-Sicht einer Anforderungsspezifikation
(13)   */
(14)  transformation prm2cim_proz(refmdl : bpmn, prm : uml, target : bpmn) {
(15)
(16)    -- Schlüssel für Identität der Elemente festlegen
(17)    key bpmn::SequenceEdge{iD};
(18)    key bpmn::Lane{iD};
(19)    key bpmn::Activity{iD};
(20)    key bpmn::DataObject{iD};
(21)
(22)    /*
(23)     * Transformation des Diagramms
(24)     */
(25)    top relation diagram{
(26)      modelName : String;
(27)
(28)      checkonly domain refmdl refd : bpmn::BpmnDiagram {
(29)        pools = refpool : bpmn::Pool {}
(30)      };
(31)
(32)      checkonly domain prm mdl : uml::Model {
```

```
(33)          name = modelName,
(34)          packagedElement = pack : uml::Package { }
(35)       };
(36)
(37)       enforce domain target diag : bpmn::BpmnDiagram {
(38)         name = modelName,
(39)         pools = pool : bpmn::Pool {}
(40)       };
(41)
(42)       where {
(43)         -- Transformation der im Diagramm enthaltenen Artefakte
(44)         diagramArtifact(refd, pack, diag);
(45)         -- Transformation der im Diagramm enthaltenen Pools
(46)         diagramPool(refpool, pack, pool);
(47)       }
(48)     }
(49)
(50)     /*
(51)      * Transformation der im Diagramm enthaltenen Artefakte
(52)      */
(53)     relation diagramArtifact {
(54)       checkonly domain refmdl refd : bpmn::BpmnDiagram {
(55)         artifacts = refa : bpmn::Artifact { }
(56)       };
(57)
(58)       primitive domain pack : uml::Package;
(59)
(60)       enforce domain target diag : bpmn::BpmnDiagram {
(61)         artifacts = a : bpmn::Artifact { }
(62)       };
(63)
(64)       where {
(65)         -- Semantische konkretisierung der im Diagramm enthaltenen eLoGo-Notiz
(66)         --
      prm2cim_proz_allgemein::diagramArtifactTextAnnotation_eLoGoNotiz(refa, pack,
      a);
(67)       }
(68)
(69)     }
(70)
(71)     /*
(72)      * Transformation der im Diagram enthaltenen Pools inkl. ihrer Elemente
(73)      */
(74)     relation diagramPool{
(75)       pname : String;
(76)
(77)       checkonly domain refmdl refp : bpmn::Pool {  };
(78)
(79)       checkonly domain prm pack : uml::Package {
(80)           name = pname
(81)       };
(82)
(83)       enforce domain target p : bpmn::Pool {
(84)         name = pname
(85)       };
(86)
(87)       where {
(88)         -- Transformation der im Pool enthaltenen Artefakte
(89)         diagramPoolArtifact(refp, pack, p);
(90)         -- Transformation der im Pool enthaltenen Aktivitäten
(91)         diagramPoolActivity(refp,pack,p);
(92)       }
(93)     }
(94)
(95)     /*
(96)      *  Transformation der im Diagramm enthaltenen Artifacts
(97)      */
(98)     relation diagramPoolArtifact {
(99)       ident : String;
(100)      checkonly domain refmdl refp : bpmn::Pool{
(101)        artifacts = refd : bpmn::DataObject {
(102)          iD = ident
(103)        }
(104)      };
(105)
```

```
(106)     primitive domain pack : uml::Package;
(107)
(108)     enforce domain target p : bpmn::Pool{
(109)       artifacts = d : bpmn::DataObject {
(110)         iD = ident
(111)       }
(112)     };
(113)
(114)     where {
(115)       -- Konkretisierung der Semantik von DataObjects
(116)       -- Anliegen - Bewusstwerdung
(117)
      prm2cim_proz_anliegen_bewusstwerdung::diagramPoolDataObject_Hinweis(refp,
      pack, p);
(118)
      prm2cim_proz_anliegen_bewusstwerdung::diagramPoolDataObject_Leistung(refp,
      pack, p);
(119)       -- Anliegen - Vorbereitung
(120)       -- ...
(121)       -- Anliegen - Antragstellung
(122)       -- ...
(123)       -- Anliegen - Bearbeitung verfolgen
(124)       -- ...
(125)       -- Leistung - Bearbeitung
(126)       -- ...
(127)       -- Leistung - Ergebnismitteilung
(128)       -- ...
(129)       -- Leistung - Nachbereitung
(130)       -- ...
(131)       -- Gesamtprozess
(132)       -- ...
(133)       -- Transformation von DataObjects ohne Konkretisierung der Semantik
(134)       prm2cim_proz_allgemein::diagramPoolDataObject_Undefined(refp, pack,
      p);
(135)     }
(136)
(137)   }
(138)
(139)   /*
(140)    * Transformation der im Pool enthaltenen Aktivitäten inkl. ihrer
      eingehenden
(141)    * und ausgehenden Kanten sowie Referenzen auf ihre Schwimmbahn (Lane)
(142)    */
(143)   relation diagramPoolActivity{
(144)     ident : String;
(145)     t : bpmn::ActivityType;
(146)     asso : OrderedSet(bpmn::Association);
(147)
(148)     checkonly domain refmdl refp : bpmn::Pool {
(149)       vertices = refv : bpmn::Activity {
(150)         iD = ident,
(151)         activityType = t
(152)       }
(153)     };
(154)
(155)     checkonly domain prm pack : uml::Package {
(156)
(157)     };
(158)
(159)     enforce domain target p : bpmn::Pool {
(160)       vertices = v : bpmn::Activity {
(161)         iD = ident,
(162)         activityType = t,
(163)         associations = asso->select(a | refv.associations->exists(refa|
      refa.source.iD = a.source.iD))
(164)       }
(165)     };
(166)
(167)     where {
(168)       asso =
      p.artifacts.associations.oclAsType(OrderedSet(bpmn::Association));
(169)
(170)       -- Eingehenden und ausgehenden Sequenzfluss der Aktivität
      transformieren
(171)       diagramPoolActivityOutEdges(refp, refv, p, v);
```

```
(172)       diagramPoolActivityOInEdges(refp, refv, p, v);
(173)
(174)       -- Transformation der Schwimmbahnen des Pools
(175)       diagramPoolLane(refp,p);
(176)
(177)       -- Konkretisierung der Semantik von Aktivitäten
(178)       -- Anliegen - Bewusstwerdung
(179)
      prm2cim_proz_anliegen_bewusstwerdung::diagramPoolActivity_BewusstwerdungAufz
      eigen(refv, pack, v);
(180)       -- Anliegen - Antragstellung
(181)       -- ...
(182)       -- Anliegen - Vorbereitung
(183)       -- ...
(184)       -- Anliegen - Bearbeitung verfolgen
(185)       -- ...
(186)       -- Leistung - Bearbeitung
(187)       -- ...
(188)       -- Leistung - Ergebnismitteilung
(189)       -- ...
(190)       -- Leistung - Nachbereitung
(191)       -- ...
(192)       -- Gesamtprozess
(193)       -- ...
(194)
(195)       -- Transformation von Aktivitäten ohne Konkretisierung der Semantik
(196)       prm2cim_proz_allgemein::diagramPoolActivity_Undefined(refv, pack, v);
(197)     }
(198)   }
(199)
(200)   /*
(201)    * Erzeugt die ausgehenden Kanten einer Aktivität
(202)    *
(203)    * Hinweis: Muss im Kontext des Pools erfolgen, über den oben definierten
      key
(204)    * werden doppelte Edges verhindert
(205)    */
(206)   relation diagramPoolActivityOutEdges {
(207)     ident : String;
(208)
(209)     checkonly domain refmdl refp : bpmn::Pool {
(210)       sequenceEdges = refse : bpmn::SequenceEdge {
(211)         iD = ident
(212)       }
(213)     };
(214)
(215)     checkonly domain refmdl refa : bpmn::Activity {
(216)       outgoingEdges = refoute : bpmn::SequenceEdge {}
(217)     };
(218)
(219)     enforce domain target p : bpmn::Pool  {
(220)       sequenceEdges = se : bpmn::SequenceEdge {
(221)         iD = ident
(222)       }
(223)     };
(224)
(225)     enforce domain target a : bpmn::Activity {
(226)       -- Hier wird die Schnittmenge aus den Kanten gebildet, die es
      einerseits
(227)       -- im Pool insgesamt gibt und den Kanten, die andererseits eingehende
      Kanten
(228)       -- der Aktivität im Referenzprozess sind!
(229)       outgoingEdges = p.sequenceEdges->intersection(refa.outgoingEdges)
(230)     };
(231)
(232)   }
(233)
(234)   /*
(235)    * Erzeugt die eingehenden Kanten einer Aktivität
(236)    *
(237)    * Hinweis: Muss im Kontext des Pools erfolgen, über den oben definierten
      key
(238)    * werden doppelte Edges verhindert
(239)    */
(240)   relation diagramPoolActivityOInEdges {
```

```
(241)     ident : String;
(242)
(243)     checkonly domain refmdl refp : bpmn::Pool {
(244)       sequenceEdges = refse : bpmn::SequenceEdge {
(245)         iD = ident
(246)       }
(247)     };
(248)
(249)     checkonly domain refmdl refa : bpmn::Activity {
(250)       incomingEdges = refine : bpmn::SequenceEdge {}
(251)     };
(252)
(253)     enforce domain target p : bpmn::Pool  {
(254)       sequenceEdges = se : bpmn::SequenceEdge {
(255)         iD = ident
(256)       }
(257)     };
(258)
(259)     enforce domain target a : bpmn::Activity {
(260)       -- Hier wird die Schnittmenge aus den Kanten gebildet, die es
      einerseits
(261)       -- im Pool insgesamt gibt und den Kanten, die andererseits eingehende
      Kanten
(262)       -- der Aktivität im Referenzprozess sind!
(263)       incomingEdges = p.sequenceEdges->intersection(refa.incomingEdges)
(264)     };
(265)
(266)   }
(267)
(268)   /*
(269)    * Transformation der im Pool enthaltenen Schwimmbahnen inkl. der
      Referenzen auf enthaltene Aktivitäten
(270)    */
(271)   relation diagramPoolLane {
(272)     ident : String;
(273)     n : String;
(274)
(275)     checkonly domain refmdl refp : bpmn::Pool {
(276)       lanes = refl : bpmn::Lane {
(277)         name = n,
(278)         iD = ident
(279)       }
(280)     };
(281)
(282)     enforce domain target p : bpmn::Pool {
(283)       lanes = l : bpmn::Lane {
(284)         name = n,
(285)         iD = ident
(286)       }
(287)     };
(288)
(289)     where {
(290)       -- Vervollständigung der Schwimmbahnen um Refeferenzen auf enthaltene
      Aktivititen
(291)       diagramPoolLaneRefActivities(refp, refl, p, l);
(292)     }
(293)   }
(294)
(295)   /*
(296)    * Vervollständigung der Schwimmbahnen um Refeferenzen auf enthaltene
      Aktivititen
(297)    */
(298)   relation diagramPoolLaneRefActivities{
(299)     ident : String;
(300)     act : OrderedSet(bpmn::Activity);
(301)
(302)     checkonly domain refmdl refp : bpmn::Pool {
(303)       vertices = refv : bpmn::Activity {
(304)         iD = ident
(305)       }
(306)     };
(307)
(308)     checkonly domain refmdl refl : bpmn::Lane {
(309)       activities = refa : bpmn::Activity {
(310)         iD = ident
```

```
(311)       }
(312)     };
(313)
(314)     enforce domain target p : bpmn::Pool {
(315)       vertices = v : bpmn::Activity {
(316)         iD = ident
(317)       }
(318)     };
(319)
(320)     enforce domain target l : bpmn::Lane {
(321)       activities = p.vertices.oclAsType(OrderedSet(bpmn::Activity))-
      >intersection(refl.activities)
(322)     };
(323)
(324)     where {
(325)       act = refl.activities;
(326)     }
(327)   }
(328)
(329) }
```

Listing 54 – Transformation „prm2cim_proz“ (prm2cim_proz.qvt)

```
(330) import prm2cim_proz_allgemein;
(331)
(332) transformation prm2cim_proz_anliegen_bewusstwerdung(refmdl : bpmn, prm :
      uml, target : bpmn) {
(333)
(334)   /*
(335)    * Transformation der Aktivität "BewusstwerdungAufzeigen"
(336)    */
(337)   relation diagramPoolActivity_BewusstwerdungAufzeigen {
(338)     n : String;
(339)     d : String;
(340)     r : String;
(341)
(342)     checkonly domain refmdl refa : bpmn::Activity {
(343)       name = n,
(344)       documentation = d,
(345)       ncname = 'BewusstwerdungAufzeigen'
(346)     };
(347)
(348)     checkonly domain prm pack : uml::Package {
(349)
(350)     };
(351)
(352)     enforce domain target a: bpmn::Activity {
(353)       name = n.replace('Verwaltungskontakt', r),
(354)       documentation = d.replace('Verwaltungskontakt', r),
(355)       ncname = 'BewusstwerdungAufzeigen'
(356)     };
(357)
(358)     where {
(359)       r = pack.packagedElement->select(inst |
      inst.oclAsType(uml::InstanceSpecification).classifier->exists(clf |
      clf.name='VwAntrag'))->first().name;
(360)
(361)     }
(362)   }
(363)
(364)   /*
(365)    * Transformation des Data Objects "Hinweis"
(366)    */
(367)   relation diagramPoolDataObject_Hinweis {
(368)     vn, ident : String;
(369)
(370)     checkonly domain refmdl refp : bpmn::Pool {
(371)       artifacts = refd : bpmn::DataObject{
(372)       ncname = 'Hinweis',
(373)       iD = ident
(374)       }
(375)     };
(376)
(377)     primitive domain pack : uml::Package;
(378)
```

```
(379)     enforce domain target p : bpmn::Pool {
(380)       artifacts = d : bpmn::DataObject {
(381)         name = refd.name + ' auf ' + vn,
(382)         ncname = 'Hinweis',
(383)         iD = ident
(384)       }
(385)     };
(386)
(387)     where {
(388)       vn = pack.packagedElement->select(p |
      p.oclAsType(uml::InstanceSpecification).classifier->exists( cls | cls.name =
      'VwProdukt'))->first().name;
(389)       prm2cim_proz_allgemein::diagramPoolDataObjectAssociation(refd, refp,
      d, p);
(390)     }
(391)   }
(392)
(393)   /*
(394)    * Transformation des Data Objects "Verwaltungsleistung"
(395)    */
(396)   relation diagramPoolDataObject_Leistung {
(397)     vn, ident: String;
(398)
(399)     checkonly domain refmdl refp : bpmn::Pool {
(400)       artifacts = refd : bpmn::DataObject {
(401)         ncname = 'Verwaltungsleistung',
(402)         iD = ident
(403)       }
(404)     };
(405)
(406)     primitive domain pack : uml::Package;
(407)
(408)     enforce domain target p : bpmn::Pool {
(409)       artifacts = d : bpmn::DataObject {
(410)         name = vn,
(411)         ncname = 'Verwaltungsleistung',
(412)         iD = ident
(413)       }
(414)     };
(415)
(416)     where {
(417)       vn = pack.packagedElement->select(p |
      p.oclAsType(uml::InstanceSpecification).classifier->exists( cls | cls.name =
      'VwProdukt'))->first().name;
(418)       prm2cim_proz_allgemein::diagramPoolDataObjectAssociation(refd, refp,
      d, p);
(419)     }
(420)
(421)   }
(422)
(423)
(424) }
```

Listing 55 – Transformation „prm2cim_proz_anliegen_bewusstwerdung“ (prm2cim_proz_anliegen_bewusstwerdung.qvt)

```
(1)   transformation prm2cim_proz_allgemein(refmdl : bpmn, prm : uml, target :
      bpmn) {
(2)
(3)     /*
(4)      * Semantische konkretisierung der im Diagramm enthaltenen eLoGo-Notiz
(5)      */
(6)     relation diagramArtifactTextAnnotation_eLoGoNotiz {
(7)
(8)       checkonly domain refmdl refa : bpmn::TextAnnotation {
(9)         ncname = 'eLoGoNotiz'
(10)      };
(11)
(12)      primitive domain pack : uml::Package;
(13)
(14)      enforce domain target a : bpmn::TextAnnotation {
(15)        name = refa.name.replace('eLoGo-Referenzprozessmodell', pack.name + '-
      Prozesmodell')
(16)      };
(17)
```

```
(18)    }
(19)
(20)    /*
(21)     * Transformation von Aktivitäten, für die keine Konkretisierung der
       Semantik vorgesehen ist
(22)     */
(23)    relation diagramPoolActivity_Undefined {
(24)      n, cn : String;
(25)
(26)      checkonly domain refmdl refa : bpmn::Activity {
(27)        name = n,
(28)        ncname = cn
(29)      };
(30)
(31)      primitive domain pack : uml::Package;
(32)
(33)      enforce domain target a : bpmn::Activity {
(34)        name = n,
(35)        ncname = cn
(36)      };
(37)
(38)      when {
(39)        cn.oclIsUndefined();
(40)      }
(41)
(42)    }
(43)
(44)    /*
(45)     * Transformation von DataObjects, für die keine Konkretisierung der
       Semantik vorgesehen ist
(46)     */
(47)    relation diagramPoolDataObject_Undefined {
(48)      n, cn, ident : String;
(49)
(50)      checkonly domain refmdl refp : bpmn::Pool {
(51)        artifacts = refd : bpmn::DataObject {
(52)          name = n,
(53)          ncname = cn,
(54)          iD = ident
(55)        }
(56)      };
(57)
(58)      primitive domain pack : uml::Package;
(59)
(60)      enforce domain target p : bpmn::Pool {
(61)        artifacts = d : bpmn::DataObject {
(62)          name = n,
(63)          ncname = cn,
(64)          iD = ident
(65)        }
(66)      };
(67)
(68)      when {
(69)        cn.oclIsUndefined();
(70)      }
(71)
(72)      where {
(73)        diagramPoolDataObjectAssociation(refd, refp, d, p);
(74)      }
(75)
(76)    }
(77)
(78)    relation diagramPoolDataObjectAssociation {
(79)
(80)      checkonly domain refmdl refd : bpmn::DataObject {
(81)        associations = refass : bpmn::Association {  }
(82)      };
(83)
(84)      primitive domain refp : bpmn::Pool;
(85)
(86)      enforce domain target d: bpmn::DataObject{
(87)          associations = ass : bpmn::Association {
(88)          }
(89)      };
(90)
```

```
(91)      primitive domain p : bpmn::Pool;
(92)
(93)    }
(94)
(95)  }
```

Listing 56 – Transformation „prm2cim_proz_allgemein“ (prm2cim_proz_allgemein.qvt)

F.2 Anforderungsmodelle

```
(1)   import prm2cim_anf_anliegen_bewusstwerdung;
(2)   import prm2cim_anf_anliegen_initiierung;
(3)   import prm2cim_anf_anliegen_vorbereitung;
(4)   import prm2cim_anf_leistung_beendigung;
(5)   import prm2cim_anf_leistung_erbringung;
(6)   import prm2cim_anf_leistung_initiierung;
(7)   import prm2cim_anf_leistung_massnahmenfestlegung;
(8)   import prm2cim_anf_leistung_mitteilung;
(9)   import prm2cim_anf_leistung_mitteilungserbringung;
(10)  import prm2cim_anf_leistung_mitwirkung;
(11)  import prm2cim_anf_leistung_sachverhaltsermittlung;
(12)  import prm2cim_anf_leistung_tatbestandspruefung;
(13)  import prm2cim_anf_allgemein;
(14)  import prm2cim_anf_allgemein_begleitung;
(15)
(16)  /*
(17)   * Transformation von PRM nach CIM basierend auf dem UML-Profil
(18)   * des eLoGo-Referenzanforderungsmodells für die Anwendungsfall-
(19)   * und Klassendiagramme einer Anforderungsspezifikation
(20)   */
(21)
(22)  transformation prm2cim_anf(prm: uml, profiles: uml, target : uml) {
(23)
(24)    key Package {name};
(25)
(26)    top relation model {
(27)
(28)      n : String;
(29)
(30)      checkonly domain prm srcMdl :uml::Model{
(31)        name = n,
(32)        packagedElement = srcPack : uml::Package { }
(33)      };
(34)
(35)      enforce domain target dstMdl: uml::Model{
(36)        name = n,
(37)        packagedElement = packAnliegen : uml::Package {
(38)          name = n + ' Anliegen'
(39)        },
(40)        packagedElement = packVerwaltungsleistung : uml::Package {
(41)          name = n + ' Verwaltungsleistung'
(42)        },
(43)        packagedElement = packQuerschnittsfnkt : uml::Package {
(44)          name = n + ' Querschnittsfunktionen'
(45)        }
(46)      };
(47)
(48)      where {
(49)        -- Package "Anliegen"
(50)        -- Abbildungsregeln für den Anwendungsfall "Verwaltungleistung bewusst
      machen"
(51)        prm2cim_anf_anliegen_bewusstwerdung::model(srcPack, packAnliegen);
(52)        -- Abbildungsregeln für den Anwendungsfall "Bearbeitung beobachten"
(53)        -- ...
(54)        -- Abbildungsregeln für den Anwendungsfall "Verwaltungsleistung
      bewusst machen"
(55)        -- ...
(56)        -- Abbildungsregeln für den Anwendungsfall "Verwaltungsleistung
      initiieren"
(57)        -- ...
(58)        -- Abbildungsregeln für den Anwendungsfall "Verwaltungskontakt
      vorbereiten"
(59)        -- ...
(60)
(61)        -- Package "Verwaltungsleistung"
```

```
(62)        -- Abbildungsregeln für den Anwendungsfall "Bearbeitung beenden"
(63)        -- ...
(64)        -- Abbildungsregeln für den Anwendungsfall "Leistung erbringen"
(65)        prm2cim_anf_leistung_erbringung::model(srcPack,
      packVerwaltungsleistung);
(66)
(67)        -- Abbildungsregeln für den Anwendungsfall "Leistungsbearbeitung
      initiieren"
(68)        -- ...
(69)        -- Abbildungsregeln für den Anwendungsfall "Maßnahmen festlegen"
(70)        -- ...
(71)        -- Abbildungsregeln für den Anwendungsfall "Ergebnis mitteilen"
(72)        -- ...
(73)        -- Abbildungsregeln für den Anwendungsfall "Ergebnis mitteilen und
      ggf. Leistung erbringen"
(74)        -- ...
(75)        -- Abbildungsregeln für den Anwendungsfall "Mitwirkung durchführen"
(76)        -- ...
(77)        -- Abbildungsregeln für den Anwendungsfall "Sachverhalt sichten,
      prüfen und erfassen"
(78)        -- ...
(79)        -- Abbildungsregeln für den Anwendungsfall "Tatbestand prüfen"
(80)        -- ...
(81)
(82)
(83)        -- Package "Querschnittsfunktionen"
(84)        -- Abbildungsregeln für den Anwendungsfall "Bearbeitung begleiten"
(85)        -- ...
(86)
(87)        -- Profil
(88)        profile(srcMdl, dstMdl);
(89)      }
(90)
(91)    }
(92)
(93)
      /*************************************************************************
(94)     *
(95)     * Domänenobjektmodell
(96)     * Stereotypen
(97)     *
(98)
      *************************************************************************/
(99)
(100)   top relation stereotype_Verwaltungsleistung {
(101)     n : String;
(102)     checkonly domain prm srcInst : uml::InstanceSpecification {
(103)       name = n,
(104)       classifier = clsf : uml::Class {
(105)         name = 'VwLeistung'
(106)       }
(107)     };
(108)
(109)     enforce domain target stype : eGovRefAnf::Verwaltungsleistung {
(110)       base_Class = cls : uml::Class{ }
(111)     };
(112)
(113)     when {
(114)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementClass_Verwaltungsleistung
      (srcInst, cls);
(115)     }
(116)   }
(117)
(118)   top relation stereotype_Verwaltungsprodukt {
(119)     n : String;
(120)     checkonly domain prm srcInst : uml::InstanceSpecification {
(121)       name = n,
(122)       classifier = clsf : uml::Class {
(123)         name = 'VwRechtsfolge'
(124)       }
(125)     };
(126)
(127)     enforce domain target bewh : eGovRefAnf::Rechtsfolge{
(128)       base_Class = cls :uml::Class { }
```

```
(129)     };
(130)
(131)     when {
(132)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementClass_Verwaltungsprodukt(
      srcInst, cls);
(133)
(134)     }
(135)   }
(136)
(137)   top relation stereotype_Nachfrager {
(138)     n : String;
(139)     checkonly domain prm srcInst : uml::InstanceSpecification {
(140)       name = n,
(141)       classifier = clsf : uml::Class {
(142)         name = 'VwAntragsteller'
(143)       }
(144)     };
(145)
(146)     enforce domain target bewh : eGovRefAnf::Nachfrager{
(147)       base_Class = cls :uml::Class { }
(148)     };
(149)
(150)     when {
(151)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementClass_Nachfrager(srcInst,
      cls);
(152)
(153)     }
(154)   }
(155)
(156)   top relation stereotype_HinweisInformation {
(157)     n : String;
(158)     checkonly domain prm srcInst : uml::InstanceSpecification {
(159)       name = n,
(160)       classifier = clsf : uml::Class {
(161)         name = 'VwHinweisInformation'
(162)       }
(163)     };
(164)
(165)     enforce domain target bewh : eGovRefAnf::HinweisInformation{
(166)       base_Class = cls :uml::Class {   }
(167)     };
(168)
(169)     when {
(170)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementClass_HinweisInformation(
      srcInst, cls);
(171)
(172)     }
(173)   }
(174)
(175)   top relation stereotype_Lebenslage {
(176)     n : String;
(177)     checkonly domain prm srcInst : uml::InstanceSpecification {
(178)       name = n,
(179)       classifier = clsf : uml::Class {
(180)         name = 'VwLebenslage'
(181)       }
(182)     };
(183)
(184)     enforce domain target bewh : eGovRefAnf::Lebenslage {
(185)       base_Class = cls :uml::Class {   }
(186)     };
(187)
(188)     when {
(189)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementClass_Lebenslage(srcInst,
      cls);
(190)
(191)     }
(192)   }
(193)
(194)
      /*************************************************************************
```

```
(195)    *
(196)    * Anwendungsfallmodell
(197)    * Stereotypen
(198)    *
(199)
      ***************************************************************************/
(200)
(201)
(202)   top relation stereotype_VerwaltungsleistungBewusstMachen {
(203)     n : String;
(204)     checkonly domain prm srcInst : uml::InstanceSpecification {
(205)       name = n,
(206)       classifier = clsf : uml::Class {
(207)         name = 'VwLeistung'
(208)       }
(209)     };
(210)
(211)     enforce domain target bewh :
      eGovRefAnf::Verwaltungsleistungbewusstmachen {
(212)       base_UseCase = uc :uml::UseCase { }
(213)     };
(214)
(215)     when {
(216)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementUseCase_Verwaltungsleistu
      ngBewusstMachen(srcInst, uc);
(217)
(218)     }
(219)   }
(220)
(221)   top relation stereotype_LeistungErbringen {
(222)     n : String;
(223)     checkonly domain prm srcInst : uml::InstanceSpecification {
(224)       name = n,
(225)       classifier = clsf : uml::Class {
(226)         name = 'VwLeistung'
(227)       }
(228)     };
(229)
(230)     enforce domain target bewh : eGovRefAnf::Leistungerbringen {
(231)       base_UseCase = uc :uml::UseCase { }
(232)     };
(233)
(234)     when {
(235)
      prm2cim_anf_leistung_erbringung::packageElementUseCase_LeistungErbringen(src
      Inst, uc);
(236)
(237)     }
(238)   }
(239)
(240)   top relation stereotype_NachfragerAkteur {
(241)     n : String;
(242)     checkonly domain prm srcInst : uml::InstanceSpecification {
(243)       name = n,
(244)       classifier = clsf : uml::Class {
(245)         name = 'VwAntragsteller'
(246)       }
(247)     };
(248)
(249)     enforce domain target bewh : eGovRefAnf::NachfragerAkteur {
(250)       base_Actor = act : uml::Actor { }
(251)     };
(252)
(253)     when {
(254)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementActor_Nachfrager(srcInst,
      act);
(255)     }
(256)   }
(257)
(258)   top relation stereotype_BearbeiterFrontOffice {
(259)     n : String;
(260)     checkonly domain prm srcInst : uml::InstanceSpecification {
(261)       name = n,
```

```
(262)       classifier = clsf : uml::Class {
(263)         name = 'VwAnwenderFrontOffice'
(264)       }
(265)     };
(266)
(267)     enforce domain target bewh : eGovRefAnf::BearbeiterFrontOffice{
(268)       base_Actor = act : uml::Actor { }
(269)     };
(270)
(271)     when {
(272)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementActor_BearbeiterFrontOffi
      ce(srcInst, act);
(273)     }
(274)   }
(275)
(276)   top relation stereotype_Bearbeiter {
(277)     n : String;
(278)     checkonly domain prm srcInst : uml::InstanceSpecification {
(279)       name = n,
(280)       classifier = clsf : uml::Class {
(281)         name = 'VwAnwender'
(282)       }
(283)     };
(284)
(285)     enforce domain target bewh : eGovRefAnf::Bearbeiter{
(286)       base_Actor = act : uml::Actor { }
(287)     };
(288)
(289)     when {
(290)
      prm2cim_anf_anliegen_bewusstwerdung::packageElementActor_Bearbeiter(srcInst,
      act);
(291)     }
(292)   }
(293)
(294)   top relation stereotype_Ausfuehrender {
(295)     n : String;
(296)     checkonly domain prm srcInst : uml::InstanceSpecification {
(297)       name = n,
(298)       classifier = clsf : uml::Class {
(299)         name = 'VwAusfuehrender'
(300)       }
(301)     };
(302)
(303)     enforce domain target bewh : eGovRefAnf::Bearbeiter{
(304)       base_Actor = act : uml::Actor { }
(305)     };
(306)
(307)     when {
(308)
      prm2cim_anf_leistung_erbringung::packageElementActor_Ausfuehrender(srcInst,
      act);
(309)     }
(310)   }
(311)
(312)   top relation stereotype_AusfuehrenderFrontOffice {
(313)     n : String;
(314)     checkonly domain prm srcInst : uml::InstanceSpecification {
(315)       name = n,
(316)       classifier = clsf : uml::Class {
(317)         name = 'VwAusfuehrenderFrontOffice'
(318)       }
(319)     };
(320)
(321)     enforce domain target bewh : eGovRefAnf::BearbeiterFrontOffice{
(322)       base_Actor = act : uml::Actor { }
(323)     };
(324)
(325)     when {
(326)
      prm2cim_anf_leistung_erbringung::packageElementActor_AusfuehrenderFrontOffic
      e(srcInst, act);
(327)     }
(328)   }
```

```
(329)
(330)   top relation stereotype_AusfuehrenderBackOffice {
(331)     n : String;
(332)     checkonly domain prm srcInst : uml::InstanceSpecification {
(333)       name = n,
(334)       classifier = clsf : uml::Class {
(335)         name = 'VwAusfuehrenderBackOffice'
(336)       }
(337)     };
(338)
(339)     enforce domain target bewh : eGovRefAnf::BearbeiterBackOffice{
(340)       base_Actor = act : uml::Actor { }
(341)     };
(342)
(343)     when {
(344)
      prm2cim_anf_leistung_erbringung::packageElementActor_AusfuehrenderBackOffice
      (srcInst, act);
(345)     }
(346)   }
(347)
(348)   top relation stereotype_RechtsfolgenMerkmalNF {
(349)
(350)     checkonly domain prm srcInst1 : uml::InstanceSpecification {
(351)       classifier = clsf1 : uml::Class {
(352)         name = 'VwRechtsfolgenMerkmal'
(353)       }
(354)     };
(355)
(356)     checkonly domain prm srcInst2 : uml::InstanceSpecification {
(357)       classifier = clsf2 : uml::Class {
(358)         name = 'VwLeistung'
(359)       }
(360)     };
(361)
(362)     enforce domain target bewh : eGovRefAnf::Rahmenbedingung{
(363)       ausfuehrungsbezogen =
      prm2cim_anf_leistung_erbringung::hasRuntimeNF(srcInst1),
(364)       entwicklungsbezogen =
      prm2cim_anf_leistung_erbringung::hasDesigntimeNF(srcInst1),
(365)       base_Comment =  cmt : uml::Comment { }
(366)     };
(367)
(368)     when {
(369)
      prm2cim_anf_leistung_erbringung::packageElementComment_RechtsfolgeMerkmalNF(
      srcInst1, srcInst2, cmt);
(370)     }
(371)   }
(372)
(373)
      /**************************************************************************
(374)    *
(375)    * Erzeugung des UML-Profils für die eLoGo-Referenzmodelle für E-
      Government
(376)    *
(377)
      **************************************************************************/
(378)   -- Bestmögliche Annäherung an die korrekte Definition des UML-Profils und
      seiner Anwendung
(379)   -- basierend auf EMF 2.6 muss in Verbindung mit apply-profile.xsl um die
      Bestandteile
(380)   -- erweitert werden, die per QVT-R nicht angelegt werden können.
(381)   relation profile {
(382)     n : String;
(383)
(384)     checkonly domain prm p : uml::Model {
(385)       name = n
(386)     };
(387)
(388)     enforce domain target t : uml::Model {
(389)       name = n,
(390)       profileApplication = pa : uml::ProfileApplication {
(391)         eAnnotations = ano : ecore::EAnnotation {
(392)           source = 'http://www.eclipse.org/uml2/2.0.0/UML'
```

```
(393)         }
(394)       }
(395)     };
(396)
(397)   }
(398)
(399) }
```

Listing 57 – Transformation „prm2cim_anf“ (prm2cim_anf.qvt)

```
(1)   -- Anwendungsfall "Verwaltungsleistung bewusst machen"
(2)   -- (Hinweis: Signatur dieser Transformation muss mit der Signatur der
      importierenden
(3)   -- Transformation übereinstimmen, da sonst trotz definierter Schlüssel
      Elemente doppel
(4)   -- angelegt werden)
(5)   transformation prm2cim_anf_anliegen_bewusstwerdung(prm: uml, profiles: uml,
      target : uml) {
(6)
(7)     -- Schlüssel die die Identität des Elementes bestimmen
(8)     key uml::Class {name};
(9)     key uml::UseCase {name};
(10)
(11)    /*
(12)     * Relation zur Transformation des Modells in den Anwendungsfall
(13)     * "Verwaltungsleistung bewusst machen"
(14)     */
(15)    relation model {
(16)
(17)      checkonly domain prm srcPack :uml::Package{ };
(18)
(19)      enforce domain target packAnliegen : uml::Package {  };
(20)      /*
(21)      n : String;
(22)
(23)      checkonly domain prm srcMdl :uml::Model{
(24)        name = n,
(25)        packagedElement = srcPack : uml::Package { }
(26)      };
(27)
(28)      enforce domain target dstMdl: uml::Model{
(29)        name = n,
(30)        packagedElement = packAnliegen : uml::Package {
(31)          name = n + ' Anliegen'
(32)        },
(33)        packagedElement = packVerwaltungsleistung : uml::Package {
(34)          name = n + ' Verwaltungsleistung'
(35)        },
(36)        packagedElement = packVerwaltungsleistung : uml::Package {
(37)          name = n + ' Querschnittsfunktionen'
(38)        }
(39)      };
(40)      */
(41)      where {
(42)
(43)        -- Domänenobjektmodell
(44)        -- Klassen
(45)        packageElement_Verwaltungsprodukt(srcPack, packAnliegen);
(46)        packageElement_Verwaltungsleistung(srcPack, packAnliegen);
(47)        packageElement_Nachfrager(srcPack, packAnliegen);
(48)        packageElement_Lebenslage(srcPack, packAnliegen);
(49)        packageElement_HinweisInformation(srcPack, packAnliegen);
(50)        packageElement_BenutzerProfil(packAnliegen);
(51)        -- Beziehungen
(52)
      packageElementAssoziation_Verwaltungsleistung_Verwaltungsprodukt(srcPack,
      packAnliegen);
(53)        packageElementAssoziation_Nachfrager_Verwaltungsleistung(srcPack,
      packAnliegen);
(54)        packageElementAssoziation_Nachfrager_Lebenslage(srcPack,
      packAnliegen);
(55)
      packageElementAssoziation_HinweisInformation_Verwaltungsleistung(srcPack,
      packAnliegen);
```

```
(56)
      packageElementAssoziation_HinweisInformation_Verwaltungsprodukt(srcPack,
      packAnliegen);
(57)        packageElementDependency_Lebenslage_HinweisInformation(srcPack,
      packAnliegen);
(58)        packageElementAssoziation_HinweisInformation_Benutzerprofil(srcPack,
      packAnliegen);
(59)
(60)        -- Anwendungsfallmodell
(61)        -- Anwendungsfall
(62)        packageElement_VerwaltungsleistungBewusstMachen(srcPack,
      packAnliegen);
(63)        packageElementAlt_VerwaltungsleistungBewusstMachen(srcPack,
      packAnliegen);
(64)        -- Akteure
(65)        packageElement_NachfragerAkteur(srcPack, packAnliegen);
(66)        packageElement_BearbeiterFrontOffice(srcPack, packAnliegen);
(67)        packageElement_Bearbeiter(srcPack, packAnliegen);
(68)        -- Beziehungen
(69)
      packageElementAssoziation_Nachfrager_VerwaltungsleistungBewusstMachen(srcPac
      k, packAnliegen);
(70)
      packageElementAssoziation_Bearbeiter_VerwaltungsleistungBewusstMachen(srcPac
      k, packAnliegen);
(71)
      packageElementAssoziation_BearbeiterFrontOffice_VerwaltungsleistungBewusstMa
      chen(srcPack, packAnliegen);
(72)
      packageElementAssoziationAlt_Nachfrager_VerwaltungsleistungBewusstMachen(src
      Pack, packAnliegen);
(73)
      packageElementAssoziationAlt_Bearbeiter_VerwaltungsleistungBewusstMachen(src
      Pack, packAnliegen);
(74)
      packageElementAssoziationAlt_BearbeiterFrontOffice_VerwaltungsleistungBewuss
      tMachen(srcPack, packAnliegen);
(75)
(76)      }
(77)    }
(78)
(79)
(80)
      /*************************************************************************
(81)       *
(82)       * Domänenobjektmodell
(83)       * Klassen
(84)       *
(85)
      *************************************************************************/
(86)
(87)    /*
(88)     *
(89)     */
(90)    relation packageElement_BenutzerProfil {
(91)
(92)      enforce domain target  pack : uml::Package {
(93)        packagedElement = cls1 : uml::Class {
(94)          name = 'BenutzerProfil'
(95)        },
(96)        packagedElement = cls2 : uml::Class {
(97)          name = 'Benutzer'
(98)        },
(99)        packagedElement = ass : uml::Association {
(100)         name = '',
(101)         ownedEnd = p1 : uml::Property {
(102)           type = cls1 : uml::Class { },
(103)           name = cls1.name,
(104)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(105)             value = 1
(106)           },
(107)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(108)             value = 1
(109)           }
(110)
```

```
(111)         },
(112)         ownedEnd = p2 : uml::Property {
(113)           type = cls2 : uml::Class { },
(114)           name = cls2.name,
(115)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(116)             value = 1
(117)           },
(118)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(119)             value = 1
(120)           }
(121)         }
(122)       }
(123)
(124)     };
(125)
(126)   }
(127)
(128)   /*
(129)    * Lebenslage als Element des Packages
(130)    */
(131)   relation packageElement_HinweisInformation {
(132)
(133)     checkonly domain prm srcPack :uml::Package {
(134)       packagedElement = srcInst : uml::InstanceSpecification {
(135)         classifier = clsf : uml::Class {
(136)           name = 'VwHinweisInformation'
(137)         }
(138)             }
(139)     };
(140)
(141)     enforce domain target  pack : uml::Package {
(142)       packagedElement = cls : uml::Class { }
(143)     };
(144)
(145)     when {
(146)       not(hasRefinement(srcInst));
(147)     }
(148)
(149)     where {
(150)       packageElementClass_HinweisInformation(srcInst, cls);
(151)     }
(152)
(153)   }
(154)
(155)   /*
(156)    * HinweisInformation als Klasse
(157)    */
(158)   relation packageElementClass_HinweisInformation {
(159)     n,b,a,q : String;
(160)     checkonly domain prm srcInst : uml::InstanceSpecification {
(161)       name = n,
(162)       classifier = clsf : uml::Class {
(163)         name = 'VwHinweisInformation'
(164)       }
(165)     };
(166)
(167)     enforce domain target cls : uml::Class {
(168)       name = n,
(169)       -- Dokumentation
(170)       eAnnotations = doc : ecore::EAnnotation {
(171)         source = 'http://www.topcased.org/documentation',
(172)         eModelElement = cls,
(173)         details = dtls : ecore::EStringToStringMapEntry {
(174)           value = '1. Beschreibung: ' + b + '; 2. Arbeitsnotiz: ' + a + '; 3. Quelle: ' + q,
(175)           _key = 'documentation'
(176)         }
(177)       }
(178)     };
(179)
(180)     when {
(181)       not(hasRefinement(srcInst));
(182)     }
(183)
(184)     where {
```

```
(185)       q = getQuelle(srcInst);
(186)       a = getArbeitsnotiz(srcInst);
(187)       b = getBeschreibung(srcInst);
(188)     }
(189)   }
(190)
(191)
(192)   /*
(193)    * Lebenslage als Element des Packages
(194)    */
(195)   relation packageElement_Lebenslage {
(196)
(197)     checkonly domain prm srcPack :uml::Package {
(198)       packagedElement = srcInst : uml::InstanceSpecification {
(199)         classifier = clsf : uml::Class {
(200)           name = 'VwLebenslage'
(201)         }
(202)       }
(203)     };
(204)
(205)     enforce domain target  pack : uml::Package {
(206)       packagedElement = cls : uml::Class { }
(207)
(208)     };
(209)
(210)     when {
(211)       not(hasRefinement(srcInst));
(212)     }
(213)
(214)     where {
(215)       packageElementClass_Lebenslage(srcInst, cls);
(216)     }
(217)   }
(218)
(219)   /*
(220)    * Lebenslage als Klasse
(221)    */
(222)   relation packageElementClass_Lebenslage {
(223)     n,a,b,q : String;
(224)     checkonly domain prm srcInst : uml::InstanceSpecification {
(225)       name = n,
(226)       classifier = clsf : uml::Class {
(227)         name = 'VwLebenslage'
(228)       }
(229)     };
(230)
(231)     enforce domain target cls : uml::Class {
(232)       name = n,
(233)       -- Dokumentation
(234)       eAnnotations = doc : ecore::EAnnotation {
(235)         source = 'http://www.topcased.org/documentation',
(236)         eModelElement = cls,
(237)         details = dtls : ecore::EStringToStringMapEntry {
(238)           value = '1. Beschreibung: ' + b + '; 2. Arbeitsnotiz: ' + a + '; 
      3. Quelle: ' + q,
(239)           _key = 'documentation'
(240)         }
(241)       }
(242)     };
(243)
(244)     when {
(245)       not(hasRefinement(srcInst));
(246)     }
(247)
(248)     where {
(249)       q = getQuelle(srcInst);
(250)       a = getArbeitsnotiz(srcInst);
(251)       b = getBeschreibung(srcInst);
(252)     }
(253)   }
(254)
(255)   /*
(256)    * Nachfrager als Element des Packages
(257)    */
(258)   relation packageElement_Nachfrager {
```

```
(259)
(260)     checkonly domain prm srcPack :uml::Package {
(261)       packagedElement = srcInst : uml::InstanceSpecification {
(262)         classifier = clsf : uml::Class {
(263)           name = 'VwAntragsteller'
(264)         }
(265)       }
(266)     };
(267)
(268)     enforce domain target  pack : uml::Package {
(269)       packagedElement = cls : uml::Class { }
(270)     };
(271)
(272)     when {
(273)       not(hasRefinement(srcInst));
(274)     }
(275)
(276)     where {
(277)       packageElementClass_Nachfrager(srcInst, cls);
(278)     }
(279)   }
(280)
(281)   /*
(282)    * Nachfrager als Klasse
(283)    */
(284)   relation packageElementClass_Nachfrager {
(285)     n,b,a,q : String;
(286)     checkonly domain prm srcInst : uml::InstanceSpecification {
(287)       name = n,
(288)       classifier = clsf : uml::Class {
(289)         name = 'VwAntragsteller'
(290)       }
(291)     };
(292)
(293)     enforce domain target cls : uml::Class {
(294)       name = n,
(295)       -- Spezialisierung der Klasse Benutzer
(296)       superClass = scls : uml::Class {
(297)         name = 'Benutzer'
(298)       },
(299)       -- Dokumentation
(300)       eAnnotations = doc : ecore::EAnnotation {
(301)         source = 'http://www.topcased.org/documentation',
(302)         eModelElement = cls,
(303)         details = dtls : ecore::EStringToStringMapEntry {
(304)           value = '1. Beschreibung: ' + b + '; 2. Arbeitsnotiz: ' + a + '; 
      3. Quelle: ' + q,
(305)           _key = 'documentation'
(306)         }
(307)       }
(308)     };
(309)
(310)     when {
(311)       not(hasRefinement(srcInst));
(312)     }
(313)
(314)     where {
(315)       q = getQuelle(srcInst);
(316)       a = getArbeitsnotiz(srcInst);
(317)       b = getBeschreibung(srcInst);
(318)     }
(319)   }
(320)
(321)   /*
(322)    * Verwaltungsleistung als Element des Packages
(323)    */
(324)   relation packageElement_Verwaltungsleistung {
(325)
(326)     checkonly domain prm srcPack :uml::Package {
(327)       packagedElement = srcInst : uml::InstanceSpecification {
(328)         classifier = clsf : uml::Class {
(329)           name = 'VwLeistung'
(330)         }
(331)       }
(332)     };
```

```
(333)
(334)     enforce domain target  pack : uml::Package {
(335)       packagedElement = cls : uml::Class { }
(336)
(337)     };
(338)
(339)     when {
(340)       not(hasRefinement(srcInst));
(341)     }
(342)
(343)     where {
(344)       packageElementClass_Verwaltungsleistung(srcInst, cls);
(345)     }
(346)
(347)   }
(348)
(349)   /*
(350)    * Verwaltungsleistung als Klasse
(351)    */
(352)   relation packageElementClass_Verwaltungsleistung {
(353)     n,b,a,q : String;
(354)     checkonly domain prm srcInst : uml::InstanceSpecification {
(355)       name = n,
(356)       classifier = clsf : uml::Class {
(357)         name = 'VwLeistung'
(358)       }
(359)     };
(360)
(361)     enforce domain target cls : uml::Class {
(362)       name = n,
(363)       eAnnotations = doc : ecore::EAnnotation {
(364)         source = 'http://www.topcased.org/documentation',
(365)         eModelElement = cls,
(366)         details = dtls : ecore::EStringToStringMapEntry {
(367)           value = '1. Beschreibung: ' + b + '; 2. Arbeitsnotiz: ' + a + '; 3. Quelle: ' + q,
(368)           _key = 'documentation'
(369)         }
(370)       }
(371)     };
(372)
(373)     when {
(374)       not(hasRefinement(srcInst));
(375)     }
(376)
(377)     where {
(378)       q = getQuelle(srcInst);
(379)       a = getArbeitsnotiz(srcInst);
(380)       b = getBeschreibung(srcInst);
(381)     }
(382)   }
(383)
(384)
(385)   /*
(386)    * Verwaltungsprodukt als Element des Packages
(387)    */
(388)   relation packageElement_Verwaltungsprodukt {
(389)     n : String;
(390)
(391)     checkonly domain prm srcPack :uml::Package {
(392)       packagedElement = srcInst : uml::InstanceSpecification {
(393)         name = n,
(394)         classifier = clsf : uml::Class {
(395)           name = 'VwRechtsfolge'
(396)         }
(397)       }
(398)     };
(399)
(400)     enforce domain target  pack : uml::Package {
(401)       packagedElement = cls : uml::Class {
(402)       }
(403)     };
(404)
(405)     when {
(406)       not(hasRefinement(srcInst));
```

```
(407)     }
(408)
(409)     where {
(410)       packageElementClass_Verwaltungsprodukt(srcInst, cls);
(411)     }
(412)
(413)   }
(414)
(415)   /*
(416)    * Verwaltungsprodukt als Klasse
(417)    */
(418)   relation packageElementClass_Verwaltungsprodukt {
(419)     n, b, a, q : String;
(420)     checkonly domain prm srcInst : uml::InstanceSpecification {
(421)       name = n,
(422)       classifier = clsf : uml::Class {
(423)         name = 'VwRechtsfolge'
(424)       }
(425)     };
(426)
(427)     enforce domain target cls : uml::Class {
(428)       name = n,
(429)       eAnnotations = doc : ecore::EAnnotation {
(430)         source = 'http://www.topcased.org/documentation',
(431)         eModelElement = cls,
(432)         details = dtls : ecore::EStringToStringMapEntry {
(433)           value = '1. Beschreibung: ' + b + '; 2. Arbeitsnotiz: ' + a + '; 3. Quelle: ' + q,
(434)           _key = 'documentation'
(435)         }
(436)       }
(437)     };
(438)
(439)     when {
(440)       not(hasRefinement(srcInst));
(441)     }
(442)
(443)     where {
(444)       q = getQuelle(srcInst);
(445)       a = getArbeitsnotiz(srcInst);
(446)       b = getBeschreibung(srcInst);
(447)     }
(448)
(449)   }
(450)
(451)
      /*****************************************************************************
(452)    *
(453)    * Domänenobjektmodell
(454)    * Beziehungen
(455)    *
(456)
      *****************************************************************************/
(457)
(458)   /*
(459)    * Beziehung von Verwaltungsleistung zum Verwaltungsprodukt
(460)    */
(461)   relation packageElementAssoziation_Verwaltungsleistung_Verwaltungsprodukt {
(462)
(463)     checkonly domain prm srcPack :uml::Package {
(464)       packagedElement = srcInstVwl : uml::InstanceSpecification {
(465)         classifier = clsf1 : uml::Class {
(466)           name = 'VwLeistung'
(467)         }
(468)       },
(469)
(470)       packagedElement = srcInstVwp : uml::InstanceSpecification {
(471)         classifier = clsf2 : uml::Class {
(472)           name = 'VwRechtsfolge'
(473)         }
(474)       }
(475)
(476)     };
(477)
```

```
(478)     enforce domain target pack : uml::Package {
(479)       packagedElement = ass : uml::Association {
(480)         name = 'istVomTyp',
(481)         ownedEnd = p1 : uml::Property {
(482)           type = cls1 : uml::Class { },
(483)           name = srcInstVwl.name,
(484)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(485)             value = 1
(486)           },
(487)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(488)             value = 1
(489)           }
(490)
(491)         },
(492)         ownedEnd = p2 : uml::Property {
(493)           type = cls2 : uml::Class { },
(494)           name = srcInstVwp.name,
(495)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(496)             value = 1
(497)           },
(498)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(499)             value = 1
(500)           }
(501)         }
(502)       }
(503)     };
(504)
(505)     when {
(506)       not(hasRefinement(srcInstVwl));
(507)       not(hasRefinement(srcInstVwp));
(508)       packageElementClass_Verwaltungsleistung(srcInstVwl, cls1);
(509)       packageElementClass_Verwaltungsprodukt(srcInstVwp, cls2);
(510)     }
(511)   }
(512)
(513)
(514)   /*
(515)    * Beziehung vom Nachfrager zur Verwaltungsleistung
(516)    */
(517)   relation packageElementAssoziation_Nachfrager_Verwaltungsleistung {
(518)
(519)     checkonly domain prm srcPack :uml::Package {
(520)       packagedElement = srcInst1 : uml::InstanceSpecification {
(521)         classifier = clsf1 : uml::Class {
(522)           name = 'VwAntragsteller'
(523)         }
(524)       },
(525)
(526)       packagedElement = srcInst2 : uml::InstanceSpecification {
(527)         classifier = clsf2 : uml::Class {
(528)           name = 'VwLeistung'
(529)         }
(530)       }
(531)
(532)     };
(533)
(534)     enforce domain target pack : uml::Package {
(535)       packagedElement = ass : uml::Association {
(536)         name = 'erkenntMoeglichkeitNotwendigkeitVon',
(537)         ownedEnd = p1 : uml::Property {
(538)           type = cls1 : uml::Class { },
(539)           name = srcInst1.name,
(540)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(541)             value = 1
(542)           },
(543)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(544)             value = 1
(545)           }
(546)
(547)         },
(548)         ownedEnd = p2 : uml::Property {
(549)           type = cls2 : uml::Class { },
(550)           name = srcInst2.name,
(551)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(552)             value = 1
```

```
(553)           },
(554)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(555)             value = 1
(556)           }
(557)         }
(558)       }
(559)     };
(560)
(561)     when {
(562)       not(hasRefinement(srcInst1));
(563)       not(hasRefinement(srcInst2));
(564)       packageElementClass_Nachfrager(srcInst1, cls1);
(565)       packageElementClass_Verwaltungsleistung(srcInst2, cls2);
(566)     }
(567)   }
(568)
(569)   /*
(570)    * Beziehung vom Nachfrager zur Lebenslage
(571)    */
(572)   relation packageElementAssoziation_Nachfrager_Lebenslage {
(573)
(574)     checkonly domain prm srcPack :uml::Package {
(575)       packagedElement = srcInst1 : uml::InstanceSpecification {
(576)         classifier = clsf1 : uml::Class {
(577)           name = 'VwAntragsteller'
(578)         }
(579)       },
(580)
(581)       packagedElement = srcInst2 : uml::InstanceSpecification {
(582)         classifier = clsf2 : uml::Class {
(583)           name = 'VwLebenslage'
(584)         }
(585)       }
(586)
(587)     };
(588)
(589)     enforce domain target pack : uml::Package {
(590)       packagedElement = ass : uml::Association {
(591)         name = 'hat',
(592)         ownedEnd = p1 : uml::Property {
(593)           type = cls1 : uml::Class { },
(594)           name = srcInst1.name,
(595)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(596)             value = 1
(597)           },
(598)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(599)             value = 1
(600)           }
(601)
(602)         },
(603)         ownedEnd = p2 : uml::Property {
(604)           type = cls2 : uml::Class { },
(605)           name = srcInst2.name,
(606)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(607)             value = 1
(608)           },
(609)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(610)             value = 1
(611)           }
(612)         }
(613)       }
(614)     };
(615)
(616)     when {
(617)       not(hasRefinement(srcInst1));
(618)       not(hasRefinement(srcInst2));
(619)       packageElementClass_Nachfrager(srcInst1, cls1);
(620)       packageElementClass_Lebenslage(srcInst2, cls2);
(621)     }
(622)   }
(623)
(624)   /*
(625)    * Beziehung vom Nachfrager zur Lebenslage
(626)    */
(627)   relation packageElementDependency_Lebenslage_HinweisInformation {
```

```
(628)
(629)     checkonly domain prm srcPack :uml::Package {
(630)       packagedElement = srcInst1 : uml::InstanceSpecification {
(631)         classifier = clsf1 : uml::Class {
(632)           name = 'VwLebenslage'
(633)         }
(634)       },
(635)
(636)       packagedElement = srcInst2 : uml::InstanceSpecification {
(637)         classifier = clsf2 : uml::Class {
(638)           name = 'VwHinweisInformation'
(639)         }
(640)       }
(641)
(642)     };
(643)
(644)     enforce domain target pack : uml::Package {
(645)       packagedElement = cls1 : uml::Class {},
(646)       packagedElement = cls2 : uml::Class {},
(647)       packagedElement = dep : uml::Dependency {
(648)         supplier = cls2.oclAsType(OrderedSet(uml::Class)),
(649)         client = cls1.oclAsType(OrderedSet(uml::Class))
(650)       }
(651)
(652)     };
(653)
(654)     when {
(655)       not(hasRefinement(srcInst1));
(656)       not(hasRefinement(srcInst2));
(657)       packageElementClass_Lebenslage(srcInst1, cls1);
(658)       packageElementClass_HinweisInformation(srcInst2, cls2);
(659)     }
(660)   }
(661)
(662)   /*
(663)    * Beziehung vom Hinweis/Information zur Verwaltungsleistung
(664)    */
(665)   relation packageElementAssoziation_HinweisInformation_Verwaltungsleistung
      {
(666)
(667)     checkonly domain prm srcPack :uml::Package {
(668)       packagedElement = srcInst1 : uml::InstanceSpecification {
(669)         classifier = clsf1 : uml::Class {
(670)           name = 'VwHinweisInformation'
(671)         }
(672)       },
(673)
(674)       packagedElement = srcInst2 : uml::InstanceSpecification {
(675)         classifier = clsf2 : uml::Class {
(676)           name = 'VwLeistung'
(677)         }
(678)       }
(679)
(680)     };
(681)
(682)     enforce domain target pack : uml::Package {
(683)       packagedElement = ass : uml::Association {
(684)         name = 'beziehtSichAuf',
(685)         ownedEnd = p1 : uml::Property {
(686)           type = cls1 : uml::Class { },
(687)           name = srcInst1.name,
(688)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(689)             value = 1
(690)           },
(691)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(692)             value = 1
(693)           }
(694)
(695)         },
(696)         ownedEnd = p2 : uml::Property {
(697)           type = cls2 : uml::Class { },
(698)           name = srcInst2.name,
(699)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(700)             value = 1
(701)           },
```

```
(702)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(703)             value = 1
(704)           }
(705)         }
(706)       }
(707)     };
(708)
(709)     when {
(710)       not(hasRefinement(srcInst1));
(711)       not(hasRefinement(srcInst2));
(712)       packageElementClass_HinweisInformation(srcInst1, cls1);
(713)       packageElementClass_Verwaltungsleistung(srcInst2, cls2);
(714)     }
(715)   }
(716)
(717)   /*
(718)    * Beziehung vom Hinweis/Information zum Verwaltungsprodukt
(719)    */
(720)   relation packageElementAssoziation_HinweisInformation_Verwaltungsprodukt {
(721)
(722)     checkonly domain prm srcPack :uml::Package {
(723)       packagedElement = srcInst1 : uml::InstanceSpecification {
(724)         classifier = clsf1 : uml::Class {
(725)           name = 'VwHinweisInformation'
(726)         }
(727)       },
(728)
(729)       packagedElement = srcInst2 : uml::InstanceSpecification {
(730)         classifier = clsf2 : uml::Class {
(731)           name = 'VwRechtsfolge'
(732)         }
(733)       }
(734)
(735)     };
(736)
(737)     enforce domain target pack : uml::Package {
(738)       packagedElement = ass : uml::Association {
(739)         name = 'verweistAuf',
(740)         ownedEnd = p1 : uml::Property {
(741)           type = cls1 : uml::Class { },
(742)           name = srcInst1.name,
(743)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(744)             value = 1
(745)           },
(746)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(747)             value = 1
(748)           }
(749)
(750)         },
(751)         ownedEnd = p2 : uml::Property {
(752)           type = cls2 : uml::Class { },
(753)           name = srcInst2.name,
(754)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(755)             value = 1
(756)           },
(757)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(758)             value = 1
(759)           }
(760)         }
(761)       }
(762)     };
(763)
(764)     when {
(765)       not(hasRefinement(srcInst1));
(766)       not(hasRefinement(srcInst2));
(767)       packageElementClass_HinweisInformation(srcInst1, cls1);
(768)       packageElementClass_Verwaltungsprodukt(srcInst2, cls2);
(769)     }
(770)   }
(771)
(772)   /*
(773)    * Beziehung vom Hinweis/Information zum Benutzerprofil
(774)    */
(775)   relation packageElementAssoziation_HinweisInformation_Benutzerprofil {
(776)
```

```
(777)     checkonly domain prm srcPack :uml::Package {
(778)       packagedElement = srcInst1 : uml::InstanceSpecification {
(779)         classifier = clsf1 : uml::Class {
(780)           name = 'VwHinweisInformation'
(781)         }
(782)       }
(783)     };
(784)
(785)     enforce domain target pack : uml::Package {
(786)       packagedElement = ass : uml::Association {
(787)         name = '',
(788)         ownedEnd = p1 : uml::Property {
(789)           type = cls1 : uml::Class { },
(790)           name = srcInst1.name,
(791)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(792)             value = 1
(793)           },
(794)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(795)             value = 1
(796)           }
(797)
(798)         },
(799)         ownedEnd = p2 : uml::Property {
(800)           type = cls2 : uml::Class {
(801)             name='BenutzerProfil'
(802)           },
(803)           name = cls2.name,
(804)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(805)             value = 1
(806)           },
(807)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(808)             value = 1
(809)           }
(810)         }
(811)       }
(812)     };
(813)
(814)     when {
(815)       not(hasRefinement(srcInst1));
(816)       packageElementClass_HinweisInformation(srcInst1, cls1);
(817)     }
(818)   }
(819)
(820)
(821)
      /*************************************************************************
(822)    *
(823)    * Anwendungsfallmodell
(824)    * Anwendungsfälle
(825)    *
(826)
      *************************************************************************/
(827)
(828)   /*
(829)    * Verwaltungsleistung bewusst machen als Element ausgehend von der
      Verwaltungshandlung "VwBewusstwerdung"
(830)    */
(831)   relation packageElement_VerwaltungsleistungBewusstMachen {
(832)
(833)
(834)     checkonly domain prm srcPack :uml::Package {
(835)       packagedElement = srcInst1 : uml::InstanceSpecification {
(836)         classifier = clsf1 : uml::Class {
(837)           name = 'VwBewusstwerdung'
(838)         }
(839)       }
(840)     };
(841)
(842)     enforce domain target  pack : uml::Package {
(843)       packagedElement = uc : uml::UseCase { }
(844)
(845)     };
(846)
(847)     when {
(848)       not(hasRefinement(srcInst1));
```

```
(849)     }
(850)
(851)     where {
(852)       packageElementUseCase_VerwaltungsleistungBewusstMachen(srcInst1, uc);
(853)     }
(854)   }
(855)
(856)   /*
(857)    * "Verwaltungsleistung bewusst machen" als Element (Alternative)
      ausgehend
(858)    * von Verwaltungshandlungsgegenstand "VwLeistung"
(859)    */
(860)   relation packageElementAlt_VerwaltungsleistungBewusstMachen {
(861)
(862)     checkonly domain prm srcPack :uml::Package {
(863)       packagedElement = srcInst1 : uml::InstanceSpecification {
(864)         classifier = clsf1 : uml::Class {
(865)           name = 'VwLeistung'
(866)         }
(867)       }
(868)     };
(869)
(870)     enforce domain target  pack : uml::Package {
(871)       packagedElement = uc : uml::UseCase { }
(872)
(873)     };
(874)
(875)     when {
(876)       not(hasRefinement(srcInst1));
(877)       not(existsVerwaltungshandlung('VwBewusstwerdung', srcPack));
(878)     }
(879)
(880)     where {
(881)       packageElementUseCaseAlt_VerwaltungsleistungBewusstMachen(srcInst1,
      uc);
(882)     }
(883)   }
(884)
(885)   /*
(886)    * UseCase "Verwaltungsleistung bewusst machen" ausgehend von der
      Verwaltungshandlung "VwBewusstwerdung"
(887)    */
(888)   relation packageElementUseCase_VerwaltungsleistungBewusstMachen {
(889)     nvbm, q, a, z, vb, nb, ab : String;
(890)
(891)     checkonly domain prm srcInst1 : uml::InstanceSpecification {
(892)       name = nvbm,
(893)       classifier = clsf : uml::Class {
(894)         name = 'VwBewusstwerdung'
(895)       }
(896)     };
(897)
(898)     enforce domain target uc : uml::UseCase{
(899)       name = nvbm,
(900)       -- Dokumentation
(901)       eAnnotations = doc : ecore::EAnnotation {
(902)         source = 'http://www.topcased.org/documentation',
(903)         eModelElement = uc,
(904)         details = dtls : ecore::EStringToStringMapEntry {
(905)           value = '1. Ziel: ' + z + '; 2. Vorbedingung: ' + vb + '; 3.
      Nachbedingung: ' + nb + '4. Ablauf: ' + ab + '; 5. Quelle: ' + q,
(906)           _key = 'documentation'
(907)         }
(908)       }
(909)     };
(910)
(911)     where {
(912)       q = getQuelle(srcInst1);
(913)       a = getArbeitsnotiz(srcInst1);
(914)       ab = getBeschreibung(srcInst1);
(915)       z = getAnwendungsfallZielValue(srcInst1);
(916)       vb = getAnwendungsfallVorbedingungValue(srcInst1);
(917)       nb = getAnwendungsfallNachbedingungValue(srcInst1);
(918)     }
(919)   }
```

```
(920)
(921)   /*
(922)    * UseCase "Verwaltungsleistung bewusst machen" (Alternative) ausgehend
(923)    * von Verwaltungshandlungsgegenstand "VwLeistung"
(924)    */
(925)
(926)   relation packageElementUseCaseAlt_VerwaltungsleistungBewusstMachen {
(927)     nvl, nvp, nvhi : String;
(928)     b, a, q : String;
(929)     vbp, nbp, zlp, abp : String;
(930)     vb, nb, zl, ab : String;
(931)     elem : OrderedSet(uml::Element);
(932)     inst : OrderedSet(uml::InstanceSpecification);
(933)
(934)     checkonly domain prm srcInst1 : uml::InstanceSpecification {
(935)       name = nvl,
(936)       classifier = clsf : uml::Class {
(937)         name = 'VwLeistung'
(938)       }
(939)     };
(940)
(941)     enforce domain target uc : uml::UseCase{
(942)       name = nvl + ' bewusst machen',
(943)       -- Dokumentation
(944)       eAnnotations = doc : ecore::EAnnotation {
(945)         source = 'http://www.topcased.org/documentation',
(946)         eModelElement = uc,
(947)         details = dtls : ecore::EStringToStringMapEntry {
(948)           value = '1. Ziel: ' + zl + '; 2. Vorbedingung: ' + vb + '; 3.
      Nachbedingung: ' + nb + '4. Ablauf: ' + ab + '; 5. Quelle: ' + q,
(949)           _key = 'documentation'
(950)         }
(951)       }
(952)     };
(953)
(954)     where {
(955)       -- Quelle, Arbeitsnotiz und Beschreibung aus der Instanz ermitteln
(956)       q = getQuelle(srcInst1);
(957)       a = getArbeitsnotiz(srcInst1);
(958)       b = getBeschreibung(srcInst1);
(959)
(960)       -- Inforrmationen im PRM auswerten (hier prototypischer Zugriff
(961)       -- auf alle Insatnzen von HinweisInformtaion und Rechtsfolge
(962)       -- ohne die Beziehungen zwischen den Elementen zu beachten)
(963)       elem = srcInst1.allOwningPackages()->first().allOwnedElements();
(964)       inst = elem->select(e | e.oclIsTypeOf(uml::InstanceSpecification));
(965)       nvhi = concatNames(inst->select(i | i.classifier->exists(c |
      c.name='VwHinweisInformation')));
(966)       nvp = concatNames(inst->select(i | i.classifier->exists(c |
      c.name='VwRechtsfolge')));
(967)
(968)       -- Ziel
(969)       zlp = getAnwendungsfallZiel('Verwaltungsleistung bewusst machen');
(970)       zl = zlp.replace('einer Verwaltungsleistung', 'von ' + nvp);
(971)       -- Vorbedingung
(972)       vbp = getAnwendungsfallVorbedingung('Verwaltungsleistung bewusst
      machen');
(973)       vb = vbp;
(974)       -- Nachbedingung
(975)       nbp = getAnwendungsfallNachbedingung('Verwaltungsleistung bewusst
      machen');
(976)       nb = nbp.replace('Verwaltungsleistungen wurden', 'Verwaltungsleistung
      ' + nvl + ' wurde(n)').replace('Hinweise', 'Hinweise auf ' + nvhi);
(977)       -- Ablauf
(978)       abp = getAnwendungsfallAblauf('Verwaltungsleistung bewusst machen');
(979)       ab = abp.replace('aus den Hinweisen', 'aus ' +
      nvhi).replace('Verwaltungsprodukte und', 'Verwaltungsprodukte (' + nvp + ')
      und');
(980)     }
(981)   }
(982)
(983)
(984)
      /*********************************************************************
(985)    *
```

```
(986)    * Anwendungsfallmodell
(987)    * Akteure
(988)    *
(989)
      ***************************************************************************/
(990)
(991)   /*
(992)    * Nachfrager (Akteur) als Element
(993)    */
(994)   relation packageElement_NachfragerAkteur {
(995)
(996)     checkonly domain prm srcPack :uml::Package {
(997)       packagedElement = srcInst : uml::InstanceSpecification {
(998)         classifier = clsf : uml::Class {
(999)           name = 'VwAntragsteller'
(1000)          }
(1001)        }
(1002)     };
(1003)
(1004)     enforce domain target  pack : uml::Package {
(1005)       packagedElement = act : uml::Actor { }
(1006)     };
(1007)
(1008)     when {
(1009)       not(hasRefinement(srcInst));
(1010)     }
(1011)
(1012)     where {
(1013)       packageElementActor_Nachfrager(srcInst, act);
(1014)     }
(1015)   }
(1016)
(1017)   /*
(1018)    * Nachfrager als Akteur
(1019)    */
(1020)   relation packageElementActor_Nachfrager {
(1021)     n,b,a,q : String;
(1022)     checkonly domain prm srcInst : uml::InstanceSpecification {
(1023)       name = n,
(1024)       classifier = clsf : uml::Class {
(1025)         name = 'VwAntragsteller'
(1026)       }
(1027)     };
(1028)
(1029)     enforce domain target act : uml::Actor{
(1030)       name = n,
(1031)       -- Dokumentation
(1032)       eAnnotations = doc : ecore::EAnnotation {
(1033)         source = 'http://www.topcased.org/documentation',
(1034)         eModelElement = act,
(1035)         details = dtls : ecore::EStringToStringMapEntry {
(1036)           value = '1. Beschreibung: ' + b + '; 2. Arbeitsnotiz: ' + a + ';
      3. Quelle: ' + q,
(1037)           _key = 'documentation'
(1038)         }
(1039)       }
(1040)     };
(1041)
(1042)     when {
(1043)       not(hasRefinement(srcInst));
(1044)     }
(1045)
(1046)     where {
(1047)       q = getQuelle(srcInst);
(1048)       a = getArbeitsnotiz(srcInst);
(1049)       b = getBeschreibung(srcInst);
(1050)     }
(1051)   }
(1052)
(1053)   /*
(1054)    * Bearbeiter Front Office als Element
(1055)    */
(1056)   relation packageElement_BearbeiterFrontOffice {
(1057)
(1058)     checkonly domain prm srcPack :uml::Package {
```

```
(1059)        packagedElement = srcInst : uml::InstanceSpecification {
(1060)          classifier = clsf : uml::Class {
(1061)            name = 'VwAnwenderFrontOffice'
(1062)          }
(1063)        }
(1064)      };
(1065)
(1066)      enforce domain target  pack : uml::Package {
(1067)        packagedElement = act : uml::Actor { }
(1068)      };
(1069)
(1070)      where {
(1071)        packageElementActor_BearbeiterFrontOffice(srcInst, act);
(1072)      }
(1073)    }
(1074)
(1075)    /*
(1076)     * Bearbeiter Front Office als Akteur
(1077)     */
(1078)    relation packageElementActor_BearbeiterFrontOffice {
(1079)      n,b,a,q : String;
(1080)      checkonly domain prm srcInst : uml::InstanceSpecification {
(1081)        name = n,
(1082)        classifier = clsf : uml::Class {
(1083)          name = 'VwAnwenderFrontOffice'
(1084)        }
(1085)      };
(1086)
(1087)      enforce domain target act : uml::Actor{
(1088)        name = n,
(1089)        -- Dokumentation
(1090)        eAnnotations = doc : ecore::EAnnotation {
(1091)          source = 'http://www.topcased.org/documentation',
(1092)          eModelElement = act,
(1093)          details = dtls : ecore::EStringToStringMapEntry {
(1094)            value = '1. Beschreibung: ' + b + '; 2. Arbeitsnotiz: ' + a + ';
      3. Quelle: ' + q,
(1095)            _key = 'documentation'
(1096)          }
(1097)        }
(1098)      };
(1099)
(1100)      when {
(1101)        not(hasRefinement(srcInst));
(1102)      }
(1103)
(1104)      where {
(1105)        q = getQuelle(srcInst);
(1106)        a = getArbeitsnotiz(srcInst);
(1107)        b = getBeschreibung(srcInst);
(1108)      }
(1109)    }
(1110)
(1111)    /*
(1112)     * Bearbeiter als Element
(1113)     */
(1114)    relation packageElement_Bearbeiter {
(1115)
(1116)      checkonly domain prm srcPack :uml::Package {
(1117)        packagedElement = srcInst : uml::InstanceSpecification {
(1118)          classifier = clsf : uml::Class {
(1119)            name = 'VwAnwender'
(1120)          }
(1121)        }
(1122)      };
(1123)
(1124)      enforce domain target  pack : uml::Package {
(1125)        packagedElement = act : uml::Actor { }
(1126)      };
(1127)
(1128)      when {
(1129)        not(hasRefinement(srcInst));
(1130)      }
(1131)
(1132)      where {
```

```
(1133)        packageElementActor_Bearbeiter(srcInst, act);
(1134)      }
(1135)    }
(1136)
(1137)    /*
(1138)     * Bearbeiter als Akteur
(1139)     */
(1140)    relation packageElementActor_Bearbeiter {
(1141)      n,b,a,q : String;
(1142)      checkonly domain prm srcInst : uml::InstanceSpecification {
(1143)        name = n,
(1144)        classifier = clsf : uml::Class {
(1145)          name = 'VwAnwender'
(1146)        }
(1147)      };
(1148)
(1149)      enforce domain target act : uml::Actor{
(1150)        name = n,
(1151)        -- Dokumentation
(1152)        eAnnotations = doc : ecore::EAnnotation {
(1153)          source = 'http://www.topcased.org/documentation',
(1154)          eModelElement = act,
(1155)          details = dtls : ecore::EStringToStringMapEntry {
(1156)            value = '1. Beschreibung: ' + b + '; 2. Arbeitsnotiz: ' + a + '; 3. Quelle: ' + q,
(1157)            _key = 'documentation'
(1158)          }
(1159)        }
(1160)      };
(1161)
(1162)      when {
(1163)        not(hasRefinement(srcInst));
(1164)      }
(1165)
(1166)      where {
(1167)        q = getQuelle(srcInst);
(1168)        a = getArbeitsnotiz(srcInst);
(1169)        b = getBeschreibung(srcInst);
(1170)      }
(1171)    }
(1172)
(1173)    /*************************************************************************
(1174)     *
(1175)     * Anwendungsfallmodell
(1176)     * Kommunikationsbeziehungen
(1177)     *
(1178)     *************************************************************************/
(1179)
(1180)    /*
(1181)     * Beziehung vom Nachfrager zum Anwendungsfall Verwaltungsleistung bewusst machen
(1182)     */
(1183)    relation packageElementAssoziation_Nachfrager_VerwaltungsleistungBewusstMachen {
(1184)
(1185)      checkonly domain prm srcPack :uml::Package {
(1186)        packagedElement = srcInst1 : uml::InstanceSpecification {
(1187)          classifier = clsf1 : uml::Class {
(1188)            name = 'VwAntragsteller'
(1189)          }
(1190)        },
(1191)
(1192)        packagedElement = srcInst2 : uml::InstanceSpecification {
(1193)          classifier = clsf2 : uml::Class {
(1194)            name = 'VwBewusstwerdung'
(1195)          }
(1196)        }
(1197)      };
(1198)
(1199)      enforce domain target pack : uml::Package {
(1200)        packagedElement = ass : uml::Association {
(1201)          ownedEnd = p1 : uml::Property {
(1202)            type = act : uml::Actor { },
```

```
(1203)            name = srcInst1.name,
(1204)            upperValue = uval1 : uml::LiteralUnlimitedNatural {
(1205)              value = 1
(1206)            },
(1207)            lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(1208)              value = 1
(1209)            }
(1210)
(1211)          },
(1212)          ownedEnd = p2 : uml::Property {
(1213)            type = uc : uml::UseCase { },
(1214)            name = srcInst2.name,
(1215)            upperValue = uval2 : uml::LiteralUnlimitedNatural {
(1216)              value = 1
(1217)            },
(1218)            lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(1219)              value = 1
(1220)            }
(1221)          },
(1222)          navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(1223)        }
(1224)      };
(1225)
(1226)      when {
(1227)        not(hasRefinement(srcInst1));
(1228)        not(hasRefinement(srcInst2));
(1229)        packageElementActor_Nachfrager(srcInst1, act);
(1230)        packageElementUseCase_VerwaltungsleistungBewusstMachen(srcInst2, uc);
(1231)      }
(1232)    }
(1233)
(1234)    /*
(1235)     * Beziehung vom Bearbeiter Front Office zum Anwendungsfall
      Verwaltungsleistung bewusst machen
(1236)     */
(1237)    relation
      packageElementAssoziation_BearbeiterFrontOffice_VerwaltungsleistungBewusstMa
      chen {
(1238)
(1239)      checkonly domain prm srcPack :uml::Package {
(1240)        packagedElement = srcInst1 : uml::InstanceSpecification {
(1241)          classifier = clsf1 : uml::Class {
(1242)            name = 'VwAnwenderFrontOffice'
(1243)          }
(1244)        },
(1245)
(1246)        packagedElement = srcInst2 : uml::InstanceSpecification {
(1247)          classifier = clsf2 : uml::Class {
(1248)            name = 'VwBewusstwerdung'
(1249)          }
(1250)        }
(1251)      };
(1252)
(1253)      enforce domain target pack : uml::Package {
(1254)        packagedElement = ass : uml::Association {
(1255)          ownedEnd = p1 : uml::Property {
(1256)            type = act : uml::Actor { },
(1257)            name = srcInst1.name,
(1258)            upperValue = uval1 : uml::LiteralUnlimitedNatural {
(1259)              value = 1
(1260)            },
(1261)            lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(1262)              value = 1
(1263)            }
(1264)
(1265)          },
(1266)          ownedEnd = p2 : uml::Property {
(1267)            type = uc : uml::UseCase { },
(1268)            name = srcInst2.name,
(1269)            upperValue = uval2 : uml::LiteralUnlimitedNatural {
(1270)              value = 1
(1271)            },
(1272)            lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(1273)              value = 1
(1274)            }
```

```
(1275)          },
(1276)          navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(1277)        }
(1278)      };
(1279)
(1280)      when {
(1281)        not(hasRefinement(srcInst1));
(1282)        not(hasRefinement(srcInst2));
(1283)        packageElementActor_BearbeiterFrontOffice(srcInst1, act);
(1284)        packageElementUseCase_VerwaltungsleistungBewusstMachen(srcInst2, uc);
(1285)      }
(1286)    }
(1287)
(1288)    /*
(1289)     * Beziehung vom Bearbeiter Front Office zum Anwendungsfall
      Verwaltungsleistung bewusst machen
(1290)     */
(1291)    relation
      packageElementAssoziation_Bearbeiter_VerwaltungsleistungBewusstMachen {
(1292)
(1293)      checkonly domain prm srcPack :uml::Package {
(1294)        packagedElement = srcInst1 : uml::InstanceSpecification {
(1295)          classifier = clsf1 : uml::Class {
(1296)            name = 'VwAnwender'
(1297)          }
(1298)        },
(1299)
(1300)        packagedElement = srcInst2 : uml::InstanceSpecification {
(1301)          classifier = clsf2 : uml::Class {
(1302)            name = 'VwBewusstwerdung'
(1303)          }
(1304)        }
(1305)      };
(1306)
(1307)      enforce domain target pack : uml::Package {
(1308)        packagedElement = ass : uml::Association {
(1309)          ownedEnd = p1 : uml::Property {
(1310)            type = act : uml::Actor { },
(1311)            name = srcInst1.name,
(1312)            upperValue = uval1 : uml::LiteralUnlimitedNatural {
(1313)              value = 1
(1314)            },
(1315)            lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(1316)              value = 1
(1317)            }
(1318)
(1319)          },
(1320)          ownedEnd = p2 : uml::Property {
(1321)            type = uc : uml::UseCase { },
(1322)            name = srcInst2.name,
(1323)            upperValue = uval2 : uml::LiteralUnlimitedNatural {
(1324)              value = 1
(1325)            },
(1326)            lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(1327)              value = 1
(1328)            }
(1329)          },
(1330)          navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(1331)        }
(1332)      };
(1333)
(1334)      when {
(1335)        not(hasRefinement(srcInst1));
(1336)        not(hasRefinement(srcInst2));
(1337)        packageElementActor_Bearbeiter(srcInst1, act);
(1338)        packageElementUseCase_VerwaltungsleistungBewusstMachen(srcInst2, uc);
(1339)      }
(1340)    }
(1341)
(1342)    /*
(1343)     * Alternative Beziehung vom Nachfrager zum Anwendungsfall
      Verwaltungsleistung bewusst machen
(1344)     */
(1345)    relation
      packageElementAssoziationAlt_Nachfrager_VerwaltungsleistungBewusstMachen {
```

```
(1346)
(1347)      checkonly domain prm srcPack :uml::Package {
(1348)        packagedElement = srcInst1 : uml::InstanceSpecification {
(1349)          classifier = clsf1 : uml::Class {
(1350)            name = 'VwAntragsteller'
(1351)          }
(1352)        },
(1353)
(1354)        packagedElement = srcInst2 : uml::InstanceSpecification {
(1355)          classifier = clsf2 : uml::Class {
(1356)            name = 'VwLeistung'
(1357)          }
(1358)        }
(1359)      };
(1360)
(1361)      enforce domain target pack : uml::Package {
(1362)        packagedElement = ass : uml::Association {
(1363)          ownedEnd = p1 : uml::Property {
(1364)            type = act : uml::Actor { },
(1365)            name = srcInst1.name,
(1366)            upperValue = uval1 : uml::LiteralUnlimitedNatural {
(1367)              value = 1
(1368)            },
(1369)            lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(1370)              value = 1
(1371)            }
(1372)
(1373)          },
(1374)          ownedEnd = p2 : uml::Property {
(1375)            type = uc : uml::UseCase { },
(1376)            name = srcInst2.name,
(1377)            upperValue = uval2 : uml::LiteralUnlimitedNatural {
(1378)              value = 1
(1379)            },
(1380)            lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(1381)              value = 1
(1382)            }
(1383)          },
(1384)          navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(1385)        }
(1386)      };
(1387)
(1388)      when {
(1389)        not(hasRefinement(srcInst1));
(1390)        not(hasRefinement(srcInst2));
(1391)        not(existsVerwaltungshandlung('VwBewusstwerdung', srcPack));
(1392)        packageElementActor_Nachfrager(srcInst1, act);
(1393)        packageElementUseCaseAlt_VerwaltungsleistungBewusstMachen(srcInst2,
      uc);
(1394)      }
(1395)    }
(1396)
(1397)    /*
(1398)     * Alternative Beziehung vom Bearbeiter Front Office zum Anwendungsfall
      Verwaltungsleistung bewusst machen
(1399)     */
(1400)    relation
      packageElementAssoziationAlt_BearbeiterFrontOffice_VerwaltungsleistungBewuss
      tMachen {
(1401)
(1402)      checkonly domain prm srcPack :uml::Package {
(1403)        packagedElement = srcInst1 : uml::InstanceSpecification {
(1404)          classifier = clsf1 : uml::Class {
(1405)            name = 'VwAnwenderFrontOffice'
(1406)          }
(1407)        },
(1408)
(1409)        packagedElement = srcInst2 : uml::InstanceSpecification {
(1410)          classifier = clsf2 : uml::Class {
(1411)            name = 'VwLeistung'
(1412)          }
(1413)        }
(1414)      };
(1415)
(1416)      enforce domain target pack : uml::Package {
```

```
(1417)        packagedElement = ass : uml::Association {
(1418)          ownedEnd = p1 : uml::Property {
(1419)            type = act : uml::Actor { },
(1420)            name = srcInst1.name,
(1421)            upperValue = uval1 : uml::LiteralUnlimitedNatural {
(1422)              value = 1
(1423)            },
(1424)            lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(1425)              value = 1
(1426)            }
(1427)
(1428)          },
(1429)          ownedEnd = p2 : uml::Property {
(1430)            type = uc : uml::UseCase { },
(1431)            name = srcInst2.name,
(1432)            upperValue = uval2 : uml::LiteralUnlimitedNatural {
(1433)              value = 1
(1434)            },
(1435)            lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(1436)              value = 1
(1437)            }
(1438)          },
(1439)          navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(1440)        }
(1441)      };
(1442)
(1443)      when {
(1444)        not(hasRefinement(srcInst1));
(1445)        not(hasRefinement(srcInst2));
(1446)        not(existsVerwaltungshandlung('VwBewusstwerdung', srcPack));
(1447)        packageElementActor_BearbeiterFrontOffice(srcInst1, act);
(1448)        packageElementUseCaseAlt_VerwaltungsleistungBewusstMachen(srcInst2,
      uc);
(1449)      }
(1450)    }
(1451)
(1452)    /*
(1453)     * Alternative Beziehung vom Bearbeiter Front Office zum Anwendungsfall
      Verwaltungsleistung bewusst machen
(1454)     */
(1455)    relation
      packageElementAssoziationAlt_Bearbeiter_VerwaltungsleistungBewusstMachen {
(1456)
(1457)      checkonly domain prm srcPack :uml::Package {
(1458)        packagedElement = srcInst1 : uml::InstanceSpecification {
(1459)          classifier = clsf1 : uml::Class {
(1460)            name = 'VwAnwender'
(1461)          }
(1462)        },
(1463)
(1464)        packagedElement = srcInst2 : uml::InstanceSpecification {
(1465)          classifier = clsf2 : uml::Class {
(1466)            name = 'VwLeistung'
(1467)          }
(1468)        }
(1469)      };
(1470)
(1471)      enforce domain target pack : uml::Package {
(1472)        packagedElement = ass : uml::Association {
(1473)          ownedEnd = p1 : uml::Property {
(1474)            type = act : uml::Actor { },
(1475)            name = srcInst1.name,
(1476)            upperValue = uval1 : uml::LiteralUnlimitedNatural {
(1477)              value = 1
(1478)            },
(1479)            lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(1480)              value = 1
(1481)            }
(1482)
(1483)          },
(1484)          ownedEnd = p2 : uml::Property {
(1485)            type = uc : uml::UseCase { },
(1486)            name = srcInst2.name,
(1487)            upperValue = uval2 : uml::LiteralUnlimitedNatural {
(1488)              value = 1
```

```
(1489)            },
(1490)            lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(1491)              value = 1
(1492)            }
(1493)          },
(1494)          navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(1495)        }
(1496)      };
(1497)
(1498)      when {
(1499)        not(hasRefinement(srcInst1));
(1500)        not(hasRefinement(srcInst2));
(1501)        packageElementActor_Bearbeiter(srcInst1, act);
(1502)        not(existsVerwaltungshandlung('VwBewusstwerdung', srcPack));
(1503)        packageElementUseCaseAlt_VerwaltungsleistungBewusstMachen(srcInst2,
      uc);
(1504)      }
(1505)    }
(1506)
(1507)
(1508)
(1509)
(1510)
(1511)
      /*************************************************************************
(1512)     *
(1513)     * Hilfsfunktionen
(1514)     *
(1515)
      *************************************************************************/
(1516)
(1517)    /*
(1518)     * Liefert falsch, wenn es keine ausgehenden Dependencies mit dem
      Stereotyp
(1519)     * <<refine>> gibt. Andernfalls liefert es wahr.
(1520)     */
(1521)    query hasRefinement(inst : uml::InstanceSpecification) : Boolean {
(1522)      if (inst.clientDependency->oclIsUndefined() or inst.clientDependency-
      >isEmpty()) then
(1523)        false
(1524)      else
(1525)        true
(1526)        -- inst.clientDependency->asSequence()->exists(d |
      d.isStereotypeApplied(getStereotype('refine')))
(1527)      endif
(1528)    }
(1529)
(1530)
(1531)    /*
(1532)     * Liefert den Wert des Attributes "Beschreibung" der übergebenen Instanz
      zurück
(1533)     */
(1534)    query getBeschreibung(inst : uml::InstanceSpecification) :  String {
(1535)      getSlotValue(inst, 'Beschreibung')
(1536)    }
(1537)
(1538)    /*
(1539)     * Liefert den Wert des Attributes "Quelle" der übergebenen Instanz
      zurück
(1540)     */
(1541)    query getQuelle(inst : uml::InstanceSpecification) :  String {
(1542)      getSlotValue(inst, 'Quelle')
(1543)    }
(1544)
(1545)    /*
(1546)     * Liefert den Wert des Attributes "Arbeitsnotiz" der übergebenen Instanz
      zurück
(1547)     */
(1548)    query getArbeitsnotiz(inst : uml::InstanceSpecification) :  String {
(1549)      getSlotValue(inst, 'Arbeitsnotiz')
(1550)    }
(1551)
(1552)    /*
(1553)     * Liefert den Wert des Attributes "HandlungVorher" der übergebenen
      Instanz zurück
```

```
(1554)     */
(1555)    query getAnwendungsfallVorbedingungValue(inst :
      uml::InstanceSpecification) :  String {
(1556)      getSlotValue(inst, 'HandlungVorher')
(1557)    }
(1558)
(1559)    /*
(1560)     * Liefert den Wert des Attributes "HandlungNacher" der übergebenen
      Instanz zurück
(1561)     */
(1562)    query getAnwendungsfallNachbedingungValue(inst :
      uml::InstanceSpecification) :  String {
(1563)      getSlotValue(inst, 'HandlungNacher')
(1564)    }
(1565)
(1566)    /*
(1567)     * Liefert den Wert des Attributes "HandlungsZiel" der übergebenen
      Instanz zurück
(1568)     */
(1569)    query getAnwendungsfallZielValue(inst : uml::InstanceSpecification) :
      String {
(1570)      getSlotValue(inst, 'HandlungsZiel')
(1571)    }
(1572)
(1573)    /*
(1574)     * Liefert den Default-Wert des Attributes "Profil" zum Stereotyp mit dem
      übergebenen Namen für einen Akteur
(1575)     */
(1576)    query getAkteurProfil(stname : String) :  String {
(1577)
      getAttributesDefaultValue(getOwnedAttribute(getStereotypeFromProfile(stname,
      'eGovRefAnf'), 'Profil'))
(1578)    }
(1579)
(1580)    /*
(1581)     * Liefert den Default-Wert des Attributes "Ziel" zum Stereotyp mit dem
      übergebenen Namen für einen Akteur
(1582)     */
(1583)    query getAkteurZiel(stname : String) :  String {
(1584)
      getAttributesDefaultValue(getOwnedAttribute(getStereotypeFromProfile(stname,
      'eGovRefAnf'), 'Ziel'))
(1585)    }
(1586)
(1587)    /*
(1588)     * Liefert den Default-Wert des Attributes "Ziel" zum Stereotyp mit dem
      übergebenen Namen für einen Anwendungsfall
(1589)     */
(1590)    query getAnwendungsfallZiel(stname : String) :  String {
(1591)
      getAttributesDefaultValue(getOwnedAttribute(getStereotypeFromProfile(stname,
      'eGovRefAnf'), 'Ziel'))
(1592)    }
(1593)
(1594)    /*
(1595)     * Liefert den Default-Wert des Attributes "Vorbedingung" zum Stereotyp
      mit dem übergebenen Namen für einen Anwendungsfall
(1596)     */
(1597)    query getAnwendungsfallVorbedingung(stname : String) :  String {
(1598)
      getAttributesDefaultValue(getOwnedAttribute(getStereotypeFromProfile(stname,
      'eGovRefAnf'), 'Vorbedingung'))
(1599)    }
(1600)
(1601)    /*
(1602)     * Liefert den Default-Wert des Attributes "Nachbedingung" zum Stereotyp
      mit dem übergebenen Namen für einen Anwendungsfall
(1603)     */
(1604)    query getAnwendungsfallNachbedingung(stname : String) :  String {
(1605)
      getAttributesDefaultValue(getOwnedAttribute(getStereotypeFromProfile(stname,
      'eGovRefAnf'), 'Nachbedingung'))
(1606)    }
(1607)
(1608)    /*
```

```
(1609)     * Liefert den Default-Wert des Attributes "Ablauf" zum Stereotyp mit dem
      übergebenen Namen für einen Anwendungsfall
(1610)     */
(1611)    query getAnwendungsfallAblauf(stname : String) :  String {
(1612)
      getAttributesDefaultValue(getOwnedAttribute(getStereotypeFromProfile(stname,
      'eGovRefAnf'), 'Ablauf'))
(1613)    }
(1614)
(1615)    /*
(1616)     * Liefert den Wert des Attributes mit dem übergebenen Namen. Wenn es
      kein Attribut mit
(1617)     * diesem Namen gibt oder der Wert nicht gesetzt ist, wird ein fester
      Text zurückgeliefert.
(1618)     */
(1619)    query getSlotValue(inst : uml::InstanceSpecification, feature : String) :
      String {
(1620)      if (inst.slot->exists(x : uml::Slot | x.definingFeature.name =
      feature)) then
(1621)       inst.slot->asSequence()->select(x : uml::Slot | x.definingFeature.name
      = feature)->first().value->first().stringValue()
(1622)      else
(1623)       '- keine -'
(1624)      endif
(1625)    }
(1626)
(1627)
(1628)    /*
(1629)     * Liefert den Stereotyp mit dem angegebenen Namen aus dem Profil mit dem
      angegebenen Namen
(1630)     */
(1631)    query getStereotypeFromProfile ( stName : String, prName : String ) :
      uml::Stereotype
(1632)    {
(1633)      Stereotype.allInstances()->select
(1634)        (x : uml::Stereotype | (x.name = stName and x.getProfile().name =
      prName))->asSequence()->first()
(1635)    }
(1636)
(1637)    /*
(1638)     * Liefert wahr, wenn die Instanz den Stereotyp "owlIndividual" hat.
      Andernfalls
(1639)     * liefert sie falsch.
(1640)     */
(1641)    query isOwlIndividual(inst: InstanceSpecification) : Boolean {
(1642)       if (inst.isStereotypeApplied(getStereotype('owlIndividual'))) then
(1643)         true
(1644)       else
(1645)         false
(1646)       endif
(1647)     }
(1648)
(1649)    /*
(1650)     * Liefert das Attribut mit dem übergebenen Namen der übergebenen Klasse
(1651)     */
(1652)    query getOwnedAttribute(cls : Class, an : String) : Property {
(1653)      cls.ownedAttribute->asSequence()->select(p : Property | p.name = an)-
      >first()
(1654)    }
(1655)
(1656)    /*
(1657)     * Liefert den Default-Wert des übergebenen Attributes
(1658)     */
(1659)    query getAttributesDefaultValue(p : Property) : String {
(1660)      if (p.oclIsUndefined()) then
(1661)        ''
(1662)      else if (p.defaultValue->isEmpty()) then
(1663)        ''
(1664)        else
(1665)          p.defaultValue->asSequence()->first().stringValue()
(1666)        endif
(1667)      endif
(1668)    }
(1669)
(1670)     /*
```

```
(1671)      * Liefert den ersten Stereotyp mit dem angegebenen Namen
(1672)      * siehe: Siegfried Nolte, QVT-Releation Language, Springer, 2009, pp.
      165
(1673)      */
(1674)    query getStereotype ( stName : String ) : uml::Stereotype {
(1675)      Stereotype.allInstances()->select
(1676)           (x : uml::Stereotype | x.name = stName)->asSequence()->first()
(1677)    }
(1678)
(1679)     /*
(1680)      * Liefert die Namen der InstanceSpecifications verbunden durch Komma
      und "und" zurück
(1681)      */
(1682)     query concatNames(instSet : OrderedSet(uml::InstanceSpecification)) :
      String {
(1683)      if instSet.isEmpty() then
(1684)           ''
(1685)         else if (instSet->size()=1) then
(1686)              instSet->first().name
(1687)            else if (instSet->size()=2) then
(1688)                instSet->first().name + ' und ' + concatNames(instSet-
      >excluding(instSet->first()))
(1689)              else
(1690)                instSet->first().name + ', ' + concatNames(instSet-
      >excluding(instSet->first()))
(1691)              endif
(1692)            endif
(1693)          endif
(1694)     }
(1695)
(1696)     /*
(1697)      * Prüft, ob eine Verwaltungshandlung mit dem übergebenen Namen im
      übergebenen Packages existiert
(1698)      */
(1699)     query existsVerwaltungshandlung(vhname : String, srcPack : Package) :
      Boolean {
(1700)       if (vhname.oclIsUndefined() or srcPack.oclIsUndefined()) then
(1701)         false
(1702)       else
(1703)         srcPack.packagedElement->select(elem|
      elem.oclIsTypeOf(uml::InstanceSpecification)).oclAsType(OrderedSet(uml::Inst
      anceSpecification))->exists(inst | inst.classifier->exists(c|
      c.name=vhname))
(1704)       endif
(1705)     }
(1706)  }
```

Listing 58 – Transformation „prm2cim_anf_anliegen_bewusstwerdung“ (prm2cim_anf_anliegen_bewusstwerdung.qvt)

```
(1)   -- Anwendungsfall "Leistung erbringen"
(2)   transformation prm2cim_anf_leistung_erbringung(prm : uml, target : uml) {
(3)
(4)     -- Schlüssel die die Identität des Elementes bestimmen
(5)     key uml::UseCase {name};
(6)
(7)     /*
(8)      * Relation zur Transformation des Modells in den Anwendungsfall
(9)      * "Verwaltungsleistung bewusst machen"
(10)     */
(11)    relation model {
(12)
(13)      checkonly domain prm srcPack :uml::Package{ };
(14)
(15)      enforce domain target packVerwaltungsleistung : uml::Package {  };
(16)
(17)      where {
(18)
(19)        -- Anwendungsfallmodell
(20)        -- Anwendungsfall
(21)        packageElement_LeistungErbringen(srcPack, packVerwaltungsleistung);
(22)
(23)        -- Akteure
(24)        packageElement_AusfuehrenderFrontOffice(srcPack,
      packVerwaltungsleistung);
```

```
(25)        packageElement_AusfuehrenderBackOffice(srcPack,
      packVerwaltungsleistung);
(26)        packageElement_Ausfuehrender(srcPack, packVerwaltungsleistung);
(27)
(28)        -- Beziehungen
(29)
      packageElementAssoziation_AusfuehrenderFrontOffice_LeistungErbringen(srcPack
      , packVerwaltungsleistung);
(30)
      packageElementAssoziation_AusfuehrenderBackOffice_LeistungErbringen(srcPack,
      packVerwaltungsleistung);
(31)        packageElementAssoziation_Ausfuehrender_LeistungErbringen(srcPack,
      packVerwaltungsleistung);
(32)
(33)        -- Nicht-funktionale Anforderungen
(34)        packageElement_RechtsfolgeMerkmalNF(srcPack, packVerwaltungsleistung);
(35)
(36)      }
(37)    }
(38)
(39)
      /*************************************************************************
(40)     *
(41)     * Anwendungsfallmodell
(42)     * Anwendungsfälle
(43)     *
(44)
      *************************************************************************/
(45)
(46)    /*
(47)     * Leistung erbringen als Element
(48)     * ausgehend vom Verwaltungshandlungsgegenstand "VwLeistung"
(49)     */
(50)    relation packageElement_LeistungErbringen{
(51)
(52)      checkonly domain prm srcPack :uml::Package {
(53)        packagedElement = srcInst1 : uml::InstanceSpecification {
(54)          classifier = clsf1 : uml::Class {
(55)            name = 'VwLeistung'
(56)          }
(57)        }
(58)      };
(59)
(60)      enforce domain target  pack : uml::Package {
(61)        packagedElement = uc : uml::UseCase { }
(62)
(63)      };
(64)
(65)      when {
(66)        not(hasRefinement(srcInst1));
(67)        -- Alternative Implementierung basierend auf einer Instanz von
(68)        -- VwLeistungserbringung ist hier nicht realisiert.
(69)        -- not(existsVerwaltungshandlung('VwLeistungserbringung', srcPack));
(70)      }
(71)
(72)      where {
(73)        packageElementUseCase_LeistungErbringen(srcInst1, uc);
(74)      }
(75)    }
(76)
(77)    /*
(78)     * UseCase "Leistung erbringen" ausgehend
(79)     * von Verwaltungshandlungsgegenstand "VwLeistung"
(80)     */
(81)
(82)    relation packageElementUseCase_LeistungErbringen {
(83)      nvl : String;
(84)
(85)      checkonly domain prm srcInst1 : uml::InstanceSpecification {
(86)        name = nvl,
(87)        classifier = clsf : uml::Class {
(88)          name = 'VwLeistung'
(89)        }
(90)      };
(91)
```

```
(92)      enforce domain target uc : uml::UseCase{
(93)        name = nvl + ' erbringen',
(94)        -- Dokumentation
(95)        eAnnotations = doc : ecore::EAnnotation {
(96)          source = 'http://www.topcased.org/documentation',
(97)          eModelElement = uc,
(98)          details = dtls : ecore::EStringToStringMapEntry {
(99)            value = 'Uebernahme der Beschreibung innerhalb der prototypischen
      Umsetzung diese Use Case nicht realisiert.',
(100)           _key = 'documentation'
(101)         }
(102)       }
(103)     };
(104)
(105)   }
(106)
(107)
      /*******************************************************************************
(108)    *
(109)    * Anwendungsfallmodell
(110)    * Akteure
(111)    *
(112)
      *******************************************************************************/
(113)
(114)   /*
(115)    * Ausfuehrender als Element
(116)    */
(117)   relation packageElement_Ausfuehrender {
(118)
(119)     checkonly domain prm srcPack :uml::Package {
(120)       packagedElement = srcInst : uml::InstanceSpecification {
(121)         classifier = clsf : uml::Class {
(122)           name = 'VwAusfuehrender'
(123)         }
(124)       }
(125)     };
(126)
(127)     enforce domain target  pack : uml::Package {
(128)       packagedElement = act : uml::Actor { }
(129)     };
(130)
(131)     when {
(132)       not(hasRefinement(srcInst));
(133)     }
(134)
(135)     where {
(136)       packageElementActor_Ausfuehrender(srcInst, act);
(137)     }
(138)   }
(139)
(140)   /*
(141)    * Ausfeuhrender im Front Office als Akteur
(142)    */
(143)   relation packageElementActor_Ausfuehrender {
(144)     n : String;
(145)     checkonly domain prm srcInst : uml::InstanceSpecification {
(146)       name = n,
(147)       classifier = clsf : uml::Class {
(148)         name = 'VwAusfuehrender'
(149)       }
(150)     };
(151)
(152)     enforce domain target act : uml::Actor{
(153)       name = n,
(154)       -- Dokumentation
(155)       eAnnotations = doc : ecore::EAnnotation {
(156)         source = 'http://www.topcased.org/documentation',
(157)         eModelElement = act,
(158)         details = dtls : ecore::EStringToStringMapEntry {
(159)           value = 'Uebernahme der Beschreibung innerhalb der prototypischen
      Umsetzung diese Actors nicht realisiert.',
(160)           _key = 'documentation'
(161)         }
(162)       }
```

```
(163)     };
(164)
(165)   }
(166)
(167)   /*
(168)    * Ausfuehrender im Front Office als Element
(169)    */
(170)   relation packageElement_AusfuehrenderFrontOffice {
(171)
(172)     checkonly domain prm srcPack :uml::Package {
(173)       packagedElement = srcInst : uml::InstanceSpecification {
(174)         classifier = clsf : uml::Class {
(175)           name = 'VwAusfuehrenderFrontOffice'
(176)         }
(177)       }
(178)     };
(179)
(180)     enforce domain target  pack : uml::Package {
(181)       packagedElement = act : uml::Actor { }
(182)     };
(183)
(184)     when {
(185)       not(hasRefinement(srcInst));
(186)     }
(187)
(188)     where {
(189)       packageElementActor_AusfuehrenderFrontOffice(srcInst, act);
(190)     }
(191)   }
(192)
(193)   /*
(194)    * Ausfeuhrender im Front Office als Akteur
(195)    */
(196)   relation packageElementActor_AusfuehrenderFrontOffice {
(197)     n : String;
(198)     checkonly domain prm srcInst : uml::InstanceSpecification {
(199)       name = n,
(200)       classifier = clsf : uml::Class {
(201)         name = 'VwAusfuehrenderFrontOffice'
(202)       }
(203)     };
(204)
(205)     enforce domain target act : uml::Actor{
(206)       name = n,
(207)       -- Dokumentation
(208)       eAnnotations = doc : ecore::EAnnotation {
(209)         source = 'http://www.topcased.org/documentation',
(210)         eModelElement = act,
(211)         details = dtls : ecore::EStringToStringMapEntry {
(212)           value = 'Uebernahme der Beschreibung innerhalb der prototypischen
      Umsetzung diese Actors nicht realisiert.',
(213)           _key = 'documentation'
(214)         }
(215)       }
(216)     };
(217)
(218)   }
(219)
(220)   /*
(221)    * Ausfuehrender im Back Office als Element
(222)    */
(223)   relation packageElement_AusfuehrenderBackOffice {
(224)
(225)     checkonly domain prm srcPack :uml::Package {
(226)       packagedElement = srcInst : uml::InstanceSpecification {
(227)         classifier = clsf : uml::Class {
(228)           name = 'VwAusfuehrenderBackOffice'
(229)         }
(230)       }
(231)     };
(232)
(233)     enforce domain target  pack : uml::Package {
(234)       packagedElement = act : uml::Actor { }
(235)     };
(236)
```

```
(237)     when {
(238)       not(hasRefinement(srcInst));
(239)     }
(240)
(241)     where {
(242)       packageElementActor_AusfuehrenderBackOffice(srcInst, act);
(243)     }
(244)   }
(245)
(246)   /*
(247)    * Ausfeuhrender im Back Office als Akteur
(248)    */
(249)   relation packageElementActor_AusfuehrenderBackOffice {
(250)     n : String;
(251)     checkonly domain prm srcInst : uml::InstanceSpecification {
(252)       name = n,
(253)       classifier = clsf : uml::Class {
(254)         name = 'VwAusfuehrenderBackOffice'
(255)       }
(256)     };
(257)
(258)     enforce domain target act : uml::Actor{
(259)       name = n,
(260)       -- Dokumentation
(261)       eAnnotations = doc : ecore::EAnnotation {
(262)         source = 'http://www.topcased.org/documentation',
(263)         eModelElement = act,
(264)         details = dtls : ecore::EStringToStringMapEntry {
(265)           value = 'Uebernahme der Beschreibung innerhalb der prototypischen
      Umsetzung diese Actors nicht realisiert.',
(266)           _key = 'documentation'
(267)         }
(268)       }
(269)     };
(270)
(271)   }
(272)
(273)
      /*************************************************************************
(274)    *
(275)    * Anwendungsfallmodell
(276)    * Kommunikationsbeziehungen
(277)    *
(278)
      *************************************************************************/
(279)
(280)   /*
(281)    * Beziehung vom Ausfuehrenden im Front Office zum Anwendungsfall Leistung
      erbringen
(282)    */
(283)   relation
      packageElementAssoziation_AusfuehrenderFrontOffice_LeistungErbringen {
(284)     checkonly domain prm srcPack :uml::Package {
(285)       packagedElement = srcInst1 : uml::InstanceSpecification {
(286)         classifier = clsf1 : uml::Class {
(287)           name = 'VwAusfuehrenderFrontOffice'
(288)         }
(289)       },
(290)
(291)       packagedElement = srcInst2 : uml::InstanceSpecification {
(292)         classifier = clsf2 : uml::Class {
(293)           name = 'VwLeistung'
(294)         }
(295)       }
(296)     };
(297)
(298)     enforce domain target pack : uml::Package {
(299)       packagedElement = ass : uml::Association {
(300)         ownedEnd = p1 : uml::Property {
(301)           type = act : uml::Actor { },
(302)           name = srcInst1.name,
(303)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(304)             value = 1
(305)           },
(306)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
```

```
(307)             value = 1
(308)           }
(309)
(310)         },
(311)         ownedEnd = p2 : uml::Property {
(312)           type = uc : uml::UseCase { },
(313)           name = srcInst2.name + ' erbringen',
(314)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(315)             value = 1
(316)           },
(317)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(318)             value = 1
(319)           }
(320)         },
(321)         navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(322)       }
(323)     };
(324)
(325)     when {
(326)       not(hasRefinement(srcInst1));
(327)       not(hasRefinement(srcInst2));
(328)
(329)       packageElementActor_AusfuehrenderFrontOffice(srcInst1, act);
(330)       packageElementUseCase_LeistungErbringen(srcInst2, uc);
(331)     }
(332)
(333)   }
(334)
(335)   /*
(336)    * Beziehung vom Ausfuehrenden im Back Office zum Anwendungsfall Leistung
      erbringen
(337)    */
(338)   relation
      packageElementAssoziation_AusfuehrenderBackOffice_LeistungErbringen {
(339)     checkonly domain prm srcPack :uml::Package {
(340)       packagedElement = srcInst1 : uml::InstanceSpecification {
(341)         classifier = clsf1 : uml::Class {
(342)           name = 'VwAusfuehrenderBackOffice'
(343)         }
(344)       },
(345)
(346)       packagedElement = srcInst2 : uml::InstanceSpecification {
(347)         classifier = clsf2 : uml::Class {
(348)           name = 'VwLeistung'
(349)         }
(350)       }
(351)     };
(352)
(353)     enforce domain target pack : uml::Package {
(354)       packagedElement = ass : uml::Association {
(355)         ownedEnd = p1 : uml::Property {
(356)           type = act : uml::Actor { },
(357)           name = srcInst1.name,
(358)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(359)             value = 1
(360)           },
(361)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(362)             value = 1
(363)           }
(364)
(365)         },
(366)         ownedEnd = p2 : uml::Property {
(367)           type = uc : uml::UseCase { },
(368)           name = srcInst2.name + ' erbringen',
(369)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(370)             value = 1
(371)           },
(372)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(373)             value = 1
(374)           }
(375)         },
(376)         navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(377)       }
(378)     };
(379)
```

```
(380)     when {
(381)       not(hasRefinement(srcInst1));
(382)       not(hasRefinement(srcInst2));
(383)
(384)       packageElementActor_AusfuehrenderBackOffice(srcInst1, act);
(385)       packageElementUseCase_LeistungErbringen(srcInst2, uc);
(386)     }
(387)
(388)   }
(389)
(390)   /*
(391)    * Beziehung vom Ausfuehrenden zum Anwendungsfall Leistung erbringen
(392)    */
(393)   relation packageElementAssoziation_Ausfuehrender_LeistungErbringen {
(394)     checkonly domain prm srcPack :uml::Package {
(395)       packagedElement = srcInst1 : uml::InstanceSpecification {
(396)         classifier = clsf1 : uml::Class {
(397)           name = 'VwAusfuehrender'
(398)         }
(399)       },
(400)
(401)       packagedElement = srcInst2 : uml::InstanceSpecification {
(402)         classifier = clsf2 : uml::Class {
(403)           name = 'VwLeistung'
(404)         }
(405)       }
(406)     };
(407)
(408)     enforce domain target pack : uml::Package {
(409)       packagedElement = ass : uml::Association {
(410)         ownedEnd = p1 : uml::Property {
(411)           type = act : uml::Actor { },
(412)           name = srcInst1.name,
(413)           upperValue = uval1 : uml::LiteralUnlimitedNatural {
(414)             value = 1
(415)           },
(416)           lowerValue = lval1 : uml::LiteralUnlimitedNatural {
(417)             value = 1
(418)           }
(419)
(420)         },
(421)         ownedEnd = p2 : uml::Property {
(422)           type = uc : uml::UseCase { },
(423)           name = srcInst2.name + ' erbringen',
(424)           upperValue = uval2 : uml::LiteralUnlimitedNatural {
(425)             value = 1
(426)           },
(427)           lowerValue = lval2 : uml::LiteralUnlimitedNatural {
(428)             value = 1
(429)           }
(430)         },
(431)         navigableOwnedEnd = p2.oclAsType(OrderedSet(uml::Property))
(432)       }
(433)     };
(434)
(435)     when {
(436)       not(hasRefinement(srcInst1));
(437)       not(hasRefinement(srcInst2));
(438)
(439)       packageElementActor_Ausfuehrender(srcInst1, act);
(440)       packageElementUseCase_LeistungErbringen(srcInst2, uc);
(441)     }
(442)
(443)   }
(444)
(445)
      /*****************************************************************************
(446)    *
(447)    * Anwendungsfallmodell
(448)    * Nicht-funktionale Anforderungen
(449)    *
(450)
      *****************************************************************************/
(451)
(452)   /*
```

```
(453)    * Nicht-funktionale Anforderungen des RechtsfolgenMerkmal als Element
(454)    */
(455)   relation packageElement_RechtsfolgeMerkmalNF {
(456)
(457)     checkonly domain prm srcPack :uml::Package {
(458)       packagedElement = srcInst1 : uml::InstanceSpecification {
(459)         classifier = clsf : uml::Class {
(460)           name = 'VwRechtsfolgenMerkmal'
(461)         }
(462)       },
(463)
(464)       packagedElement = srcInst2 : uml::InstanceSpecification {
(465)         classifier = clsf2 : uml::Class {
(466)           name = 'VwLeistung'
(467)         }
(468)       }
(469)     };
(470)
(471)     enforce domain target  pack : uml::Package {
(472)       ownedComment = cmt : uml::Comment {
(473)       }
(474)     };
(475)
(476)     when {
(477)       not(hasRefinement(srcInst1));
(478)       hasDesigntimeNF(srcInst1) or hasRuntimeNF(srcInst1);
(479)     }
(480)
(481)     where {
(482)       packageElementComment_RechtsfolgeMerkmalNF(srcInst1, srcInst2, cmt);
(483)     }
(484)
(485)   }
(486)
(487)   /*
(488)    * * Nicht-funktionale Anforderungen des RechtsfolgenMerkmal als Kommentar
(489)    */
(490)   relation packageElementComment_RechtsfolgeMerkmalNF {
(491)
(492)     checkonly domain prm srcInst1 : uml::InstanceSpecification {
(493)       classifier = clsf1 : uml::Class {
(494)         name = 'VwRechtsfolgenMerkmal'
(495)       }
(496)     };
(497)
(498)     checkonly domain prm srcInst2 : uml::InstanceSpecification {
(499)       classifier = clsf2 : uml::Class {
(500)         name = 'VwLeistung'
(501)       }
(502)     };
(503)
(504)     enforce domain target cmt : uml::Comment{
(505)         _body = getNF(srcInst1),
(506)         annotatedElement = uc : uml::UseCase{ }
(507)     };
(508)
(509)     when {
(510)       packageElementUseCase_LeistungErbringen(srcInst2, uc);
(511)     }
(512)
(513)   }
(514)
(515)
      /*************************************************************************
(516)    *
(517)    * Hilfsfunktionen
(518)    *
(519)
      *************************************************************************/
(520)
(521)   /*
(522)    * Liefert falsch, wenn es keine ausgehenden Dependencies mit dem
      Stereotyp
(523)    * <<refine>> gibt. Andernfalls liefert es wahr.
(524)    */
```

```
(525)   query hasRefinement(inst : uml::InstanceSpecification) : Boolean {
(526)     if (inst.clientDependency->oclIsUndefined() or inst.clientDependency-
      >isEmpty()) then
(527)       false
(528)     else
(529)       true
(530)       -- inst.clientDependency->asSequence()->exists(d |
      d.isStereotypeApplied(getStereotype('refine')))
(531)     endif
(532)   }
(533)
(534)    /*
(535)     * Prüft, ob eine Verwaltungshandlung mit dem übergebenen Namen im
      übergebenen Packages existiert
(536)     */
(537)    query existsVerwaltungshandlung(vhname : String, srcPack : Package) :
      Boolean {
(538)      if (vhname.oclIsUndefined() or srcPack.oclIsUndefined()) then
(539)        false
(540)      else
(541)        srcPack.packagedElement->select(elem|
      elem.oclIsTypeOf(uml::InstanceSpecification)).oclAsType(OrderedSet(uml::Inst
      anceSpecification))->exists(inst | inst.classifier->exists(c|
      c.name=vhname))
(542)      endif
(543)    }
(544)
(545)   /*
(546)    * Liefert den Wert des Attributes 'istRahmenbedingungEinsatz'. Wenn es
      kein Attribut mit
(547)    * diesem Namen gibt oder der Wert nicht gesetzt ist, wird _false
      zurückgeliefert.
(548)    */
(549)   query hasRuntimeNF(inst : uml::InstanceSpecification) :  Boolean {
(550)     if (inst.slot->exists(x : uml::Slot | x.definingFeature.name =
      'istRahmenbedingungEinsatz')) then
(551)      inst.slot->asSequence()->select(x : uml::Slot | x.definingFeature.name
      = 'istRahmenbedingungEinsatz')->first().value->first().booleanValue()
(552)     else
(553)      false
(554)     endif
(555)   }
(556)
(557)   /*
(558)    * Liefert den Wert des Attributes 'istRahmenbedingungUmsetzung'. Wenn es
      kein Attribut mit
(559)    * diesem Namen gibt oder der Wert nicht gesetzt ist, wird _false
      zurückgeliefert.
(560)    */
(561)   query hasDesigntimeNF(inst : uml::InstanceSpecification) :  Boolean {
(562)     if (inst.slot->exists(x : uml::Slot | x.definingFeature.name =
      'istRahmenbedingungUmsetzung')) then
(563)      inst.slot->asSequence()->select(x : uml::Slot | x.definingFeature.name
      = 'istRahmenbedingungUmsetzung')->first().value->first().booleanValue()
(564)     else
(565)      false
(566)     endif
(567)   }
(568)
(569)   /*
(570)    * Liefert den Wert des Attributes 'istRahmenbedingungUmsetzung'. Wenn es
      kein Attribut mit
(571)    * diesem Namen gibt oder der Wert nicht gesetzt ist, wird _false
      zurückgeliefert.
(572)    */
(573)   query getNF(inst : uml::InstanceSpecification) :  String {
(574)     if (inst.slot->exists(x : uml::Slot | x.definingFeature.name =
      'Rahmenbedingung')) then
(575)      inst.slot->asSequence()->select(x : uml::Slot | x.definingFeature.name
      = 'Rahmenbedingung')->first().value->first().stringValue()
(576)     else
(577)      '- keine -'
(578)     endif
(579)   }
(580)
```

```
(581) }
```

Listing 59 – Transformation „prm2cim_anf_leistung_erbringung“ (prm2cim_anf_leistung_erbringung.qvt)

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <!-- XSLT to apply UML-Profile -->
(3)   <xsl:stylesheet version="2.0"
(4)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(5)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(6)     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(7)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(8)     xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
(9)     xmlns:eGovRefAnf="http:///eGovRefAnf.ecore"
(10)    xmlns:eGovRefAnf_1="http:///schemas/eGovRefAnf/_xwmDUMThEeCj2LiIEZ2r4Q/120"
(11)    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
(12)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
(13)    xsi:schemaLocation="http:///schemas/eGovRefAnf/_xwmDUMThEeCj2LiIEZ2r4Q/120
      ../ProfileModels/eGovRefAnf.uml#_xwofkMThEeCj2LiIEZ2r4Q">
(14)
(15)    <xsl:namespace-alias stylesheet-prefix="eGovRefAnf" result-
      prefix="eGovRefAnf_1"/>
(16)
(17)    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
(18)
(19)    <xsl:template match="/">
(20)            <xsl:apply-templates/>
(21)    </xsl:template>
(22)
(23)  <xsl:template match="@*|node()">
(24)     <xsl:copy>
(25)       <xsl:apply-templates select="@*|node()"/>
(26)     </xsl:copy>
(27)  </xsl:template>
(28)
(29)  <xsl:template match="xmi:XMI">
(30)     <xsl:copy>
(31)    <xsl:attribute name="xsi:schemaLocation">
(32)            <xsl:text>http:///schemas/eGovRefAnf/_xwmDUMThEeCj2LiIEZ2r4Q/120
      ../ProfileModels/eGovRefAnf.uml#_xwofkMThEeCj2LiIEZ2r4Q</xsl:text>
(33)    </xsl:attribute>
(34)       <xsl:apply-templates select="@*|node()"/>
(35)     </xsl:copy>
(36)  </xsl:template>
(37)
(38)  <xsl:template match="uml:Model">
(39)     <xsl:copy>
(40)       <xsl:apply-templates select="@*|node()"/>
(41)     </xsl:copy>
(42)     <xsl:call-template name="Stereotypes"/>
(43)  </xsl:template>
(44)
(45)
(46)  <xsl:template match="profileApplication">
(47)      <profileApplication xmi:id="_wVnL4MUJEeClgL7iIc7saQ">
(48)        <eAnnotations xmi:id="_wVnL4cUJEeClgL7iIc7saQ"
      source="http://www.eclipse.org/uml2/2.0.0/UML">
(49)          <references xmi:type="ecore:EPackage"
      href="../ProfileModels/eGovRefAnf.uml#_xwofkMThEeCj2LiIEZ2r4Q"/>
(50)        </eAnnotations>
(51)        <appliedProfile href="../ProfileModels/eGovRefAnf.uml#_XkGiwB07Ed-
      QQ4mYkrb7Gg"/>
(52)      </profileApplication>
(53)  </xsl:template>
(54)
(55)  <!-- Stereotypen dieses Namespace werden übersprungen, weil sie erst nach
      dem uml::Model folgen dürfen -->
(56)  <xsl:template match="eGovRefAnf:*"/>
(57)
(58)  <xsl:template name="Stereotypes">
(59)    <xsl:for-each select="//eGovRefAnf:*">
(60)            <xsl:element name="eGovRefAnf_1:{local-name()}">
(61)                    <xsl:apply-templates select="@*|node()"/>
(62)            </xsl:element>
(63)    </xsl:for-each>
(64)  </xsl:template>
```

```
(65)
(66)  </xsl:stylesheet>
```

Listing 60 – Zuweisung des UML-Profils für RDF und OWL per XSLT

G Verfolgbarkeitsdokumentation

Dieser Abschnitt stellt die XSL-Stylesheets dar, mit denen die Verfolgbarkeitsdokumentation des Ausgangsdokumentes (Anhang G.1 und G.2), der OWL-Ontologie (Anhang G.3), des PRM (Anhang G.4) und des CIM (Anhang G.5) erzeugt werden. Das Ausgangsdokument sind zwei Stylesheets angegeben. Ein Stylesheet verarbetiet ein Ausgangsdokument, das mit dem OntoMat annotiert wurde, das andere ein RDFa-annotiertes Ausgangsdokument. Der Abschnitt stellt weiterhin das Stylesheet zur Erweiterung der UML-Modelle um Referenzen auf die Abschnitte der Verfolgbarkeitsdokumentation dar (G.6) . Zum Abschluss des Abschnittes werden gemeinsam benutzt Funktionen dargestellt, die von mehreren Stylesheets verwendet werden (Anhang G.7).

G.1 OntoMat-Ausgangsdokument

```
(1)   <?xml version="1.0" encoding="ISO-8859-1"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(4)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(5)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(6)     xmlns:html="http://www.w3.org/TR/REC-html40"
(7)     xmlns:xhtml="http://www.w3.org/1999/xhtml"
(8)     xmlns="http://www.w3.org/1999/xhtml"
(9)     xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
(10)    xmlns:dc="http://purl.org/dc/elements/1.1/"
(11)
      xmlns:ontomat="http://annotation.semanticweb.org/ontologies/cream/ontomat#"
(12)    xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(13)    xmlns:rss="http://purl.org/rss/1.0/"
(14)    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
(15)    xmlns:govobjects="http://govobjects.thomasoff.de/bewohnerparken#"
(16)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(17)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(18)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(19)    xmlns:owl="http://www.w3.org/2002/07/owl#"
(20)    xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
(21)    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(22)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(23)    xmlns:rdfodm="urn:rdf2odm.ecore"
(24)    xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
(25)    xmlns:owl2prm_annotation="urn:owl2prm_annotation.ecore"
(26)    xmlns:exslt="http://exslt.org/common"
(27)    extension-element-prefixes="exslt">
(28)
(29)    <xsl:include href="commonGovObjTraces.xsl"/>
(30)
(31)    <xsl:output method="xhtml" version="1.0" encoding="ISO-8859-1"
      indent="yes"/>
(32)
(33)    <xsl:param name="param-trace-rdf2prm-file"
      select="'../in/trace.rdf2prm'"/>
(34)
(35)    <xsl:template match="/html:html">
(36)      <!-- Grundstruktur für xhtml-Dokumente -->
(37)      <xsl:element name="html" namespace="http://www.w3.org/1999/xhtml"
      inherit-namespaces="yes">
(38)        <xsl:element name="head">
(39)          <xsl:element name="meta">
(40)            <xsl:attribute name="http-equiv">
(41)              <xsl:text>content-type</xsl:text>
(42)            </xsl:attribute>
(43)            <xsl:attribute name="content">
(44)              <xsl:text>text/html; charset=ISO-8859-1</xsl:text>
(45)            </xsl:attribute>
(46)          </xsl:element>
(47)          <xsl:element name="title">
(48)            <xsl:text>GovObj Traceability: PRM - Text</xsl:text>
```

```
(49)          </xsl:element>
(50)          <xsl:element name="script">
(51)            <xsl:attribute name="type">
(52)              <xsl:text>text/javascript</xsl:text>
(53)            </xsl:attribute>
(54)            <xsl:text disable-output-escaping="yes">
(55)            <![CDATA[
(56)            // <!--
(57)
(58)            function showRange(start, end, id) {
(59)              var range = document.body.createTextRange();
(60)              var elem = document.getElementById("document-src");
(61)
(62)              range.moveToElementText(elem);
(63)              range.collapse(true);
(64)
(65)              range.moveStart('character', start+1);
(66)
(67)              range.moveEnd('character', end-start);
(68)
(69)              range.expand('word');
(70)              range.select();
(71)
(72)              range.pasteHTML("<a href='#" + id + "'> <img
      src='../favorites_16.gif'/>" + range.htmlText + "</a>");
(73)
(74)              range.scrollIntoView(true);
(75)            }
(76)
(77)            function showRanges(from, to, id) {
(78)
(79)              var elem = document.getElementById("document-src");
(80)
(81)              for (i = 0; i < id.length; i++) {
(82)                showRange(from[i], to[i], id[i]);
(83)              }
(84)
(85)            }
(86)
(87)            // -->
(88)            ]]>
(89)            </xsl:text>
(90)          </xsl:element>
(91)        </xsl:element>
(92)        <xsl:element name="body">
(93)          <xsl:variable name="document-rdf">
(94)            <xsl:value-of select="substring-after(//comment(), 'ONTOMAT-
      ANNOTATION-BEGIN')"/>
(95)          </xsl:variable>
(96)
(97)          <xsl:variable name="node-rdf" select="saxon:parse($document-rdf)"
      xmlns:saxon="http://saxon.sf.net/"/>
(98)
(99)          <xsl:element name="div">
(100)
(101)           <xsl:attribute name="id">
(102)             <xsl:text>owl-instances</xsl:text>
(103)           </xsl:attribute>
(104)
(105)           <xsl:element name="h1">Verfolgbarkeit von Annotationen des
      Ausgangsdokumentes</xsl:element>
(106)           <xsl:element name="p">
(107)             <xsl:text>Die folgenden OntoMat-Annotationen sind im
      Ausgangsdokument enthalten.</xsl:text>
(108)           </xsl:element>
(109)
(110)           <xsl:element name="p">
(111)             <xsl:element name="input">
(112)               <xsl:attribute name="type">
(113)                 <xsl:text>button</xsl:text>
(114)               </xsl:attribute>
(115)               <xsl:attribute name="value">
(116)                 <xsl:text>Alle Annotationen anzeigen</xsl:text>
(117)               </xsl:attribute>
(118)               <xsl:attribute name="onclick">
```

```
(119)                 <xsl:text>configureRanges()</xsl:text>
(120)               </xsl:attribute>
(121)             </xsl:element>
(122)           </xsl:element>
(123)
(124)           <xsl:for-each select="$node-rdf//govobjects-
      ontology:*[@rdf:about]">
(125)             <xsl:variable name="about" select="@rdf:about"/>
(126)             <xsl:variable name="range" select="substring-after($node-
      rdf//ontomat:ReificationDataIndividual[ontomat:AboutIndividual =
      $about]/ontomat:CreationSource, '#')"/>
(127)             <xsl:variable name="range-from" select="substring-
      before(substring-after($range, 'xpointer(//point()['), ']')"/>
(128)             <xsl:variable name="range-to" select="substring-
      before(substring-after($range, 'range-to(//point()['), ']')"/>
(129)             <xsl:variable name="anchor" select="substring-after($about,
      '#')"/>
(130)             <xsl:variable name="name" select="govobj:rdf-name(.)"/>
(131)             <xsl:variable name="label" select="govobj:rdf-label(.)"/>
(132)             <xsl:variable name="type" select="govobj:rdf-type(.)"/>
(133)
(134)             <xsl:element name="div">
(135)               <xsl:attribute name="class">
(136)                 <xsl:text>div-element</xsl:text>
(137)               </xsl:attribute>
(138)
(139)               <!-- Achnor für Navigation im Dokument anlegen -->
(140)               <xsl:element name="a">
(141)                 <xsl:attribute name="name">
(142)                   <xsl:value-of select="$anchor"/>
(143)                 </xsl:attribute>
(144)               </xsl:element>
(145)
(146)               <!-- Überschrift des Elementes anlegen -->
(147)               <xsl:element name="h2">
(148)                 <xsl:value-of select="$label"/>
(149)               </xsl:element>
(150)
(151)               <xsl:element name="p">
(152)                 <xsl:element name="input">
(153)                   <xsl:attribute name="type">
(154)                     <xsl:text>button</xsl:text>
(155)                   </xsl:attribute>
(156)                   <xsl:attribute name="value">
(157)                     <xsl:text>Annotierte Textpassage anzeigen</xsl:text>
(158)                   </xsl:attribute>
(159)                   <xsl:attribute name="onclick">
(160)                     <xsl:text>showRange(</xsl:text>
(161)                     <xsl:value-of select="$range-from"/>
(162)                     <xsl:text>, </xsl:text>
(163)                     <xsl:value-of select="$range-to"/>
(164)                     <xsl:text>, '</xsl:text>
(165)                     <xsl:value-of select="$anchor"/>
(166)                     <xsl:text>')</xsl:text>
(167)                   </xsl:attribute>
(168)                 </xsl:element>
(169)               </xsl:element>
(170)
(171)               <xsl:element name="div">
(172)                 <xsl:attribute name="class">
(173)                   <xsl:text>div-eigenschaften</xsl:text>
(174)                 </xsl:attribute>
(175)                 <xsl:element name="h3">
(176)                   <xsl:text>Eigenschaften</xsl:text>
(177)                 </xsl:element>
(178)
(179)                 <xsl:element name="p">Das Individuum hat die folgenden
      Eigenschaften:</xsl:element>
(180)
(181)                 <xsl:element name="ul">
(182)                   <xsl:attribute name="class">
(183)                     <xsl:text>ul-eigenschaft</xsl:text>
(184)                   </xsl:attribute>
(185)
(186)                   <xsl:element name="li">
```

```
(187)                     <xsl:attribute name="class">
(188)                       <xsl:text>li-eigenschaft</xsl:text>
(189)                     </xsl:attribute>
(190)                     <xsl:element name="span">
(191)                       <xsl:attribute name="class">
(192)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(193)                       </xsl:attribute>
(194)                       <xsl:text>Name:</xsl:text>
(195)                     </xsl:element>
(196)                     <xsl:element name="span">
(197)                       <xsl:attribute name="class">
(198)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(199)                       </xsl:attribute>
(200)                       <xsl:value-of select="$name"/>
(201)                     </xsl:element>
(202)                   <!-- end: li -->
(203)                   </xsl:element>
(204)
(205)                   <xsl:element name="li">
(206)                     <xsl:attribute name="class">
(207)                       <xsl:text>li-eigenschaft</xsl:text>
(208)                     </xsl:attribute>
(209)                     <xsl:element name="span">
(210)                       <xsl:attribute name="class">
(211)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(212)                       </xsl:attribute>
(213)                       <xsl:text>Ontologie-Klasse:</xsl:text>
(214)                     </xsl:element>
(215)                     <xsl:element name="span">
(216)                       <xsl:attribute name="class">
(217)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(218)                       </xsl:attribute>
(219)                       <xsl:value-of select="$type"/>
(220)                     </xsl:element>
(221)                   <!-- end: li -->
(222)                   </xsl:element>
(223)
(224)                   <xsl:element name="li">
(225)                     <xsl:attribute name="class">
(226)                       <xsl:text>li-eigenschaft</xsl:text>
(227)                     </xsl:attribute>
(228)                     <xsl:element name="span">
(229)                       <xsl:attribute name="class">
(230)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(231)                       </xsl:attribute>
(232)                       <xsl:text>Annotierte Textpassage: von Zeichen
      </xsl:text>
(233)                     </xsl:element>
(234)                     <xsl:element name="span">
(235)                       <xsl:attribute name="class">
(236)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(237)                       </xsl:attribute>
(238)                       <xsl:value-of select="$range-from"/>
(239)                       <xsl:text> bis </xsl:text>
(240)                       <xsl:value-of select="$range-to"/>
(241)                     </xsl:element>
(242)                   <!-- end: li -->
(243)                   </xsl:element>
(244)
(245)                   <!-- end: ul -->
(246)                   </xsl:element>
(247)
(248)                 <!-- end: div - eigenschaften -->
(249)                 </xsl:element>
(250)
(251)
(252)               <!-- Eigenschaften (DataProperties) -->
(253)               <xsl:element name="div">
(254)                 <xsl:attribute name="class">
(255)                   <xsl:text>div-attribute</xsl:text>
(256)                 </xsl:attribute>
(257)
(258)                 <xsl:element name="h3">Data Properties</xsl:element>
(259)                 <xsl:element name="p">Das Individuum hat die folgenden
      Werteausprägungen von Data Properties.</xsl:element>
```

```
(260)
(261)                 <!-- Prüfen, ob ohne Eigenschaften -->
(262)                 <xsl:if test="count(govobjects-ontology:*[@rdf:datatype]) =
      0">
(263)                   <xsl:element name="p">
(264)                     <xsl:text>- Keine -</xsl:text>
(265)                   </xsl:element>
(266)                 </xsl:if>
(267)
(268)                 <!-- Eigenschaft -->
(269)                 <xsl:for-each select="govobjects-ontology:*[@rdf:datatype]">
(270)                   <xsl:element name="div">
(271)                     <xsl:attribute name="class">
(272)                       <xsl:text>div-attribut</xsl:text>
(273)                     </xsl:attribute>
(274)
(275)                     <xsl:element name="h4">
(276)                       <xsl:text>Data Property '</xsl:text>
(277)                       <xsl:value-of select="local-name()"/>
(278)                       <xsl:text>'</xsl:text>
(279)                     </xsl:element>
(280)
(281)                     <xsl:element name="ul">
(282)                       <xsl:attribute name="class">
(283)                         <xsl:text>ul-attribut</xsl:text>
(284)                       </xsl:attribute>
(285)
(286)                       <xsl:element name="li">
(287)                         <xsl:attribute name="class">
(288)                           <xsl:text>li-attribut</xsl:text>
(289)                         </xsl:attribute>
(290)                         <xsl:element name="span">
(291)                           <xsl:attribute name="class">
(292)                             <xsl:text>span-attribut-titel</xsl:text>
(293)                           </xsl:attribute>
(294)                           <xsl:text>Name:</xsl:text>
(295)                         </xsl:element>
(296)                         <xsl:element name="span">
(297)                           <xsl:attribute name="class">
(298)                             <xsl:text>span-attribut-wert</xsl:text>
(299)                           </xsl:attribute>
(300)                           <xsl:value-of select="local-name()"/>
(301)                         </xsl:element>
(302)                       <!-- end: li -->
(303)                       </xsl:element>
(304)
(305)                       <xsl:element name="li">
(306)                         <xsl:attribute name="class">
(307)                           <xsl:text>li-attribut</xsl:text>
(308)                         </xsl:attribute>
(309)                         <xsl:element name="span">
(310)                           <xsl:attribute name="class">
(311)                             <xsl:text>span-attribut-titel</xsl:text>
(312)                           </xsl:attribute>
(313)                           <xsl:text>Datentyp:</xsl:text>
(314)                         </xsl:element>
(315)                         <xsl:element name="span">
(316)                           <xsl:attribute name="class">
(317)                             <xsl:text>span-attribut-wert</xsl:text>
(318)                           </xsl:attribute>
(319)                           <xsl:value-of select="substring-
      after(@rdf:datatype, '#')"/>
(320)                         </xsl:element>
(321)                       <!-- end: li -->
(322)                       </xsl:element>
(323)
(324)                       <xsl:element name="li">
(325)                         <xsl:attribute name="class">
(326)                           <xsl:text>li-attribut</xsl:text>
(327)                         </xsl:attribute>
(328)                         <xsl:element name="span">
(329)                           <xsl:attribute name="class">
(330)                             <xsl:text>span-attribut-titel</xsl:text>
(331)                           </xsl:attribute>
(332)                           <xsl:text>Wert:</xsl:text>
```

```
(333)                          </xsl:element>
(334)                          <xsl:element name="span">
(335)                            <xsl:attribute name="class">
(336)                              <xsl:text>span-attribut-wert</xsl:text>
(337)                            </xsl:attribute>
(338)                            <xsl:choose>
(339)                              <xsl:when test="string-length(.) > 0">
(340)                                <xsl:text>'</xsl:text>
(341)                                <xsl:value-of select="."/>
(342)                                <xsl:text>'</xsl:text>
(343)                              </xsl:when>
(344)                              <xsl:otherwise>
(345)                                <xsl:text>- Undefiniert -</xsl:text>
(346)                              </xsl:otherwise>
(347)                            </xsl:choose>
(348)                          </xsl:element>
(349)                        <!-- end: li -->
(350)                        </xsl:element>
(351)
(352)                      <!-- end: ul - attribut -->
(353)                      </xsl:element>
(354)                    <!-- end: div - attribut -->
(355)                    </xsl:element>
(356)
(357)                  <!-- end: Attribute-->
(358)                  </xsl:for-each>
(359)
(360)                <!-- end: div-attribute -->
(361)                </xsl:element>
(362)
(363)                <!-- Beziehungen (ObjectProperties) -->
(364)                <xsl:element name="div">
(365)                  <xsl:attribute name="class">
(366)                    <xsl:text>div-beziehungen</xsl:text>
(367)                  </xsl:attribute>
(368)
(369)                  <xsl:element name="h3">Object Properties</xsl:element>
(370)                  <xsl:element name="p">Das Individuum hat die folgenden
      Ausprägungen von Object Properties.</xsl:element>
(371)
(372)                  <!-- Prüfen, ob ohne Beziehungen -->
(373)                  <xsl:if test="count(govobjects-
      ontology:*[not(@rdf:datatype)]) = 0">
(374)                    <xsl:element name="p">
(375)                      <xsl:text>- Keine -</xsl:text>
(376)                    </xsl:element>
(377)                  </xsl:if>
(378)
(379)                <!-- Beziehung -->
(380)                <xsl:for-each select="govobjects-
      ontology:*[not(@rdf:datatype)]">
(381)                    <xsl:element name="div">
(382)                    <xsl:attribute name="class">
(383)                      <xsl:text>div-beziehung</xsl:text>
(384)                    </xsl:attribute>
(385)
(386)                    <xsl:element name="h4">
(387)                      <xsl:text>Object Property '</xsl:text>
(388)                      <xsl:value-of select="local-name()"/>
(389)                      <xsl:text>'</xsl:text>
(390)                    </xsl:element>
(391)
(392)                    <xsl:element name="p">
(393)                      <xsl:text>Das Individuum hat die Beziehung '</xsl:text>
(394)                      <xsl:value-of select="local-name()"/>
(395)                      <xsl:text>' zu dem/den Individuen: </xsl:text>
(396)
(397)                      <!-- Inverse Beziehungen zu einem Individuum -->
(398)                      <xsl:if test="not(govobjects-ontology:*)">
(399)                        <xsl:variable name="anchor-dst">
(400)                          <xsl:value-of select="@rdf:resource"/>
(401)                        </xsl:variable>
(402)                        <xsl:variable name="name-dst">
(403)                          <xsl:value-of select="$node-rdf//govobjects-
      ontology:*[@rdf:about = $anchor-dst]/rdfs:label"/>
```

```
(404)                       </xsl:variable>
(405)
(406)                       <xsl:element name="a">
(407)                         <xsl:attribute name="href">
(408)                           <xsl:value-of select="concat('#', substring-
      after($anchor-dst, '#'))"/>
(409)                         </xsl:attribute>
(410)                         <xsl:value-of select="$name-dst"/>
(411)                       </xsl:element>
(412)                     </xsl:if>
(413)
(414)                     <!-- Beziehung zu anderen Individuen -->
(415)                     <xsl:for-each select="govobjects-ontology:*">
(416)                       <xsl:variable name="anchor-dst">
(417)                         <xsl:value-of select="@rdf:about"/>
(418)                       </xsl:variable>
(419)                       <xsl:variable name="name-dst">
(420)                         <xsl:value-of select="rdfs:label"/>
(421)                       </xsl:variable>
(422)
(423)                       <xsl:element name="a">
(424)                         <xsl:attribute name="href">
(425)                           <xsl:value-of select="concat('#', substring-
      after($anchor-dst, '#'))"/>
(426)                         </xsl:attribute>
(427)                         <xsl:value-of select="$name-dst"/>
(428)                       </xsl:element>
(429)
(430)                       <xsl:if test="position() &lt; last() - 1">
(431)                         <xsl:text>, </xsl:text>
(432)                       </xsl:if>
(433)                       <xsl:if test="position() = last() - 1">
(434)                         <xsl:text> und </xsl:text>
(435)                       </xsl:if>
(436)                       <xsl:if test="position() = last()">
(437)                         <xsl:text/>
(438)                       </xsl:if>
(439)                     </xsl:for-each>
(440)                     <xsl:text>.</xsl:text>
(441)
(442)                   <!-- End: p -->
(443)                   </xsl:element>
(444)
(445)                 <!-- end: div - beziehung -->
(446)                 </xsl:element>
(447)
(448)               <!-- end: beziehung -->
(449)               </xsl:for-each>
(450)
(451)               <!-- end: div-beziehungen -->
(452)               </xsl:element>
(453)
(454)               <!-- Zielelement(e) im PRM -->
(455)               <xsl:element name="div">
(456)                 <xsl:attribute name="class">
(457)                   <xsl:text>div-verfolgbarkeit-vorwaerts</xsl:text>
(458)                 </xsl:attribute>
(459)
(460)                 <xsl:element name="h3">
(461)                   <xsl:text>Zielelemente im PRM</xsl:text>
(462)                 </xsl:element>
(463)
(464)                 <xsl:element name="p">Das Element wird in den folgenden
      Abbildungsregeln verwendet.</xsl:element>
(465)
(466)                 <xsl:variable name="id" select="$about"/>
(467)                 <xsl:variable name="document-trace-rdf2prm"
      select="document($param-trace-rdf2prm-file)"/>
(468)
(469)                 <xsl:if test="count($document-trace-
      rdf2prm//owl2prm_annotation:*[child::*[@href=$id]]) = 0">
(470)                   <xsl:element name="p">- Keine -</xsl:element>
(471)                 </xsl:if>
(472)
```

```
(473)                 <xsl:for-each select="$document-trace-
      rdf2prm//owl2prm_annotation:*[child::*[@href=$id]]">
(474)
(475)                   <xsl:element name="h4">Abbildungsregel '<xsl:value-of
      select="local-name(.)"/>'</xsl:element>
(476)
(477)                   <xsl:element name="p">Das Element wird in die folgenden
      Elemente des PRM überführt:</xsl:element>
(478)
(479)                   <xsl:element name="ul">
(480)                     <xsl:attribute name="class">
(481)                       <xsl:text>ul-zielelemente-prm</xsl:text>
(482)                     </xsl:attribute>
(483)
(484)                     <xsl:for-each select="child::*[starts-with(name(),
      'dst')]">
(485)                       <xsl:variable name="file-prm" select="substring-
      before(@href, '#')"/>
(486)                       <xsl:variable name="id-prm" select="substring-
      after(@href, '#')"/>
(487)                       <xsl:variable name="document-prm"
      select="document($file-prm)"/>
(488)                       <xsl:variable name="elem-prm" select="$document-
      prm//*[@xmi:id = $id-prm]"/>
(489)                       <xsl:variable name="label-prm"
      select="govobj:label($elem-prm, $id-prm)"/>
(490)
(491)                       <xsl:element name="li">
(492)                         <xsl:attribute name="class">
(493)                           <xsl:text>li-zielelement-prm</xsl:text>
(494)                         </xsl:attribute>
(495)                         <xsl:element name="a">
(496)                           <xsl:attribute name="href">
(497)                             <xsl:value-of select="replace(@href, '.uml',
      '.html')"/>
(498)                           </xsl:attribute>
(499)                           <xsl:value-of select="$label-prm"/>
(500)                         </xsl:element>
(501)                       </xsl:element>
(502)
(503)                     <!--  end: child -->
(504)                     </xsl:for-each>
(505)
(506)                   <!-- end: ul-zielelemente-prm -->
(507)                   </xsl:element>
(508)
(509)                 <!-- end: owl2prm_annotation -->
(510)                 </xsl:for-each>
(511)
(512)               <!-- end: div-verfolgbarkeit-vorwaerts -->
(513)               </xsl:element>
(514)
(515)             <!-- End: div -->
(516)             </xsl:element>
(517)
(518)           <!-- End: Individuum -->
(519)           </xsl:for-each>
(520)
(521)         </xsl:element>
(522)
(523)         <xsl:element name="script">
(524)           <xsl:attribute name="type">
(525)             <xsl:text>text/javascript</xsl:text>
(526)           </xsl:attribute>
(527)           <xsl:text disable-output-escaping="yes">
(528)           <![CDATA[
(529)           // <!--
(530)
(531)           function configureRanges() {
(532)
(533)             var from = new Array();
(534)             var to = new Array();
(535)             var id = new Array();
(536)           ]]>
(537)           </xsl:text>
```

```
(538)
(539)           <xsl:for-each select="$node-rdf//govobjects-
      ontology:*[@rdf:about]">
(540)             <xsl:variable name="about" select="@rdf:about"/>
(541)             <xsl:variable name="range" select="substring-after($node-
      rdf//ontomat:ReificationDataIndividual[ontomat:AboutIndividual =
      $about]/ontomat:CreationSource, '#')"/>
(542)             <xsl:variable name="range-from" select="substring-
      before(substring-after($range, 'xpointer(//point()['), ']')"/>
(543)             <xsl:variable name="range-to" select="substring-
      before(substring-after($range, 'range-to(//point()['), ']')"/>
(544)             <xsl:variable name="anchor" select="substring-after($about,
      '#')"/>
(545)
(546)             <xsl:value-of select="concat('from[', position(), ']=', $range-
      from, ';')"/>
(547)             <xsl:value-of select="concat('to[', position(), ']=', $range-
      to), ';'"/>
(548)             <xsl:value-of select="concat('id[', position(), ']=')"/>
(549)             <xsl:text>"</xsl:text>
(550)             <xsl:value-of select="$anchor"/>
(551)             <xsl:text>";</xsl:text>
(552)           </xsl:for-each>
(553)
(554)           <xsl:text disable-output-escaping="yes">
(555)           <![CDATA[
(556)
(557)             showRanges(from, to, id);
(558)           }
(559)
(560)           // -->
(561)           ]]>
(562)           </xsl:text>
(563)         </xsl:element>
(564)
(565)         <xsl:element name="div">
(566)           <xsl:attribute name="id">
(567)             <xsl:text>document-src</xsl:text>
(568)           </xsl:attribute>
(569)
(570)           <xsl:copy-of select="html:body/*" copy-namespaces="no"/>
(571)         </xsl:element>
(572)       </xsl:element>
(573)     </xsl:element>
(574)   </xsl:template>
(575)
(576) </xsl:stylesheet>
```

Listing 61 – Erzeugung der OntoMat-Verfolgbarkeitsdokumentation (create-GovObjTraces_SourceDocument.xsl)

G.2 RDFa-Ausgangsdokument

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(4)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(5)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(6)     xmlns:xhtml="http://www.w3.org/1999/xhtml"
(7)     xmlns="http://www.w3.org/1999/xhtml"
(8)     xmlns:owl="http://www.w3.org/2002/07/owl#"
(9)     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(10)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(11)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(12)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(13)    xmlns:bewohnerparken="http://govobjects.thomasoff.de/bewohnerparken#"
(14)    xmlns:rdfa2prm_annotation="urn:rdfa2prm_annotation.ecore"
(15)    xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
(16)    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(17)    xsi:schemaLocation="urn:rdfa2prm_annotation.ecore
      rdfa2prm_annotation.ecore" >
(18)
(19)    <xsl:include href="commonGovObjTraces.xsl"/>
(20)    <xsl:include href="convert-OwlCommon.xsl"/>
(21)
```

```
(22)    <xsl:output method="xhtml" version="1.0" encoding="UTF-8" indent="yes"/>
(23)
(24)    <xsl:param name="param-trace-rdf2prm-file"
      select="'../in/trace.rdf2prm'"/>
(25)
(26)    <xsl:template match="/xhtml:html">
(27)
(28)      <!-- Grundstruktur für xhtml-Dokumente -->
(29)      <xsl:element name="html" namespace="http://www.w3.org/1999/xhtml"
      inherit-namespaces="yes">
(30)
(31)        <xsl:element name="head">
(32)          <xsl:element name="meta">
(33)            <xsl:attribute name="http-equiv">
(34)              <xsl:text>content-type</xsl:text>
(35)            </xsl:attribute>
(36)            <xsl:attribute name="content">
(37)              <xsl:text>text/html; charset=ISO-8859-1</xsl:text>
(38)            </xsl:attribute>
(39)          </xsl:element>
(40)          <xsl:element name="title">
(41)            <xsl:text>GovObj Traceability: PRM - Text</xsl:text>
(42)          </xsl:element>
(43)        <!-- end: head -->
(44)        </xsl:element>
(45)
(46)        <xsl:element name="body">
(47)
(48)          <xsl:element name="div">
(49)
(50)            <xsl:attribute name="id">
(51)              <xsl:text>owl-instances</xsl:text>
(52)            </xsl:attribute>
(53)
(54)            <xsl:element name="h1">Verfolgbarkeit von Annotationen des
      Ausgangsdokumentes</xsl:element>
(55)
(56)            <xsl:element name="p">
(57)              <xsl:text>Die folgenden RDFa-Annotationen sind im
      Ausgangsdokument enthalten.</xsl:text>
(58)            </xsl:element>
(59)
(60)            <xsl:for-each select="//xhtml:span[@typeof][not(@rel)]">
(61)              <xsl:variable name="about" select="@about"/>
(62)              <xsl:variable name="anchor" select="substring-after($about,
      '#')"/>
(63)              <xsl:variable name="name" select="."/>
(64)              <xsl:variable name="label" select="."/>
(65)              <xsl:variable name="type" select="substring-after(@typeof,
      ':')"/>
(66)
(67)              <xsl:element name="div">
(68)                <xsl:attribute name="class">
(69)                  <xsl:text>div-element</xsl:text>
(70)                </xsl:attribute>
(71)
(72)                <!-- Achnor für Navigation im Dokument anlegen -->
(73)                <xsl:element name="a">
(74)                  <xsl:attribute name="name">
(75)                    <xsl:value-of select="$anchor"/>
(76)                  </xsl:attribute>
(77)                </xsl:element>
(78)
(79)                <!-- Überschrift des Elementes anlegen -->
(80)                <xsl:element name="h2">
(81)                  <xsl:text>Individuum '</xsl:text>
(82)                  <xsl:value-of select="$label"/>
(83)                  <xsl:text>' der Ontologie-Klasse '</xsl:text>
(84)                  <xsl:value-of select="$type"/>
(85)                  <xsl:text>'</xsl:text>
(86)                </xsl:element>
(87)
(88)                <xsl:element name="div">
(89)                  <xsl:attribute name="class">
(90)                    <xsl:text>div-eigenschaften</xsl:text>
```

```
(91)                  </xsl:attribute>
(92)                  <xsl:element name="h3">
(93)                    <xsl:text>Eigenschaften</xsl:text>
(94)                  </xsl:element>
(95)
(96)                  <xsl:element name="p">Das Individuum hat die folgenden
      Eigenschaften:</xsl:element>
(97)
(98)                  <xsl:element name="ul">
(99)                    <xsl:attribute name="class">
(100)                     <xsl:text>ul-eigenschaft</xsl:text>
(101)                   </xsl:attribute>
(102)
(103)                   <xsl:element name="li">
(104)                     <xsl:attribute name="class">
(105)                       <xsl:text>li-eigenschaft</xsl:text>
(106)                     </xsl:attribute>
(107)                     <xsl:element name="span">
(108)                       <xsl:attribute name="class">
(109)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(110)                       </xsl:attribute>
(111)                       <xsl:text>Name:</xsl:text>
(112)                     </xsl:element>
(113)                     <xsl:element name="span">
(114)                       <xsl:attribute name="class">
(115)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(116)                       </xsl:attribute>
(117)                       <xsl:value-of select="$label"/>
(118)                     </xsl:element>
(119)                   <!-- end: li -->
(120)                   </xsl:element>
(121)
(122)                   <xsl:element name="li">
(123)                     <xsl:attribute name="class">
(124)                       <xsl:text>li-eigenschaft</xsl:text>
(125)                     </xsl:attribute>
(126)                     <xsl:element name="span">
(127)                       <xsl:attribute name="class">
(128)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(129)                       </xsl:attribute>
(130)                       <xsl:text>Ontologie-Klasse:</xsl:text>
(131)                     </xsl:element>
(132)                     <xsl:element name="span">
(133)                       <xsl:attribute name="class">
(134)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(135)                       </xsl:attribute>
(136)                       <xsl:value-of select="$type"/>
(137)                     </xsl:element>
(138)                   <!-- end: li -->
(139)                   </xsl:element>
(140)
(141)                   <xsl:element name="li">
(142)                     <xsl:attribute name="class">
(143)                       <xsl:text>li-eigenschaft</xsl:text>
(144)                     </xsl:attribute>
(145)                     <xsl:element name="span">
(146)                       <xsl:attribute name="class">
(147)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(148)                       </xsl:attribute>
(149)                       <xsl:text>Annotierte Textpassage: </xsl:text>
(150)                     </xsl:element>
(151)                     <xsl:element name="span">
(152)                       <xsl:attribute name="class">
(153)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(154)                       </xsl:attribute>
(155)                       <xsl:element name="a">
(156)                         <xsl:attribute name="href" select="concat('#',
      $anchor, '_source')"/>
(157)                           <xsl:value-of select="$name"/>
(158)                       </xsl:element>
(159)                     </xsl:element>
(160)                   <!-- end: li -->
(161)                   </xsl:element>
(162)
(163)                   <!-- end: ul -->
```

```
(164)                   </xsl:element>
(165)
(166)                 <!-- end: div - eigenschaften -->
(167)                 </xsl:element>
(168)
(169)               <!-- Eigenschaften (DataProperties) -->
(170)               <xsl:element name="div">
(171)                 <xsl:attribute name="class">
(172)                   <xsl:text>div-attribute</xsl:text>
(173)                 </xsl:attribute>
(174)
(175)                 <xsl:element name="h3">Data Properties</xsl:element>
(176)                 <xsl:element name="p">Das Individuum hat die folgenden
      Werteausprägungen von Data Properties.</xsl:element>
(177)
(178)                 <!-- Prüfen, ob ohne Eigenschaften -->
(179)                 <xsl:if test="count(//xhtml:span[@about=$about]) = 0">
(180)                   <xsl:element name="p">
(181)                     <xsl:text>- Keine -</xsl:text>
(182)                   </xsl:element>
(183)                 </xsl:if>
(184)
(185)                 <!-- Eigenschaft -->
(186)                 <xsl:for-each
      select="//xhtml:span[@about=$about][@property][not(@property='rdfs:label')]"
      >
(187)                   <xsl:element name="div">
(188)                     <xsl:attribute name="class">
(189)                       <xsl:text>div-attribut</xsl:text>
(190)                     </xsl:attribute>
(191)
(192)                     <xsl:element name="h4">
(193)                       <xsl:text>Data Property '</xsl:text>
(194)                       <xsl:value-of select="substring-after(@property,
      ':')"/>
(195)                       <xsl:text>'</xsl:text>
(196)                     </xsl:element>
(197)
(198)                     <xsl:element name="ul">
(199)                       <xsl:attribute name="class">
(200)                         <xsl:text>ul-attribut</xsl:text>
(201)                       </xsl:attribute>
(202)
(203)                       <xsl:element name="li">
(204)                         <xsl:attribute name="class">
(205)                           <xsl:text>li-attribut</xsl:text>
(206)                         </xsl:attribute>
(207)                         <xsl:element name="span">
(208)                           <xsl:attribute name="class">
(209)                             <xsl:text>span-attribut-titel</xsl:text>
(210)                           </xsl:attribute>
(211)                           <xsl:text>Name:</xsl:text>
(212)                         </xsl:element>
(213)                         <xsl:element name="span">
(214)                           <xsl:attribute name="class">
(215)                             <xsl:text>span-attribut-wert</xsl:text>
(216)                           </xsl:attribute>
(217)                           <xsl:value-of select="substring-after(@property,
      ':')"/>
(218)                         </xsl:element>
(219)                       <!-- end: li -->
(220)                       </xsl:element>
(221)
(222)                       <xsl:element name="li">
(223)                         <xsl:attribute name="class">
(224)                           <xsl:text>li-attribut</xsl:text>
(225)                         </xsl:attribute>
(226)                         <xsl:element name="span">
(227)                           <xsl:attribute name="class">
(228)                             <xsl:text>span-attribut-titel</xsl:text>
(229)                           </xsl:attribute>
(230)                           <xsl:text>Wert:</xsl:text>
(231)                         </xsl:element>
(232)                         <xsl:element name="span">
(233)                           <xsl:attribute name="class">
```

```
(234)                             <xsl:text>span-attribut-wert</xsl:text>
(235)                           </xsl:attribute>
(236)                           <xsl:value-of select="@content"/>
(237)                         </xsl:element>
(238)                       <!-- end: li -->
(239)                       </xsl:element>
(240)
(241)                     <!-- end: ul - attribut -->
(242)                     </xsl:element>
(243)                   <!-- end: div - attribut -->
(244)                   </xsl:element>
(245)
(246)                 <!-- end: Attribute-->
(247)                 </xsl:for-each>
(248)
(249)               <!-- end: div-attribute -->
(250)               </xsl:element>
(251)
(252)               <!-- Beziehungen (ObjectProperties) -->
(253)               <xsl:element name="div">
(254)                 <xsl:attribute name="class">
(255)                   <xsl:text>div-beziehungen</xsl:text>
(256)                 </xsl:attribute>
(257)
(258)                 <xsl:element name="h3">Object Properties</xsl:element>
(259)                 <xsl:element name="p">Das Individuum hat die folgenden
      Ausprägungen von Object Properties.</xsl:element>
(260)
(261)                 <!-- Prüfen, ob ohne Beziehungen -->
(262)                 <xsl:if
      test="count(//xhtml:span[@about=$about][@rel][@resource]) = 0">
(263)                   <xsl:element name="p">
(264)                     <xsl:text>- Keine -</xsl:text>
(265)                   </xsl:element>
(266)                 </xsl:if>
(267)
(268)                 <!-- Beziehung -->
(269)                 <xsl:for-each
      select="//xhtml:span[@about=$about][@rel][@resource]">
(270)                     <xsl:element name="div">
(271)                     <xsl:attribute name="class">
(272)                       <xsl:text>div-beziehung</xsl:text>
(273)                     </xsl:attribute>
(274)
(275)                     <xsl:element name="h4">
(276)                       <xsl:text>Object Property '</xsl:text>
(277)                       <xsl:value-of select="substring-after(@rel, ':')"/>
(278)                       <xsl:text>'</xsl:text>
(279)                     </xsl:element>
(280)
(281)                     <xsl:element name="p">
(282)                       <xsl:text>Das Individuum hat die Beziehung
      '</xsl:text>
(283)                       <xsl:value-of select="substring-after(@rel, ':')"/>
(284)                       <xsl:text>' zu dem/den Individuen: </xsl:text>
(285)
(286)                       <!-- Inverse Beziehungen zu einem Individuum über @rev
      sind nicht implementiert -->
(287)
(288)                       <!-- Beziehung zu anderen Individuen -->
(289)
(290)                       <xsl:variable name="resource-dst" select="@resource"/>
(291)                       <xsl:variable name="anchor-dst" select="substring-
      after($resource-dst, '#')"/>
(292)                       <xsl:variable name="name-dst"
      select="//xhtml:span[@about=$resource-dst][@property='rdfs:label']"/>
(293)
(294)                       <xsl:element name="a">
(295)                         <xsl:attribute name="href" select="concat('#',
      $anchor-dst)"/>
(296)                         <xsl:value-of select="$name-dst"/>
(297)                       </xsl:element>
(298)
(299)                       <xsl:text>.</xsl:text>
(300)
```

```
(301)                     <!-- End: p -->
(302)                     </xsl:element>
(303)
(304)                   <!-- end: div - beziehung -->
(305)                   </xsl:element>
(306)
(307)                 <!-- end: beziehung -->
(308)                 </xsl:for-each>
(309)
(310)               <!-- end: div-beziehungen -->
(311)               </xsl:element>
(312)
(313)               <!-- Zielelement(e) im PRM -->
(314)               <xsl:element name="div">
(315)                 <xsl:attribute name="class">
(316)                   <xsl:text>div-verfolgbarkeit-vorwaerts</xsl:text>
(317)                 </xsl:attribute>
(318)
(319)                 <xsl:element name="h3">
(320)                   <xsl:text>Zielelemente im PRM</xsl:text>
(321)                 </xsl:element>
(322)
(323)                 <xsl:element name="p">Das Element wird in den folgenden
      Abbildungsregeln verwendet.</xsl:element>
(324)
(325)                 <xsl:variable name="id" select="concat(govobj:generate-
      id(.), '_individual')"/>
(326)                 <xsl:variable name="document-trace-rdf2prm"
      select="document($param-trace-rdf2prm-file)"/>
(327)
(328)                 <xsl:if test="count($document-trace-
      rdf2prm//rdfa2prm_annotation:*[child::*[contains(@href, $id)]]) = 0">
(329)                   <xsl:element name="p">- Keine -</xsl:element>
(330)                 </xsl:if>
(331)
(332)                 <xsl:for-each select="$document-trace-
      rdf2prm//rdfa2prm_annotation:*[child::*[contains(@href, $id)]]">
(333)
(334)                 <xsl:element name="h4">Abbildungsregel '<xsl:value-of
      select="local-name(.)"/>'</xsl:element>
(335)
(336)                 <xsl:element name="p">Das Element wird in die folgenden
      Elemente des PRM überführt:</xsl:element>
(337)
(338)                 <xsl:element name="ul">
(339)                   <xsl:attribute name="class">
(340)                     <xsl:text>ul-zielelemente-prm</xsl:text>
(341)                   </xsl:attribute>
(342)
(343)                   <xsl:for-each select="child::*[starts-with(name(),
      'dst')]">
(344)                     <xsl:variable name="file-prm" select="substring-
      before(@href, '#')"/>
(345)                     <xsl:variable name="id-prm" select="substring-
      after(@href, '#')"/>
(346)                     <xsl:variable name="document-prm"
      select="document($file-prm)"/>
(347)                     <xsl:variable name="elem-prm" select="$document-
      prm//*[@xmi:id = $id-prm]"/>
(348)                     <xsl:variable name="label-prm"
      select="govobj:label($elem-prm, $id-prm)"/>
(349)
(350)                     <xsl:element name="li">
(351)                       <xsl:attribute name="class">
(352)                         <xsl:text>li-zielelement-prm</xsl:text>
(353)                       </xsl:attribute>
(354)                       <xsl:element name="a">
(355)                         <xsl:attribute name="href">
(356)                           <xsl:value-of select="replace(@href, '.uml',
      '.html')"/>
(357)                         </xsl:attribute>
(358)                         <xsl:value-of select="$label-prm"/>
(359)                       </xsl:element>
(360)                     </xsl:element>
(361)
```

```
(362)                   <!--  end: child -->
(363)                   </xsl:for-each>
(364)
(365)                 <!-- end: ul-zielelemente-prm -->
(366)                 </xsl:element>
(367)
(368)                 <!-- end: rdfa2prm_annotation -->
(369)                 </xsl:for-each>
(370)
(371)               <!-- end: div-verfolgbarkeit-vorwaerts -->
(372)               </xsl:element>
(373)
(374)             <!--  end: div (div-element) -->
(375)             </xsl:element>
(376)
(377)           </xsl:for-each>
(378)
(379)         <!-- end: div (owl-instances) -->
(380)         </xsl:element>
(381)
(382)         <xsl:element name="div">
(383)           <xsl:attribute name="id">
(384)             <xsl:text>document-src</xsl:text>
(385)           </xsl:attribute>
(386)
(387)           <xsl:variable name="src-body" select="xhtml:body/*"/>
(388)
(389)           <xsl:apply-templates select="$src-body" mode="src-body"/>
(390)
(391)         </xsl:element>
(392)       </xsl:element>
(393)     <!-- end: html -->
(394)     </xsl:element>
(395)
(396)   </xsl:template>
(397)
(398)   <xsl:template match="@*|node()" mode="src-body">
(399)     <xsl:copy copy-namespaces="no">
(400)       <xsl:apply-templates select="@*|node()" mode="src-body"/>
(401)     </xsl:copy>
(402)   </xsl:template>
(403)
(404)   <xsl:template match="xhtml:span" mode="src-body">
(405)     <xsl:if test="not(@rel)">
(406)       <xsl:element name="a">
(407)         <xsl:attribute name="name" select="concat(substring-after(@about,
      '#'), '_source')"/>
(408)       </xsl:element>
(409)       <xsl:element name="a">
(410)         <xsl:attribute name="href" select="concat('#', substring-
      after(@about, '#'))"/>
(411)         <xsl:element name="img">
(412)           <xsl:attribute name="src" select="'../favorites_16.gif'"/>
(413)         </xsl:element>
(414)         <xsl:value-of select="normalize-space(.)"/>
(415)       </xsl:element>
(416)     </xsl:if>
(417)   </xsl:template>
(418)
(419) </xsl:stylesheet>
```

Listing 62 – Erzeugung der RDFa-Verfolgbarkeitsdokumentation (create-GovObjTraces_RdfaDocument.xsl)

G.3 OWL-Ontologie

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(4)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(5)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(6)     xmlns:xhtml="http://www.w3.org/1999/xhtml"
(7)     xmlns="http://www.w3.org/1999/xhtml"
(8)     xmlns:owl="http://www.w3.org/2002/07/owl#"
(9)     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(10)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
```

```
(11)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(12)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(13)    xmlns:bewohnerparken="http://govobjects.thomasoff.de/bewohnerparken#"
(14)    xmlns:owl2prm_ontology="urn:owl2prm_ontology.ecore"
(15)    xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
(16)    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
(17)
(18)    <xsl:include href="commonGovObjTraces.xsl"/>
(19)
(20)    <xsl:output method="xhtml" version="1.0" encoding="UTF-8" indent="yes"/>
(21)
(22)    <xsl:param name="param-trace-rdf2prm-file"
      select="'../in/trace.owl2prm'"/>
(23)
(24)    <xsl:template match="/rdf:RDF">
(25)
(26)      <!-- Grundstruktur für xhtml-Dokumente -->
(27)      <xsl:element name="html" namespace="http://www.w3.org/1999/xhtml"
      inherit-namespaces="yes">
(28)
(29)        <xsl:element name="head">
(30)          <xsl:element name="meta">
(31)            <xsl:attribute name="http-equiv">
(32)              <xsl:text>content-type</xsl:text>
(33)            </xsl:attribute>
(34)            <xsl:attribute name="content">
(35)              <xsl:text>text/html; charset=ISO-8859-1</xsl:text>
(36)            </xsl:attribute>
(37)          </xsl:element>
(38)          <xsl:element name="title">
(39)            <xsl:text>GovObj Traceability: PRM - Ontology</xsl:text>
(40)          </xsl:element>
(41)        <!-- end: head -->
(42)        </xsl:element>
(43)
(44)        <xsl:element name="body">
(45)
(46)          <xsl:element name="div">
(47)
(48)            <xsl:attribute name="id">
(49)              <xsl:text>owl-ontology</xsl:text>
(50)            </xsl:attribute>
(51)
(52)            <xsl:element name="h1">Verfolgbarkeit der OWL-
      Ontologie</xsl:element>
(53)
(54)            <xsl:element name="p">
(55)              <xsl:text>Die folgenden Klassen, Object Properties und Datatype
      Properties sind in der Ontologie enthalten.</xsl:text>
(56)            </xsl:element>
(57)
(58)            <xsl:for-each select="//owl:Class[@rdf:about]">
(59)              <xsl:variable name="about" select="@rdf:about"/>
(60)              <xsl:variable name="anchor" select="substring-after($about,
      '#')"/>
(61)              <xsl:variable name="name" select="$anchor"/>
(62)              <xsl:variable name="label" select="govobj:rdf-name(.)"/>
(63)              <xsl:variable name="comment" select="rdfs:comment"/>
(64)
(65)              <xsl:element name="div">
(66)                <xsl:attribute name="class">
(67)                  <xsl:text>div-element</xsl:text>
(68)                </xsl:attribute>
(69)
(70)                <!-- Achnor für Navigation im Dokument anlegen -->
(71)                <xsl:element name="a">
(72)                  <xsl:attribute name="name">
(73)                    <xsl:value-of select="$anchor"/>
(74)                  </xsl:attribute>
(75)                </xsl:element>
(76)
(77)                <!-- Überschrift des Elementes anlegen -->
(78)                <xsl:element name="h2">
(79)                  <xsl:text>Klasse '</xsl:text>
```

```
(80)                  <xsl:value-of select="$label"/>
(81)                  <xsl:text>'</xsl:text>
(82)                </xsl:element>
(83)
(84)                <xsl:element name="div">
(85)                  <xsl:attribute name="class">
(86)                    <xsl:text>div-eigenschaften</xsl:text>
(87)                  </xsl:attribute>
(88)                  <xsl:element name="h3">
(89)                    <xsl:text>Eigenschaften</xsl:text>
(90)                  </xsl:element>
(91)
(92)                  <xsl:element name="p">Die Klasse hat die folgenden
      Eigenschaften:</xsl:element>
(93)
(94)                  <xsl:element name="ul">
(95)                    <xsl:attribute name="class">
(96)                      <xsl:text>ul-eigenschaft</xsl:text>
(97)                    </xsl:attribute>
(98)
(99)                    <xsl:element name="li">
(100)                     <xsl:attribute name="class">
(101)                       <xsl:text>li-eigenschaft</xsl:text>
(102)                     </xsl:attribute>
(103)                     <xsl:element name="span">
(104)                       <xsl:attribute name="class">
(105)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(106)                       </xsl:attribute>
(107)                       <xsl:text>Name:</xsl:text>
(108)                     </xsl:element>
(109)                     <xsl:element name="span">
(110)                       <xsl:attribute name="class">
(111)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(112)                       </xsl:attribute>
(113)                       <xsl:value-of select="$label"/>
(114)                     </xsl:element>
(115)                   <!-- end: li -->
(116)                   </xsl:element>
(117)
(118)                   <xsl:element name="li">
(119)                     <xsl:attribute name="class">
(120)                       <xsl:text>li-eigenschaft</xsl:text>
(121)                     </xsl:attribute>
(122)                     <xsl:element name="span">
(123)                       <xsl:attribute name="class">
(124)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(125)                       </xsl:attribute>
(126)                       <xsl:text>Oberklasse:</xsl:text>
(127)                     </xsl:element>
(128)                     <xsl:element name="span">
(129)                       <xsl:attribute name="class">
(130)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(131)                       </xsl:attribute>
(132)                       <xsl:if test="count(rdfs:subClassOf[@rdf:resource]) =
      0">
(133)                         <xsl:text>- Keine -</xsl:text>
(134)                       </xsl:if>
(135)                       <xsl:for-each select="rdfs:subClassOf[@rdf:resource]">
(136)                         <xsl:element name="a">
(137)                           <xsl:attribute name="href"
      select="@rdf:resource"/>
(138)                           <xsl:value-of select="substring-
      after(@rdf:resource, '#')"/>
(139)                         </xsl:element>
(140)                         <xsl:if test="position() != last()">
(141)                           <xsl:text>, </xsl:text>
(142)                         </xsl:if>
(143)                       </xsl:for-each>
(144)                     </xsl:element>
(145)                   <!-- end: li -->
(146)                   </xsl:element>
(147)
(148)                   <xsl:element name="li">
(149)                     <xsl:attribute name="class">
(150)                       <xsl:text>li-eigenschaft</xsl:text>
```

```
(151)                     </xsl:attribute>
(152)                     <xsl:element name="span">
(153)                       <xsl:attribute name="class">
(154)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(155)                       </xsl:attribute>
(156)                       <xsl:text>Beschreibung:</xsl:text>
(157)                     </xsl:element>
(158)                     <xsl:element name="span">
(159)                       <xsl:attribute name="class">
(160)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(161)                       </xsl:attribute>
(162)                       <xsl:value-of select="$comment"/>
(163)                     </xsl:element>
(164)                   <!-- end: li -->
(165)                   </xsl:element>
(166)
(167)                   <!-- end: ul -->
(168)                   </xsl:element>
(169)
(170)                 <!-- end: div - eigenschaften -->
(171)                 </xsl:element>
(172)
(173)                 <!-- Eigenschaften (Datatype Properties) -->
(174)                 <xsl:element name="div">
(175)                   <xsl:attribute name="class">
(176)                     <xsl:text>div-attribute</xsl:text>
(177)                   </xsl:attribute>
(178)
(179)                   <xsl:element name="h3">Datatype Properties</xsl:element>
(180)                   <xsl:element name="p">Die Klasse hat die folgenden
      Datatype Properties.</xsl:element>
(181)
(182)                   <!-- Prüfen, ob ohne Eigenschaften -->
(183)                   <xsl:if
      test="count(//owl:DatatypeProperty[rdfs:domain[@rdf:resource=$about]]) = 0">
(184)                     <xsl:element name="p">
(185)                       <xsl:text>- Keine -</xsl:text>
(186)                     </xsl:element>
(187)                   </xsl:if>
(188)
(189)                   <!-- Eigenschaft -->
(190)                   <xsl:for-each
      select="//owl:DatatypeProperty[rdfs:domain[@rdf:resource=$about]]">
(191)                     <xsl:element name="div">
(192)                       <xsl:attribute name="class">
(193)                         <xsl:text>div-attribut</xsl:text>
(194)                       </xsl:attribute>
(195)
(196)                       <xsl:element name="h4">
(197)                         <xsl:text>Data Property '</xsl:text>
(198)                         <xsl:value-of select="rdfs:label"/>
(199)                         <xsl:text>'</xsl:text>
(200)                       </xsl:element>
(201)
(202)                       <xsl:element name="ul">
(203)                         <xsl:attribute name="class">
(204)                           <xsl:text>ul-attribut</xsl:text>
(205)                         </xsl:attribute>
(206)
(207)                         <xsl:element name="li">
(208)                           <xsl:attribute name="class">
(209)                             <xsl:text>li-attribut</xsl:text>
(210)                           </xsl:attribute>
(211)                           <xsl:element name="span">
(212)                             <xsl:attribute name="class">
(213)                               <xsl:text>span-attribut-titel</xsl:text>
(214)                             </xsl:attribute>
(215)                             <xsl:text>Name:</xsl:text>
(216)                           </xsl:element>
(217)                           <xsl:element name="span">
(218)                             <xsl:attribute name="class">
(219)                               <xsl:text>span-attribut-wert</xsl:text>
(220)                             </xsl:attribute>
(221)                             <xsl:element name="a">
```

```
(222)                                <xsl:attribute name="href"
      select="@rdf:about"/>
(223)                                <xsl:value-of select="rdfs:label"/>
(224)                              </xsl:element>
(225)                            </xsl:element>
(226)                          <!-- end: li -->
(227)                          </xsl:element>
(228)
(229)                          <xsl:element name="li">
(230)                            <xsl:attribute name="class">
(231)                              <xsl:text>li-attribut</xsl:text>
(232)                            </xsl:attribute>
(233)                            <xsl:element name="span">
(234)                              <xsl:attribute name="class">
(235)                                <xsl:text>span-attribut-titel</xsl:text>
(236)                              </xsl:attribute>
(237)                              <xsl:text>Domain:</xsl:text>
(238)                            </xsl:element>
(239)                            <xsl:element name="span">
(240)                              <xsl:attribute name="class">
(241)                                <xsl:text>span-attribut-wert</xsl:text>
(242)                              </xsl:attribute>
(243)                              <xsl:for-each select="rdfs:domain">
(244)                                <xsl:element name="a">
(245)                                  <xsl:attribute name="href"
      select="@rdf:resource"/>
(246)                                  <xsl:value-of select="substring-
      after(@rdf:resource, '#')"/>
(247)                                </xsl:element>
(248)                                <xsl:if test="position() != last()">
(249)                                  <xsl:text>, </xsl:text>
(250)                                </xsl:if>
(251)                              </xsl:for-each>
(252)                            </xsl:element>
(253)                          <!-- end: li -->
(254)                          </xsl:element>
(255)
(256)                          <xsl:element name="li">
(257)                            <xsl:attribute name="class">
(258)                              <xsl:text>li-attribut</xsl:text>
(259)                            </xsl:attribute>
(260)                            <xsl:element name="span">
(261)                              <xsl:attribute name="class">
(262)                                <xsl:text>span-attribut-titel</xsl:text>
(263)                              </xsl:attribute>
(264)                              <xsl:text>Range:</xsl:text>
(265)                            </xsl:element>
(266)                            <xsl:element name="span">
(267)                              <xsl:attribute name="class">
(268)                                <xsl:text>span-attribut-wert</xsl:text>
(269)                              </xsl:attribute>
(270)                              <xsl:element name="a">
(271)                                <xsl:attribute name="href"
      select="rdfs:range/@rdf:resource"/>
(272)                                <xsl:value-of select="substring-
      after(rdfs:range/@rdf:resource, '#')"/>
(273)                              </xsl:element>
(274)                            </xsl:element>
(275)                          <!-- end: li -->
(276)                          </xsl:element>
(277)
(278)                          <xsl:element name="li">
(279)                            <xsl:attribute name="class">
(280)                              <xsl:text>li-attribut</xsl:text>
(281)                            </xsl:attribute>
(282)                            <xsl:element name="span">
(283)                              <xsl:attribute name="class">
(284)                                <xsl:text>span-attribut-titel</xsl:text>
(285)                              </xsl:attribute>
(286)                              <xsl:text>Beschreibung:</xsl:text>
(287)                            </xsl:element>
(288)                            <xsl:element name="span">
(289)                              <xsl:attribute name="class">
(290)                                <xsl:text>span-attribut-wert</xsl:text>
(291)                              </xsl:attribute>
```

```
(292)                             <xsl:value-of select="rdfs:comment"/>
(293)                           </xsl:element>
(294)                         <!-- end: li -->
(295)                         </xsl:element>
(296)
(297)                       <!-- end: ul - attribut -->
(298)                       </xsl:element>
(299)                     <!-- end: div - attribut -->
(300)                     </xsl:element>
(301)
(302)                   <!-- end: Attribute-->
(303)                   </xsl:for-each>
(304)
(305)                 <!-- end: div-attribute -->
(306)                 </xsl:element>
(307)
(308)                 <!-- Beziehungen (Object Properties) -->
(309)                 <xsl:element name="div">
(310)                   <xsl:attribute name="class">
(311)                     <xsl:text>div-attribute</xsl:text>
(312)                   </xsl:attribute>
(313)
(314)                   <xsl:element name="h3">Object Properties</xsl:element>
(315)                   <xsl:element name="p">Die Klasse hat die folgenden Object
      Properties.</xsl:element>
(316)
(317)                   <!-- Prüfen, ob ohne Eigenschaften -->
(318)                   <xsl:if
      test="count(//owl:ObjectProperty[rdfs:domain[@rdf:resource=$about]]) = 0">
(319)                     <xsl:element name="p">
(320)                       <xsl:text>- Keine -</xsl:text>
(321)                     </xsl:element>
(322)                   </xsl:if>
(323)
(324)                   <!-- Eigenschaft -->
(325)                   <xsl:for-each
      select="//owl:ObjectProperty[rdfs:domain[@rdf:resource=$about]]">
(326)                     <xsl:element name="div">
(327)                       <xsl:attribute name="class">
(328)                         <xsl:text>div-attribut</xsl:text>
(329)                       </xsl:attribute>
(330)
(331)                       <xsl:element name="h4">
(332)                         <xsl:text>Object Property '</xsl:text>
(333)                         <xsl:value-of select="rdfs:label"/>
(334)                         <xsl:text>'</xsl:text>
(335)                       </xsl:element>
(336)
(337)                       <xsl:element name="ul">
(338)                         <xsl:attribute name="class">
(339)                           <xsl:text>ul-attribut</xsl:text>
(340)                         </xsl:attribute>
(341)
(342)                         <xsl:element name="li">
(343)                           <xsl:attribute name="class">
(344)                             <xsl:text>li-attribut</xsl:text>
(345)                           </xsl:attribute>
(346)                           <xsl:element name="span">
(347)                             <xsl:attribute name="class">
(348)                               <xsl:text>span-attribut-titel</xsl:text>
(349)                             </xsl:attribute>
(350)                             <xsl:text>Name:</xsl:text>
(351)                           </xsl:element>
(352)                           <xsl:element name="span">
(353)                             <xsl:attribute name="class">
(354)                               <xsl:text>span-attribut-wert</xsl:text>
(355)                             </xsl:attribute>
(356)                             <xsl:element name="a">
(357)                               <xsl:attribute name="href"
      select="@rdf:about"/>
(358)                               <xsl:value-of select="rdfs:label"/>
(359)                             </xsl:element>
(360)                           </xsl:element>
(361)                         <!-- end: li -->
(362)                         </xsl:element>
```

```
(363)
(364)                         <xsl:element name="li">
(365)                           <xsl:attribute name="class">
(366)                             <xsl:text>li-attribut</xsl:text>
(367)                           </xsl:attribute>
(368)                           <xsl:element name="span">
(369)                             <xsl:attribute name="class">
(370)                               <xsl:text>span-attribut-titel</xsl:text>
(371)                             </xsl:attribute>
(372)                             <xsl:text>Domain:</xsl:text>
(373)                           </xsl:element>
(374)                           <xsl:element name="span">
(375)                             <xsl:attribute name="class">
(376)                               <xsl:text>span-attribut-wert</xsl:text>
(377)                             </xsl:attribute>
(378)                             <xsl:value-of
      select="rdfs:domain/@rdf:resource"/>
(379)                           </xsl:element>
(380)                         <!-- end: li -->
(381)                         </xsl:element>
(382)
(383)                         <xsl:element name="li">
(384)                           <xsl:attribute name="class">
(385)                             <xsl:text>li-attribut</xsl:text>
(386)                           </xsl:attribute>
(387)                           <xsl:element name="span">
(388)                             <xsl:attribute name="class">
(389)                               <xsl:text>span-attribut-titel</xsl:text>
(390)                             </xsl:attribute>
(391)                             <xsl:text>Range:</xsl:text>
(392)                           </xsl:element>
(393)                           <xsl:element name="span">
(394)                             <xsl:attribute name="class">
(395)                               <xsl:text>span-attribut-wert</xsl:text>
(396)                             </xsl:attribute>
(397)                             <xsl:value-of
      select="rdfs:range/@rdf:resource"/>
(398)                           </xsl:element>
(399)                         <!-- end: li -->
(400)                         </xsl:element>
(401)
(402)                         <xsl:if test="owl:inverseOf">
(403)                           <xsl:element name="li">
(404)                             <xsl:attribute name="class">
(405)                               <xsl:text>li-attribut</xsl:text>
(406)                             </xsl:attribute>
(407)                             <xsl:element name="span">
(408)                               <xsl:attribute name="class">
(409)                                 <xsl:text>span-attribut-titel</xsl:text>
(410)                               </xsl:attribute>
(411)                               <xsl:text>Name:</xsl:text>
(412)                             </xsl:element>
(413)                             <xsl:element name="span">
(414)                               <xsl:attribute name="class">
(415)                                 <xsl:text>span-attribut-wert</xsl:text>
(416)                               </xsl:attribute>
(417)                               <xsl:element name="a">
(418)                                 <xsl:attribute name="href"
      select="owl:inverseOf/@rdf:resource"/>
(419)                                 <xsl:value-of select="substring-
      before(owl:inverseOf/@rdf:resource, '#')"/>
(420)                               </xsl:element>
(421)                             </xsl:element>
(422)                           <!-- end: li -->
(423)                           </xsl:element>
(424)                         </xsl:if>
(425)
(426)                         <xsl:element name="li">
(427)                           <xsl:attribute name="class">
(428)                             <xsl:text>li-attribut</xsl:text>
(429)                           </xsl:attribute>
(430)                           <xsl:element name="span">
(431)                             <xsl:attribute name="class">
(432)                               <xsl:text>span-attribut-titel</xsl:text>
(433)                             </xsl:attribute>
```

```
(434)                             <xsl:text>Beschreibung:</xsl:text>
(435)                           </xsl:element>
(436)                           <xsl:element name="span">
(437)                             <xsl:attribute name="class">
(438)                               <xsl:text>span-attribut-wert</xsl:text>
(439)                             </xsl:attribute>
(440)                             <xsl:value-of select="rdfs:comment"/>
(441)                           </xsl:element>
(442)                         <!-- end: li -->
(443)                         </xsl:element>
(444)
(445)                       <!-- end: ul - attribut -->
(446)                       </xsl:element>
(447)                     <!-- end: div - attribut -->
(448)                     </xsl:element>
(449)
(450)                   <!-- end: Attribute-->
(451)                   </xsl:for-each>
(452)
(453)                 <!-- end: div-attribute -->
(454)                 </xsl:element>
(455)
(456)               <!-- Zielelement(e) im PRM -->
(457)               <xsl:element name="div">
(458)                 <xsl:attribute name="class">
(459)                   <xsl:text>div-verfolgbarkeit-vorwaerts</xsl:text>
(460)                 </xsl:attribute>
(461)
(462)                 <xsl:element name="h3">
(463)                   <xsl:text>Zielelemente im PRM</xsl:text>
(464)                 </xsl:element>
(465)
(466)                 <xsl:element name="p">Das Element wird in den folgenden
      Abbildungsregeln verwendet.</xsl:element>
(467)
(468)                 <xsl:variable name="id" select="@rdf:about"/>
(469)                 <xsl:variable name="document-trace-rdf2prm"
      select="document($param-trace-rdf2prm-file)"/>
(470)
(471)                 <xsl:if test="count($document-trace-
      rdf2prm//owl2prm_ontology:*[child::*[contains(@href, $id)]]) = 0">
(472)                   <xsl:element name="p">- Keine -</xsl:element>
(473)                 </xsl:if>
(474)
(475)                 <xsl:for-each select="$document-trace-
      rdf2prm//owl2prm_ontology:*[child::*[contains(@href, $id)]]">
(476)
(477)                 <xsl:element name="h4">Abbildungsregel '<xsl:value-of
      select="local-name(.)"/>'</xsl:element>
(478)
(479)                 <xsl:element name="p">Das Element wird in die folgenden
      Elemente des PRM überführt:</xsl:element>
(480)
(481)                 <xsl:element name="ul">
(482)                   <xsl:attribute name="class">
(483)                     <xsl:text>ul-zielelemente-prm</xsl:text>
(484)                   </xsl:attribute>
(485)
(486)                   <xsl:for-each select="child::*[starts-with(name(),
      'dst')]">
(487)                     <xsl:variable name="file-prm" select="substring-
      before(@href, '#')"/>
(488)                     <xsl:variable name="id-prm" select="substring-
      after(@href, '#')"/>
(489)                     <xsl:variable name="document-prm"
      select="document($file-prm)"/>
(490)                     <xsl:variable name="elem-prm" select="$document-
      prm//*[@xmi:id = $id-prm]"/>
(491)                     <xsl:variable name="label-prm"
      select="govobj:label($elem-prm, $id-prm)"/>
(492)
(493)                     <xsl:element name="li">
(494)                       <xsl:attribute name="class">
(495)                         <xsl:text>li-zielelement-prm</xsl:text>
(496)                       </xsl:attribute>
```

```
(497)                       <xsl:element name="a">
(498)                         <xsl:attribute name="href">
(499)                           <xsl:value-of select="replace(@href, '.uml',
      '.html')"/>
(500)                         </xsl:attribute>
(501)                         <xsl:value-of select="$label-prm"/>
(502)                       </xsl:element>
(503)                     </xsl:element>
(504)
(505)                   <!--  end: child -->
(506)                   </xsl:for-each>
(507)
(508)                 <!-- end: ul-zielelemente-prm -->
(509)                 </xsl:element>
(510)
(511)                 <!-- end: rdfa2prm_annotation -->
(512)                 </xsl:for-each>
(513)
(514)               <!-- end: div-verfolgbarkeit-vorwaerts -->
(515)               </xsl:element>
(516)
(517)             <!--  end: div (div-element) -->
(518)             </xsl:element>
(519)
(520)           </xsl:for-each>
(521)
(522)         <!-- end: div (owl-classes) -->
(523)         </xsl:element>
(524)
(525)       <!-- end: body -->
(526)       </xsl:element>
(527)
(528)     <!-- end: html -->
(529)     </xsl:element>
(530)
(531)   </xsl:template>
(532)
(533) </xsl:stylesheet>
```

Listing 63 – Verfolgbarkeitsdokumentation für die OWL-Ontologie (create-GovObjTraces_OwlOntology.xsl)

G.4 Pre-Requirements Model

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(4)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(5)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(6)     xmlns:html="http://www.w3.org/TR/REC-html40"
(7)     xmlns:xhtml="http://www.w3.org/1999/xhtml"
(8)     xmlns="http://www.w3.org/1999/xhtml"
(9)     xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
(10)    xmlns:dc="http://purl.org/dc/elements/1.1/"
(11)
      xmlns:ontomat="http://annotation.semanticweb.org/ontologies/cream/ontomat#"
(12)    xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(13)    xmlns:rss="http://purl.org/rss/1.0/"
(14)    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
(15)    xmlns:govobjects="http://govobjects.thomasoff.de/bewohnerparken#"
(16)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(17)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(18)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(19)    xmlns:owl="http://www.w3.org/2002/07/owl#"
(20)    xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
(21)    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(22)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(23)    xmlns:rdfodm="urn:rdf2odm.ecore"
(24)    xmlns:xmi="http://www.omg.org/XMI"
(25)    xmlns:xmi21="http://schema.omg.org/spec/XMI/2.1"
(26)    xmlns:saxon="http://saxon.sf.net/"
(27)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
(28)
      xmlns:OwlUmlProfile="http:///schemas/OwlUmlProfile/_QwFB4CezEeC9_fKmAwBwrw/2
      8"
```

```
(29)    extension-element-prefixes="saxon"
(30)    xmlns:owl2prm_annotation="urn:owl2prm_annotation.ecore">
(31)
(32)    <xsl:include href="commonGovObjTraces.xsl"/>
(33)
(34)    <xsl:output method="xhtml" version="1.0" encoding="UTF-8" indent="yes"/>
(35)
(36)    <xsl:param name="param-prm-model-file" select="'../Model/prm_stvg_vwv-
        stvo.uml'"/>
(37)    <xsl:param name="param-trace-prm2cim-file1"
        select="'../in/trace.prm2cim_anf'"/>
(38)    <xsl:param name="param-trace-prm2cim-file2"
        select="'../in/trace.prm2cim_proz'"/>
(39)    <xsl:param name="param-trace-rdf2prm-file3"
        select="'../in/trace.rdf2prm'"/>
(40)
(41)    <xsl:param name="param-prefix-rdf" select="'Rdf'"/>
(42)    <xsl:param name="param-prefix-prm" select="'prm'"/>
(43)    <xsl:param name="param-prefix-cim" select="'cim'"/>
(44)
(45)    <xsl:variable name="prm-model-file" select="replace($param-prm-model-file,
        '\\', '/')"/>
(46)
(47)    <xsl:variable name="document-trace-rdf2prm" select="document($param-trace-
        rdf2prm-file3)"/>
(48)    <!-- Hier zunächst nur die Betrachtung der Traces zum Anforderungsmodell -
        ->
(49)    <xsl:variable name="document-trace-prm2cim" select="document($param-trace-
        prm2cim-file1)"/>
(50)
(51)    <xsl:variable name="elements-missing" select="0" saxon:assignable="yes"/>
(52)    <xsl:variable name="transformation-forward" select="0"
        saxon:assignable="yes"/>
(53)    <xsl:variable name="transformation-backward" select="0"
        saxon:assignable="yes"/>
(54)
(55)    <xsl:template match="/xmi21:XMI">
(56)      <!-- Grundstruktur für xhtml-Dokumente -->
(57)      <xsl:element name="html" namespace="http://www.w3.org/1999/xhtml"
        inherit-namespaces="yes">
(58)        <xsl:element name="head">
(59)          <xsl:element name="meta">
(60)            <xsl:attribute name="http-equiv">
(61)              <xsl:text>content-type</xsl:text>
(62)            </xsl:attribute>
(63)            <xsl:attribute name="content">
(64)              <xsl:text>text/html; charset=UTF-8</xsl:text>
(65)            </xsl:attribute>
(66)          </xsl:element>
(67)          <xsl:element name="title">
(68)            <xsl:text>GovObj Traceability: PRM</xsl:text>
(69)          </xsl:element>
(70)          <xsl:element name="script">
(71)            <xsl:attribute name="type" select="'text/javascript'"/>
(72)            <xsl:attribute name="src" select="'../govobj-filtern.js'"/>
(73)          </xsl:element>
(74)        </xsl:element>
(75)        <xsl:element name="body">
(76)          <xsl:attribute name="onload" select="'resetForm()'"/>
(77)
(78)          <xsl:element name="div">
(79)            <xsl:attribute name="id">
(80)              <xsl:text>uml-instances</xsl:text>
(81)            </xsl:attribute>
(82)
(83)            <!-- Im PRM vorhandene Modellelemente -->
(84)            <xsl:element name="h1">Verfolgbarkeit von Modellelementen des
        PRM</xsl:element>
(85)
(86)            <xsl:element name="div">
(87)              <xsl:attribute name="name" select="'filter'"/>
(88)              <xsl:attribute name="id" select="'div-filter'"/>
(89)
(90)              <xsl:element name="h2">
(91)                <xsl:text>Modellelemente filtern</xsl:text>
```

```
(92)              </xsl:element>
(93)
(94)              <xsl:element name="form">
(95)                <xsl:attribute name="name" select="'filter'"/>
(96)                <xsl:element name="div">
(97)                  <xsl:attribute name="name" select="'filterkriterien'"/>
(98)                  <xsl:element name="div">
(99)                    <xsl:attribute name="name" select="'fk_name'"/>
(100)                   <xsl:element name="span">
(101)                     <xsl:attribute name="name" select="'fk_name_label'"/>
(102)                     <xsl:text>Name</xsl:text>
(103)                   </xsl:element>
(104)                   <xsl:element name="br"/>
(105)                   <xsl:element name="select">
(106)                     <xsl:attribute name="name" select="'fk_name_select'"/>
(107)                     <xsl:attribute name="title" select="'Name'"/>
(108)                     <xsl:attribute name="size" select="'4'"/>
(109)                     <xsl:attribute name="multiple" select="'multiple'"/>
(110)                   </xsl:element>
(111)                 </xsl:element>
(112)
(113)                 <xsl:element name="div">
(114)                   <xsl:attribute name="name" select="'fk_typ'"/>
(115)                   <xsl:element name="span">
(116)                     <xsl:attribute name="name" select="'fk_typ_label'"/>
(117)                     <xsl:text>Typ</xsl:text>
(118)                   </xsl:element>
(119)                   <xsl:element name="br"/>
(120)                   <xsl:element name="select">
(121)                     <xsl:attribute name="name" select="'fk_typ_select'"/>
(122)                     <xsl:attribute name="title" select="'Typ'"/>
(123)                     <xsl:attribute name="size" select="'4'"/>
(124)                     <xsl:attribute name="multiple" select="'multiple'"/>
(125)                   </xsl:element>
(126)                 </xsl:element>
(127)
(128)                 <xsl:element name="div">
(129)                   <xsl:attribute name="name" select="'fk_abbregel'"/>
(130)                   <xsl:element name="span">
(131)                     <xsl:attribute name="name"
      select="'fk_abbregel_label'"/>
(132)                     <xsl:text>Abbildungsregel</xsl:text>
(133)                   </xsl:element>
(134)                   <xsl:element name="br"/>
(135)                   <xsl:element name="select">
(136)                     <xsl:attribute name="name"
      select="'fk_abbregel_select'"/>
(137)                     <xsl:attribute name="title"
      select="'Abbildungsregel(n)'"/>
(138)                     <xsl:attribute name="size" select="'4'"/>
(139)                     <xsl:attribute name="multiple" select="'multiple'"/>
(140)                   </xsl:element>
(141)                 </xsl:element>
(142)
(143)               <!--  end: div - filterkriterien -->
(144)               </xsl:element>
(145)               <xsl:element name="div">
(146)                 <xsl:attribute name="name" select="'filteraktionen'"/>
(147)                 <xsl:element name="input">
(148)                   <xsl:attribute name="type" select="'button'"/>
(149)                   <xsl:attribute name="value" select="'Zurücksetzen'"/>
(150)                   <xsl:attribute name="onclick" select="'resetFilter()'"/>
(151)                 </xsl:element>
(152)                 <xsl:element name="input">
(153)                   <xsl:attribute name="type" select="'button'"/>
(154)                   <xsl:attribute name="value" select="'Filtern'"/>
(155)                   <xsl:attribute name="onclick" select="'filtern()'"/>
(156)                 </xsl:element>
(157)               </xsl:element>
(158)             <!--  end: form - filter -->
(159)             </xsl:element>
(160)           <!--  end: div - filter -->
(161)           </xsl:element>
(162)
(163)           <xsl:element name="div">
```

```
(164)             <xsl:attribute name="name" select="'div-filtermessage'"/>
(165)             <xsl:element name="span">
(166)               <xsl:attribute name="name" select="'span-filtermessage'"/>
(167)               <xsl:attribute name="id" select="'span-filtermessage'"/>
(168)             </xsl:element>
(169)           </xsl:element>
(170)
(171)           <xsl:element name="p">
(172)             <xsl:text>Die folgenden Modellemente sind Bestandteil des
      PRM.</xsl:text>
(173)           </xsl:element>
(174)
(175)           <xsl:for-each select="//uml:Model|//packagedElement">
(176)
(177)             <xsl:variable name="id" select="@xmi21:id"/>
(178)
(179)             <xsl:variable name="type">
(180)               <xsl:choose>
(181)                 <xsl:when test="@xmi21:type">
(182)                   <xsl:value-of select="@xmi21:type"/>
(183)                 </xsl:when>
(184)                 <xsl:otherwise>
(185)                   <xsl:value-of select="name()"/>
(186)                 </xsl:otherwise>
(187)               </xsl:choose>
(188)               <xsl:if test="classifier[@xmi21:type='uml:Association']">
(189)                 <xsl:text> Link</xsl:text>
(190)               </xsl:if>
(191)             </xsl:variable>
(192)             <xsl:variable name="name" select="@name"/>
(193)             <xsl:variable name="label">
(194)               <xsl:value-of select="$type"/>
(195)               <xsl:text> '</xsl:text>
(196)               <xsl:value-of select="$name"/>
(197)               <xsl:text>'</xsl:text>
(198)             </xsl:variable>
(199)             <xsl:variable name="documentation">
(200)               <xsl:choose>
(201)                 <xsl:when
      test="eAnnotations[1][@source='http://www.topcased.org/documentation']">
(202)                   <xsl:value-of select="eAnnotations[1]/details/@value"/>
(203)                 </xsl:when>
(204)                 <xsl:otherwise>
(205)                   <xsl:text>- Keine -</xsl:text>
(206)                 </xsl:otherwise>
(207)               </xsl:choose>
(208)             </xsl:variable>
(209)
(210)             <xsl:element name="div">
(211)               <xsl:attribute name="class">
(212)                 <xsl:text>div-element</xsl:text>
(213)               </xsl:attribute>
(214)
(215)               <!-- Achnor für Navigation im Dokument anlegen -->
(216)               <xsl:element name="a">
(217)                 <xsl:attribute name="name">
(218)                   <xsl:value-of select="$id"/>
(219)                 </xsl:attribute>
(220)               </xsl:element>
(221)
(222)               <!-- Filtereigenschaften -->
(223)               <xsl:element name="div">
(224)                 <xsl:attribute name="style" select="'{display: block;}'"/>
(225)                 <xsl:attribute name="class" select="'div-filterkriterien'"/>
(226)                 <xsl:element name="span">
(227)                   <xsl:attribute name="name" select="'fk_name_value'"/>
(228)                   <xsl:value-of select="$label"/>
(229)                 </xsl:element>
(230)                 <xsl:element name="span">
(231)                   <xsl:attribute name="name" select="'fk_typ_value'"/>
(232)                   <xsl:value-of select="$type"/>
(233)                 </xsl:element>
(234)                 <xsl:for-each select="$document-trace-
      prm2cim/xmi:XMI/node()[child::*[substring-after(@href, '#') = $id]]">
(235)                   <xsl:element name="span">
```

```
(236)                     <xsl:attribute name="name" select="'fk_abbregel_multi-
      value'"/>
(237)                       <xsl:value-of select="name()"/>
(238)                   </xsl:element>
(239)                 </xsl:for-each>
(240)                 <xsl:for-each select="$document-trace-
      rdf2prm//owl2prm_annotation:*[child::*[@href=$id]]">
(241)                   <xsl:element name="span">
(242)                     <xsl:attribute name="name" select="'fk_abbregel_multi-
      value'"/>
(243)                       <xsl:value-of select="local-name(.)"/>
(244)                   </xsl:element>
(245)                 </xsl:for-each>
(246)               </xsl:element>
(247)
(248)               <!-- Überschrift des Elementes anlegen -->
(249)               <xsl:element name="h2">
(250)                 <xsl:value-of select="$label"/>
(251)               </xsl:element>
(252)
(253)               <xsl:element name="div">
(254)                 <xsl:attribute name="class">
(255)                   <xsl:text>div-eigenschaften</xsl:text>
(256)                 </xsl:attribute>
(257)                 <xsl:element name="h3">
(258)                   <xsl:text>Eigenschaften</xsl:text>
(259)                 </xsl:element>
(260)
(261)                 <xsl:element name="p">Das Element hat die folgenden
      Eigenschaften:</xsl:element>
(262)
(263)                 <xsl:element name="ul">
(264)                   <xsl:attribute name="class">
(265)                     <xsl:text>ul-eigenschaft</xsl:text>
(266)                   </xsl:attribute>
(267)                   <xsl:element name="li">
(268)                     <xsl:attribute name="class">
(269)                       <xsl:text>li-eigenschaft</xsl:text>
(270)                     </xsl:attribute>
(271)                     <xsl:element name="span">
(272)                       <xsl:attribute name="class">
(273)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(274)                       </xsl:attribute>
(275)                       <xsl:text>Name:</xsl:text>
(276)                     </xsl:element>
(277)                     <xsl:element name="span">
(278)                       <xsl:attribute name="class">
(279)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(280)                       </xsl:attribute>
(281)                       <xsl:value-of select="$name"/>
(282)                     </xsl:element>
(283)                   </xsl:element>
(284)
(285)                   <xsl:element name="li">
(286)                     <xsl:attribute name="class">
(287)                       <xsl:text>li-eigenschaft</xsl:text>
(288)                     </xsl:attribute>
(289)                     <xsl:element name="span">
(290)                       <xsl:attribute name="class">
(291)                         <xsl:text>span-eigenschaft-titel</xsl:text>
(292)                       </xsl:attribute>
(293)                       <xsl:text>Typ:</xsl:text>
(294)                     </xsl:element>
(295)                     <xsl:element name="span">
(296)                       <xsl:attribute name="class">
(297)                         <xsl:text>span-eigenschaft-wert</xsl:text>
(298)                       </xsl:attribute>
(299)                       <xsl:value-of select="$type"/>
(300)                     </xsl:element>
(301)                   </xsl:element>
(302)
(303)                   <xsl:element name="li">
(304)                     <xsl:attribute name="class">
(305)                       <xsl:text>li-eigenschaft</xsl:text>
(306)                     </xsl:attribute>
```

```
(307)                       <xsl:element name="span">
(308)                         <xsl:attribute name="class">
(309)                           <xsl:text>span-eigenschaft-titel</xsl:text>
(310)                         </xsl:attribute>
(311)                         <xsl:text>ID:</xsl:text>
(312)                       </xsl:element>
(313)                       <xsl:element name="span">
(314)                         <xsl:attribute name="class">
(315)                           <xsl:text>span-eigenschaft-wert</xsl:text>
(316)                         </xsl:attribute>
(317)                         <xsl:value-of select="$id"/>
(318)                       </xsl:element>
(319)                     </xsl:element>
(320)
(321)                     <xsl:element name="li">
(322)                       <xsl:attribute name="class">
(323)                         <xsl:text>li-eigenschaft</xsl:text>
(324)                       </xsl:attribute>
(325)                       <xsl:element name="span">
(326)                         <xsl:attribute name="class">
(327)                           <xsl:text>span-eigenschaft-titel</xsl:text>
(328)                         </xsl:attribute>
(329)                         <xsl:text>Dokumentation:</xsl:text>
(330)                       </xsl:element>
(331)                       <xsl:element name="span">
(332)                         <xsl:attribute name="class">
(333)                           <xsl:text>span-eigenschaft-wert</xsl:text>
(334)                         </xsl:attribute>
(335)                         <xsl:value-of select="$documentation"/>
(336)                       </xsl:element>
(337)                     </xsl:element>
(338)
(339)                     <!-- end: ul -->
(340)                     </xsl:element>
(341)
(342)                 <!-- end: div - eigenschaften -->
(343)                 </xsl:element>
(344)
(345)                 <!-- Attribute -->
(346)                 <xsl:if test="classifier/@xmi21:type='uml:Class'">
(347)                   <xsl:element name="div">
(348)                     <xsl:attribute name="class">
(349)                       <xsl:text>div-attribute</xsl:text>
(350)                     </xsl:attribute>
(351)
(352)                     <xsl:element name="h3">Attribute</xsl:element>
(353)                     <xsl:element name="p">Die Instanz hat die folgenden
      Werteausprägungen von Attributen.</xsl:element>
(354)
(355)                     <!-- Prüfen, ob ohne Attribute -->
(356)                     <xsl:if test="count(slot) = 0">
(357)                       <xsl:element name="p">
(358)                         <xsl:text>- Keine -</xsl:text>
(359)                       </xsl:element>
(360)                     </xsl:if>
(361)
(362)                     <!-- Attribut -->
(363)                     <xsl:for-each select="slot">
(364)                       <xsl:element name="div">
(365)                         <xsl:attribute name="class">
(366)                           <xsl:text>div-attribut</xsl:text>
(367)                         </xsl:attribute>
(368)
(369)                         <xsl:variable name="prm-attribut-name"
      select="substring-before(substring-after(definingFeature/@href, '#'),
      '_attribute')"/>
(370)                         <xsl:variable name="prm-attribut-type"
      select="value/@xmi21:type"/>
(371)                         <xsl:variable name="prm-attribut-wert"
      select="value/@value"/>
(372)
(373)                         <xsl:element name="h4">
(374)                           <xsl:text>Attribut '</xsl:text>
(375)                           <xsl:value-of select="$prm-attribut-name"/>
(376)                           <xsl:text>'</xsl:text>
```

```
(377)                       </xsl:element>
(378)
(379)                       <xsl:element name="ul">
(380)                         <xsl:attribute name="class">
(381)                           <xsl:text>ul-attribut</xsl:text>
(382)                         </xsl:attribute>
(383)
(384)                         <!-- Name -->
(385)                         <xsl:element name="li">
(386)                           <xsl:attribute name="class">
(387)                             <xsl:text>li-attribut</xsl:text>
(388)                           </xsl:attribute>
(389)                           <xsl:element name="span">
(390)                             <xsl:attribute name="class">
(391)                               <xsl:text>span-attribut-titel</xsl:text>
(392)                             </xsl:attribute>
(393)                             <xsl:text>Name:</xsl:text>
(394)                           </xsl:element>
(395)                           <xsl:element name="span">
(396)                             <xsl:attribute name="class">
(397)                               <xsl:text>span-attribut-wert</xsl:text>
(398)                             </xsl:attribute>
(399)                             <xsl:value-of select="$prm-attribut-name"/>
(400)                           </xsl:element>
(401)                         <!-- end: li -->
(402)                         </xsl:element>
(403)
(404)                         <!-- Datentyp -->
(405)                         <xsl:element name="li">
(406)                           <xsl:attribute name="class">
(407)                             <xsl:text>li-attribut</xsl:text>
(408)                           </xsl:attribute>
(409)                           <xsl:element name="span">
(410)                             <xsl:attribute name="class">
(411)                               <xsl:text>span-attribut-titel</xsl:text>
(412)                             </xsl:attribute>
(413)                             <xsl:text>Datentyp:</xsl:text>
(414)                           </xsl:element>
(415)                           <xsl:element name="span">
(416)                             <xsl:attribute name="class">
(417)                               <xsl:text>span-attribut-wert</xsl:text>
(418)                             </xsl:attribute>
(419)                             <xsl:value-of select="$prm-attribut-type"/>
(420)                           </xsl:element>
(421)                         <!-- end: li -->
(422)                         </xsl:element>
(423)
(424)                         <!-- Wert -->
(425)                         <xsl:element name="li">
(426)                           <xsl:attribute name="class">
(427)                             <xsl:text>li-attribut</xsl:text>
(428)                           </xsl:attribute>
(429)                           <xsl:element name="span">
(430)                             <xsl:attribute name="class">
(431)                               <xsl:text>span-attribut-titel</xsl:text>
(432)                             </xsl:attribute>
(433)                             <xsl:text>Wert:</xsl:text>
(434)                           </xsl:element>
(435)                           <xsl:element name="span">
(436)                             <xsl:attribute name="class">
(437)                               <xsl:text>span-attribut-wert</xsl:text>
(438)                             </xsl:attribute>
(439)                             <xsl:choose>
(440)                               <xsl:when test="string-length($prm-attribut-
      wert) > 0">
(441)                                 <xsl:text>'</xsl:text>
(442)                                 <xsl:value-of select="$prm-attribut-wert"/>
(443)                                 <xsl:text>'</xsl:text>
(444)                               </xsl:when>
(445)                               <xsl:otherwise>
(446)                                 <xsl:text>- Undefiniert -</xsl:text>
(447)                               </xsl:otherwise>
(448)                             </xsl:choose>
(449)                           </xsl:element>
(450)                         <!-- end: li -->
```

```
(451)                          </xsl:element>
(452)
(453)                        <!-- end: ul - attribut -->
(454)                        </xsl:element>
(455)                      <!-- end: div - attribut -->
(456)                      </xsl:element>
(457)
(458)                    <!-- end: attribut -->
(459)                    </xsl:for-each>
(460)
(461)                  <!-- end: div-attribute-->
(462)                  </xsl:element>
(463)                </xsl:if>
(464)
(465)                <!-- Beziehungen -->
(466)                <xsl:element name="div">
(467)                  <xsl:attribute name="class">
(468)                    <xsl:text>div-beziehungen</xsl:text>
(469)                  </xsl:attribute>
(470)
(471)                  <xsl:element name="h3">Assoziationen</xsl:element>
(472)                  <xsl:element name="p">Das Individuum hat die folgenden
      Ausprägungen von Assoziationen.</xsl:element>
(473)
(474)                  <!-- Prüfen, ob ohne Beziehungen -->
(475)                  <xsl:if
      test="count(//packagedElement[classifier/@xmi21:type='uml:Association'][slot
      /value/@instance=$id]) = 0">
(476)                    <xsl:element name="p">
(477)                      <xsl:text>- Keine -</xsl:text>
(478)                    </xsl:element>
(479)                  </xsl:if>
(480)
(481)                <!-- Beziehung -->
(482)                <xsl:for-each
      select="//packagedElement[classifier/@xmi21:type='uml:Association'][slot/val
      ue/@instance=$id]">
(483)                    <xsl:element name="div">
(484)                    <xsl:attribute name="class">
(485)                      <xsl:text>div-beziehung</xsl:text>
(486)                    </xsl:attribute>
(487)
(488)                    <xsl:variable name="prm-beziehung-id" select="@xmi21:id"/>
(489)                    <xsl:variable name="prm-beziehung-name" select="@name"/>
(490)                    <xsl:variable name="prm-beziehung-ziel-id"
      select="slot[not(value/@instance=$id)]/value/@instance"/>
(491)                    <xsl:variable name="prm-beziehung-ziel-elem"
      select="//packagedElement[@xmi21:id = $prm-beziehung-ziel-id]"/>
(492)
(493)                    <xsl:element name="h4">
(494)                      <xsl:text>Assozitation '</xsl:text>
(495)                      <xsl:value-of select="$prm-beziehung-name"/>
(496)                      <xsl:text>'</xsl:text>
(497)                    </xsl:element>
(498)
(499)                    <xsl:element name="p">
(500)                      <xsl:text>Die Instanz '</xsl:text>
(501)                      <xsl:value-of select="$name"/>
(502)                      <xsl:text>' hat die Beziehung '</xsl:text>
(503)                      <xsl:element name="a">
(504)                        <xsl:attribute name="href">
(505)                          <xsl:value-of select="concat('#', $prm-beziehung-
      id)"/>
(506)                        </xsl:attribute>
(507)                        <xsl:value-of select="$prm-beziehung-name"/>
(508)                      </xsl:element>
(509)                      <xsl:text>' zur Instanz '</xsl:text>
(510)                      <xsl:choose>
(511)                        <xsl:when test="$prm-beziehung-ziel-elem/@name">
(512)                          <xsl:element name="a">
(513)                            <xsl:attribute name="href">
(514)                              <xsl:value-of select="concat('#', $prm-
      beziehung-ziel-id)"/>
(515)                            </xsl:attribute>
```

```
(516)                           <xsl:value-of select="$prm-beziehung-ziel-
      elem/@name"/>
(517)                         </xsl:element>
(518)                       </xsl:when>
(519)                       <xsl:otherwise>
(520)                         <xsl:text>- Undefiniert -</xsl:text>
(521)                       </xsl:otherwise>
(522)                     </xsl:choose>
(523)                     <xsl:text>'</xsl:text>
(524)
(525)                   <!-- End: p -->
(526)                   </xsl:element>
(527)
(528)                 <!-- end: div - beziehung -->
(529)                 </xsl:element>
(530)
(531)               <!-- end: beziehung -->
(532)               </xsl:for-each>
(533)
(534)               <!-- end: div-eigenschaften -->
(535)               </xsl:element>
(536)
(537)               <!-- Verfolgbarkeit - Rückwärts zu Annotationen -->
(538)               <xsl:element name="div">
(539)                 <xsl:attribute name="class">
(540)                   <xsl:text>div-verfolgbarkeit-rueckwaerts</xsl:text>
(541)                 </xsl:attribute>
(542)
(543)                 <xsl:element name="h3">Annotationen im
      Ausgangsdokument</xsl:element>
(544)                 <xsl:element name="p">Das Element ist aus der/den folgenden
      Transformation(en) hervorgegangen.</xsl:element>
(545)
(546)                 <xsl:for-each select="$document-trace-
      rdf2prm//rdfodm:InstanceSpecification[substring-after(dstIndv/@href, '#') =
      $id]">
(547)                   <xsl:element name="div">
(548)                     <xsl:attribute name="class">
(549)                       <xsl:text>div-verfolgbarkeit-transformation</xsl:text>
(550)                     </xsl:attribute>
(551)                     <xsl:element name="h4">
(552)                       <xsl:value-of select="name()"/>
(553)                     </xsl:element>
(554)                     <xsl:element name="p">
(555)                       <xsl:text>Das Element ist im Rahmen der Transformation
      aus dem folgenden Element hervorgegangen:</xsl:text>
(556)                     </xsl:element>
(557)                     <xsl:element name="ul">
(558)                       <xsl:attribute name="class">
(559)                         <xsl:text>ul-verfolgbarkeit-transformation-
      elemente</xsl:text>
(560)                       </xsl:attribute>
(561)                       <xsl:for-each select="*[contains(@href, $param-prefix-
      rdf)]">
(562)                         <xsl:variable name="rdf-model-file"
      select="substring-before(@href, '#')"/>
(563)                         <xsl:variable name="rdf-elem-id" select="substring-
      after(@href, '#')"/>
(564)                         <xsl:variable name="rdf-elem" select="document($rdf-
      model-file)//node()[substring-after(@rdf:about, '#') = $rdf-elem-id]"/>
(565)
(566)                         <xsl:element name="li">
(567)                           <xsl:attribute name="class">
(568)                             <xsl:text>li-verfolgbarkeit-transformation-
      element</xsl:text>
(569)                           </xsl:attribute>
(570)                           <xsl:text>Individuum '</xsl:text>
(571)                           <xsl:element name="a">
(572)                             <xsl:attribute name="href">
(573)                               <xsl:value-of select="replace(@href, '.rdf',
      '.html')"/>
(574)                             </xsl:attribute>
(575)                             <xsl:choose>
(576)                               <xsl:when test="$rdf-elem/rdfs:label">
```

```
(577)                                  <xsl:value-of select="normalize-space($rdf-
      elem/rdfs:label)"/>
(578)                                </xsl:when>
(579)                                <xsl:otherwise>
(580)                                  <xsl:value-of select="substring-after(@href,
      '#')"/>
(581)                                </xsl:otherwise>
(582)                              </xsl:choose>
(583)                            </xsl:element>
(584)                            <xsl:text>' (Instanz der Klasse '</xsl:text>
(585)                            <xsl:value-of select="local-name($rdf-elem)"/>
(586)                            <xsl:text>')</xsl:text>
(587)                          </xsl:element>
(588)                        </xsl:for-each>
(589)                      </xsl:element>
(590)                    </xsl:element>
(591)                    <!-- saxon:assign name="transformation-backward"
      select="$transformation-backward+1"/ -->
(592)                  </xsl:for-each>
(593)
(594)                  <xsl:if test="count($document-trace-
      rdf2prm//rdfodm:InstanceSpecification[substring-after(dstIndv/@href, '#') =
      $id]) = 0">
(595)                    <xsl:element name="p">- Keine -</xsl:element>
(596)                  </xsl:if>
(597)
(598)                <!-- end: div-verfolgbarkeit-rueckwaerts -->
(599)                </xsl:element>
(600)
(601)                <xsl:element name="div">
(602)                  <xsl:attribute name="class">
(603)                    <xsl:text>div-verfolgbarkeit-vorwaerts</xsl:text>
(604)                  </xsl:attribute>
(605)
(606)                  <xsl:element name="h3">Zielelemente im CIM</xsl:element>
(607)
(608)                  <xsl:element name="p">Das Element wird in der/den folgenden
      Transformation(en) verwendet.</xsl:element>
(609)
(610)                  <xsl:for-each select="$document-trace-
      prm2cim/xmi:XMI/node()[child::*[substring-after(@href, '#') = $id]]">
(611)
(612)                    <xsl:element name="div">
(613)                      <xsl:attribute name="class">
(614)                        <xsl:text>div-verfolgbarkeit-transformation</xsl:text>
(615)                      </xsl:attribute>
(616)                      <xsl:element name="h4">
(617)                        <xsl:text>Transformation '</xsl:text>
(618)                        <xsl:value-of select="name()"/>
(619)                        <xsl:text>'</xsl:text>
(620)                      </xsl:element>
(621)                      <xsl:text>Die Transformation '</xsl:text>
(622)                      <xsl:value-of select="name()"/>
(623)                      <xsl:text>' erzeugt die folgenden Zielelemente und
      verwendet die folgenden Geschwisterelemente.</xsl:text>
(624)
(625)                      <xsl:element name="div">
(626)                        <xsl:attribute name="class">
(627)                          <xsl:text>div-verfolgbarkeit-transformation-
      ziel</xsl:text>
(628)                        </xsl:attribute>
(629)                        <xsl:element name="h5">Zielelemente</xsl:element>
(630)                        <xsl:element name="p">
(631)                          <xsl:text>Das Element </xsl:text>
(632)                          <xsl:value-of select="$label"/>
(633)                          <xsl:text> wird im Rahmen der Transformation
      '</xsl:text>
(634)                          <xsl:value-of select="name()"/>
(635)                          <xsl:text>' in folgende Elemente
      überführt:</xsl:text>
(636)                        </xsl:element>
(637)                        <xsl:element name="ul">
(638)                          <xsl:attribute name="class">
(639)                            <xsl:text>ul-verfolgbarkeit-transformation-
      elemente</xsl:text>
```

```
(640)                         </xsl:attribute>
(641)                         <xsl:for-each select="*[contains(@href, $param-
      prefix-cim)]">
(642)                           <xsl:variable name="cim-model-file"
      select="substring-before(@href, '#')"/>
(643)                           <xsl:variable name="cim-elem-id"
      select="substring-after(@href, '#')"/>
(644)                           <xsl:variable name="cim-elem"
      select="document($cim-model-file)//node()[@xmi21:id = $cim-elem-id]"/>
(645)
(646)                           <xsl:choose>
(647)                             <xsl:when test="$cim-elem/@name">
(648)                               <xsl:element name="li">
(649)                                 <xsl:attribute name="class">
(650)                                   <xsl:text>li-verfolgbarkeit-
      transformation-element</xsl:text>
(651)                                 </xsl:attribute>
(652)                                 <xsl:element name="a">
(653)                                   <xsl:attribute name="href">
(654)                                     <xsl:value-of select="replace(@href,
      '.uml', '.html')"/>
(655)                                   </xsl:attribute>
(656)                                   <xsl:choose>
(657)                                     <xsl:when test="$cim-elem/@xmi21:type">
(658)                                       <xsl:value-of select="$cim-
      elem/@xmi21:type"/>
(659)                                     </xsl:when>
(660)                                     <xsl:otherwise>
(661)                                       <xsl:value-of select="name($cim-
      elem)"/>
(662)                                     </xsl:otherwise>
(663)                                   </xsl:choose>
(664)                                   <xsl:text> '</xsl:text>
(665)                                   <xsl:value-of select="$cim-elem/@name"/>
(666)                                   <xsl:text>'</xsl:text>
(667)                                 </xsl:element>
(668)                                 <xsl:text> (als Werteausprägung der
      Variablen '</xsl:text>
(669)                                 <xsl:value-of select="local-name()"/>
(670)                                 <xsl:text>')</xsl:text>
(671)                               </xsl:element>
(672)                             </xsl:when>
(673)                             <xsl:when test="$cim-elem">
(674)                               <!-- Diese Modellelemente sind hier nicht
      relevant (z.B. Upper und Lower-Value eines Assoziation) -->
(675)                             </xsl:when>
(676)                             <xsl:otherwise>
(677)                               <xsl:element name="li">
(678)                                 <xsl:attribute name="class">
(679)                                   <xsl:text>li-verfolgbarkeit-
      transformation-element</xsl:text>
(680)                                 </xsl:attribute>
(681)                                 <xsl:text>Nicht mehr im CIM vorhandenes
      Element mit xmi:id = '</xsl:text>
(682)                                 <xsl:value-of select="$cim-elem-id"/>
(683)                                 <xsl:text>'.</xsl:text>
(684)                               </xsl:element>
(685)                             </xsl:otherwise>
(686)                           </xsl:choose>
(687)                         </xsl:for-each>
(688)
(689)                       <!-- end: ul -->
(690)                       </xsl:element>
(691)                     <!-- end: div-verfolgbarkeit-transformation-ziel -->
(692)                     </xsl:element>
(693)
(694)                     <xsl:element name="div">
(695)                       <xsl:attribute name="class">
(696)                         <xsl:text>div-verfolgbarkeit-transformation-
      geschwister</xsl:text>
(697)                       </xsl:attribute>
(698)
(699)                       <xsl:element
      name="h5">Geschwisterelemente</xsl:element>
(700)                       <xsl:element name="p">
```

```
(701)                         <xsl:text>Das Element wurde zusammen mit folgenden
      Geschwisterelementen transformiert:</xsl:text>
(702)                       </xsl:element>
(703)                       <xsl:element name="ul">
(704)                         <xsl:attribute name="class">
(705)                           <xsl:text>li-verfolgbarkeit-transformation-
      elemente</xsl:text>
(706)                         </xsl:attribute>
(707)                         <xsl:for-each select="*[contains(@href, $param-
      prefix-prm)]">
(708)                           <xsl:variable name="prm-model-file"
      select="substring-before(@href, '#')"/>
(709)                           <xsl:variable name="prm-elem-id"
      select="substring-after(@href, '#')"/>
(710)                           <xsl:variable name="prm-elem"
      select="document($prm-model-file)//node()[@xmi21:id = $prm-elem-id]"/>
(711)
(712)                           <xsl:choose>
(713)                             <xsl:when test="$prm-elem/@name">
(714)                               <xsl:element name="li">
(715)                                 <xsl:attribute name="class">
(716)                                   <xsl:text>li-verfolgbarkeit-
      transformation-element</xsl:text>
(717)                                 </xsl:attribute>
(718)                                 <xsl:element name="a">
(719)                                   <xsl:attribute name="href">
(720)                                     <xsl:value-of select="concat('#', $prm-
      elem-id)"/>
(721)                                   </xsl:attribute>
(722)                                   <xsl:choose>
(723)                                     <xsl:when test="$prm-elem/@xmi21:type">
(724)                                       <xsl:value-of select="$prm-
      elem/@xmi21:type"/>
(725)                                     </xsl:when>
(726)                                     <xsl:otherwise>
(727)                                       <xsl:value-of select="name($prm-
      elem)"/>
(728)                                     </xsl:otherwise>
(729)                                   </xsl:choose>
(730)                                   <xsl:text> '</xsl:text>
(731)                                   <xsl:value-of select="$prm-elem/@name"/>
(732)                                   <xsl:text>'</xsl:text>
(733)                                 </xsl:element>
(734)                                 <xsl:text> (als Werteausprägung der
      Variablen '</xsl:text>
(735)                                 <xsl:value-of select="local-name()"/>
(736)                                 <xsl:text>')</xsl:text>
(737)                               </xsl:element>
(738)                             </xsl:when>
(739)                             <xsl:when test="$prm-elem">
(740)                               <!-- Diese Modellelemente sind hier nicht
      relevant (z.B. Upper und Lower-Value eines Assoziation) -->
(741)                             </xsl:when>
(742)                             <xsl:otherwise>
(743)                               <xsl:element name="li">
(744)                                 <xsl:attribute name="class">
(745)                                   <xsl:text>li-verfolgbarkeit-
      transformation-element</xsl:text>
(746)                                 </xsl:attribute>
(747)                                 <xsl:text>Nicht mehr im PRM vorhandenes
      Element mit xmi:id = '</xsl:text>
(748)                                 <xsl:value-of select="$prm-elem-id"/>
(749)                                 <xsl:text>'.</xsl:text>
(750)                               </xsl:element>
(751)                             </xsl:otherwise>
(752)                           </xsl:choose>
(753)
(754)                         </xsl:for-each>
(755)
(756)                         <!-- Prüfen, ob ohne Geschwister -->
(757)                         <xsl:if test="count(*[contains(@href, $param-prefix-
      prm)]) = 0">
(758)                           <xsl:element name="p">- Keine -</xsl:element>
(759)                         </xsl:if>
(760)                       </xsl:element>
```

```
(761)
(762)                       <!-- end: div-verfolgbarkeit-transformation-
      geschwister -->
(763)                       </xsl:element>
(764)                   <!-- end: div-verfolgbarkeit-transformation -->
(765)                   </xsl:element>
(766)
(767)                 <!-- end: Transformatonen -->
(768)                 </xsl:for-each>
(769)
(770)                 <xsl:if test="count($document-trace-
      prm2cim/xmi:XMI/node()[child::*[substring-after(@href, '#') = $id]])=0">
(771)                   <xsl:element name="p">- Keine -</xsl:element>
(772)                 </xsl:if>
(773)
(774)               <!-- end: div-verfolgbarkeit-vorwaerts -->
(775)               </xsl:element>
(776)
(777)             <!-- end: div - element -->
(778)             </xsl:element>
(779)
(780)           <!-- end: Modellelement -->
(781)           </xsl:for-each>
(782)
(783)           <!-- "Fehlende" Modellelemente des PRM -->
(784)           <xsl:element name="h1">Gelöschte Modellelemente PRM</xsl:element>
(785)           <xsl:element name="p">
(786)             <xsl:text>Die folgenden Modellelemente waren Bestandteil der
      Transformation zur Erzeugung des PRM, sind im Modell aber nicht mehr
      enthalten.</xsl:text>
(787)           </xsl:element>
(788)
(789)           <!-- Für jedes Element im rdf2prm-Trace prüfen, ob es noch
      Bestandteil des Modells ist,
(790)                wenn nicht, dann Hinweis geben. -->
(791)           <xsl:variable name="document-prm" select="/" />
(792)
(793)           <xsl:for-each select="$document-trace-
      rdf2prm//rdfodm:InstanceSpecification">
(794)             <xsl:variable name="trace-id" select="substring-
      after(dstIndv/@href, '#') "/>
(795)             <xsl:variable name="trace-src" select="substring-
      after(srcIndv/@href, '#') "/>
(796)
(797)             <xsl:if test="count($document-prm//*[@xmi21:id = $trace-id]) =
      0">
(798)               <xsl:element name="div">
(799)                 <xsl:attribute name="class">
(800)                   <xsl:text>div-element</xsl:text>
(801)                 </xsl:attribute>
(802)                 <xsl:element name="h2">
(803)                   <xsl:text>Unbekanntes Modellelement (xmi:id = '</xsl:text>
(804)                   <xsl:value-of select="$trace-id"/>
(805)                   <xsl:text>')</xsl:text>
(806)                 </xsl:element>
(807)                 <xsl:element name="p">
(808)                   <xsl:text>Das aus der Annotation '</xsl:text>
(809)                   <xsl:value-of select="$trace-src"/>
(810)                   <xsl:text>' hervorgegangene Modellelement mit der xmi:id =
      '</xsl:text>
(811)                   <xsl:value-of select="$trace-id"/>
(812)                   <xsl:text>' ist nach Abschluss der Transformation im
      Rahmen der PRM-Modellierung gelöscht worden.</xsl:text>
(813)                 </xsl:element>
(814)                 <saxon:assign name="elements-missing" select="$elements-
      missing+1"/>
(815)               </xsl:element>
(816)             </xsl:if>
(817)
(818)           </xsl:for-each>
(819)
(820)           <xsl:if test="$elements-missing = 0">
(821)             <xsl:element name="p">- Keine -</xsl:element>
(822)           </xsl:if>
(823)
```

```
(824)         <!-- end: div -->
(825)         </xsl:element>
(826)
(827)       <!-- end: body -->
(828)       </xsl:element>
(829)
(830)     <!-- end: html -->
(831)     </xsl:element>
(832)
(833)   </xsl:template>
(834)
(835) </xsl:stylesheet>
```

Listing 64 – Verfolgbarkeitsdokumentation des PRM (create-GovObjTraces_PRM.xsl)

G.5 Computational Independent Model

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(4)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(5)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(6)     xmlns:html="http://www.w3.org/TR/REC-html40"
(7)     xmlns:xhtml="http://www.w3.org/1999/xhtml"
(8)     xmlns="http://www.w3.org/1999/xhtml"
(9)     xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
(10)    xmlns:dc="http://purl.org/dc/elements/1.1/"
(11)
      xmlns:ontomat="http://annotation.semanticweb.org/ontologies/cream/ontomat#"
(12)    xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
(13)    xmlns:rss="http://purl.org/rss/1.0/"
(14)    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
(15)    xmlns:govobjects="http://govobjects.thomasoff.de/bewohnerparken#"
(16)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(17)    xmlns:govobjects-
      ontology="http://govobjects.thomasoff.de/ontology/govobjects-ontology.owl#"
(18)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(19)    xmlns:owl="http://www.w3.org/2002/07/owl#"
(20)    xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
(21)    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
(22)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(23)    xmlns:rdfodm="urn:rdf2odm.ecore"
(24)    xmlns:xmi="http://www.omg.org/XMI"
(25)    xmlns:xmi21="http://schema.omg.org/spec/XMI/2.1"
(26)    xmlns:saxon="http://saxon.sf.net/"
(27)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
(28)    xmlns:eGovRefAnf="http:///eGovRefAnf.ecore"
(29)    extension-element-prefixes="saxon">
(30)
(31)    <xsl:include href="commonGovObjTraces.xsl"/>
(32)
(33)    <xsl:output method="xhtml" version="1.0" encoding="UTF-8" indent="yes"/>
(34)
(35)    <!--  xsl:param name="param-cim-model-file"
      select="'../Result/cim_stvg_vwv-stvo.uml'"/ -->
(36)    <xsl:param name="param-trace-prm2cim-file1"
      select="'../in/trace.prm2cim_anf'"/>
(37)    <xsl:param name="param-trace-prm2cim-file2"
      select="'../in/trace.prm2cim_proz'"/>
(38)
(39)    <xsl:param name="param-prefix-prm" select="'prm'"/>
(40)    <xsl:param name="param-prefix-cim" select="'cim'"/>
(41)
(42)    <!--  xsl:variable name="cim-model-file" select="replace($param-cim-model-
      file, '\\', '/')"/ -->
(43)
(44)    <xsl:variable name="document-trace-prm2cim" select="document($param-trace-
      prm2cim-file1)"/>
(45)
(46)    <xsl:variable name="elements-missing" select="0" saxon:assignable="yes"/>
(47)    <xsl:variable name="transformation-backward" select="0"
      saxon:assignable="yes"/>
(48)
(49)    <xsl:template match="/xmi21:XMI">
(50)      <!-- Grundstruktur für xhtml-Dokumente -->
```

```
(51)      <xsl:element name="html" namespace="http://www.w3.org/1999/xhtml"
      inherit-namespaces="yes">
(52)        <xsl:element name="head">
(53)          <xsl:element name="meta">
(54)            <xsl:attribute name="http-equiv">
(55)              <xsl:text>content-type</xsl:text>
(56)            </xsl:attribute>
(57)            <xsl:attribute name="content">
(58)              <xsl:text>text/html; charset=UTF-8</xsl:text>
(59)            </xsl:attribute>
(60)          </xsl:element>
(61)          <xsl:element name="title">
(62)            <xsl:text>GovObj Traceability: CIM</xsl:text>
(63)          </xsl:element>
(64)          <xsl:element name="script">
(65)            <xsl:attribute name="type" select="'text/javascript'"/>
(66)            <xsl:attribute name="src" select="'../govobj-filtern.js'"/>
(67)          </xsl:element>
(68)        </xsl:element>
(69)        <xsl:element name="body">
(70)          <xsl:attribute name="onload" select="'resetForm()'"/>
(71)
(72)          <xsl:element name="div">
(73)            <xsl:attribute name="id">
(74)              <xsl:text>uml-instances</xsl:text>
(75)            </xsl:attribute>
(76)
(77)            <!-- Im PRM vorhandene Modellelemente -->
(78)            <xsl:element name="h1">Verfolgbarkeit von Modellelementen des
      CIM</xsl:element>
(79)
(80)            <xsl:element name="div">
(81)              <xsl:attribute name="name" select="'filter'"/>
(82)              <xsl:attribute name="id" select="'div-filter'"/>
(83)
(84)              <xsl:element name="h2">
(85)                <xsl:text>Modellelemente filtern</xsl:text>
(86)              </xsl:element>
(87)
(88)              <xsl:element name="form">
(89)                <xsl:attribute name="name" select="'filter'"/>
(90)                <xsl:element name="div">
(91)                  <xsl:attribute name="name" select="'filterkriterien'"/>
(92)                  <xsl:element name="div">
(93)                    <xsl:attribute name="name" select="'fk_name'"/>
(94)                    <xsl:element name="span">
(95)                      <xsl:attribute name="name" select="'fk_name_label'"/>
(96)                      <xsl:text>Name</xsl:text>
(97)                    </xsl:element>
(98)                    <xsl:element name="br"/>
(99)                    <xsl:element name="select">
(100)                     <xsl:attribute name="name" select="'fk_name_select'"/>
(101)                     <xsl:attribute name="title" select="'Name'"/>
(102)                     <xsl:attribute name="size" select="'4'"/>
(103)                     <xsl:attribute name="multiple" select="'multiple'"/>
(104)                   </xsl:element>
(105)                 </xsl:element>
(106)
(107)                 <xsl:element name="div">
(108)                   <xsl:attribute name="name" select="'fk_typ'"/>
(109)                   <xsl:element name="span">
(110)                     <xsl:attribute name="name" select="'fk_typ_label'"/>
(111)                     <xsl:text>Typ</xsl:text>
(112)                   </xsl:element>
(113)                   <xsl:element name="br"/>
(114)                   <xsl:element name="select">
(115)                     <xsl:attribute name="name" select="'fk_typ_select'"/>
(116)                     <xsl:attribute name="title" select="'Typ'"/>
(117)                     <xsl:attribute name="size" select="'4'"/>
(118)                     <xsl:attribute name="multiple" select="'multiple'"/>
(119)                   </xsl:element>
(120)                 </xsl:element>
(121)
(122)                 <xsl:element name="div">
(123)                   <xsl:attribute name="name" select="'fk_abbregel'"/>
```

```
(124)                   <xsl:element name="span">
(125)                     <xsl:attribute name="name"
      select="'fk_abbregel_label'"/>
(126)                     <xsl:text>Abbildungsregel</xsl:text>
(127)                   </xsl:element>
(128)                   <xsl:element name="br"/>
(129)                   <xsl:element name="select">
(130)                     <xsl:attribute name="name"
      select="'fk_abbregel_select'"/>
(131)                     <xsl:attribute name="title"
      select="'Abbildungsregel(n)'"/>
(132)                     <xsl:attribute name="size" select="'4'"/>
(133)                     <xsl:attribute name="multiple" select="'multiple'"/>
(134)                   </xsl:element>
(135)                 </xsl:element>
(136)
(137)               <!--  end: div - filterkriterien -->
(138)               </xsl:element>
(139)               <xsl:element name="div">
(140)                 <xsl:attribute name="name" select="'filteraktionen'"/>
(141)                 <xsl:element name="input">
(142)                   <xsl:attribute name="type" select="'button'"/>
(143)                   <xsl:attribute name="value" select="'Zurücksetzen'"/>
(144)                   <xsl:attribute name="onclick" select="'resetFilter()'"/>
(145)                 </xsl:element>
(146)                 <xsl:element name="input">
(147)                   <xsl:attribute name="type" select="'button'"/>
(148)                   <xsl:attribute name="value" select="'Filtern'"/>
(149)                   <xsl:attribute name="onclick" select="'filtern()'"/>
(150)                 </xsl:element>
(151)               </xsl:element>
(152)             <!--  end: form - filter -->
(153)             </xsl:element>
(154)           <!--  end: div - filter -->
(155)           </xsl:element>
(156)
(157)           <xsl:element name="div">
(158)             <xsl:attribute name="name" select="'div-filtermessage'"/>
(159)             <xsl:element name="span">
(160)               <xsl:attribute name="name" select="'span-filtermessage'"/>
(161)               <xsl:attribute name="id" select="'span-filtermessage'"/>
(162)             </xsl:element>
(163)           </xsl:element>
(164)
(165)           <xsl:element name="p">
(166)             <xsl:text>Die folgenden Modellemente sind Bestandteil des
      CIM.</xsl:text>
(167)           </xsl:element>
(168)
(169)           <xsl:for-each
      select="//uml:Model|//packagedElement|//eGovRefAnf:*">
(170)
(171)             <xsl:variable name="id" select="@xmi21:id"/>
(172)
(173)             <xsl:variable name="type">
(174)               <xsl:choose>
(175)                 <xsl:when test="@xmi21:type">
(176)                   <xsl:value-of select="@xmi21:type"/>
(177)                 </xsl:when>
(178)                 <xsl:otherwise>
(179)                   <xsl:value-of select="name()"/>
(180)                 </xsl:otherwise>
(181)               </xsl:choose>
(182)             </xsl:variable>
(183)
(184)             <xsl:variable name="name">
(185)               <xsl:choose>
(186)                 <xsl:when test="@name">
(187)                   <xsl:value-of select="@name"/>
(188)                 </xsl:when>
(189)                 <xsl:otherwise>
(190)                   <xsl:value-of select="local-name()"/>
(191)                 </xsl:otherwise>
(192)               </xsl:choose>
(193)             </xsl:variable>
```

```
(194)
(195)             <xsl:variable name="label">
(196)               <xsl:value-of select="$type"/>
(197)               <xsl:text> '</xsl:text>
(198)               <xsl:value-of select="$name"/>
(199)               <xsl:text>'</xsl:text>
(200)             </xsl:variable>
(201)
(202)             <xsl:variable name="documentation">
(203)               <xsl:choose>
(204)                 <xsl:when
      test="eAnnotations[1][@source='http://www.topcased.org/documentation']">
(205)                   <xsl:value-of
      select="eAnnotations[1][@source='http://www.topcased.org/documentation']/det
      ails/@value"/>
(206)                 </xsl:when>
(207)                 <xsl:otherwise>
(208)                   <xsl:text>- Keine -</xsl:text>
(209)                 </xsl:otherwise>
(210)               </xsl:choose>
(211)             </xsl:variable>
(212)
(213)             <xsl:element name="div">
(214)               <xsl:attribute name="class">
(215)                 <xsl:text>div-element</xsl:text>
(216)               </xsl:attribute>
(217)
(218)               <!-- Achnor für Navigation im Dokument anlegen -->
(219)               <xsl:element name="a">
(220)                 <xsl:attribute name="name" select="$id"/>
(221)               </xsl:element>
(222)
(223)               <!-- Filtereigenschaften -->
(224)               <xsl:element name="div">
(225)                 <xsl:attribute name="style" select="'{display: none;}'"/>
(226)                 <xsl:attribute name="class" select="'div-filterkriterien'"/>
(227)                 <xsl:element name="span">
(228)                   <xsl:attribute name="name" select="'fk_name_value'"/>
(229)                   <xsl:value-of select="$label"/>
(230)                 </xsl:element>
(231)                 <xsl:element name="span">
(232)                   <xsl:attribute name="name" select="'fk_typ_value'"/>
(233)                   <xsl:value-of select="$type"/>
(234)                 </xsl:element>
(235)                 <xsl:for-each select="$document-trace-
      prm2cim/xmi:XMI/node()[child::*[substring-after(@href, '#') = $id]]">
(236)                   <xsl:element name="span">
(237)                     <xsl:attribute name="name" select="'fk_abbregel_multi-
      value'"/>
(238)                       <xsl:value-of select="name()"/>
(239)                   </xsl:element>
(240)                 </xsl:for-each>
(241)               </xsl:element>
(242)
(243)               <!-- Überschrift des Elementes anlegen -->
(244)               <xsl:element name="h2">
(245)                 <xsl:value-of select="$label"/>
(246)               </xsl:element>
(247)
(248)               <xsl:element name="div">
(249)                 <xsl:attribute name="class">
(250)                   <xsl:text>div-eigenschaften</xsl:text>
(251)                 </xsl:attribute>
(252)                 <xsl:element name="h3">
(253)                   <xsl:text>Eigenschaften</xsl:text>
(254)                 </xsl:element>
(255)
(256)                 <xsl:element name="p">Das Element hat die folgenden
      Eigenschaften:</xsl:element>
(257)
(258)                 <xsl:element name="ul">
(259)                   <xsl:attribute name="class">
(260)                     <xsl:text>ul-eigenschaft</xsl:text>
(261)                   </xsl:attribute>
(262)
```

```
(263)                     <!-- Name -->
(264)                     <xsl:element name="li">
(265)                       <xsl:attribute name="class">
(266)                         <xsl:text>li-eigenschaft</xsl:text>
(267)                       </xsl:attribute>
(268)                       <xsl:element name="span">
(269)                         <xsl:attribute name="class">
(270)                           <xsl:text>span-eigenschaft-titel</xsl:text>
(271)                         </xsl:attribute>
(272)                         <xsl:text>Name:</xsl:text>
(273)                       </xsl:element>
(274)                       <xsl:element name="span">
(275)                         <xsl:attribute name="class">
(276)                           <xsl:text>span-eigenschaft-wert</xsl:text>
(277)                         </xsl:attribute>
(278)                         <xsl:value-of select="$name"/>
(279)                       </xsl:element>
(280)                     </xsl:element>
(281)
(282)                     <!-- Typ -->
(283)                     <xsl:element name="li">
(284)                       <xsl:attribute name="class">
(285)                         <xsl:text>li-eigenschaft</xsl:text>
(286)                       </xsl:attribute>
(287)                       <xsl:element name="span">
(288)                         <xsl:attribute name="class">
(289)                           <xsl:text>span-eigenschaft-titel</xsl:text>
(290)                         </xsl:attribute>
(291)                         <xsl:text>Typ:</xsl:text>
(292)                       </xsl:element>
(293)                       <xsl:element name="span">
(294)                         <xsl:attribute name="class">
(295)                           <xsl:text>span-eigenschaft-wert</xsl:text>
(296)                         </xsl:attribute>
(297)                         <xsl:value-of select="$type"/>
(298)                       </xsl:element>
(299)                     </xsl:element>
(300)
(301)                     <!-- ID -->
(302)                     <xsl:element name="li">
(303)                       <xsl:attribute name="class">
(304)                         <xsl:text>li-eigenschaft</xsl:text>
(305)                       </xsl:attribute>
(306)                       <xsl:element name="span">
(307)                         <xsl:attribute name="class">
(308)                           <xsl:text>span-eigenschaft-titel</xsl:text>
(309)                         </xsl:attribute>
(310)                         <xsl:text>ID:</xsl:text>
(311)                       </xsl:element>
(312)                       <xsl:element name="span">
(313)                         <xsl:attribute name="class">
(314)                           <xsl:text>span-eigenschaft-wert</xsl:text>
(315)                         </xsl:attribute>
(316)                         <xsl:value-of select="$id"/>
(317)                       </xsl:element>
(318)                     </xsl:element>
(319)
(320)                     <!-- Dokumentation -->
(321)                     <xsl:element name="li">
(322)                       <xsl:attribute name="class">
(323)                         <xsl:text>li-eigenschaft</xsl:text>
(324)                       </xsl:attribute>
(325)                       <xsl:element name="span">
(326)                         <xsl:attribute name="class">
(327)                           <xsl:text>span-eigenschaft-titel</xsl:text>
(328)                         </xsl:attribute>
(329)                         <xsl:text>Dokumentation:</xsl:text>
(330)                       </xsl:element>
(331)                       <xsl:element name="span">
(332)                         <xsl:attribute name="class">
(333)                           <xsl:text>span-eigenschaft-wert</xsl:text>
(334)                         </xsl:attribute>
(335)                         <xsl:value-of select="$documentation"/>
(336)                       </xsl:element>
(337)                     </xsl:element>
```

```
(338)
(339)                   <!-- end: ul -->
(340)                   </xsl:element>
(341)
(342)               <!-- end: div - eigenschaften -->
(343)               </xsl:element>
(344)
(345)               <!-- Attribute -->
(346)               <xsl:if test="classifier/@xmi21:type='uml:Class'">
(347)                 <xsl:element name="div">
(348)                   <xsl:attribute name="class">
(349)                     <xsl:text>div-attribute</xsl:text>
(350)                   </xsl:attribute>
(351)
(352)                   <xsl:element name="h3">Attribute</xsl:element>
(353)                   <xsl:element name="p">Die Instanz hat die folgenden
      Werteausprägungen von Attributen.</xsl:element>
(354)
(355)                   <!-- Prüfen, ob ohne Attribute -->
(356)                   <xsl:if test="count(slot) = 0">
(357)                     <xsl:element name="p">
(358)                       <xsl:text>- Keine -</xsl:text>
(359)                     </xsl:element>
(360)                   </xsl:if>
(361)
(362)                   <!-- Attribut -->
(363)                   <xsl:for-each select="slot">
(364)                     <xsl:element name="div">
(365)                       <xsl:attribute name="class">
(366)                         <xsl:text>div-attribut</xsl:text>
(367)                       </xsl:attribute>
(368)
(369)                       <xsl:variable name="prm-attribut-name"
      select="substring-before(substring-after(definingFeature/@href, '#'),
      '_attribute')"/>
(370)                       <xsl:variable name="prm-attribut-type"
      select="value/@xmi21:type"/>
(371)                       <xsl:variable name="prm-attribut-wert"
      select="value/@value"/>
(372)
(373)                       <xsl:element name="h4">
(374)                         <xsl:text>Attribut '</xsl:text>
(375)                         <xsl:value-of select="$prm-attribut-name"/>
(376)                         <xsl:text>'</xsl:text>
(377)                       </xsl:element>
(378)
(379)                       <xsl:element name="ul">
(380)                         <xsl:attribute name="class">
(381)                           <xsl:text>ul-attribut</xsl:text>
(382)                         </xsl:attribute>
(383)
(384)                         <!-- Name -->
(385)                         <xsl:element name="li">
(386)                           <xsl:attribute name="class">
(387)                             <xsl:text>li-attribut</xsl:text>
(388)                           </xsl:attribute>
(389)                           <xsl:element name="span">
(390)                             <xsl:attribute name="class">
(391)                               <xsl:text>span-attribut-titel</xsl:text>
(392)                             </xsl:attribute>
(393)                             <xsl:text>Name:</xsl:text>
(394)                           </xsl:element>
(395)                           <xsl:element name="span">
(396)                             <xsl:attribute name="class">
(397)                               <xsl:text>span-attribut-wert</xsl:text>
(398)                             </xsl:attribute>
(399)                             <xsl:value-of select="$prm-attribut-name"/>
(400)                           </xsl:element>
(401)                         <!-- end: li -->
(402)                         </xsl:element>
(403)
(404)                         <!-- Datentyp -->
(405)                         <xsl:element name="li">
(406)                           <xsl:attribute name="class">
(407)                             <xsl:text>li-attribut</xsl:text>
```

```
(408)                           </xsl:attribute>
(409)                           <xsl:element name="span">
(410)                             <xsl:attribute name="class">
(411)                               <xsl:text>span-attribut-titel</xsl:text>
(412)                             </xsl:attribute>
(413)                             <xsl:text>Datentyp:</xsl:text>
(414)                           </xsl:element>
(415)                           <xsl:element name="span">
(416)                             <xsl:attribute name="class">
(417)                               <xsl:text>span-attribut-wert</xsl:text>
(418)                             </xsl:attribute>
(419)                             <xsl:value-of select="$prm-attribut-type"/>
(420)                           </xsl:element>
(421)                         <!-- end: li -->
(422)                         </xsl:element>
(423)
(424)                         <!-- Wert -->
(425)                         <xsl:element name="li">
(426)                           <xsl:attribute name="class">
(427)                             <xsl:text>li-attribut</xsl:text>
(428)                           </xsl:attribute>
(429)                           <xsl:element name="span">
(430)                             <xsl:attribute name="class">
(431)                               <xsl:text>span-attribut-titel</xsl:text>
(432)                             </xsl:attribute>
(433)                             <xsl:text>Wert:</xsl:text>
(434)                           </xsl:element>
(435)                           <xsl:element name="span">
(436)                             <xsl:attribute name="class">
(437)                               <xsl:text>span-attribut-wert</xsl:text>
(438)                             </xsl:attribute>
(439)                             <xsl:choose>
(440)                               <xsl:when test="string-length($prm-attribut-
      wert) > 0">
(441)                                 <xsl:text>'</xsl:text>
(442)                                 <xsl:value-of select="$prm-attribut-wert"/>
(443)                                 <xsl:text>'</xsl:text>
(444)                               </xsl:when>
(445)                               <xsl:otherwise>
(446)                                 <xsl:text>- Undefiniert -</xsl:text>
(447)                               </xsl:otherwise>
(448)                             </xsl:choose>
(449)                           </xsl:element>
(450)                         <!-- end: li -->
(451)                         </xsl:element>
(452)
(453)                       <!-- end: ul - attribut -->
(454)                       </xsl:element>
(455)                     <!-- end: div - attribut -->
(456)                     </xsl:element>
(457)
(458)                   <!-- end: attribut -->
(459)                   </xsl:for-each>
(460)
(461)                 <!-- end: div-attribute-->
(462)                 </xsl:element>
(463)               </xsl:if>
(464)
(465)               <!-- Für Assoziationen: Enden -->
(466)               <xsl:if test="@xmi21:type='uml:Association'">
(467)                 <xsl:element name="div">
(468)                   <xsl:attribute name="class">
(469)                     <xsl:text>div-enden</xsl:text>
(470)                   </xsl:attribute>
(471)
(472)                   <xsl:element name="h3">Assoziationsenden</xsl:element>
(473)                   <xsl:element name="p">Die Assoziation verbindet die
      folgenden Klassen.</xsl:element>
(474)
(475)                   <xsl:element name="ul">
(476)                     <xsl:attribute name="class">
(477)                       <xsl:text>ul-enden</xsl:text>
(478)                     </xsl:attribute>
(479)                     <xsl:for-each select="ownedEnd">
```

```
(480)                       <xsl:variable name="cim-assoziation-rolle-id"
      select="@type"/>
(481)                       <xsl:element name="li">
(482)                         <xsl:attribute name="class">
(483)                           <xsl:text>li-ende</xsl:text>
(484)                         </xsl:attribute>
(485)                         <xsl:text>Klasse '</xsl:text>
(486)                         <xsl:value-of select="//*[@xmi21:id = $cim-
      assoziation-rolle-id]/@name"/>
(487)                         <xsl:text>' in der Rolle '</xsl:text>
(488)                         <xsl:value-of select="@name"/>
(489)                         <xsl:text>'</xsl:text>
(490)                       </xsl:element>
(491)                     </xsl:for-each>
(492)                   <!-- end: ul-enden -->
(493)                   </xsl:element>
(494)                 <!-- end: div-enden-->
(495)                 </xsl:element>
(496)               </xsl:if>
(497)
(498)               <!-- Für Dependency: Enden -->
(499)               <xsl:if test="@xmi21:type='uml:Dependency'">
(500)                 <xsl:element name="div">
(501)                   <xsl:attribute name="class">
(502)                     <xsl:text>div-enden</xsl:text>
(503)                   </xsl:attribute>
(504)
(505)                   <xsl:element name="h3">Dependency-Enden</xsl:element>
(506)                   <xsl:element name="p">Die Dependency verbindet die
      folgenden Klassen.</xsl:element>
(507)
(508)                   <xsl:element name="ul">
(509)                     <xsl:attribute name="class">
(510)                       <xsl:text>ul-enden</xsl:text>
(511)                     </xsl:attribute>
(512)                       <xsl:variable name="cim-dependency-supplier-id"
      select="@supplier"/>
(513)                       <xsl:variable name="cim-dependency-client-id"
      select="@client"/>
(514)                       <xsl:element name="li">
(515)                         <xsl:attribute name="class">
(516)                           <xsl:text>li-ende</xsl:text>
(517)                         </xsl:attribute>
(518)                         <xsl:text>Quelle '</xsl:text>
(519)                         <xsl:value-of select="//*[@xmi21:id = $cim-
      dependency-client-id]/@name"/>
(520)                         <xsl:text>'</xsl:text>
(521)                       </xsl:element>
(522)                       <xsl:element name="li">
(523)                         <xsl:attribute name="class">
(524)                           <xsl:text>li-ende</xsl:text>
(525)                         </xsl:attribute>
(526)                         <xsl:text>Ziel '</xsl:text>
(527)                         <xsl:value-of select="//*[@xmi21:id = $cim-
      dependency-supplier-id]/@name"/>
(528)                         <xsl:text>'</xsl:text>
(529)                       </xsl:element>
(530)
(531)                   <!-- end: ul-enden -->
(532)                   </xsl:element>
(533)                 <!-- end: div-enden-->
(534)                 </xsl:element>
(535)               </xsl:if>
(536)
(537)               <!-- Für Klassen, Akteure, UseCases: Beziehungen -->
(538)               <xsl:if test="not(@xmi21:type='uml:Association') and
      not(@xmi21:type='uml:Dependency')">
(539)                 <xsl:element name="div">
(540)                   <xsl:attribute name="class">
(541)                     <xsl:text>div-beziehungen</xsl:text>
(542)                   </xsl:attribute>
(543)
(544)                   <xsl:element name="h3">Assoziationen</xsl:element>
(545)                   <xsl:element name="p">Das Element hat die folgenden
      Assoziationen.</xsl:element>
```

```
(546)
(547)                   <!-- Prüfen, ob ohne Beziehungen -->
(548)                   <xsl:if
      test="count(//packagedElement[@xmi21:type='uml:Association'][ownedEnd/@type=
      $id]) = 0">
(549)                     <xsl:element name="p">
(550)                       <xsl:text>- Keine -</xsl:text>
(551)                     </xsl:element>
(552)                   </xsl:if>
(553)
(554)                 <!-- Beziehung -->
(555)                 <xsl:for-each
      select="//packagedElement[@xmi21:type='uml:Association'][ownedEnd/@type=$id]
      ">
(556)                     <xsl:element name="div">
(557)                     <xsl:attribute name="class">
(558)                       <xsl:text>div-beziehung</xsl:text>
(559)                     </xsl:attribute>
(560)
(561)                     <xsl:variable name="cim-beziehung-id"
      select="@xmi21:id"/>
(562)                     <xsl:variable name="cim-beziehung-name">
(563)                       <xsl:choose>
(564)                         <xsl:when test="@name">
(565)                           <xsl:value-of select="@name"/>
(566)                         </xsl:when>
(567)                         <xsl:otherwise>
(568)                           <xsl:text>OhneName</xsl:text>
(569)                         </xsl:otherwise>
(570)                       </xsl:choose>
(571)                     </xsl:variable>
(572)                     <xsl:variable name="cim-beziehung-quelle-rolle"
      select="ownedEnd[@type=$id]/@name"/>
(573)                     <xsl:variable name="cim-beziehung-ziel-id"
      select="ownedEnd[not(@type=$id)]/@type"/>
(574)                     <xsl:variable name="cim-beziehung-ziel-rolle"
      select="ownedEnd[not(@type=$id)]/@name"/>
(575)                     <xsl:variable name="cim-beziehung-ziel-elem"
      select="//packagedElement[@xmi21:id = $cim-beziehung-ziel-id]"/>
(576)
(577)                     <xsl:element name="h4">
(578)                       <xsl:text>Assozitation '</xsl:text>
(579)                       <xsl:value-of select="$cim-beziehung-name"/>
(580)                       <xsl:text>'</xsl:text>
(581)                     </xsl:element>
(582)
(583)                     <xsl:element name="p">
(584)                       <xsl:text>Die Klasse '</xsl:text>
(585)                       <xsl:value-of select="$name"/>
(586)                       <xsl:text>' hat in der Rolle '</xsl:text>
(587)                       <xsl:value-of select="$cim-beziehung-quelle-rolle"/>
(588)                       <xsl:text>' die Beziehung '</xsl:text>
(589)                       <xsl:element name="a">
(590)                         <xsl:attribute name="href">
(591)                           <xsl:value-of select="concat('#', $cim-beziehung-
      id)"/>
(592)                         </xsl:attribute>
(593)                         <xsl:value-of select="$cim-beziehung-name"/>
(594)                       </xsl:element>
(595)                       <xsl:text>' zur Klasse '</xsl:text>
(596)                       <xsl:choose>
(597)                         <xsl:when test="$cim-beziehung-ziel-elem/@name">
(598)                           <xsl:element name="a">
(599)                             <xsl:attribute name="href">
(600)                               <xsl:value-of select="concat('#', $cim-
      beziehung-ziel-id)"/>
(601)                             </xsl:attribute>
(602)                             <xsl:value-of select="$cim-beziehung-ziel-
      elem/@name"/>
(603)                           </xsl:element>
(604)                         </xsl:when>
(605)                         <xsl:otherwise>
(606)                           <xsl:text>- Undefiniert -</xsl:text>
(607)                         </xsl:otherwise>
(608)                       </xsl:choose>
```

```
(609)                       <xsl:text>' in der Rolle '</xsl:text>
(610)                       <xsl:value-of select="$cim-beziehung-ziel-rolle"/>
(611)                       <xsl:text>'</xsl:text>
(612)
(613)                     <!-- End: p -->
(614)                     </xsl:element>
(615)
(616)                   <!-- end: div - beziehung -->
(617)                   </xsl:element>
(618)
(619)                 <!-- end: beziehung -->
(620)                 </xsl:for-each>
(621)
(622)                 <!-- end: div-eigenschaften -->
(623)                 </xsl:element>
(624)
(625)               </xsl:if>
(626)               <!-- Verfolgbarkeit - Rückwärts zum PRM  -->
(627)               <xsl:element name="div">
(628)                 <xsl:attribute name="class">
(629)                   <xsl:text>div-verfolgbarkeit-rueckwaerts</xsl:text>
(630)                 </xsl:attribute>
(631)
(632)                 <xsl:element name="h3">Ausgangselemente im PRM</xsl:element>
(633)
(634)                 <xsl:element name="p">Das Element wird in der/den folgenden
      Transformation(en) verwendet.</xsl:element>
(635)
(636)                 <xsl:for-each select="$document-trace-
      prm2cim/xmi:XMI/node()[child::*[substring-after(@href, '#') = $id]]">
(637)
(638)                   <xsl:element name="div">
(639)                     <xsl:attribute name="class">
(640)                       <xsl:text>div-verfolgbarkeit-transformation</xsl:text>
(641)                     </xsl:attribute>
(642)                     <xsl:element name="h4">
(643)                       <xsl:text>Transformation '</xsl:text>
(644)                       <xsl:value-of select="name()"/>
(645)                       <xsl:text>'</xsl:text>
(646)                     </xsl:element>
(647)                     <xsl:text>Die Transformation '</xsl:text>
(648)                     <xsl:value-of select="name()"/>
(649)                     <xsl:text>' nutzte die folgenden Ausgangs- und
      Geschwisterelemente.</xsl:text>
(650)
(651)                     <xsl:element name="div">
(652)                       <xsl:attribute name="class">
(653)                         <xsl:text>div-verfolgbarkeit-transformation-
      ziel</xsl:text>
(654)                       </xsl:attribute>
(655)                       <xsl:element name="h5">Ausgangselemente</xsl:element>
(656)                       <xsl:element name="p">
(657)                         <xsl:text>Das Element </xsl:text>
(658)                         <xsl:value-of select="$label"/>
(659)                         <xsl:text> ist im Rahmen der Transformation
      '</xsl:text>
(660)                         <xsl:value-of select="name()"/>
(661)                         <xsl:text>' aus den folgenden Elementen
      hervorgegangen:</xsl:text>
(662)                       </xsl:element>
(663)                       <xsl:element name="ul">
(664)                         <xsl:attribute name="class">
(665)                           <xsl:text>ul-verfolgbarkeit-transformation-
      elemente</xsl:text>
(666)                         </xsl:attribute>
(667)                         <xsl:for-each select="*[contains(@href, $param-
      prefix-prm)]">
(668)                           <xsl:variable name="prm-model-file"
      select="substring-before(@href, '#')"/>
(669)                           <xsl:variable name="prm-elem-id"
      select="substring-after(@href, '#')"/>
(670)                           <xsl:variable name="prm-elem"
      select="document($prm-model-file)//node()[@xmi21:id = $prm-elem-id]"/>
(671)
(672)                           <xsl:choose>
```

```
(673)                             <xsl:when test="$prm-elem/@name">
(674)                               <xsl:element name="li">
(675)                                 <xsl:attribute name="class">
(676)                                   <xsl:text>li-verfolgbarkeit-
      transformation-element</xsl:text>
(677)                                 </xsl:attribute>
(678)                                 <xsl:element name="a">
(679)                                   <xsl:attribute name="href">
(680)                                     <xsl:value-of select="replace(@href,
      '.uml', '.html')"/>
(681)                                   </xsl:attribute>
(682)                                   <xsl:choose>
(683)                                     <xsl:when test="$prm-elem/@xmi21:type">
(684)                                       <xsl:value-of select="$prm-
      elem/@xmi21:type"/>
(685)                                     </xsl:when>
(686)                                     <xsl:otherwise>
(687)                                       <xsl:value-of select="name($prm-
      elem)"/>
(688)                                     </xsl:otherwise>
(689)                                   </xsl:choose>
(690)                                   <xsl:text> '</xsl:text>
(691)                                   <xsl:value-of select="$prm-elem/@name"/>
(692)                                   <xsl:text>'</xsl:text>
(693)                                 </xsl:element>
(694)                                 <xsl:text> (als Werteausprägung der
      Variablen '</xsl:text>
(695)                                 <xsl:value-of select="local-name()"/>
(696)                                 <xsl:text>')</xsl:text>
(697)                               </xsl:element>
(698)                             </xsl:when>
(699)                             <xsl:when test="$prm-elem">
(700)                               <!-- Diese Modellelemente sind hier nicht
      relevant (z.B. Upper und Lower-Value eines Assoziation) -->
(701)                             </xsl:when>
(702)                             <xsl:otherwise>
(703)                               <xsl:element name="li">
(704)                                 <xsl:attribute name="class">
(705)                                   <xsl:text>li-verfolgbarkeit-
      transformation-element</xsl:text>
(706)                                 </xsl:attribute>
(707)                                 <xsl:text>Nicht mehr im PRM vorhandenes
      Element mit xmi:id = '</xsl:text>
(708)                                 <xsl:value-of select="$prm-elem-id"/>
(709)                                 <xsl:text>'.</xsl:text>
(710)                               </xsl:element>
(711)                             </xsl:otherwise>
(712)                           </xsl:choose>
(713)                         </xsl:for-each>
(714)
(715)                       <!-- end: ul -->
(716)                       </xsl:element>
(717)                     <!-- end: div-verfolgbarkeit-transformation-ziel -->
(718)                     </xsl:element>
(719)
(720)                     <xsl:element name="div">
(721)                       <xsl:attribute name="class">
(722)                         <xsl:text>div-verfolgbarkeit-transformation-
      geschwister</xsl:text>
(723)                       </xsl:attribute>
(724)
(725)                       <xsl:element
      name="h5">Geschwisterelemente</xsl:element>
(726)                       <xsl:element name="p">
(727)                         <xsl:text>Das Element wurde zusammen mit folgenden
      Geschwisterelementen transformiert:</xsl:text>
(728)                       </xsl:element>
(729)                       <xsl:element name="ul">
(730)                         <xsl:attribute name="class">
(731)                           <xsl:text>li-verfolgbarkeit-transformation-
      elemente</xsl:text>
(732)                         </xsl:attribute>
(733)                         <xsl:for-each select="*[contains(@href, $param-
      prefix-cim)]">
```

```
(734)                             <xsl:variable name="cim-model-file"
      select="substring-before(@href, '#')"/>
(735)                             <xsl:variable name="cim-elem-id"
      select="substring-after(@href, '#')"/>
(736)                             <xsl:variable name="cim-elem"
      select="document($cim-model-file)//node()[@xmi21:id = $cim-elem-id]"/>
(737)
(738)                             <xsl:choose>
(739)                               <xsl:when test="$cim-elem/@name">
(740)                                 <xsl:element name="li">
(741)                                   <xsl:attribute name="class">
(742)                                     <xsl:text>li-verfolgbarkeit-
      transformation-element</xsl:text>
(743)                                   </xsl:attribute>
(744)                                   <xsl:element name="a">
(745)                                     <xsl:attribute name="href">
(746)                                       <xsl:value-of select="replace(@href,
      '.uml', '.html')"/>
(747)                                     </xsl:attribute>
(748)                                     <xsl:choose>
(749)                                       <xsl:when test="$cim-elem/@xmi21:type">
(750)                                         <xsl:value-of select="$cim-
      elem/@xmi21:type"/>
(751)                                       </xsl:when>
(752)                                       <xsl:otherwise>
(753)                                         <xsl:value-of select="name($cim-
      elem)"/>
(754)                                       </xsl:otherwise>
(755)                                     </xsl:choose>
(756)                                     <xsl:text> '</xsl:text>
(757)                                     <xsl:value-of select="$cim-elem/@name"/>
(758)                                     <xsl:text>'</xsl:text>
(759)                                   </xsl:element>
(760)                                   <xsl:text> (als Werteausprägung der
      Variablen '</xsl:text>
(761)                                   <xsl:value-of select="local-name()"/>
(762)                                   <xsl:text>')</xsl:text>
(763)                                 </xsl:element>
(764)                               </xsl:when>
(765)                               <xsl:when test="$cim-elem">
(766)                                 <!-- Diese Modellelemente sind hier nicht
      relevant (z.B. Upper und Lower-Value eines Assoziation) -->
(767)                               </xsl:when>
(768)                               <xsl:otherwise>
(769)                                 <xsl:element name="li">
(770)                                   <xsl:attribute name="class">
(771)                                     <xsl:text>li-verfolgbarkeit-
      transformation-element</xsl:text>
(772)                                   </xsl:attribute>
(773)                                   <xsl:text>Nicht mehr im PRM vorhandenes
      Element mit xmi:id = '</xsl:text>
(774)                                   <xsl:value-of select="$cim-elem-id"/>
(775)                                   <xsl:text>'.</xsl:text>
(776)                                 </xsl:element>
(777)                               </xsl:otherwise>
(778)                             </xsl:choose>
(779)
(780)                           </xsl:for-each>
(781)
(782)                           <!-- Prüfen, ob ohne Geschwister -->
(783)                           <xsl:if test="count(*[contains(@href, $param-prefix-
      cim)]) = 0">
(784)                             <xsl:element name="p">- Keine -</xsl:element>
(785)                           </xsl:if>
(786)                         </xsl:element>
(787)
(788)                         <!-- end: div-verfolgbarkeit-transformation-
      geschwister -->
(789)                       </xsl:element>
(790)                     <!-- end: div-verfolgbarkeit-transformation -->
(791)                     </xsl:element>
(792)
(793)                 <!-- end: Transformatonen -->
(794)                 </xsl:for-each>
(795)
```

```
(796)                 <xsl:if test="count($document-trace-
      prm2cim/xmi:XMI/node()[child::*[substring-after(@href, '#') = $id]])=0">
(797)                   <xsl:element name="p">- Keine -</xsl:element>
(798)                 </xsl:if>
(799)
(800)               <!-- end: div-verfolgbarkeit-vorwaerts -->
(801)               </xsl:element>
(802)
(803)             <!-- end: div - element -->
(804)             </xsl:element>
(805)
(806)           <!-- end: Modellelement -->
(807)           </xsl:for-each>
(808)
(809)           <!-- "Fehlende" Modellelemente des PRM -->
(810)           <xsl:element name="h1">Gelöschte Modellelemente CIM</xsl:element>
(811)           <xsl:element name="p">
(812)             <xsl:text>Die folgenden Modellelemente waren Bestandteil der
      Transformation zur Erzeugung des CIM, sind im Modell aber nicht mehr
      enthalten.</xsl:text>
(813)           </xsl:element>
(814)
(815)           <!-- Für jedes Element im rdf2prm-Trace prüfen, ob es noch
      Bestandteil des Modells ist,
(816)              wenn nicht, dann Hinweis geben. -->
(817)           <xsl:variable name="document-cim" select="/" />
(818)
(819)           <xsl:for-each select="$document-trace-
      prm2cim/xmi:XMI/node()/node()">
(820)             <xsl:variable name="trace-id" select="substring-after(@href,
      '#') "/>
(821)
(822)             <xsl:if test="count($document-cim//*[@xmi21:id = $trace-id]) =
      0">
(823)               <xsl:element name="div">
(824)                 <xsl:attribute name="class">
(825)                   <xsl:text>div-element</xsl:text>
(826)                 </xsl:attribute>
(827)
(828)               <xsl:element name="div">
(829)                 <xsl:attribute name="style" select="'{display: none;}'"/>
(830)                 <xsl:attribute name="class" select="'div-filterkriterien'"/>
(831)                 <xsl:element name="span">
(832)                   <xsl:attribute name="name" select="'fk_name_value'"/>
(833)                 </xsl:element>
(834)                 <xsl:element name="span">
(835)                   <xsl:attribute name="name" select="'fk_typ_value'"/>
(836)                 </xsl:element>
(837)                 <xsl:element name="span">
(838)                   <xsl:attribute name="name" select="'fk_abbregel_multi-
      value'"/>
(839)                     <xsl:value-of select="name(parent::*)"/>
(840)                 </xsl:element>
(841)               </xsl:element>
(842)
(843)                 <xsl:element name="h2">
(844)                   <xsl:text>Unbekanntes Modellelement (xmi:id = '</xsl:text>
(845)                   <xsl:value-of select="$trace-id"/>
(846)                   <xsl:text>')</xsl:text>
(847)                 </xsl:element>
(848)                 <xsl:element name="p">
(849)                   <xsl:text>Das in der Transformation '</xsl:text>
(850)                   <xsl:value-of select="name(parent::*)"/>
(851)                   <xsl:text>' erzeugte Modellelement mit der xmi:id =
      '</xsl:text>
(852)                   <xsl:value-of select="$trace-id"/>
(853)                   <xsl:text>' ist nach Abschluss der Transformation im
      Rahmen der CIM-Modellierung gelöscht worden.</xsl:text>
(854)                 </xsl:element>
(855)                 <saxon:assign name="elements-missing" select="$elements-
      missing+1"/>
(856)               </xsl:element>
(857)             </xsl:if>
(858)
(859)           </xsl:for-each>
```

```
(860)
(861)           <xsl:if test="$elements-missing = 0">
(862)             <xsl:element name="p">- Keine -</xsl:element>
(863)           </xsl:if>
(864)
(865)         <!-- end: div -->
(866)         </xsl:element>
(867)
(868)       <!-- end: body -->
(869)       </xsl:element>
(870)
(871)     <!-- end: html -->
(872)     </xsl:element>
(873)
(874)   </xsl:template>
(875)
(876) </xsl:stylesheet>
```

Listing 65 – Verfolgbarkeitsdokumentation des CIM (create-GovObjTraces_CIM.xsl)

G.6 Erweiterung der UML-Modelle

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(4)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(5)     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(6)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(7)     xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
(8)     xmlns:eGovRefAnf="http:///eGovRefAnf.ecore"
(9)     xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
(10)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML">
(11)
(12)    <xsl:include href="commonGovObjTraces.xsl"/>
(13)
(14)    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
(15)
(16)    <!-- xsl:param name="model-file" select="'unbekannte-datei.uml'"/ -->
(17)    <xsl:param name="param-model-file" select="'..\Result\cim_stvg_vwv-
      stvo.uml'"/>
(18)    <!--  Referenz nur auf absolute Pfade möglich -->
(19)    <xsl:param name="param-root-dir"
      select="'file:///C:/Daten/Entwicklung%20%2864%20bit%29/Promotion/GovObj_0_2/
      GovObj/bat/'"/>
(20)
(21)    <xsl:variable name="model-file" select="replace($param-model-file,'\\',
      '/')"/>
(22)    <xsl:variable name="trace-file" select="replace($model-file, '.uml',
      '.html')"/>
(23)    <xsl:variable name="root-dir">
(24)      <xsl:choose>
(25)        <xsl:when test="ends-with(replace($param-root-dir, '\\', '/'), '/')">
(26)          <xsl:value-of select="replace(replace(concat('file:///', encode-for-
      uri(replace($param-root-dir, '\\', '/'))), '%2F', '/'), '%3A', ':')"/>
(27)        </xsl:when>
(28)        <xsl:otherwise>
(29)          <xsl:value-of select="replace(replace(concat('file:///', encode-for-
      uri(replace($param-root-dir, '\\', '/')), '/'), '%2F', '/'), '%3A', ':')"/>
(30)        </xsl:otherwise>
(31)      </xsl:choose>
(32)    </xsl:variable>
(33)
(34)    <xsl:template match="/">
(35)      <xsl:apply-templates/>
(36)    </xsl:template>
(37)
(38)    <xsl:template match="@*|node()">
(39)      <xsl:copy>
(40)      <xsl:apply-templates select="@*|node()"/>
(41)      </xsl:copy>
(42)    </xsl:template>
(43)
(44)    <xsl:template match="packagedElement">
(45)      <xsl:copy>
(46)      <xsl:apply-templates select="@*|node()"/>
```

```
(47)      <xsl:element name="eAnnotations">
(48)        <xsl:attribute name="xmi:id" select="concat(generate-id(.),
      '_govobj')"/>
(49)        <xsl:attribute name="source">
(50)          <xsl:text>http://www.topcased.org/resources</xsl:text>
(51)        </xsl:attribute>
(52)        <xsl:element name="details">
(53)          <xsl:attribute name="xmi:id" select="concat(generate-
      id(.),'_dtls')"/>
(54)          <xsl:attribute name="key">
(55)            <xsl:text>RR0</xsl:text>
(56)          </xsl:attribute>
(57)          <xsl:attribute name="value" select="concat($root-dir, $trace-file ,
      '#', @xmi:id)" xmlns:govobj="http://govobjects.thomasoff.de/ns"/>
(58)          <!--  xsl:attribute name="value" select="concat($root-dir, $trace-
      file , '#', govobj:anchor($model-file, @xmi:id))"
      xmlns:govobj="http://govobjects.thomasoff.de/ns"/ -->
(59)          <!--  xsl:attribute name="value"
      select="concat(replace(replace($model-file, '.uml', '.html'),'\\', '/') ,
      '#', govobj:anchor($model-file, @xmi:id))"
      xmlns:govobj="http://govobjects.thomasoff.de/ns"/ -->
(60)          <!--  xsl:attribute name="value" select="concat(replace($model-file,
      '.uml','.html'), '#', replace($model-file,'\.','_'),'_', @xmi:id)"/ -->
(61)        </xsl:element>
(62)      </xsl:element>
(63)      </xsl:copy>
(64)    </xsl:template>
(65)
(66)  </xsl:stylesheet>
```

Listing 66 – Datei „add-GovObjTraces.xsl“

G.7 Gemeinsam benutzte Funktionen

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns="http://www.w3.org/1999/xhtml"
(4)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(5)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(6)     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(7)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(8)     xmlns:xmi="http://www.omg.org/XMI"
(9)     xmlns:xmi1="http://schema.omg.org/spec/XMI/2.1"
(10)    xmlns:eGovRefAnf="http:///eGovRefAnf.ecore"
(11)    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
(12)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
(13)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(14)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(15)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(16)    xmlns:exslt="http://exslt.org/common"
(17)      extension-element-prefixes="exslt">
(18)
(19)    <xsl:function name="govobj:anchor" as="xs:string">
(20)      <xsl:param name="file" as="xs:string"/>
(21)      <xsl:param name="id" as="xs:string"/>
(22)      <xsl:sequence select="concat(replace(functx:substring-after-
      last(replace($file, '\\', '/'), '/'),'\.uml','_uml'), '_', $id)"
      xmlns:functx="http://www.functx.com" />
(23)    </xsl:function>
(24)
(25)    <xsl:function name="govobj:rdf-label" as="xs:string">
(26)      <xsl:param name="elem"/>
(27)      <xsl:variable name="type" select="govobj:rdf-type($elem)"/>
(28)      <xsl:variable name="name" select="govobj:rdf-name($elem)"/>
(29)      <xsl:variable name="label">
(30)        <xsl:text>Individuum '</xsl:text>
(31)        <xsl:value-of select="$name"/>
(32)        <xsl:text>' der Ontologie-Klasse '</xsl:text>
(33)        <xsl:value-of select="$type"/>
(34)        <xsl:text>'</xsl:text>
(35)      </xsl:variable>
(36)      <xsl:sequence select="$label"/>
(37)    </xsl:function>
(38)
(39)    <xsl:function name="govobj:rdf-name" as="xs:string">
```

```
(40)      <xsl:param name="elem"/>
(41)      <xsl:variable name="name">
(42)        <xsl:choose>
(43)          <xsl:when test="$elem/rdfs:label">
(44)            <xsl:value-of  select="$elem/rdfs:label"/>
(45)          </xsl:when>
(46)          <xsl:otherwise>
(47)           <xsl:value-of select="substring-after($elem/@rdf:about, '#')"/>
(48)          </xsl:otherwise>
(49)        </xsl:choose>
(50)      </xsl:variable>
(51)      <xsl:sequence select="$name"/>
(52)    </xsl:function>
(53)
(54)    <xsl:function name="govobj:rdf-type" as="xs:string">
(55)      <xsl:param name="elem"/>
(56)      <xsl:variable name="type" select="local-name($elem[1])"/>
(57)      <xsl:sequence select="$type"/>
(58)    </xsl:function>
(59)
(60)    <xsl:function name="govobj:file" as="xs:string">
(61)      <xsl:param name="href" as="xs:string"/>
(62)      <xsl:value-of select="substring-before($href, '#')"/>
(63)    </xsl:function>
(64)
(65)    <xsl:function name="govobj:id" as="xs:string">
(66)      <xsl:param name="href" as="xs:string"/>
(67)      <xsl:value-of select="substring-after($href, '#')"/>
(68)    </xsl:function>
(69)
(70)    <xsl:function name="govobj:type" as="xs:string">
(71)      <xsl:param name="elem"/>
(72)      <xsl:variable name="type">
(73)        <xsl:choose>
(74)          <xsl:when test="$elem/@xmi1:type">
(75)            <xsl:value-of select="$elem/@xmi1:type"/>
(76)          </xsl:when>
(77)          <xsl:otherwise>
(78)            <xsl:choose>
(79)              <xsl:when test="contains(substring-before(name($elem), ':'),
      'eGovRef')">
(80)                <xsl:text>Stereotype</xsl:text>
(81)              </xsl:when>
(82)              <xsl:otherwise>
(83)                <xsl:value-of select="local-name($elem)"/>
(84)              </xsl:otherwise>
(85)            </xsl:choose>
(86)          </xsl:otherwise>
(87)        </xsl:choose>
(88)      </xsl:variable>
(89)      <xsl:sequence select="$type"/>
(90)    </xsl:function>
(91)
(92)    <xsl:function name="govobj:label" as="xs:string">
(93)      <xsl:param name="elem"/>
(94)      <xsl:param name="id" as="xs:string"/>
(95)      <xsl:variable name="type" select="govobj:type($elem)"/>
(96)      <xsl:variable name="name">
(97)        <xsl:choose>
(98)          <xsl:when test="$elem/@name">
(99)            <xsl:value-of select="$type"/>
(100)           <xsl:text> '</xsl:text>
(101)           <xsl:value-of select="$elem/@name"/>
(102)           <xsl:text>'</xsl:text>
(103)         </xsl:when>
(104)         <xsl:otherwise>
(105)           <xsl:choose>
(106)             <xsl:when test="contains(substring-before(name($elem[1]), ':'),
      'OwlUmlProfile')">
(107)               <xsl:value-of select="$type"/>
(108)               <xsl:text> '</xsl:text>
(109)               <xsl:value-of select="local-name($elem)"/>
(110)               <xsl:text>'</xsl:text>
(111)             </xsl:when>
(112)             <xsl:otherwise>
```

```
(113)               <xsl:value-of select="$type"/>
(114)               <xsl:text> 'Ohne Name' (xmi:id = '</xsl:text>
(115)               <xsl:value-of select="$id"/>
(116)               <xsl:text>')</xsl:text>
(117)             </xsl:otherwise>
(118)           </xsl:choose>
(119)         </xsl:otherwise>
(120)       </xsl:choose>
(121)     </xsl:variable>
(122)     <xsl:sequence select="$name"/>
(123)   </xsl:function>
(124)
(125)   <xsl:function name="govobj:name" as="xs:string">
(126)     <xsl:param name="elem"/>
(127)     <xsl:param name="id" as="xs:string"/>
(128)     <xsl:variable name="type" select="govobj:type($elem)"/>
(129)     <xsl:variable name="name">
(130)       <xsl:choose>
(131)         <xsl:when test="$elem/@name">
(132)           <xsl:value-of select="$elem/@name"/>
(133)         </xsl:when>
(134)         <xsl:otherwise>
(135)           <xsl:choose>
(136)             <xsl:when test="contains(substring-before(name($elem), ':'),
      'OwlUmlProfile')">
(137)               <xsl:value-of select="local-name($elem)"/>
(138)             </xsl:when>
(139)             <xsl:otherwise>
(140)               <xsl:text> Ohne Name (xmi:id = '</xsl:text>
(141)               <xsl:value-of select="$id"/>
(142)               <xsl:text>')</xsl:text>
(143)             </xsl:otherwise>
(144)           </xsl:choose>
(145)         </xsl:otherwise>
(146)       </xsl:choose>
(147)     </xsl:variable>
(148)     <xsl:sequence select="$name"/>
(149)   </xsl:function>
(150)
(151)   <xsl:function name="govobj:documentation" as="xs:string">
(152)     <xsl:param name="elem"/>
(153)     <xsl:variable name="doc">
(154)       <xsl:choose>
(155)         <xsl:when
      test="$elem/eAnnotations[1][@source='http://www.topcased.org/documentation']
      ">
(156)           <xsl:value-of select="$elem/eAnnotations[1]/details/@value"/>
(157)         </xsl:when>
(158)         <xsl:otherwise>
(159)           <xsl:text>- Keine -</xsl:text>
(160)         </xsl:otherwise>
(161)       </xsl:choose>
(162)     </xsl:variable>
(163)     <xsl:sequence select="$doc"/>
(164)   </xsl:function>
(165)
(166)   <!--
(167)   Autor: Priscilla Walmsley, Datypic, pwalmsley@datypic.com,
      http://www.datypic.com
(168)   Quelle: http://www.xsltfunctions.com/xsl/functx_substring-after-last-
      match.html
(169)   The source code for this function is available under the GNU LGPL license.
(170)   -->
(171)   <xsl:function name="functx:substring-after-last" as="xs:string"
      xmlns:functx="http://www.functx.com" >
(172)     <xsl:param name="arg" as="xs:string?"/>
(173)     <xsl:param name="delim" as="xs:string"/>
(174)     <xsl:sequence select="replace ($arg,concat('^.*',functx:escape-for-
      regex($delim)),'')"/>
(175)   </xsl:function>
(176)
(177)   <!--
(178)   Autor: Priscilla Walmsley, Datypic, pwalmsley@datypic.com,
      http://www.datypic.com
(179)   Quelle: http://www.xsltfunctions.com/xsl/functx_escape-for-regex.html
```

```
(180)   The source code for this function is available under the GNU LGPL license.
(181)   -->
(182)   <xsl:function name="functx:escape-for-regex" as="xs:string"
      xmlns:functx="http://www.functx.com" >
(183)       <xsl:param name="arg" as="xs:string?"/>
(184)       <xsl:sequence select="replace($arg,'(\.|\[|\]|\\|\||\-
      |\^|\$|\?|\*|\+|\{|\}|\(|\))','\\$1')"/>
(185)   </xsl:function>
(186)
(187) </xsl:stylesheet>
```

Listing 67 – Gemeinsam genutzt Funktionen in der Datei „commonGovObjTraces.xsl“

```
(1)   <?xml version="1.0" encoding="UTF-8"?>
(2)   <xsl:stylesheet version="2.0"
(3)     xmlns="http://www.w3.org/1999/xhtml"
(4)     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(5)     xmlns:xs="http://www.w3.org/2001/XMLSchema"
(6)     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
(7)     xmlns:fn="http://www.w3.org/2005/xpath-functions"
(8)     xmlns:xmi="http://www.omg.org/XMI"
(9)     xmlns:xmi1="http://schema.omg.org/spec/XMI/2.1"
(10)    xmlns:eGovRefAnf="http:///eGovRefAnf.ecore"
(11)    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
(12)    xmlns:uml="http://www.eclipse.org/uml2/3.0.0/UML"
(13)    xmlns:govobj="http://govobjects.thomasoff.de/ns"
(14)    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(15)    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
(16)    xmlns:exslt="http://exslt.org/common"
(17)      extension-element-prefixes="exslt">
(18)
(19)    <!--
(20)      Verkürzt lange Zeichenketten
(21)
(22)      Ist die übergebene Zeichenkette länger als 7 Zeichen, wird sie
(23)      auf 5 Zeichen gekürzt und ihre ursprüngliche Gesamtlänge angehängt.
(24)     -->
(25)    <xsl:function name="govobj:shorten" as="xs:string">
(26)      <xsl:param name="str"/>
(27)      <xsl:variable name="short">
(28)        <xsl:choose>
(29)          <xsl:when test="string-length($str) &gt; 7">
(30)            <xsl:value-of select="concat(substring($str, 1, 5), string-
      length($str))"/>
(31)          </xsl:when>
(32)          <xsl:otherwise>
(33)            <xsl:value-of select="$str"/>
(34)          </xsl:otherwise>
(35)        </xsl:choose>
(36)      </xsl:variable>
(37)      <xsl:sequence select="$short"/>
(38)    </xsl:function>
(39)
(40)    <!--
(41)      Ersetzt die Umlaute
(42)
(43)      Im übergebenen String werden nach folgendem Muster die Umlaute
(44)      ersetzt: ä = ae, Ä = Ae, ß = ss.
(45)     -->
(46)    <xsl:function name="govobj:replace-umlaute" as="xs:string">
(47)      <xsl:param name="str"/>
(48)      <xsl:sequence
      select="replace(replace(replace(replace(replace(replace(replace($str, 'ß',
      'ss'), 'ä', 'ae'), 'ö', 'oe'), 'ü', 'ue'), 'Ä', 'Ae'), 'Ö', 'oe'), 'Ü',
      'Ue')"/>
(49)    </xsl:function>
(50)
(51)    <!--
(52)    Autor: Priscilla Walmsley, Datypic, pwalmsley@datypic.com,
      http://www.datypic.com
(53)    Quelle: http://www.xsltfunctions.com/xsl/functx_substring-after-last-
      match.html
(54)    The source code for this function is available under the GNU LGPL license.
(55)    -->
```

```
(56)    <xsl:function name="functx:substring-after-last" as="xs:string"
      xmlns:functx="http://www.functx.com" >
(57)      <xsl:param name="arg" as="xs:string?"/>
(58)      <xsl:param name="delim" as="xs:string"/>
(59)      <xsl:sequence select="replace ($arg,concat('^.*',functx:escape-for-
      regex($delim)),'')"/>
(60)    </xsl:function>
(61)
(62)
(63)    <!--
(64)    Autor: Priscilla Walmsley, Datypic, pwalmsley@datypic.com,
      http://www.datypic.com
(65)    Quelle: http://www.xsltfunctions.com/xsl/functx_escape-for-regex.html
(66)    The source code for this function is available under the GNU LGPL license.
(67)    -->
(68)    <xsl:function name="functx:escape-for-regex" as="xs:string"
      xmlns:functx="http://www.functx.com" >
(69)        <xsl:param name="arg" as="xs:string?"/>
(70)        <xsl:sequence select="replace($arg,'(\.|\[|\]|\\|\||\-
      |\^|\$|\?|\*|\+|\{|\}|\(|\))','\\$1')"/>
(71)    </xsl:function>
(72)
(73)  </xsl:stylesheet>
```

Listing 68 – Gemeinsam genutzt Funktionen in der Datei „commo.xsl“

```
(1)   /**
(2)    * Füllen der Filterkriterien mit den tatsächlich vorhandenen Werten
(3)    */
(4)    function populateFilter() {
(5)      var form = document.forms['filter'];
(6)      var selName = form.elements['fk_name_select'];
(7)      var selTyp = form.elements['fk_typ_select'];
(8)      var selAbbRegel = form.elements['fk_abbregel_select'];
(9)
(10)    var elems= document.getElementsByTagName("SPAN");
(11)    var alleNamen = new Array();
(12)    var alleTypen = new Array();
(13)    var alleAbbRegeln = new Array();
(14)
(15)    alleNamen[0] = "";
(16)    alleTypen[0] = "";
(17)    alleAbbRegeln[0] = "";
(18)
(19)    if (elems != null) {
(20)      if (elems.length != null) {
(21)        for (i = 0; i < elems.length; i++) {
(22)          if (elems[i].name == 'fk_name_value') {
(23)             alleNamen[alleNamen.length] = elems[i].innerHTML;
(24)          }
(25)
(26)          if (elems[i].name == 'fk_typ_value') {
(27)            alleTypen[alleTypen.length] = elems[i].innerHTML;
(28)          }
(29)
(30)          if (elems[i].name == 'fk_abbregel_multi-value') {
(31)            alleAbbRegeln[alleAbbRegeln.length] = elems[i].innerHTML;
(32)          }
(33)
(34)        }
(35)      }
(36)    }
(37)    var namen = removeDuplicates(alleNamen);
(38)    var typen = removeDuplicates(alleTypen);
(39)    var abbRegeln = removeDuplicates(alleAbbRegeln);
(40)
(41)     for (i = 0; i < namen.length; i++) {
(42)       selName.options[selName.options.length] = new Option(namen[i]);
(43)     }
(44)
(45)     for (i = 0; i < typen.length; i++) {
(46)       selTyp.options[selTyp.options.length] = new Option(typen[i]);
(47)     }
(48)
(49)     for (i = 0; i < typen.length; i++) {
```

```
(50)       selAbbRegel.options[selAbbRegel.options.length] = new
      Option(abbRegeln[i]);
(51)     }
(52)
(53)   }
(54)
(55)   /**
(56)    * Hilfsfunktion zum Entfernen doppelter Einträge eines Arrays
(57)    */
(58)   function removeDuplicates(array) {
(59)    var result=new Array();
(60)
(61)    for(i=0; i<array.length; i++) {
(62)      var contains = 0;
(63)    for(j=0; j<result.length; j++ ) {
(64)
(65)      if(result[j]==array[i]) {
(66)        contains = 1;
(67)        break;
(68)       }
(69)
(70)     }
(71)     if (contains == 0) {
(72)       result[result.length] = array[i];
(73)     }
(74)
(75)    }
(76)    return result;
(77)  }
(78)
(79)
(80)   /**
(81)    * Zurücksetzen des Filters (inkl. des Formulars und der Elementauswahl)
(82)    */
(83)   function resetFilter() {
(84)     resetForm();
(85)     resetElements();
(86)     document.getElementById('span-filtermessage').innerHTML = 'Filter
      zur&uuml;ckgesetzt.';
(87)   }
(88)
(89)   /**
(90)    * Zurücksetzen des Formulars und Befüllen der Auswahlkriterien
(91)    */
(92)   function resetForm() {
(93)      populateFilter();
(94)     var form = document.forms['filter'];
(95)     form.elements['fk_name_select'].selectedIndex = 0;
(96)     form.elements['fk_typ_select'].selectedIndex = 0;
(97)     form.elements['fk_abbregel_select'].selectedIndex = 0;
(98)   }
(99)
(100)  /**
(101)   * Zurücksetzen der Filternachricht.
(102)   */
(103)  function resetMessage() {
(104)    document.getElementById('span-filtermessage').innerHTML = '';
(105)  }
(106)
(107)  /**
(108)   * Zurcüksetzen der gefilterten Elemente
(109)   */
(110)  function resetElements() {
(111)    var elems = document.getElementsByTagName("SPAN");
(112)
(113)    if (elems != null) {
(114)      if (elems.length != null) {
(115)        for (i = 0; i < elems.length; i++) {
(116)          if (elems[i].name != null) {
(117)            if (elems[i].name.indexOf("_value") > -1) {
(118)              elems[i].parentNode.parentNode.style.display = 'block';
(119)            }
(120)          }
(121)        }
(122)      }
```

```
(123)    }
(124)
(125)  }
(126)
(127)  /**
(128)   * Filtern der Elemente nach allen Kriterien
(129)   */
(130)  function filtern() {
(131)
(132)    resetMessage();
(133)
(134)    resetElements();
(135)
(136)    msg = 'Gefiltert werden Elemente mit ';
(137)    msg1 = '';
(138)    msg2 = '';
(139)    msg3 = '';
(140)
(141)    var form = document.forms['filter'];
(142)    var sel1 = form.elements['fk_name_select'];
(143)    var idx1 = sel1.selectedIndex;
(144)    var ttl1 = sel1.title;
(145)    var val1 = sel1.options[idx1].text;
(146)
(147)    if (val1.length>0) {
(148)      msg1 = ttl1 + " = '" + val1 + "'";
(149)      filterElement(sel1.name, val1);
(150)    }
(151)
(152)
(153)    var sel2 = form.elements['fk_typ_select'];
(154)    var idx2 = sel2.selectedIndex;
(155)    var ttl2 = sel2.title;
(156)    var val2 = sel2.options[idx2].text;
(157)
(158)    if (val2.length>0) {
(159)      msg2 = ttl2 + " = '" + val2 + "'";
(160)      if (msg1.length > 0) {
(161)        msg2 = " und " + msg2;
(162)      }
(163)      filterElement(sel2.name, val2);
(164)    }
(165)
(166)    var sel3 = form.elements['fk_abbregel_select'];
(167)    var idx3 = sel3.selectedIndex;
(168)    var ttl3 = sel3.title;
(169)    var val3 = sel3.options[idx3].text;
(170)
(171)    if (val3.length>0) {
(172)      msg3 = ttl3 + " = '" + val3 + "'";
(173)      if (msg1.length + msg2.length > 0) {
(174)        msg3 = " und " + msg3;
(175)      }
(176)      filterMultiElement(sel3.name, val3);
(177)    }
(178)
(179)    msg = msg + msg1 + msg2 + msg3 + ".";
(180)
(181)    document.getElementById('span-filtermessage').innerHTML = msg;
(182)  }
(183)
(184)  /**
(185)   * Filtern der Elemente nach einem Kriterium
(186)   */
(187)  function filterElement(name, value) {
(188)
(189)    var elems= document.getElementsByTagName("SPAN");
(190)    var name = name.replace("_select", "_value");
(191)
(192)    if (elems != null) {
(193)      if (elems.length != null) {
(194)        for (i = 0; i < elems.length; i++) {
(195)          if (elems[i].name == name) {
(196)             if (elems[i].innerHTML.indexOf(value) == -1) {
(197)               elems[i].parentNode.parentNode.style.display = 'none';
```

```
(198)             }
(199)           }
(200)         }
(201)
(202)       }
(203)
(204)     }
(205)   }
(206)
(207)   /**
(208)    * Filtern der Elemente nach einem Kriterium
(209)    */
(210)   function filterMultiElement(name, value) {
(211)
(212)     var elems= document.getElementsByTagName("SPAN");
(213)     var name = name.replace("_select", "_multi-value");
(214)     var hide = 1;
(215)
(216)     if (elems != null) {
(217)       if (elems.length != null) {
(218)        for (i = 0; i < elems.length; i++) {
(219)          if (elems[i].name == name) {
(220)            if (elems[i].innerHTML.indexOf(value) > -1) {
(221)              hide = 0;
(222)            }
(223)
(224)            if (i > 0) {
(225)              if (elems[i].parentNode != elems[i-1].parentNode) {
(226)                if (hide == 1) {
(227)                  elems[i].parentNode.parentNode.style.display = 'none';
(228)                } else {
(229)                  hide = 1;
(230)                }
(231)              }
(232)            }
(233)          }
(234)        }
(235)       }
(236)     }
(237)   }
```

Listing 69 – JavaScript-Funktionen zum Filtern der Verfolgbarkeitsdokumentation des PRM und CIM (govobj-filtern.js)

H Eclipse-Umgebung

Die nachfolgende Tabelle gibt einen Überblick über die Versions- und Konfigurationsangaben der eingesetzten Eclipse-Umgebung.[407]

Komponente	Versionsnummer	Installationspaket
Eclipse SDK	3.6.2.M20110210-1200	org.eclipse.sdk.ide
Eclipse XML Editors and Tools	3.2.3.v201102160550-7H7AFUWDxumQJOi9ghcTb5YgkwEZ	org.eclipse.wst.xml_ui.feature.feature.group
Ecore OCL Search (Incubation)	0.7.0.v200806130939	org.eclipse.emf.search.ecore.ocl.feature.group
Ecore Tools SDK (Incubation)	0.10.0.v20100615-1639	org.eclipse.emf.ecoretools.sdk.feature.group

[407] Darüber hinaus wurden manuell die Komponenten des Werkzeugs mediniQVT in Version 1.7.0 RC2 in die Umgebung integriert, so dass innerhalb einer Eclipse-Installation mit allen Komponenten gearbeitet werden konnte. Für den Ansatz dieser Arbeit ist diese Integration jedoch nicht zwingend. Die Arbeit in getrennten Werkzeugen ist ebenfalls möglich.

EMF - Eclipse Modeling Framework SDK	2.6.1.v20100914-1218	org.eclipse.emf.sdk.feature.group
EMF Search OCL (Incubation)	0.7.0.v200806130939	org.eclipse.emf.search.ocl.feature.group
EMF Validation Framework SDK	1.4.0.v20100428-2315-67J-96SGR35WNqZRZfmQnghI6uFA	org.eclipse.emf.validation.sdk.feature.group
Graphical Editing Framework GEF SDK	3.6.2.v20110128-0100-7G7R77A5WNcHQDbhX8JWOYLOSeRJ	org.eclipse.gef.sdk.feature.group
Graphical Modeling Framework (GMF) Notation SDK	1.4.1.v20100909-1000-47B28xGC7DxS9Q69KFQ3AHJL9t84	org.eclipse.gmf.runtime.notation.sdk.feature.group
Graphical Modeling Framework (GMF) Runtime	1.4.2.v20110201-1000-7d9A7FFYnmMD7_nYQqNhbc2tsgXc	org.eclipse.gmf.feature.group
Graphical Modeling Framework (GMF) Runtime Examples	1.4.1.v20100909-1300-7B7L1F8NcJP-Jz-Qb9Sz0mY	org.eclipse.gmf.examples.runtime.feature.group
Graphical Modeling Framework (GMF) Runtime SDK	1.4.2.v20110201-1000-67G49E9QYBfkXgwmVt5ksk7w9Vqi	org.eclipse.gmf.runtime.sdk.feature.group
Java Extension For UML Editor	4.3.0.201102071539	org.topcased.uml.java.feature.group
Object Constraint Language (OCL) 2.0 Compatibility Source	1.1.101.v200901271800-208_7w311A12382911	org.eclipse.emf.ocl.source.feature.group
OCL Examples and Editors	3.0.2.R30x_v201101181635-86-7aF9Rn28mFQj3xjxEv38kn	org.eclipse.ocl.examples.feature.group
OCL Extender SDK	3.0.2.R30x_v201101181635-7J-7A8j_CDZUhIyiEJYRHlnWupBh	org.eclipse.ocl.all.sdk.feature.group
Requirement core	4.3.0.201102071538	org.topcased.requirement.feature.feature.group
Requirements for Topcased Editors	4.3.0.201102071538	org.topcased.requirement.topcased.feature.feature.group

Topcased Core Feature	4.3.0.201102071538	org.topcased.core.feature.feature.group
Topcased Document Importer	4.3.0.201102071538	org.topcased.documentimporter.feature.group
Topcased Document Importer for UML	1.0.0.201102071538	org.topcased.documentimporter.uml.feature.group
Topcased Modeler Toolkit	4.3.0.201102071538	org.topcased.modeler.toolkit.feature.group
Topcased OCL Tools	4.3.0.201102071538	org.topcased.ocl.feature.feature.group
Topcased Tramway (incubation)	1.0.0.201102071539	org.topcased.tramway.feature.group
Topcased UML2 Editors	4.3.0.201102071539	org.topcased.uml.feature.group
UML2 Diagramming (Incubation)	0.10.0.201103290534	org.eclipse.uml2.diagram.feature.group
UML2 Diagramming Samples (Incubation)	0.8.0.201103290534	org.eclipse.uml2tools.examples.feature.group
UML2 Documentation	3.1.0.v201005281015	org.eclipse.uml2.doc.feature.group
UML2 End-User Features	3.1.2.v201010261927	org.eclipse.uml2.feature.group
UML2 Examples	3.1.1.v201008191505	org.eclipse.uml2.examples.feature.group
UML2 Examples Source	3.1.1.v201008191505	org.eclipse.uml2.examples.source.feature.group
UML2 Extender SDK	3.1.2.v201010261927	org.eclipse.uml2.sdk.feature.group
UML2 OCL Search (Incubation)	0.7.0.v200806130939	org.eclipse.uml2.search.ocl.feature.group
UML2 Source	3.1.2.v201010261927	org.eclipse.uml2.source.feature.group
UML2 Tools (Incubation)	0.10.0.201103290534-7O8--cLTxI-3122143_1645	org.eclipse.uml2tools.feature.group
UML2 Tools SDK (Incubation)	0.10.0.201103290534-07Q-cLcqw-7FTE64nUr0X_F23325	org.eclipse.uml2tools.sdk.feature.group

Tabelle 58 – Versions- und Konfigurationsangaben der eingesetzten Eclipse-Komponenten

Abbildungsverzeichnis

Tabellenverzeichnis

Listingverzeichnis

Literatur

[Abel, 2007] A. Abel: Compliance Engineering. In: Claus Rautenstrauch (Hrsg.): Die Zukunft der Anwendungssoftware – die Anwendungssoftware der Zukunft, Magdeburger Schriften zur Wirtschaftsinformatik; 2007; ISBN 978-3-8322-6328-7.

[Adida, 2008] B. Adida. hGRDDL: Bridging microformats and RDFa. In: Journal of Web Semantics: Science, Services and Agents on the World Wide Web; Vol. 6, No. 1, Feb. 2008; ISN 1570-8268; S. 54-60.

[Äko, 2001] o. A.: Ergebnisprotokoll zur Sitzung der Änderungskonferenz zum Entwicklungsstandart für IT-Systeme des Bundes (EStdIT) am 31.1. und 1.2.2001; BMI; Bonn; 2001.

[Antón, 1996] A. I. Antón. Goal-Based Requirements Analysis. In: L. Macaulay (Hrsg.): Requirements for Requirements Engineering Technique. Proceedings of IEEE Second International Conference on Requirements Engineering (ICRE'96); Springer-Verlag, London, UK; 1996; ISBN 3-540-76006-7; S. 136-144

[Amsden, 2008] J. Amsden: Capturing requirements with Business Motivation Model, IBM Rational RequisitePro and IBM Rational Software Modeler - Use these tools to better understand the who, what, why, and how of business requirements; IBM developerWorks; 2008; http://www.ibm.com/developerworks/rational/library/08/0401_amsden/, letzter Aufruf: 21.08.2011.

[Amstel et al., 2008] M.F. van Amstel, M.G.J. van den Brand, Z. Protic, T. Verhoeff: Transforming Process Algebra Models into UML State Machines: Bridging a Semantic Gap?. In: A. Vallecillo, J. Gray, A. Pierantonio (Eds.): Theory and Practice of Model Transformations, First International Conference on Model Transformation (ICMT 2008), ETH Zürich, Schweiz; LNCS 5063; Springer-Verlag Berlin, Heidelberg; 2008; ISBN 978-3-540-69926-2; S. 61-75

[Baclawski et al., 2002a] K. Baclawski, M. M. Kokar, P. A. Kogut, L. Hart, J. E. Smith, J. Letkowski, P. Emery: Extending the unified modeling language for ontology development. Software and System Modeling; Vol. 1; No. 2; 2002; S. 142-156.

[Baclawski et al., 2002b] K. Baclawski, M. M. Kokar, J. E. Smith, E. Wallace, J. Letkowski, M. R. Koethe, and P. Kogut , UOL: Unified Ontology Language, Assorted papers discussed at the DC Ontology SIG meeting; 2002; http://www.omg.org/cgi-bin/doc?ontology/2002-11-02 (letzter Aufruf: 21.08.2011).

[Balzert, 1996] H. Balzert: Lehrbuch der Software-Technik: Software-Entwicklung; Spektrum, Akad. Verlag, Heidelberg, Berlin, Oxford; 1996; ISBN 3-8274-0042-2.

[BBergG] Bundesberggesetz vom 13. August 1980 (BGBl. I S. 1310), das zuletzt durch Artikel 15a des Gesetzes vom 31. Juli 2009 (BGBl. I S. 2585) geändert worden ist; http://www.gesetze-im-

internet.de/bbergg/BJNR013100980.html (letzter Aufruf: 18.08.2011).

[BDSG] Bundesdatenschutzgesetz in der Fassung der Bekanntmachung vom 14. Januar 2003 (BGBl. I S. 66), das zuletzt durch Artikel 1 des Gesetzes vom 14. August 2009 (BGBl. I S. 2814) geändert worden ist; http://www.gesetze-im-internet.de/bdsg_1990/ BJNR029550990.html (letzter Aufruf: 18.08.2011).

[Becker, 1989] B. Becker: Öffentliche Verwaltung: Lehrbuch für Wissenschaft und Praxis; Schulz; Percha, Kempfenhausen am Starnberger See; 1989; ISBN 3-7962-0346-9.

[Becker et al., 2007] J. Becker, D. Pfeiffer, M. Räckers: PICTURE – A new Approach for Domain-Specific Process Modelling. In: Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE'07), 11-15 Jun. 2007, Trondheim, Norwegen; 2007; http://sunsite.informatik.rwth-aachen.de/Publications/ CEUR-WS/Vol-247/FORUM_12.pdf (letzter Aufruf: 21.08.2011).

[Berkem, 1999] B. Berkem: Traceability Management from Business Processes to Use Cases with UML – A Proposal for Extensions to the UML's Activity Diagram Through Goal-Oriented Objects. In: Journal of Object-Oriented Programming; Vol. 12, No. 5, Sept. 1999; ISSN 0896-8438; S. 29-34

[Berenbach&Gall, 2006] B. Berenbach, M. Gall: Toward a Unified Model for Requirements Engineering. In: Proceedings of the IEEE international conference on Global Software Engineering 2006 (ICGSE'06); IEEE Computer Society Washington, DC, USA; 2006; ISBN 0-7695-2663-2; S. 237-238

[Berg, 1968] M. v. Berg: Automationsgerechte Rechts- und Verwaltungs-vorschriften. Grote; Köln, Berlin; 1968.

[Bierling, 1911] E. R. Bierling: Jurisitische Prinzipienlehre. Bd. 4, 1911 (Nachdruck Aalen 1979).

[BMJ, 1999] o.A.: Handbuch der Rechtsförmigkeit, Hsg. vom Bundesministerium der Justiz, 2. Auflage 1999, Az: 1020/3-8-2.

[BMI, 1973] Bundesministerium des Innern: Grundstäze für die Gestaltung automationsgeeigneter Rechts- und Verwaltungsvorschriften, vom 22.11.1973, Bekanntmachung des Bundesministerium des Innern (GMBl. 1973, S. 555).

[BMI SAGA, 2008] Bundesministerium des Innern (Hrsg.): SAGA - Standards und Architekturen für E-Government-Anwendungen, Version 4.0; März 2008; http://www.cio.bund.de/SharedDocs/Publikationen/DE/ Standards_und_Architekturen/saga_4_0_download.pdf?__ blob=publicationFile (letzter Aufruf: 21.08.2011).

[BMI SAGA, 2010] Bundesministerium des Innern (Hrsg.): SAGA - Standards und Architekturen für E-Government-Anwendungen,

Aktualisierte Version 4.0; Juli 2010; http://www.cio.bund.de/SharedDocs/Publikationen/DE/Standards_und_Architekturen/saga_4_0_update_kapitel_8_download.pdf?__blob=publicationFile (letzter Aufruf: 21.08.2011).

[Bode et al., 2011] S. Bode, S. Lehnert, M. Riebisch: Comprehensive Model Integration for Dependency Identification with EMFTrace. in: Proceesings of the Workshop Model-Driven Software Migration at CSMR2011, Oldenburg, Deutschland, 1.-4. März 2011. (zur Veröffentlichung vorgesehen)

[Boer et al., 2006] Alexander Boer, Marcello Di Bello, Kasper van den Berg, T. Gordon, A. Förhécz, R. Vas: Specification of the Legal Knowledge Interchange Format, Deliverable 1.1. ESTRELLA-Projekt (IST-2004-027655), 22. Januar 2007; http://www.estrellaproject.org/doc/D1.1-LKIF-Specification.pdf (letzter Aufruf: 21.08.2011).

[Botella et al., 2001] P. Botella, X. Burgués, X. Franch, M. Huerta, G. Salazar: Modeling Non-Functional Requirements. In: Proceedings of Jornadas de Ingenieria de Requisitos Aplicada JIRA; 2001.

[BPMI, 2004] Business Process Management Initiative (BPMI): Business Process Modeling Notation (BPMN) Version 1.0; 3. Mai 2004; http://www.bpmn.org/Documents/BPMN_V1-0_May_3_2004.pdf (letzter Aufruf: 21.08.2011).

[Brcina, 2006] R. Brcina: Arbeiten zur Verfolgbarkeit und Aspekte des Verfolgbarkeitsprozesses. In: Softwaretechnik-Trends; Gesellschaft für Informatik e.V.; Band 27 Heft 1; 2006; ISSN 0720-8928; http://www.theoinf.tu-ilmenau.de/~riebisch/traceability/20060128_brcina_paper_traceability.pdf (letzter Aufruf: 21.08.2011).

[Breaux, 2006] T. D. Breaux: Compliance Engineering: Aligning Software Requirements with Policies and Regulations. Doctoral Symposium at the ACM/SIGSOFT 14th Symp. on Foundations of Software Engineering (FSE-14), Portland, Oregon, USA; Nov. 2006

[Breaux, 2009] T. D. Breaux. Legal Requirements Acquisition for the Specification of Legally Compliant Information Systems. Ph.D. Thesis, North Carolina State University, USA; Apr. 2009

[Brenders-Lee et al., 2001] T. Berners-Lee, J. Hendler und O. Lassila. The semantic web. Scientific American; Mai 2001; S. 96 - 101; http://www.scientificamerican.com/article.cfm?id=the-semantic-web (letzter Aufruf: 21.08.2011).

[Breuker et al., 2002] J. Breuker, A. Elhag, E. Petkov, R. Winkels: Ontologies for Legal Information Serving and Knowledge Management. In: T.J.M. Bench-Capon, A. Daskalopulu and R.G.F. Winkels (Hrsg.): Legal Knowledge and Information Systems (JURIX 2002): The Fifteenth Annual Conference. IOS Press, Amsterdam; 2002; ISBN 1-5860-3299-2; S. 73-82; http://www.jurix.nl/pdf/j02-08.pdf (letzter Aufruf: 21.08.2011).

[Breuker et al., 2007] J. Breuker, R. Hoekstra, A. Boer (Hrsg.): OWL Ontology of Basic Legal Concepts (LKIF-Core) Deliverable 1.4 ESTRELLA-Projekt (IST-2004-027655), 22. Januar 2007, http://www.estrellaproject.org/doc/D1.4-OWL-Ontology-of-Basic-Legal-Concepts.pdf (letzter Aufruf: 21.08.2011).

[Breuker&Hoekstra, 2004a] J. Breuker, R. Hoekstra: Core concepts of law: taking common sense seriously, In: Formal Ontologies in Information Systems: Proceedings of the Third International Conference (FOIS-2004) (Frontiers in Artificial Intelligence and Applications); IOS-Press; 2004; ISBN 1-5860-3468-5; S. 210-221. http://www.lri.jur.uva.nl/docs/breuker/fois-2004-paper.pdf (letzter Aufruf: 21.08.2011).

[Breuker&Hoekstra, 2004b] J. Breuker, R. Hoekstra: DIRECT: Ontology-based Discovery of Responsibility and Causality in Legal Case Descriptions. In: T. Gordon (Hrsg.): Legal Knowledge and Information Systems (JURIX 2004): The Seventeenth Annual Conference. IOS Press, Amsterdam, Niederlande; 2004; ISBN 1-5860-3492-8; S. 59-68. http://www.jurix.nl/pdf/j04-08.pdf (letzter Aufruf: 21.08.2011).

[Brockmans et al., 2004] S. Brockmans, R. Volz, A. Eberhart, and P. Loffler. Visual modeling of OWL DL ontologies using UML. In: The Semantic Web (ISWC 2004) - Proceedings of the 3rd International Semantic Web Conference, Hiroshima, Japan, 7-11 November 2004; LNCS 3298; Springer, Berlin, Heidelberg; ISBN 978-3-540-23798-3; S. 198-213.

[Burghart, 1996] A. Burghart: Die Pflicht zum guten Gesetz, Beiträge zum Parlamentsrecht, Bd. 34. Duckner & Humboldt, Berlin; 1996; ISBN 3-4280-8669-4.

[Carcia, 2005] R. García: A Semantic Web Approach to Digital Rights Management. Department of Technologies, Universitat Pompeu Fabra; Barcelona, Spanien; 2005; http://rhizomik.net/html/~roberto/thesis/Thesis.pdf (letzter Aufruf: 21.08.2011).

[Casellas, 2008] Casellas, N.: Modelling Legal Knowledge through Ontologies. OPJK: the Ontology of Professional Judicial Knowledge; Departament de Ciència Política i Dret Públic Universitat Autònoma de Barcelona, Spanien; 2008.

[Casellas et al., 2010] N. Casellas, J.-J. Vallbé, M. Reyes de los Mozos, P. Casanovas: Ontology-Enhanced Legal Decision-Support Tools. Presentation at SubTech 2010, 11th International Conference on Substantive Technology in Legal Education and Practice, University of Zaragoza, 1.-3. Juli 2010. http://www.lefis.org/app/eportfolio/artefact/file/download.php?file=584&view=61 (letzter Aufruf: 21.08.2011).

[Ciano et al., 2010] G. Ciano, S. MacLellan, A. Perrone: Representing Non-Functional Requirements (NFRS) in Unified Modelling Language (UML); United States Patent Application Publication, Publication Number: US 2011/0113402 A1, Publication Date: 12.05.2011.

[Coad et al., 1999] P. Coad, E. Lefebvre, J. D. Luca: Java Modeling In Color With UML: Enterprise Components and Process. Prentice Hall, 1999; ISBN 0-130-115-10X.

[COMPAS, 2011] D. Schumm (Hrsg.): COMPAS - Compliance-driven Models, Languages, and Architectures for Services, Achievements & Lessons Learned; Institut für Architektur von Anwendungssystemen (IAAS), Universität Stuttgart; 25.01.2011.

[Cortellessa et al., 2007] V. Cortellessa, A. Di Marco, P. Inverardi: Integrating Performance and Reliability Analysis in a Non-Functional MDA Framework. In: M. B. Dwyer and A. Lopes (Hrsg.): Proceedings of the 10th International Conference on Fundamental Approaches to Software Engineering (FASE 2007); LNCS 4422; Springer-Verlag Berlin, Heidelberg; 2007; ISBN 978-3-540-71288-6; S. 57 - 71.

[Cranefield, 1999] S. Cranefield, M. K. Purvis. UML as an ontology modelling language. In: Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99), 31. Juli 1999, Stockholm, Schweden; CEUR Workshop Proceedings; Vol. 23; 1999; S. 46-53; http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-23/cranefield-ijcai99-iii.pdf (Letzter Aufruf: 21.08.2011).

[Cranefield, 2001] S. Cranefield. Networked Knowledge representation and exchange using UML and RDF. In: Journal of Digital Information. Vol. 1, No. 8; 2001; ISSN: 1368-7506; http://journals.tdl.org/jodi/article/viewArticle/30/31 (letzter Aufruf: 21.08.2011).

[Cysneiros&Leite, 2001] L. M. Cysneiros, J. C. Sampaio do Prado Leite: Using UML to reflect non-functional requirements. In: D. A. Stewart, J. H. Johnson (Hrsg.): Proceedings of the 2001 conference of the Centre for Advanced Studies on Collaborative research (CASCON '01), Toronto, Kanada, 2001. IBM Press; 2001; S. 2-17.

[Cyriaks&Köhler, 2008] H. Cyriaks, S. Köhler: Einsatz semantischer Werkzeuge in frühen Phasen des Software-Service-Co-Designs. In: [Fähnrich&Husen, 2008], S. 183-201.

[Decker et al., 2009] G. Decker, W. Tscheschner, J. Puchan: Migration von EPK zu BPMN. In: M. Nüttgens, F.J. Rump, J. Mendling, N. Gehrke (Hrsg.): Proceedings of EPK 2009 Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, 8. Workshop der Gesellschaft für Informatik e.V. (GI) und Treffen des Arbeitskreises „Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (WI-EPK)“. 26.-27. November 2009; Berlin; 2009; S. 91 - 109. http://www.wiso.uni-hamburg.de/fileadmin/wiso_fs_wi/EPK-Community/epk2009-proceedings.pdf (letzter Aufruf: 21.08.2011).

[Despres&Szulman, 2004] S. Despres, S. Szulman: Construction of a Legal Ontology from a European Community Legislative Text. In: T. Gordon (Hrsg.): Legal Knowledge and Information

Systems (JURIX 2004): The Seventeenth Annual Conference. IOS Press, Amsterdam, Niederlande; 2004; ISBN 1-5860-3492-8; S. 79-88; http://www.jurix.nl/pdf/j04-10.pdf (letzter Aufruf: 21.08.2011).

[Djuric et al., 2004] D. Djuric, D. Gasevic, V. Devedzic, and V. Damjanovic. A UML profile for OWL ontologies. In: U. Aßmann, M. Aksit, A. Rensink (Hrsg.): Model Driven Architecture: European MDA Workshops - Foundations and Applications, MDAFA 2003 and MDAFA 2004, Twente, Niederlande, 26-27 Juni 2003 und Linköping, Schweden; 10-11 Juni 2004; LNCS 3599; Springer, Berlin, Heidelberg; 2008; ISBN 978-3-540-28240-2; S. 204-219.

[Djuric et al., 2005] D. Djuric, D. Gasevic, V. Devedzic. Ontology Modeling and MDA. In: Journal of Object Technology. Vol. 4, No. 1, January-February 2005; ISSN 1660-1769; S. 109-128; http://www.jot.fm/ issues/issue_2005_01/article3 (letzter Aufruf: 21.08.2011).

[Dömges et al., 1996] R. Dömges, K. Pohl, M. Jarke, B. Lohmann, W. Marquardt: PRO–ART/CE - An Environment for Managing Chemical Process Simulation Models. In: Proceedings of the 10th European Simulation Multiconference. Budapest, Ungarn; Juni 1996; S. 1012–1017.

[Donzelli&Bresciani, 2003] P. Donzelli, P. Bresciani: Goal-Oriented Requirements Engineering: a Case Study in E−Government. Technical Report # P03−12−40; Istituto Trentino di Cultura; 2003.

[DudenBd4, 1998] P. Eisenberg, H. Gelhaus, H. Wellmann, H. Henne, H. Sitta: DUDEN Grammatik der deutschen Gegenwartssprache. hrsg. vom Wissenschafltichen Rat der Dudenredaktion; 6., neu bearb. Aufl., Duden Band 4; Dudenverlag Mannheim, Leipzig, Wien, Zürich; 1998; ISBN 3-4110-4046-7.

[Engel, 1999] A. Engel: IT-gestützte Vorgangsbearbeitung in der öffentlichen Verwaltung. In: Lenk, Klaus, und Roland Traunmüller (Hrsg.): Öffentliche Verwaltung und Informationstechnik - Perspektiven einer radikalen Neugestaltung der öffentlichen Verwaltung mit Informationstechnik. R.v.Decker, Heidelberg; 1999; Schriftenreihe Verwaltungsinformatik, Bd. 20; ISBN 3768514994; S. 143-176.

[Engers et al., 2001] T. M. van Engers, R. Gerrits, M. Boekenoogen, E. Glasse, P. Kordelaar: Power: using UML/OCL for modeling legislation - an application report. In: Proceedings of the 8th International Conference on Artificial Intelligence and Law (ICAIL '01); ACM Press; 2001; ISBN 1-58113-368-5; S. 157-167.

[Fähnrich&Husen, 2008] K.-P. Fähnrich, C. v. Husen (Hrsg.): Entwicklung IT-basierter Dienstleistungen. Physica-Verlag; 1. Aufl.; 2008, ISBN: 978-3-7908-1943-4.

[Fernández-López et al. 1997] M. Fernández-López, A. Gómez-Pérez, N. Juristo: METHONTOLOGY: From Ontological Art Towards Ontological Engineering. In: AAAI Technical Report SS-

97-06; 1997; S. 33 - 40; http://www.aaai.org/Papers/Symposia/Spring/1997/SS-97-06/SS97-06-005.pdf (letzter Aufruf: 21.08.2011).

[Falkovych et al., 2003] K. Falkovych, M. Sabou, H. Stuckenschmidt: UML for the Semantic Web: Transformation-Based Approaches. In: B. Omelayenko, M. Klein (Hrsg.): Knowledge Transformation for the Semantic Web: 95 (Frontiers in Artificial Intelligence and Applications). IOS Press; 2003; ISBN 1-5860-3325-5; S. 92-106.

[Fiedler, 1973] H. Fiedler: Wandlungen der Automationsgerechten Rechtssetzung. In: Datenverarbeitung und Recht (DVR) 1972/73, S. 41.

[Fink et al., 2001] F. Fink. G. Schneiderreit, S. Voß: Grundlagen der Wirtschaftsinformatik. Physica Verlag, Heideberg; 2001; ISBN 3-7908-1375-3.

[Frankel, 2003] D. Frankel: Model Driven Architecture: Applying MDA to Enterprise Computing; Wiley Publishing; 2003; ISBN 0-4713-1920-1.

[Fröhlich&Glasner, 2007] M. Fröhlich, K. Glasner: IT-Governance: Leitfaden für eine praxisgerechte Implementierung. Gabler; 1. Auf.; 2007; ISBN 3-8349-0325-6.

[Gaevic et al., 2006] D. Gaevic, D. Djuric, V. Devedic. Model Driven Architecture and Ontology Development. Springer, Berlin; 2006; ISBN 3-5403-2180-2.

[GI&ITG, 2000] GI und ITG VDE (Hrsg.): Electronic Government als Schlüssel zur Modernisierung von Staat und Verwaltung. Ein Memorandum des Fachausschusses Verwaltungsinformatik der GI und des Fachbereichs 1 der Informationstechnischen Gesellschaft im VDE, Bonn und Frankfurt; 2000.

[Giblin et al., 2005] C. Giblin, A. Y. Liu, s. Müller, B. Pfitzmann, X. Zhou: Regulations Expressed As Logical Models (REALM). In: Proceedings of the 18th Annual Conference on Legal Knowledge and Information Systems (JURIX 2005). IOS Press, 2005. Erweiterte Version verfügbar als IBM Research Report RZ 3616, 2005, http://kisogawa.inf.ethz.ch/WebBIB/publications/papers/2005/rz3616.pdf (letzter Aufruf: 21.08.2011).

[Gotel&Finkelstein, 1994] O. C. Z. Gotel, A. C. W. Finkelstein: An Analysis of the Requirements Traceability Problem. In: Proceedings of the First International Conference on Requirements Engineering. IEEE Computer Society Press; 1994; ISBN 0-8186-5480-5; S. 94-101.

[Gómez-Pérez et al., 2004] A. Gómez-Pérez, M. Fernández-López,O. Corcho: Ontological engineering: with examples from the areas of knowledge management, e-commerce and the semantic web. Springer-Verlag, London; 2004; ISBN 1-8523-3551-3.

[Gotel, 1995] O. C. Z. Gotel: Contribution Structures for Requirements Traceability. PhD thesis, Department of Computing

Imperial College of Science Technology and Medicine University of London; 1995.

[Gruber, 1993] T. R. Gruber: A Translation Approach to Portable Ontology Specifications. In: Knowledge Acquisition. Vol. 5, No. 2, 1993; ISSN 1042-8143; S. 199-220; http://tomgruber.org/writing/ontolingua-kaj-1993.pdf (letzter Aufruf: 21.08.2011).

[Grüninger&Fox, 1995] M. Grüninger, M. S. Fox: Methodology for the Design and Evaluation of Ontologies. in: Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with International Joint Conference on Artificial Intelligence (IJCAI-95); 14. April 1995; Montreal, Kanada; http://stl.mie.utoronto.ca/publications/gruninger-ijcai95.pdf (letzter Aufruf: 21.08.2011).

[Guarino, 1998] N. Guarino: Formal Ontology and Information Systems. In: N. Guarino (Hrsg.): Proceedings of the First International Conference on Formal Ontology in Information Systems (FOIS'98) (Frontiers in Artificial Intelligence and Applications). Trento, Italien; Juni 1998; IOS Press, Amsterdam; 1998; ISBN 9-0519-9399-4; S. 3-15.

[Hagen, 2001] M. Hagen: Ein Referenzmodell für Online-Transaktionssysteme im Electronic Government, Hampp, Mering; 2001; ISBN 3-8798-8612-1.

[Handschuh, 2005] S. Handschuh: Creating Ontology-based Metadata by Annotation for the Semantic Web. Genehmigte Dissertation, Universität Karlsruhe, 2005.

[Hausenblas, 2004] Michael G. Hausenblas: Semantische Darstellung und Abfrage von Rechtsnormen am Beispiel Hochschulrecht. Diplomarbeit an der Technischen Universität Graz, 2004.

[Heitkoetter, 2011] Henning Heitkoetter: Transforming PICTURE to BPMN 2.0 as Part of the Model-Driven Development of Electronic Government Systems. In: Proceedings of the 44th Hawaii International Conference on System Sciences (hicss). Koloa, Kauai, Hawaii, USA; IEEE Computer Society; 2011; ISBN 978-0-7695-4282-9; S.1-10.

[Hesse, 2002] W. Hesse: Aktuelles Schlagwort Ontologie(n). In: Informatik-Spektrum. Vol. 25, Nr. 6, 2002; ISSN 0170-6012; S. 477-480

[Hinkelmann et al., 2005] K. Hinkelmann, F. Probst, B. Thönssen: Referenzmodellierung für E-Government-Services. In: Wirtschaftsinformatik, Vol. 47, No. 5, 2005; ISSN 0937-6429; S. 356-366

[Hitzler et al., 2008] P. Hitzler, M. Krötzsch, S. Rudolph, Y. Sure: Semantic Web Grundlagen. Springer, Berlin, Heidelberg; 2008; ISBN 3-5403-3993-0.

[Hoekstra et al., 2007] R. Hoekstra, J. Breuker, M. D. Bello, and A. Boer: The LKIF Core ontology of basic legal concepts. In: P. Casanovas, M. A. Biasiotti, E. Francesconi, M. T. Sagri, (Hrsg.): Proceedings of the Workshop on Legal Ontologies and Artificial Intelligence Techniques (LOAIT 2007), Juni

2007; http://www.ittig.cnr.it/loait/LOAIT07-Proceedings.pdf (letzter Aufruf: 21.08.2011).

[Hood et al., 2005] C. Hood, A. Kreß, R. Stevenson, G. Versteegen, R. Wiebel: iX-Studie zum Thema Anforderungsmanagement - Methoden und Techniken, Einführungsszenarien und Werkzeuge im Vergleich. Heise Zeitschriften Verlag, 2005.

[Horridge et al., 2006] M. Horridge, N. Drummond, J. Goodwin, A. Rector, R. Stevens, H. Wang: The Manchester OWL Syntax. In: B. C. Grau, P. Hitzler, C. Shankey, E. Wallace (Hrsg.): Proceedings of the OWLED*06 Workshop on OWL: Experiences and Directions. Athens, Georgia, USA; 10-11 November 2006; http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-216/submission_9.pdf (letzter Aufruf: 21.08.2011).

[Horn&Off, 2004a] E. Horn, T. Off: Referenzmodelle für eGovernment; In: C. Reichard, M. Scheske, T. Schuppan (Hrsg.): Das Reformkonzept E-Government Potenziale - Ansätze - Erfahrungen; LIT Verlag, Münster; 2004; ISBN 3-8258-7686-1; S. 122-139.

[Horn&Off, 2004b] E. Horn, T. Off: eGovernment-Architekturen auf Basis der eLogo-Referenzmodelle; KWI-Projektberichte Nr. 9; Universität Potsdam, 2004, ISBN 3-937786-04-X.

[Huang et al., 2006] J. C. Huang, A. Dekhtyar, J. H. Hayes (Hrsg.): Center of Excellence for Traceability Problem Statements and Grand Challenges. Center of Excellence of Traceability Technical Report COET-GCT-06-01-0.9; 10 September 2006.

[IEEE 830, 1998] IEEE-SA Standards Board: IEEE Recommended Practice for Software Requirements Specifications; IEEE Std 830-1998; IEEE Press; 1998; ISBN 0-7381-0332-2.

[IETF URI, 1994] The Internet Engineering Task Force (IETF): Universal Resource Identifiers in WWW - A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web. 1994; http://tools.ietf.org/html/rfc1630 (letzter Aufruf: 21.08.2011).

[IETF URN, 2002] The Internet Engineering Task Force (IETF): Uniform Resource Names (URN) Namespace Definition Mechanisms. 2002; http://tools.ietf.org/html/rfc3406 (letzter Aufruf: 21.08.2011).

[ITU-T, 2008] International Telecommunication Union (ITU): User requirements notation (URN) – Language definition. ITU-T Recommendation Z.151; 11/2008; http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Z.151-200811-I!!PDF-E&type=items (letzter Aufruf: 21.08.2011).

[Jacobson et al., 1994] I. Jacobson, M. Christerson, P Jonsson, G. Övergaard: Object-Oriented Software Engineering - A Use Case Driven Approach. Addison-Wesley Publishing, Wokingham, Reading, Menlo Park, u.a.; 1994; ISBN 0-2015-4435-0.

[Jacobson et al., 1999] I. Jacobson, G. Booch, J. Rumbaugh: The Unifed Software Development Process. Addison-Wesley Longman, Amsterdam; 1999; ISBN 0-2015-7169-2.

[Jarke&Marquardt, 1995] M. Jarke, W. Marquardt: Design and Evaluation of Computer–Aided Process Modeling Tools. In: Proceedings of the First Conference on Intelligent Systems in Process Engineering, Snowmass, Colorado, USA; 1995; ISBN 0-8169-0707-2.

[Jungclaussen, 2001] H. Jungclaussen: Kausale Informatik. Einführung in die Lehre vom aktiven sprachlichen Modellieren durch Mensch und Computer. Deutscher Universitäts-Verlag; Auflage: 1. Aufl.; 2001; ISBN 3-8244-2143-7

[Kalibatiene et al., 2009] D. Kalibatiene, O. Vasilecas, G. Guizzardi: Transforming Ontology Axioms to Information Processing Rules - An MDA Based Approach, in: [Kop et al., 2009], S. 105-112, http://ceur-ws.org/Vol-460/paper10.pdf (letzter Aufruf: 21.08.2011).

[Kardoš&Drozdová, 2010] M. Kardoš, M. Drozdová: Analytical Method of CIM to PIM Transformation in Model Driven Architecture (MDA). In: Journal of Information and Organizational Sciences. Vol. 34, No. 1, 2010; e-ISSN 1846-9418, http://hrcak.srce.hr/file/83906 (letzter Aufruf: 21.08.2011).

[Kassab et al., 2009] M. Kassab, O. Ormandjieva, M. Daneva: A metamodel for Tracing Non-Functional Requirements. In: Proceedings of the 2009 World Congress on Computer Science and Information Engineering (CSIE 2009), 31. März - 2. April 2009; Los Angeles , USA. http://users.encs.concordia.ca/~moh_kass/A%20Metamodel%20for%20Tracing%20Non-Functional%20Requirements.pdf (letzter Aufruf: 21.08.2011).

[KBSt, 1997] o. A.: DOMEA - Aufbau eines Pilotsystems für Dokumentenmanagement und elektronische Archivierung im IT-gestützten Geschäftsgang; KBSt-Schriftenreihe Band 34; Koordinierungs- und Beratungsstelle der Bundesregierung für Informationstechnik in der Bundesverwaltung; Bundesministerium des Innern; Bonn; 1997.

[Kerrigan&Law, 2003] S. Kerrigan, K.H. Law: Logic-based regulation compliance-assistance. In: Proceedings of the 9th International Conference on Artificial Intelligence and Law (ICAIL '03). ACM Press; 2003; ISBN 1-58113-747-8; S. 126–135; http://eil.stanford.edu/publications/shawn_kerrigan/ICAIL2003-kerrigan.pdf (letzter Aufruf: 21.08.2011).

[KGSt, 2002] KGSt (Hrsg.): Bericht „Lebenslagen". Verwaltungs-organisation aus Bürger- und Kundensicht (Bericht 5/2002); Köln; 2002.

[Kherraf et al., 2008] S. Kherraf, E. Lefebvre, W. Suryn: Transformation From CIM to PIM Using Patterns and Archetypes. In: Proceedings of the 19th Australian Conference on Software Engineering (ASWEC 2008); 25-28 März 2008; Perth,

Australien, IEEE Computer Society; 2008; ISBN 9-780-7695-3100-7; S. 338-346.

[Kiyavitskaya et al., 2008] N. Kiyavitskaya, N. Zeni, T. D. Breaux, A. I. Antón, J. R. Cordy, L. Mich, J. Mylopoulos: Automating the Extraction of Rights and Obligations for Regulatory Compliance. In: Q. Li, S. Spaccapietra, E. Yu, A. Olivé (Hrsg.): Conceptual Modeling - Proceedings of the 27th International Conference on Conceptual Modeling (ER'08); Barcelona, Spanien, 2008; LNCS 5231; Springer-Verlag, Berlin, Heidelberg; 2008; ISBN 978-3-540-87876-6; S. 154-168.

[Kleppe et al., 2003] A. G. Kleppe, J. B. Warmer, W. Bast: MDA explained - The Model Driven Architecture: Practice and Promise; Addison-Wesley Professional, Boston, u.a.; 2003; ISBN 0-3211-9442-X.

[Klischewski, 2003] R. Klischewski: Kooperationsmodellierung in Verwaltungsprozesse mit ADONIS®. In [Schweighofer et al., 2003], S. 181-184.

[Koch et al., 2008] N. Koch, A. Knapp, G. Zhang, H. Baumeister: UML-Based Web Engineering An Approach Based on Standards. In: G. Rossi, O. Pastor, D. Schwabe, L. Olsina: Web engineering: modelling and implementing web applications. Springer, London; 2008; ISBN 1-8499-6677-X; S. 157-192.

[Kogut et al., 2002] P. Kogut, S. Cranefield, L. Hart, M. Dutra, K. Baclawski, M. Kokar, J. Smith: UML for ontology development. In: The Knowledge Engineering Review; Vol. 17, No. 1, März 2002; Cambridge University Press, New York; ISSN: 0269-8889; S. 61-64.

[Kop et al., 2009] C. Kop, M-A. Sicilia, F. Sartori, E. Dubois, P. Johannesson (Hrsg.): Proceedings of the Third International Workshop on Ontology, Conceptualization and Epistemology for Information Systems, Software Engineering and Service Science (ONTOSE'09) held in conjunction with CAiSE'09 Conference. 8. Juni 2009; Amsterdam, Niederlande; 2009; http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-460/ (letzter Aufruf: 21.08.2011).

[Kralingen, 1995] R. W. van Kralingen: Frame-based conceptual models of statute law. In: Computer/Law Series, No. 16; Kluwer Law International, Den Haag; 1995; ISBN 9-0411-0128-4.

[Kralingen et al., 1996] R.W. van Kralingen, H.J. van den Herik, J.E.J. Prins, M. Sergot, J. Zeleznikow (Hrsg.): Legal Knowledge Based Systems JURIX'96 Foundations of legal knowledge systems; Tilburg University Press; 1996; ISBN 90-361-9657-4, http://www.jurix.nl/pdf/j96-01.pdf (letzter Aufruf: 21.08.2011).

[Kübler, 1987] H. Kübler: Informationstechnik in Verwaltungs-organisationen - Einsatzbedingungen, Chancen und Risiken. Kohlhammer Verlag; Stuttgart, Berlin, Köln, Mainz; 1987; ISBN 3-1700-9568-4.

[Kurchten, 1998] P. Kruchten: The Rational Unified Process. An Introduction. Addison-Wesley Longman, 1998; ISBN 8-1775-8693-9.

[Kuhn, 2000] W. Kuhn: Ontologies from Texts. In: M. J. Egenhofer, D. M. Mark (Hrsg.): 1st International Conference on Geographic Information Science (GIScience 2000), Savannah GA, 2000.

[Kühne&Wenzel, 2006] S. Kühne, C. Welzel: Metamodellierung am Beispiel der E-Government-Domäne Meldewesen und Eclipse GMF. In: K.-P.Fähnrich, S. Kühne, A. Speck, J. Wagner (Hrsg.): Integration betrieblicher Informationssysteme: Problem-analysen und Lösungsansätze des Model-Driven Integration Engineering, Leipziger Beiträge zur Informatik, Bd. 4. Leipziger Informatik-Verbund (LIV); 2006; ISBN 3-9341-7866-9.

[Lame, 2000] G. Lame: Knowledge acquisition from texts towards an ontology of French law. In: N. Aussenac-Gilles, B. Biebow, S. Szulman (Hrsg.): Proceedings of the EKAW'2000 Workshop on Ontologies and Texts, 2.-6. Oktober 2000, Juan-les-Pins, Frankreich; 2000; http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-51/Lame.pdf (letzter Aufruf: 21.08.2011).

[Lame, 2001] G. Lame: Constructing an IR oriented legal ontology, Second International Workshop on Legal Ontologies, 13 December 2001, University of Amsterdam, Netherlands Organised in conjunction with JURIX 2001: the 14th Annual International Conference on Legal Knowledge and Information Systems (JURIX 2001); 2001. http://www.leibnizcenter.org/jurix2001/papers/lame.pdf (letzter Aufruf: 22.08.2011).

[Lamsweerde, 2000] A. van Lamsweerde: Requirements Engineering in the Year 00: A Research Perspective. In: Proceedings of the 22nd 22nd International Conference on Software Engineering (ICSE 2000), 4.-11. Juni 2000, Limerick, Ireland; ACM, New York, USA; 2000; ISBN 1-58113-206-9

[Lamsweerde, 2001] A. van Lamsweerde, Goal-Oriented Requirements Engineering: A Guided Tour. In: Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE 2001), 27-31 August 2001, Toronto, Kanada; IEEE Computer Society; 2001; ISBN 0-7695-1125-2; S. 249-263.

[Lamsweerde, 2004] A. van Lamsweerde: Goal-Oriented Requirements Engineering: A Roundtrip from Research to Practice. In: Proceedings of 12th IEEE International Conference on Requirements Engineering (RE 2004), 6-10 September 2004; Kyoto, Japan; IEEE Computer Society; 2004; ISBN 0-7695-2174-6; S. 4-8.

[Larenz, 1983] K. Larenz: Methodenlehre der Rechtswissenschaft (Enzyklopadie Der Rechts- und Staatswissenschaft), Springer, Berlin; 5. neubearb. Aufl., 1983; ISBN 354012425X.

[Larenz, 1991] K. Larenz: Methodenlehre der Rechtswissenschaft (Enzyklopadie Der Rechts- und Staatswissenschaft), Springer, Berlin; 6. neubearb. Aufl., 1991; ISBN 3-540-52872-5.

[Lenk, 2005] K. Lenk: Informationstechnik als Schlüssel zur Staatsmodernisierung: wie geht es weiter? In: H. Bonin (Hrsg.): Die Zeit nach dem E-Government. . Erschienen in der Reihe: E-Government und die Erneuerung des öffentlichen Sektors - Band 2, herausgegeben von K. Lenk, M. Brüggemeier, C. Reichard; LIT Verlag, Münster; 2005; ISBN 3-8258-8188-1; S. 95-107

[Lenk, 2010] K. Lenk: Evolution im Denken und Handeln - Verwaltungsmodernieierung im nächsten Jahrzehnt: Mehr als nur Bürgerdienste. In: VITAKO Aktuell, Zeitschrift der Bundesarbeitsgemeinschaft der Kommunalen IT-Dienstleister e.V.; Ausg. 4; November 2010; S. 6-7; http://www5.infokom-gt.de/medien/bindata/vitako_aktuell_4_10_infokom_WEB.pdf (letzter Aufruf: 21.08.2011).

[Lenk, 2011a] K. Lenk: E-Government is about Government - Verwaltungsmodernisierung mit IT im nächsten Jahrzehnt. In: K. Klenk, M. Brüggemeier: Zwischenbilanz: E-Government und Verwaltungsmodernisierung; Alcatel-Lucent Stiftung für Kommunkationsforschung; Schriften-reihe 91; 2011; ISSN 0932-156x; http://www.stiftungaktuell.de/files/sr91_e-government_und_verwaltungsmodernisierung_1.pdf (letzter Aufruf: 21.08.2011).

[Lenk, 2011b] K. Lenk: Bürgerdienste – schon eine lange Geschichte. In: [Schwabe, 2011]; S. 23-37

[Liebwald, 2009] D. Liebwald: Knowledge representation and Modelling Legal Norms: The EU Service Directive. In: N. Casellas, E. Francesconi, R. Hoekstra, S. Montemagni (Hrsg.): 3rd Workshop on Legal Ontologies and Artificial Intelligence Techniques (LOAIT 2009) joint with 2nd Workshop on Semantic Processing of Legal Text; Barcelona, Spanien; 2009; http://www.huygens.es/site/IDTSeries2_LOAIT.pdf (letzter Aufruf: 21.08.2011).

[Liu et al., 2007] Y. Liu, S. Müller, K. Xu: A static compliance-checking framework for business process models. In: IBM Systems Journal, Vol. 46, No. 2, 2007; ISSN 0018-8670; S. 335-362

[Looschelders&Roth, 1996] D. Looschelders, W. Roth: Juristische Methodik im Prozeß der Rechtsanwendung: Zugleich ein Beitrag zu den verfassungsrechtlichen Grundlagen von Gesetzesauslegung und Rechtsfortbildung; Duncker & Humblot, Berlin; 1. Auf., 1996; ISBN 3-428-08722-4.

[Lutz, 2001] M. Lutz: Ontologien für die Eingriffsregelung Modellierung eines Verfahrens zur Bewertung von Eingriffen in Natur und Landschaft und prototypische Umsetzung in einem Informationssystem für Verkehrsplaner, Diplomarbeit an der Westfälischen Wilhelms-Universität Münster, Fachbereich Geowissenschaften; November 2001; http://ifgi.uni-muenster.de/~lutzm/DA.pdf (letzter Aufruf: 21.08.2011).

[Luhmann, 1997] N. Luhmann: Legitimation durch Verfahren. Surhkamp, Wissenschaft, Frankfurt am Main; 1. Aufl., 1983; ISBN 3-518-28043-0.

[Luhmann, 1997] N. Luhmann: Recht und Automation in der öffentlichen Verwaltung - Eine verwaltungswissenschaftliche Untersuchung. Duncker & Humbolt, Berlin; 2. unv. Aufl., 1997; ISBN 3-428-00960-6.

[MacCarty, 1989] L. T. MacCarty: A language for legal discourse, I. Basic features. In: Proceedings of the Second International Conference on Artificial Intelligence and Law (ICAIL '89), Vancouver, Kanada; ACM, New York, USA; 1989; ISBN 0-89791-322-1; S. 180-189.

[MacCarty, 2002] L. T. MacCarty: Ownership: A case study in the representation of legal concepts. In: Artificial Intelligence and Law. Vol. 10, No. 1; Springer, Niederlande; 2002; ISSN 0924-8463; S. 135-161.

[Mangano, 2002] S. Mangano: XSLT Cookbook. O'Reilly, Beijing, Cambridge, u. a.; 1. Aufl., 2002; ISBN 0-5960-0372-2.

[Marcus&Maletic, 2003] A. Marcus, J. I. Maletic: Recovering documentation-to-source-code traceability links using latent semantic indexing; In: Proceedings of the 25th International Conference on Software Engineering (ICSE'03), 3-10 Mai 2003, Portland, USA. IEEE Computer Society, 2003, S. 125-135.

[Margaria&Steffen, 2009a] T. Margaria, B. Steffen: Business Process Modelling in the jABC: The One-Thing-Approach. In: J. Cardoso, W .van der Aalst (Hrsg.): Handbook of Research on Business Process Modeling. IGI Global; 2009; ISBN 1605662887; S. 1-26.

[Margaria&Steffen, 2009b] T. Margaria, B. Steffen: Continuous Model-Driven Engineering. In: Computer. Vol. 42, No. 10, Oktober 2009; ISSN 0018-9162; S. 94-97.

[Margaria&Steffen, 2009c] T. Margaria, B. Steffen: Agile IT: Thinking in User-Centric Models. In: T. Margaria, B. Steffen (Hrsg.):Leveraging Applications of Formal Methods, Verification and Validation; Communications in Computer and Information Science, Vol. 17; Springer, Berlin Heidelberg; 2009; ISBN 978-3-540-88478-1; S. 490-502.

[Matoussi&Laleau, 2008] A. Matoussi, R. Laleau: A Survey of Non-Functional Requirements in Software Development Process, Laboratory of Algorithmics, Complexity and Logic (LACL), University Paris 12 (Paris East), Technical Report TR–LACL–2008–7; 2008.

[Meinel&Sack, 2004] C. Meinel, H. Sack: WWW: Kommunikation, Internetworking, Web-Technologien (Xpert.Press); Springer, Berlin; 2004; ISBN 3-5404-4276-6.

[Menne-Haritz, 1999] A. Menne-Haritz: Geschäftsprozesse der Öffentlichen Verwaltung, Schriftenreihe Verwaltungsinformatik, Bd. 19. R. v. Decker, Heidelberg; 1999; ISBN 3-7685-1399-8.

[Meyer&Heidner, 2008] K. Meyer, S. Heidner: Entwicklung von E-Government-Dienstleistungen. In: [Fähnrich&Husen, 2008], S. 257–279.

[microTool, 1998] o.A.: actiF - Das Prozessmodell für die objektorientierte Entwicklung mit objectiF, Best Practices. microTool GmbH, 1998.

[Müller, 2011] L-S. Müller: Lebenslagen zur Strukturierung von Bürgerdiensten. In: [Schwabe, 2011]; S. 71-91.

[NKRG] Gesetz zur Einsetzung eines Nationalen Normenkontrollrates vom 14. August 2006 (BGBl. I S. 1866), das durch Artikel 1 des Gesetzes vom 16. März 2011 (BGBl. I S. 420) geändert worden ist; http://www.gesetze-im-internet.de/nkrg/BJNR186600006.html (letzter Aufruf: 18.08.2011).

[Nolte, 2009] S. Nolte: QVT- Relations Language: Modellierung mit der Query Views Transformation; Springer, Berlin, Heidelberg; 2009; ISBN 978-3-540-92170-7.

[Oertzen, 1970] H.-J. v. Oertzen: Automationsgerechtigkeit von Gesetzen. Verhandlungen des 048. Deutschen Juristentages; Band 2; Sitzungsbericht; München; 1970. Teil T, S. 29-34.

[Oertzen, 1972] H.-J. v. Oertzen: Automationsgerechte Gesetzgebung. Bonn; Godesberger Taschenbücher; 1972.

[Off et al., 2006] T. Off, E. Horn, K. Lenk: Transformation von Rechtsvorschriften in Softwareanforderungen; In: e-Staat und e-Wirtschaft aus rechtlicher Sicht; E. Schweighofer, D. Liebwald, M. Drachsler, IRIS 2006, Boorberg Verlag, Stuttgart, u.a., 2006, ISBN 3-415-03767-3; S. 310-316.

[OMG BMM, 2008] Object Management Group (OMG): Business Motivation Model (BMM) Version 1.0; OMG Document Number: formal/2008-08-02; 2008; http://www.omg.org/spec/BMM/1.0/PDF (letzter Aufruf: 21.08.2011).

[OMG BMM, 2010] Object Management Group (OMG): Business Motivation Model (BMM) Version 1.1; OMG Document Number: formal/2008-08-02; 2008; http://www.omg.org/spec/BMM/1.0/PDF (letzter Aufruf: 21.08.2011).

[OMG BPDM, 2008a] Object Management Group (OMG): Business Process Definition MetaModel (BPDM), Volume I: Formal specification, v1.0 Version 1.0; OMG Document Number: formal/2008-11-03; 2008; http://www.omg.org/spec/BPDM/1.0/volume1/PDF (letzter Aufruf: 21.08.2011).

[OMG BPDM, 2008b] Object Management Group (OMG): Business Process Definition MetaModel (BPDM), Volume II: Formal specification, v1.0 Version 1.0; OMG Document Number: formal/2008-11-04; 2008; http://www.omg.org/spec/BPDM/1.0/volume2/PDF (letzter Aufruf: 21.08.2011).

[OMG BPMN, 2006] Object Management Group (OMG) und Business Process Management Initiative (BPMI): Business Process Modelling Notation (BPMN) Version 1.0, OMG Final Adopted Specification, OMG Document Number: dtc/06-02-01, 2006; http://www.bpmn.org/Documents/

OMG_Final_Adopted_BPMN_1-0_Spec_06-02-01.pdf (letzter Aufruf: 21.08.2011).

[OMG BPMN, 2008] Object Management Group (OMG) und Business Process Management Initiative (BPMI): Business Process Modelling Notation (BPMN) Version 1.1, OMG Available Specification, OMG Document Number: formal/2008-01-17, 2009; http://www.omg.org/spec/BPMN/1.1/PDF und http://www.bpmn.org/Documents/BPMN_1-1_Specification.pdf (letzter Aufruf: 21.08.2011).

[OMG BPMN, 2009] Object Management Group (OMG): Business Process Modelling Notation (BPMN) Version 1.2 OMG Document Number: formal/2009-01-03, 2009; http://www.omg.org/spec/BPMN/1.2 (letzter Aufruf: 21.08.2011).

[OMG BPMN, 2010a] Object Management Group (OMG): Business Process Model and Notation (BPMN) Version 2.0 (Beta), OMG Document Number: dtc/2010-06-05; 2010; http://www.omg.org/spec/BPMN/2.0 (letzter Aufruf: 21.08.2011).

[OMG BPMN, 2010b] Object Management Group (OMG): UML Profile for BPMN Processes RFP, OMG Document Number: ab/10-06-01; 2010; http://www.omg.org/cgi-bin/doc?ab/10-06-01 (letzter Aufruf: 21.08.2011).

[OMG MDA, 2000] Object Management Group (OMG): Model Driven Architecture, White Paper, Draft 3.2; 2000; http://www.omg.org/cgi-bin/doc?omg/00-11-05 (letzter Aufruf: 21.08.2011).

[OMG MDA, 2001] Object Management Group (OMG): Model Driven Architecture (MDA), OMG Document Number: ormsc/2001-07-01; 2001; http://www.omg.org/cgi-bin/doc?ormsc/2001-07-01 (letzter Aufruf: 21.08.2011).

[OMG MDA, 2003a] Object Management Group (OMG): MDA Guide Version 1.0, OMG Document Number: omg/2003-05-01; 2003; http://www.omg.org/cgi-bin/doc?omg/03-05-01 (letzter Aufruf: 21.08.2011).

[OMG MDA, 2003b] Object Management Group (OMG): MDA Guide Version 1.0.1, OMG Document Number: omg/2003-06-01; 2003; http://www.omg.org/cgi-bin/doc?omg/03-06-01 (letzter Aufruf: 21.08.2011).

[OMG MOF, 2002] Object Management Group (OMG): MetaObjectFacility (MOF) Specification Version 1.4, OMG Document Number: formal/2002-04-03; 2002; http://www.omg.org/spec/MOF/1.4/ (letzter Aufruf: 21.08.2011).

[OMG MOF, 2006] Object Management Group (OMG): MOF Core specification Version 2.0, OMG Document Number: formal/2006-01-01; 2006; http://www.omg.org/spec/MOF/2.0/ (letzter Aufruf: 21.08.2011)

[OMG ODM, 2009] Object Management Group (OMG); Ontology Definition Metamodel Version 1.0. OMG Document Number: formal/2009-05-01; 2009; http://www.omg.org/spec/ODM/1.0/ (letzter Aufruf: 21.08.2011).

[OMG OCL, 2010] Object Management Group (OMG); Object Constraint Language Version 2.2; OMG Document Number: formal/2010-02-01; 2010; http://www.omg.org/spec/OCL/2.2/ (letzter Aufruf: 21.08.2011).

[OMG QVT, 2008] Object Management Group (OMG); Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification Version 1.0; OMG Document Number: formal/2008-04-03; 2008; http://www.omg.org/spec/QVT/1.0/PDF (letzter Aufruf: 21.08.2011).

[OMG QVT, 2009] Object Management Group (OMG): Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification Version 1.1 Beta 2 OMG Document Number: ptc/09-12-05; 2009, http://www.omg.org/spec/QVT/1.1/Beta2/PDF (letzter Aufruf: 21.08.2011).

[OMG QVT, 2011] Object Management Group (OMG): Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification Version 1.1 OMG Document Number: formal/2011-01-01; 2011, http://www.omg.org/spec/QVT/1.1 (letzter Aufruf: 21.08.2011).

[OMG UML, 2009a] Object Management Group (OMG): OMG Unified Modeling Language (OMG UML), Infrastructure Version 2.2, OMG Document Number: formal/2009-02-04; 2009; http://www.omg.org/cgi-bin/doc?formal/2009-02-04 (letzter Aufruf: 22.08.2011).

[OMG UML, 2009b] Object Management Group (OMG): OMG Unified Modeling Language (OMG UML), Superstructure Version 2.2, OMG Document Number: formal/2009-02-02; 2009; http://www.omg.org/cgi-bin/doc?formal/2009-02-02 (letzter Aufruf: 22.08.2011).

[OMG UML, 2010] Object Management Group (OMG): UML Profile for BPMN Processes RFP, OMG Document Number: ab/10-06-01; 2010.

[Oren et al., 2006] E. Oren, K. H. Möller, S. Scerri, S. Handschuh, M. Sintek: What are Semantic Annotations, 2006, http://www.siegfried-handschuh.net/pub/2006/whatissemannot2006.pdf (letzter Aufruf: 21.08.2011).

[Overbeek et al., 2009] S. Overbeek, M. Janssen, P. v. Bommel: An Ontological Framework for Integrated Public Service Delivery. In: [Kop et al., 2009]; S. 128-135; http://ceur-ws.org/Vol-460/paper13.pdf (letzter Aufruf: 21.08.2011).

[Padmanabhan et al., 2006] V. Padmanabhan, G. Governatori, S. Sadiq, R. Colomb and A. Rotolo. (2006) Process Modeling: The Deontic Way. In M. Stumptner, S. Hartmann, Y. Kiyoki (Hrsg.): Conceptual Modelling 2006, Third Asia-Pacific Conference on Conceptual Modelling (APCCM 2006), Hobbart, Australien, Januar 2006. CRPIT 53, Australian Computer Society; 2006; ISBN 1-920-68235-X; S. 75-84; http://eprint.uq.edu.au/archive/00003033/01/ paper2.pdf (letzter Aufruf: 21.08.2011).

[Palkovits& Karagiannis, 2003] S. Palkovits, D. Karagiannis: ADOamt® - Das ganzheitliche Modellierungswerkzeug für die öffentliche Verwaltung. In: [Schweighofer et al., 2003], S. 193-197

[Palmer, 1997] J. D. Palmer: Traceability; in: R. H. Thayer, M. Dorfman, Software Requirements Engineering, 2nd Edition, IEEE Computer Society, 1997, Seiten 412 - 422

[Parreiras et al., 2007] F. Silva Parreiras, S. Staab, A. Winter. TwoUse?: Integrating UML models and OWL ontologies. Technical Report 16/2007, Universitat Koblenz-Landau, Fachbereich Informatik, 4 2007.

[Peters et al., 2005] S.Peters, R. Brühl, J. N. Stelling: Betriebswirtschaftslehre: Einführung. 12., durchges. Aufl.; R. Oldenbourg, München, Wien; 2005; ISBN 3-486-57685-2.

[Petrasch&Meimberg, 2006] R. Petrasch, O. Meimberg: Model Driven Architecture Eine praxisorientierte Einführung in die MDA. dpunkt Verlag, Heidelberg; 2006; ISBN 3-89864-343-3.

[Pinheiro, 1996] F. A. C. Pinheiro: Design of a hyper-environment for tracing objectoriented requirements. University Oxford PhD Thesis, 1996.

[Pinheiro, 2000] F. A. C. Pinheiro: Formal and Informal Aspects of Requirements Tracing. Anais do WER00 - Workshop em Engenharia de Requisitos, Rio de Janeiro-RJ, Brasilien, 13-14 Juni 200; WER; 2000; S. 1-21.

[Pinheiro, 2004] F. A. C. Pinheiro: Requirements Traceability. In: J. C. S. do Prado Leite, J. H. Doorn (Hrsg.): Perspectives on Software Requirements. Kluwer Academic Publishers, 2004; ISBN 1-4020-7625-8; S. 91-113.

[Pohl, 1996] K. Pohl: PRO-ART - Enabling Requirements Pre-Traceability. In: Proceedings of the 2nd IEEE International Conference on Requirements Engineering (ICRE '96). Colorado, USA 1996, S. 76-85.

[Powers et al., 2004] C. Powers, S. Adler, B Wishart: EPAL translation of the The Freedom of Information and Protection of Privacy Act; IBM, Tivoli Software und Information and Privacy Commission/Ontario; 2004; http://www.ipc.on.ca/images/Resources/up-EPAL_FI1.pdf (letzter Aufruf: 21.08.2011)

[Raether, 2004] C. Raether: Automatische Generierung von semantischen Kontexten aus gesprochenen Konversationen in Gruppensitzungen. In: K.-P. Fähnrich, T. Meiren (Hrsg.): Computer Aided Engineering für IT-basierte Dienstleistungen Arbeiten aus dem Forschungsvorhaben ServCase, Leipziger Beiträge zur Informatik: Band II, Eigenverlag der Universität Leipzig, 2004, ISBN: 3-934178-39-1, S. 97-105, http://servcase.informatik.uni-leipzig.de/opencms/opencms/system/galleries/externallinks/servCASEextLinks/ServCASE_Buch_download (letzter Aufruf: 21.08.2011).

[Rath, 2009] M. Rath: Rechtliche Aspekte von IT-Compliance. in: G. Wecker, H. van Laak: Compliance in der

Unternehmerpraxis. Gabler Verlag, Wiesbaden; 2. Auf., 2009, ISBN 978-3-8349-1660-0; S. 119-167

[Rath&Sponholz, 2009] M. Rath, R. Sponholz: IT-Compliance Erfolgreiches Management regulatorischer Anforderungen. Erich Schmidt Verlag; Berlin; 2009; ISBN 978-3-503-11093-3.

[Ramesh&Jarke, 2001] B. Ramesh, M. Jarke. Towards Reference Models for Requirements Traceability. In: IEEE Transactions on Software Engineering, Vol. 27, No. 1.; 2001; ISSN: 0098-5589; S. 58-93.

[Rational, 1998] I. Spence, L. Probasco: Traceability Strategies for Managing Requirements with Use Cases. Version 1.0; Rational Software Corporation, 1998

[Ravichandar, 2008] R. Ravichandar: Capabilities Engineering - Promoting Change-Reduction and Constructing Change-Tolerant Systems, Dissertation, eingereicht am Virginia Polytechnic Institute und der Virginia State University, 2008.

[Reichard, 1987] C. Reichard: Betriebswirtschaftlehre der öffentlichen Verwaltung, 2., völlig neubearb. u. erw. Aufl. Berlin[West] u.a., De Gruyter, 1987.

[Rodríguez et al., 2007a] A. Rodríguez, E. Fernández-Medina, M. Piattini: Towards CIM to PIM: Transformation: From Secure Business Processes Defined in BPMN to Use-Cases. In: Lecture Notes in Computer Science, Springer Berlin, Heidelberg, 2007; ISBN 978-3-540-75182-3; S. 408-415.

[Rodríguez et al., 2007b] A. Rodríguez, E. Fernández-Medina, M. Piattini: Using QVT to obtain Use Cases from Secure Business Processes modeled with BPMN, 8th Workshop on Business Process Modeling, Development, and Support (BPMDS'07) im Rahmen der CAiSE'07, Trondheim, 2007, http://lams.epfl.ch/conference/bpmds07/program/Rodriguez_6.pdf (letzter Aufruf: 21.08.2011).

[Rodríguez et al., 2008] A. Rodríguez, E. Fernández-Medina, M. Piattini: CIM to PIM Transformation: A Reality. In: Research and Practical Issues of Enterprise Information Systems II, IFIP International Federation for Information Processing. Vol. 255; 2008; S. 1239-1249.

[Sadiq et al., 2007] S. Sadiq, G. Governatori, K. Naimiri: Modeling Control Objectives for Business Process Compliance. In: G. Alonso, P. Dadam, M. Rosemann (Hrsg.): Proceedings of the 5th International Conference, BPM 2007, Brisbane, Australien, 24-28. September 2007; LNCS 4714; Springer, Berlin; 2007; ISBN 978-3-540-75182-3; S. 149-164.

[Schedler&Proeller, 2003] K. Schedler, I. Proeller: New Public Management. UTB für Wissenschaft; Verlag Paul Haupt; 2. überarb. Aufl.; 2003; ISBN 325-8-065721.

[Scheer, 2001] A.-W. Scheer: ARIS - Modellierungsmethoden, Metamodelle, Anwendungen. Springer, Berlin, Heidelberg; 4. Auf., 2001; ISBN 3-5404-1601-3.

[Schleicher et al., 2010] D. Schleicher, F. Leymann, D. Schumm, M. Weidmann: Compliance Scopes: Extending the BPMN 2.0 Meta Model

to Specify Compliance Requirements. In: Proceedings of IEEE International Conference on Service-Oriented Computing and Applications (SOCA 2010), 13-15 Dezember 2010, Perth, Australien; IEEE; 2010; ISBN 978-1-4244-9802-4; S. 13-15; http://www.iaas.uni-stuttgart.de/institut/mitarbeiter/ schleicher/INPROC-2010-93%20Compliance%20Scopes.pdf (letzter Aufruf: 21.08.2011).

[Schneider, 2010a] C. Schneider: Towards an Eclipse Ontology Framework: Integrating OWL and the Eclipse Modeling Framework. In: F. S. Parreiras, J. Z. Pan, U. Assmann (Hrsg.): Proceedings of the 3rd Workshop on Transforming and Weaving Ontologies in Model Driven Engineering, Málaga, Spain, June 30, 2010, http://ceur-ws.org/Vol-604/paper9.pdf (letzter Aufruf: 21.08.2011).

[Schneider, 2010b] C. Schneider: Integration von Ontologien und modellgetriebener Softwareentwicklung: Konzeption und Entwurf eines Eclipse Ontologie Frameworks (EOF), Diplomarbeit; Universität Koblenz-Landau, Fachbereich 4: Informatik, 2010.

[Schuppan, 2005] T. Schuppan: Strukturwandel der Verwaltung mit E-Government - Eine Untersuchung am Beispiel von Kreis und Gemeinde, E-Government und die Erneuerung des öffentlichen Sektors, Bd. 7. Edition Sigma, Berlin; 2005; ISBN 3-89404-837-9.

[Schuppan, 2011] T. Schuppan: E-Government braucht neue Verpackung. In: Behörden Spiegel. April 2011; ISSN 1437-8337; S. 19.

[Schwabe, 2011] G. Schwabe (Hrsg.): Bügerservices Grundlagen - Ausprägungen - Gestaltung Potenziale, E-Government und die Erneuerung des öffentlichen Sektors, Bd. 11; Edition Sigma, Berlin; 2011; ISBN 978-3-89404-841-9.

[Schweighofer, 2010] E. Schweighofer: An Ontological Representation of EU Consular Law. In: E. Francesconi, S. Montemagni, P. Rossi, D. Tiscornia (Hrsg.): Proceedings of the 4th Workshop on Legal Ontologies and Artificial Intelligence Techniques (LOAIT 2010), 7. Juli 2010, European University Institute, Fiesole, Florence, Italien; 2010; http://ceur-ws.org/Vol-605/paper7.pdf (letzter Aufruf: 21.08.2011).

[Schweighofer et al., 2003] E. Schweighofer, T. Menzel, G. Kreuzbauer, D. Liebwald (Hrsg.): Zwischen Rechtstheorie und e-Government Aktuelle Fragen der Rechtsinformatik 2003 gewidmet Friedrich Lachmeyer; Verlag Österreich; 2003; ISBN 3-7046-4091-3.

[SemanticGov, 2007] IST STREP PROJECT FP6-2004-IST-4-027517: Providing Integrated Public Services to Citizens at the National and Pan-European level with the use of Emerging Semantic Web Technologies (SemanticGov), D.4.2 Annex I: Showcase Ontologies, Version 1.2, 2007.

[Sergot et al., 1986] M. J. Sergot, F. Sadri, F., R. A. Kowalski, F. Kriwaczek P. Hammond, H. T. Cory: The british nationality act as a logic

program. In: Communications of the ACM. Vol. 29, No. 5, 1986; ISSN 0001-0782; S. 370-386.

[Shadbolt et al., 2006] N. Shadbolt, T. Berners-Lee und W. Hall. The semantic web revisited. In: IEEE Intelligent Systems. Vol. 21, No. 3, 2006; S. 96-101; ISSN 1541-1672 ; http://eprints.ecs.soton.ac.uk/12614/1/ Semantic_Web_Revisted.pdf (letzter Aufruf: 21.08.2011).

[Siena, 2010] A. Siena: Engineering Law-Compliant Requirements The Nomos Framework. International Doctorate School in Information and Communication Technologies, DISI - University of Trento; PhD Dissertation; 2010.

[Smith et al., 2009] B. Smith, L. Vizenor, J. Schoening: Universal Core Semantic Layer. In: P. Costa, K. Laskey, L. Obrst (Hrsg.): Proceedings of the 2009 International Conference on Ontologies for the Intelligence Community (OIC-2009), Fairfax, VA, USA, October 21-22, 2009. http://ceur-ws.org/Vol-555/paper5.pdf (letzter Aufruf: 21.08.2011).

[Stamper, 1991] R. K. Stamper: The role of semantics in legal expert reasoning and legal systems. In: Ratio Juris. Vol. 4, No. 2, Juli 1991; S. 219-244; http://doc.utwente.nl/70930/1/ Stamper91role.pdf (letzter Aufruf: 21.08.2011).

[Stamper, 1996] R.K. Stamper: Signs, Information, Norms and Systems. In: B. Holmqvist, P. Andersen (Hrsg.): Signs of Work: Semiosis and Information Processing in Organizations. De Gruyter, Berlin, New York; 1996; ISBN 3-1101-4095-8.

[Steinmann&Nejdl, 2004] Steinmann, F.; Nejdl, W.: Modellierung und Ontologie, Universität Hannover, 2004.

[Stojanovic et al., 2006] L. Stojanovic, N. Stojanovic, D. Apostolou: Change management in e-government: OntoGov. In: Electronic Government. Vol. 3, No. 1, 2006; ISSN 1740-7494; S. 74-92.

[StVG] Straßenverkehrsgesetz in der Fassung der Bekanntmachung vom 5. März 2003 (BGBl. I S. 310, 919), das zuletzt durch Artikel 2 des Gesetzes vom 12. Juli 2011 (BGBl. I S. 1378) geändert worden ist; http://www.gesetze-im-internet.de/stvg/BJNR004370909.html (letzter Aufruf: 18.08.2011).

[StVO] Straßenverkehrs-Ordnung vom 16. November 1970 (BGBl. I S. 1565), die zuletzt durch Artikel 1 der Verordnung vom 1. Dezember 2010 (BGBl. I S. 1737) geändert worden ist; http://www.gesetze-im-internet.de/stvo/ BJNR015650970.html (letzter Aufruf: 18.08.2011).

[Suarez et al., 2008] E. Suarez, M. Delgado, E. Vidal: Transformation of a Process Business Model to Domain Model, in: Proceedings of the World Congress on Engineering 2008 Vol I WCE 2008, July 2 - 4, 2008, London, U.K., ISBN 978-988-98671-9-5, http://www.iaeng.org/publication/WCE2008/ WCE2008_pp165-169.pdf (letzter Aufruf: 21.08.2011).

[Sukow&Weidemann, 2007] H. Suckow, H Wiedemann: Allgemeines Verwaltungsrecht und Verwaltungsrechtschutz: Grundriss für die Aus- und

Fortbildung. Deutscher Gemeindeverlag; 15., überarb. Aufl., 2007; ISBN 3-5550-1394-7.

[Supakkul&Chung, 2005] S. Supakkul, L. Chung: A UML Profile for Goal-Oriented and Use Case-Driven Representation of NFRs and FRs. In: Proceedings of the Third ACIS International Conference on Software Engineering, Research, Management and Applications (SERA 2005), 11-13 August 2005, Mt. Pleasant, USA; IEEE Computer Society; 2005; ISBN 0-7695-2297-1; S. 112-121.

[Thieme, 1981] W. Thieme: Entscheidungen in der öffentlichen Verwaltung. Heymann, Köln u. a.; 1981; ISBN 3-452-18973-2.

[Thomas, 2006] O. Thomas: Das Referenzmodellverständnis in der Wirtschaftsinformatik: Historie, Literaturanalyse und Begriffsexplikation. In: P. Loos (Hrsg.): IWi – Veröffentlichungen des Instituts für Wirtschaftsinformatik im Deutschen Forschungszentrum für Künstliche Intelligenz, Heft 187; 2006; ISSN 1438 5678; S. 1-35; http://www.uni-saarland.de/fileadmin/user_upload/Fachrichtungen/fr13_BWL/professuren/PDF/IWi-Heft_187.pdf (letzter Aufruf: 22.08.2011).

[Thomas et al., 2004] O. Thomas, C. Seel, B. Kaffai, G. Martin: EPK-Referenzmmodelle für Verwaltungsverfahren. In: M. Nüttgens, F. J. Rump (Hrsg.): EPK 2004 - Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, Proceedings des GI-Workshops und Arbeitskreistreffens, Luxemburg, 6. Oktober 2004. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten; 2004; http://www.wiso.uni-hamburg.de/fileadmin/wiso_fs_wi/EPK-Community/epk2004-proceedings-tsskm.pdf (letzter Aufruf: 21.08.2011).

[Thönssen, 2004] B. Thönssen: Das Projekt OntoGov - Ontology enabled eGovernment Service Configuration. In: B. Bekavac, J. Herget, M. Rittberger (Hrsg.): Informationen zwischen Kultur und Marktwirtschaft. Proceedings des 9. Internationalen Symposiums für Informationswissenschaft (ISI 2004), Chur, 6.-8. Oktober 2004. Konstanz: UVK Verlagsgesellschaft mbH; 2004; ISBN 978-3-89669-706-6; S. 491-493.

[Tonu, 2006] S. A. Tonu: Incorporating Non-Functional Requirements with UML Models. A thesis presented to the University of Waterloo in fulfilment of the thesis requirement for the degree of Master of Applied Science in Electrical and Computer Engineering; Waterloo, Kanada; 2006.

[Toranzo&Castro, 1999] M. Toranzo, J. Castro: The Multiview++ Environment - Requirements Traceability from the Perspective of stakeholders. Anais do WER99 - Workshop em Engenharia de Requisitos, Buenos Aires, Argentinien, 9-10 September 1999; WER; 1999; S. 95-105.

[Torkar et al., 2011] R. Torkar, T. Gorschek, R. Feldt, M. Svahnberg, U. A. Raja, K. Kamran: Requirements Traceability: A Systematic Review and Industry Case Study; In: International Journal of Software Engineering and Knowledge Engineering, 2011 (zur Veröffentlichung vorgesehen); ISSN: 0218-1940.

[Traunmüller, 2003] R. Traunmüller: e-Government. Verwaltungsprozesse im Fokus der Rechtsinformatik. In: [Schweighofer et al., 2003], S. 80-86.

[Uschold&King, 1995] M. Uschold, M. King: Towards a Methodology for Building Ontologies. In: Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI-95; Montreal, Canada; 1995; http://staff.um.edu.mt/cabe2/lectures/webscience/docs/uschold_king_95.pdf (letzter Aufruf: 21.08.2011).

[Valente&Breuker, 1994] A. Valente, J. Breuker: A Functional Ontology of Law. In: Towards a Global Expert System in Law. Vol. 7, CEDAM Publishers, S. 201-212, 2004.

[Valente&Breuker, 1997] A. Valente, J. Breuker: ON-LINE: an architecture for modelling legal information. In: Proceedings of the 5th international conference on Artificial intelligence and law (ICAIL '95), ISBN 0-89791-758-8; S. 307-315.

[Visser&Bench-Capon, 1996] P.R.S. Visser, T.J.M. Bench-Capon: The Formal Specification of a Legal Ontology. In: Proceedings of JURIX-96; Tilburg University Press; 1996; S. 15-24.

[Visser&Bench-Capon, 1997] P.R.S. Visser, T.J.M. Bench-Capon: A Comparison of Two Legal Ontologies. In: Proceedings of the First International Workshop on Ontologies. Melbourne, Australien, 1997, S. 37-45.

[Visser&Bench-Capon, 1998] P.R.S. Visser, T.J.M. Bench-Capon: Ontologies in the Design of Legal Information Systems; Towards a Library of Legal Domain Ontologies. Conference on Applied Ontology, Buffalo University, New York, USA; 1998; S. 76-85.

[Vitvar et al., 2010] T. Vitvar, V. Peristeras, K. Tarabanis (Hrsg.): Semantic Technologies for E-Government. Springer-Verlag, Berlin, Heidelberg; 2010; ISBN 978-3-642-03506-7.

[V-Modell XT, 2006] o.A.: V-Modell XT Version 1.3 - Teil 1 bis 9. 2006; http://ftp.tu-clausthal.de/pub/institute/informatik/v-modell-xt/Releases/1.3/V-Modell-XT-Gesamt.pdf (letzter Aufruf: 21.08.2011).

[V-Modell XT Bund, 2009] IT-Stab des Bundesministerium des Innern im Auftrag des Beauftragten der Bundesregierung für Informationstechnik (Hrsg.): V-Modell XT Bund Version 1.0 - Teil 1 bis 9. 2009; http://download.4soft.de/v-modell-xt-bund/releases/1.0/V-Modell-XT-Bund-Gesamt.pdf (letzter Aufruf: 21.08.2011).

[VwVfG] Verwaltungsverfahrensgesetz in der Fassung der Bekanntmachung vom 23. Januar 2003 (BGBl. I S. 102), das zuletzt durch Artikel 2 Absatz 1 des Gesetzes vom 14. August 2009 (BGBl. I S. 2827) geändert worden ist;

http://www.gesetze-im-internet.de/vwvfg/BJNR012530976.html (letzter Aufruf: 18.08.2011)

[VwVG] Verwaltungs-Vollstreckungsgesetz in der im Bundesgesetzblatt Teil III, Gliederungsnummer 201-4, veröffentlichten bereinigten Fassung, das zuletzt durch Artikel 4 Absatz 1 des Gesetzes vom 29. Juli 2009 (BGBl. I S. 2258) geändert worden ist; http://www.gesetze-im-internet.de/vwvg/BJNR001570953.html (letzter Aufruf: 18.08.2011).

[VwV-StVO] Allgemeine Verwaltungsvorschrift zur Straßenverkehrs-Ordnung (VwV-StVO). Vom 22. Oktober 1998. In der Fassung vom 17. Juli 2009; http://www.verwaltungsvorschriften-im-internet.de/bsvwvbund_26012001_S3236420014.htm (letzter Aufruf: 18.08.2011).

[W3C GRDDL, 2007a] World Wide Web Consortium (W3C): Gleaning Resource Descriptions from Dialects of Languages (GRDDL), W3C Recommendation, 11 September 2007, http://www.w3.org/TR/grddl/ (letzter Aufruf: 21.08.2011).

[W3C GRDDL, 2007b] World Wide Web Consortium (W3C): GRDDL Primer, 3C Working Group Note, 28 June 2007, http://www.w3.org/TR/grddl-primer/ (letzter Aufruf: 21.08.2011).

[W3C HTML, 1999] World Wide Web Consortium (W3C): HTML 4.01 Specification W3C Recommendation; 24 December 1999; http://www.w3.org/TR/html401/ (letzter Aufruf: 21.08.2011).

[W3C HTML, 2011] World Wide Web Consortium (W3C): HTML5 A vocabulary and associated APIs for HTML and XHTML; W3C Working Draft 25 May 2011; http://www.w3.org/TR/html5/ (letzter Aufruf: 21.08.2011).

[W3C OWL, 2004] World Wide Web Consortium (W3C): OWL Web Ontology Language Reference, W3C Recommendation 10 February 2004, http://www.w3.org/TR/owl-ref/ (letzter Aufruf: 21.08.2011).

[W3C RDF, 2004a] World Wide Web Consortium (W3C): RDF Primer, W3C Recommendation 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-primer-20040210/ (letzter Aufruf: 21.08.2011).

[W3C RDF, 2004b] World Wide Web Consortium (W3C): Resource Description Framework (RDF): Concepts and Abstract Syntax, W3C Recommendation 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/ (letzter Aufruf: 21.08.2011).

[W3C RDF, 2004c] World Wide Web Consortium (W3C): RDF/XML Syntax Specification (Revised), W3C Recommendation 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/ (letzter Aufruf: 21.08.2011).

[W3C RDFa, 2008a] World Wide Web Consortium (W3C): RDFa Primer Bridging the Human and Data Webs. W3C Working Group Note 14, Oktober 2008; http://www.w3.org/TR/xhtml-rdfa-primer/ (letzter Aufruf: 21.08.2011).

[W3C RDFa, 2008b] World Wide Web Consortium (W3C): RDFa in XHTML: Syntax and Processing - A collection of attributes and processing rules for extending XHTML to support RDF. W3C Recommendation, 14 October 2008, http://www.w3.org/TR/rdfa-syntax/ (letzter Aufruf: 21.08.2011).

[W3C RDFS, 2004] World Wide Web Consortium (W3C): RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, 10 February 2004, http://www.w3.org/TR/rdf-schema/

[W3C XHTML, 2002] World Wide Web Consortium (W3C): XHTML 1.0 The Extensible HyperText Markup Language (Second Edition); A Reformulation of HTML 4 in XML 1.0; W3C Recommendation 26 January 2000, revised 1 August 2002; http://www.w3.org/TR/xhtml1/ (letzter Aufruf: 21.08.2011).

[W3C XHTML, 2010] World Wide Web Consortium (W3C): XHTML 2.0 W3C Working Group Note 16 December 2010; http://www.w3.org/TR/xhtml2/ (letzter Aufruf: 21.08.2011)

[W3C XSLT, 2007] World Wide Web Consortium (W3C): XSL Transformations (XSLT) Version 2.0, W3C Recommendation, 23 January 2007, http://www.w3.org/TR/ xslt20/ (letzter Aufruf: 21.08.2011).

[Weber, 2000] K. Weber (Hrsg.), D. Guntz (Bearb.): Rechtswörterbuch/begr. von Carl Creifelds. Verlag C.H. Beck, München; 16. neubearb. Aufl., 2000; ISBN 3-406-46411-4.

[Wenzel, 2007] S. Wenzel: How to Trace Model Elements? In: Softwaretechnik-Trends. Gesellschaft für Informatik e.V.; Band 27, Heft 2, 2007; ISSN 0720-8928; S. 31-32

[Wimmer&Tambouris, 2002] M. A. Wimmer, E. Tambouris: Online One-Stop Government - A working framework and requirements. In: R. Traunmüller (Hrsg.): Information systems: The e-business challenge. Kluwer Academic Publishers, Boston, Dordrecht, London; 2002; ISBN 1-4020-7174-4; S. 117-130

[Wimmer, 2002] M. A. Wimmer: Online Services für one-stop Government: Prozessmodellierung. In: E. Schweighofer, T. Menzel, G. Kreuzbauer (Hrsg.): IT in Recht und Staat: aktuelle Fragen der Rechtsinformatik (Schriftenreihe zur Rechtsinformatik, 6). Verlag Österreich, Wien; 2002; ISBN 3-7046-3827-7; S. 57-66.

[Winkel, 2004] O. Winkel: E-Government – die Konturen zeichnen sich immer deutlicher ab. In: Verwaltung und Management. Heft 3, 2004; ISSN 0947-9856; S. 126-132.

[Wolf et al., 2011] P. Wolf, M. Obermeier, H. Krcmar: Bürgerservice Engineering. In: [Schwabe, 2011], S. 189-210.

[Wolter et al., 2010] K. Wolter, M. Smialek, L. Hotz, S. Knab, J. Bojarski. W. Nowakowski: Mapping MOF-based Requirements Representations to Ontologies for Software Reuse. In: F. S. Parreiras, J. Z. Pan, U. Assmann (Hrsg.): Proceedings of the 2nd International Workshop on Transforming and Weaving Ontologies in Model Driven Engineering (TWOMDE 2009), Denver, Colorado, USA, October 4, 2009. http://ceur-ws.org/Vol-531/paper03.pdf (letzter Aufruf: 21.08.2011).

[Wright, 1991] S. Wright: Requirements traceability - What? Why? And how? In: Tools and Techniques for Maintaining Traceability During Design, IEE Colloquium, Computing and Control Division, Professional Group C1 (Software Engineering), London, U.K., Digest Number: 1991/180, 1/1–1/2; 1991

[Wulff, 2010] M. Wulff: Vorausschauende Technikfolgenabschätzung. In: Innovative Verwaltung. Das Fachmedium für erfolgreiches Verwaltungsmanagement. Heft 6, 2010; ISSN 1618-9876; S. 43.

[Wyner&Hoekstra, 2010] A. Wyner, R. Hoekstra: A Legal Case OWL Ontology with an Instantiation of Popov v. Hayashi. In: The Knowledge Engineering Review. Vol. 14, No. 2, 2010, ISSN 0269-8889 S. 1-24. http://wyner.info/research/Papers/WynerHoekstraKER2010Ontology.pdf (letzter Aufruf: 21.08.2011).

[Yu, 1994] W. D. Yu: Verifying software requirements: a requirement tracing methodology and its software tool-RADIX. In: IEEE Journal on Selected Areas in Communication. Vol. 12, No. 2, 1994; ISSN 0733-8716; S. 234-240.

[Yu&Mylopoulos, 1998] E. Yu, J. Mylopoulos: Why Goal-Oriented Requirements Engineering, University of Toronto, 1998. ftp://ftp.db.toronto.edu/pub/eric/REFSQ98.html (letzter Aufruf: 21.08.2011).

[Zhang et al., 2005] W. Zhang, H. Mei, H. Zhao, J. Yang: Transformation from CIM to PIM: A Feature-Oriented Component-Based Approach. In: L. Briand, C. Williams (Hrsg.): Proceedings of 8th International Conference on Model Driven Engineering Languages and Systems (MoDELS'05), Montego Bay, Jamaica, 2.-7. Oktober 2005, LNCS Vol. 3713; Springer, Berlin, Heidelberg; 2005; ISBN 978-3-540-29010-0; S. 248-263.

[Zhu&Gorton, 2007] L. Zhu, I. Gorton: UML Profiles for Design Decisions and Non-Functional Requirements. In: Proceedings of the Second Workshop on SHAring and Reusing architectural Knowledge Architecture, Rationale, and Design Intent (SHARK-ADI '07), Minneapolis, USA, 20.-26. Mai 2007. IEEE Computer Society, Washington, USA; 2007; ISBN:0-7695-2951-8; S. 8-16.