Commentarii informaticae didacticae | 13

Artikel erschienen in:

Jörg Desel, Simone Opel, Juliane Siegeris (Hrsg.)

Hochschuldidaktik Informatik HDI 2021

9. Fachtagung des GI-Fachbereichs Informatik und Ausbildung/Didaktik der Informatik 15.–16. September 2021 in Dortmund

(Commentarii informaticae didacticae (CID) ; 13)

2023 – 299 S.
ISBN 978-3-86956-548-4
DOI https://doi.org/10.25932/publishup-56507

Empfohlene Zitation:
Leif Bonorden: Forschendes Lernen im Bachelorseminar „Software Engineering“, In: Hochschuldidaktik Informatik HDI 2021, Jörg Desel, Simone Opel, Juliane Siegeris (Hrsg.), Potsdam, Universitätsverlag Potsdam, 2023, S. 213–230.
DOI https://doi.org/10.25932/publishup-61600

Universitätsverlag Potsdam

Forschendes Lernen im Bachelorseminar „Software Engineering“

Leif Bonorden[1]

Abstract: Forschendes Lernen ist eine Lehr-Lernform, in der Studierende einen eigenen Forschungsprozess vollständig durchlaufen. In Informatikstudiengängen und insbesondere in Informatikbachelorstudiengängen ist die Forschungsorientierung allerdings nur gering ausgeprägt: Forschendes Lernen wird kaum eingesetzt, obwohl dies möglich und sinnvoll ist. Dieser Artikel stellt ein Konzept für ein Seminar *Software Engineering* im Bachelorstudium vor und beschreibt dessen Durchführung. Abschließend wird das Konzept diskutiert und sowohl aus Studierenden- als auch aus Lehrendensicht positiv evaluiert.

Keywords: forschendes Lernen; studentische Forschung; Seminarkonzept; Bachelorstudium; wissenschaftliches Arbeiten; wissenschaftliches Schreiben

1 Einführung

Neben der Vermittlung bekannter Kenntnisse und Methoden soll ein Studium auch zum wissenschaftlichen Arbeiten hinführen. Während Masterstudiengänge der Informatik häufig eine Forschungsorientierung aufweisen, nehmen forschende Tätigkeiten im Bachelorstudium meist nur eine untergeordnete Rolle ein.

Die Lehr-Lernform des forschenden Lernens ermöglicht es Studierenden, einen vollständigen Forschungsprozess zu durchlaufen, und ist insbesondere auch in Bachelorstudiengängen anwendbar. An der Universität Hamburg wurde ein Seminar *Software Engineering* des Bachelorstudiums in der Lehr-Lernform des Forschendes Lernens konzipiert und durchgeführt. Im Folgenden wird in das Forschende Lernen eingeführt und anschließend werden das Seminarkonzept sowie die Durchführung beschrieben und evaluiert.

1 Universität Hamburg, MIN-Fakultät, Fachbereich Informatik, Vogt-Kölln-Str. 30, 22527 Hamburg, Deutschland, leif.bonorden@uni-hamburg.de https://orcid.org/0000-0002-2131-7790

2 Forschendes Lernen

2.1 Definition und Abgrenzung

Das *Forschende Lernen* ist eine Lehr-Lernform, die in ihrer heutigen Idee seit ca. 50 Jahren diskutiert wird [Bu70]. Eine moderne Definition des Begriffes stammt von Huber [Hu09]:

> Forschendes Lernen zeichnet sich vor anderen Lernformen dadurch aus, dass die Lernenden den Prozess eines Forschungsvorhabens, das auf die Gewinnung von auch für Dritte interessanten Erkenntnissen gerichtet ist, in seinen wesentlichen Phasen – von der Entwicklung der Fragen und Hypothesen über die Wahl und Ausführung der Methoden bis zur Prüfung und Darstellung der Ergebnisse in selbstständiger Arbeit oder in aktiver Mitarbeit in einem übergreifenden Projekt – (mit)gestalten, erfahren und reflektieren.

Die idealtypischen Phasen eines solchen (allgemeinen) Forschungsprozesses werden von Huber und Reinmann detaillierter beschrieben [HR19]:

1. Wahrnehmen eines Ausgangsproblems oder Rahmenthemas (Hinführung)
2. Finden einer Fragestellung, Definition des Problems
3. Erarbeiten von Informationen und theoretischen Zugängen (Forschungslage)
4. Auswahl von und Erwerb von Kenntnissen über Methoden
5. Entwickeln eines Forschungsplans bzw. Untersuchungsdesigns
6. Durchführung einer forschenden Tätigkeit
7. Erarbeitung und Präsentation der Ergebnisse
8. Reflexion des gesamten Prozesses

Abzugrenzen ist das *Forschende Lernen* gegenüber anderen Lernformen [HR19; Hu09]: Im *Forschungsnahen Lernen* werden Studierende an einen Forschungsprozess herangeführt, wobei aber die Vorstellung und Diskussion der Forschung anderer im Vordergrund stehen und die Studierenden nur wenige Teile des Prozesses selbst ausführen oder simulieren. Auch *Problemorientiertes Lernen* weist einige Gemeinsamkeiten mit *Forschendem Lernen* auf, jedoch

ist hier das Problem bzw. die Fragestellung nicht selbstgewählt und es werden im Allgemeinen keine auch für Dritte interessanten Erkenntnisse angestrebt. Ein *Projektstudium* im engeren Sinne fokussiert ebenfalls ein anderes Ziel: die Entwicklung eines Produktes bzw. ein anderes konkretes Ergebnis – im Software Engineering häufig ein Softwaresystem.

2.2 Gründe und Ziele

Forschendes Lernen folgt den Grundsätzen *Bildung durch Wissenschaft* sowie *Einheit von Lehre und Forschung*, vermittelt allgemeine (und insbesondere auch berufsbezogene) Kompetenzen und ist didaktisch fundiert [Eu05; HR19; Hu09]. Huber postuliert, dass Forschendes Lernen zu einem wissenschaftlichen Studium gehört: Forschendes Lernen sei mehr als ein „didaktischer Trick“, mache Wissenschaft als sozialen Prozess erfahrbar, und eigne sich insbesondere auch für Bachelorstudiengänge [Hu04].

Lübcke und Heudorfer beschreiben Ziele Forschenden Lernens auf mehreren Abstraktionsebenen [LH19]. Als Detailziele werden insbesondere die Berücksichtigung studentischer Interessen, das Wecken von Neugier, die Anknüpfung an bestehende Forschung, der Erwerb von Methoden- und Schreibkenntnissen, sowie das Erkennen von Zusammenhängen zwischen Studieninhalten identifiziert.

2.3 Forschendes Lernen in der Informatik

In der Informatik und nahen Disziplinen scheint Forschendes Lernen wenig verbreitet zu sein – insbesondere im Bachelorstudium. Eine Ursache liegt sicherlich in der ursprünglichen Konzeption von Bachelor- und Masterstudiengängen für die Informatik im Zuge der Bologna-Reform, bei der Informatik-Bachelorstudiengänge „grundlagen- und methodenorientiert“ sein sollten, um „Grundlagen des Faches in der Breite“ zu legen; erst die Informatik-Masterstudiengänge sollten „forschungsorientiert“ gestaltet werden [Fa04]. Auch wenn schon vier Jahre später die Empfehlung folgte, „bereits die Bachelor-Studiengänge explizit als ‚stärker forschungsorientiert‘ zu konzipieren“ [Fa08], steht dies bis heute selten im Vordergrund und die Bachelorarbeit ist meist die einzige Studienleistung, bei der ein Forschungszyklus von Fragestellung bis Ergebnispräsentation durchlaufen wird.

Als stark strukturierte Disziplin sieht die Informatik zudem einen im Vergleich zu anderen Disziplinen großen Fachkanon von Grundlagenkenntnissen und -methodik für den Beginn des Studiums vor. Zusätzlich wird auch die Employability, also der Erwerb von berufsrelevanten Kenntnissen und Fähigkeiten, stark fokussiert – insbesondere für Tätigkeiten in der Softwareentwicklung.

Im Folgenden werden einige Veröffentlichungen zum Forschenden Lernen in der Informatik kurz vorgestellt – zunächst in der allgemeineren Disziplin, schließlich konkret im Software Engineering:

Das Projekt FOLASMART hat im Lehramtsstudium der Fächer Mathematik und Informatik ein Projekt für die Entwicklung von Mathematik-Lernapps konzipiert. Für die Gestaltung dieser Lernapps haben die Studierenden dabei zu didaktischen Fragestellungen geforscht [RE12].

An der Universität Bremen bietet der Masterstudiengang *Systems Engineering* eine Studienrichtung *Forschungsvertiefung* an, in der Studierende insbesondere für wissenschaftliche Tätigkeiten ausgebildet werden. Forschendes Lernen wird dabei als Ansatz für die Studiengangsgestaltung verwendet [Ba20].

Ein interdisziplinäres Seminar der Informatik und Soziologie wurde von Knoth, Lucke und Zifonun konzipiert und durchgeführt. Studierende haben sich in Tandems zunächst mit den Methoden der beiden Disziplinen auseinandergesetzt und anschließend interdisziplinäre Forschungsanträge erarbeitet [KLZ15].

Mottok et al. haben interne (Forschungs-)Konferenzen für ein Modul *Research Methods and Strategies* im Master-Schwerpunkt *Software Engineering* konzipiert. Dabei werden Einreichungen reviewt und entweder als Vortrag zugelassen oder andernfalls als Poster präsentiert [Mo12].

Als Ergänzung zu einer Vorlesung zum Thema *Softwarearchitektur* nutzt Gast Forschendes Lernen im Rahmen kleinerer Projekte. Der Fokus liegt dabei auf praxisrelevanter Forschung und Handlungskompetenz [Ga07].

An der Universität Paderborn wurde Forschendes Lernen in ein Praxisprojekt über zwei Semester integriert. Studierende haben hier den im Projekt selbst eingesetzten Softwareentwicklungsprozess beforscht [SOF18]. Zudem wurde das diskursive Auseinandersetzen als Teil der wissenschaftlichen Forschung in einem Seminar durch semantisches Positionieren integriert [JKW15].

An der Hochschule Heilbronn haben Studierende im ersten Semester des Masterstudiengangs *Software Engineering and Management* aus vorgegebenen Forschungsthemen eines ausgewählt, bearbeitet und in Form eines möglichst publizierbaren Artikels präsentiert [HH13].

3 Kontext der Lehrveranstaltung

An der Universität Hamburg liegt das Modul *Seminar* im Pflichtbereich mehrerer Informatikstudiengänge und ist für das 5. oder 6. Fachsemester vorgesehen. Die Studierenden können in jedem Semester aus 5 bis 10 verschiedenen Seminarangeboten wählen, um diese Pflicht zu erfüllen. Laut Modulbeschreibung [Fa20] umfassen die angestrebten Lernergebnisse „die Fähigkeit zur wissenschaftlichen Recherche und zur Präsentation wissenschaftlicher Erkenntnisse“, insbesondere sollen „die Studierenden bereits im Bachelor-Studiengang in Kontakt mit Forschungsfragen und Forschungsmethodik der Informatik“ kommen. Nach dem Wissen des Autors wurde bisher keine dieser Lehrveranstaltungen in Form des Forschenden Lernens angeboten.

Das Modul umfasst 3 LP nach ECTS. Dabei werden für die Präsenz- bzw. synchronen Anteile 14 Wochen mit je 2 Stunden Arbeitsaufwand berechnet. Selbststudium und Prüfungsvorbereitung umfassen zusätzliche 62 Stunden. Die Prüfungsleistung besteht aus Referat und schriftlicher Ausarbeitung. Zusätzliche Studienleistung ist die „aktive Teilnahme“.

Formale Teilnahmevoraussetzung ist der Abschluss eines Proseminars – vorgesehen für das 2. oder 3. Fachsemester –, in dem Studierende „Schlüsselqualifikationen im Bereich des selbstständigen Recherchierens, Strukturierens, Präsentierens und Moderierens“ erlangen. Zudem müssen Studierende bereits Module im Umfang von mindestens 51 LP nach ECTS abgeschlossen haben.

4 Gestaltung der Lehrveranstaltung

Ein solches Seminarangebot zum Thema „Software Engineering“ soll nun in der Lehr-Lernform des Forschenden Lernens neu gestaltet werden. Angelehnt an [So17] wird die Lehrveranstaltung zunächst in drei grundsätzliche Phasen eingeteilt: Einstieg, Forschungsphase sowie Abschluss und Nachbereitung. Der Einstieg nimmt dabei etwa die Hälfte der 14-wöchigen Vorlesungszeit ein; die Forschungsphase beginnt bereits parallel zum Einstieg.

4.1 Einstieg

Die wöchentlichen 90-minütigen Termine in der Einstiegsphase sind als jeweils thematisch abgeschlossene Workshops konzipiert: Nach einem kurzen lehrendengeleiteten Auftakt erfolgt die Bearbeitung und Diskussion des jeweiligen Themas durch die Studierenden. Die angeleiteten studentischen Aktivitäten beschränken sich – mit einer Ausnahme – auf die Präsenz- bzw. synchronen Termine. Für ein vertiefendes Selbststudium werden Material und Anregungen zur Verfügung gestellt.

Einführung & Organisation

Im ersten Workshop werden verschiedene Sichtweisen auf Softwareentwicklung diskutiert: Softwareentwicklung als Handwerk, als Strukturwissenschaft, als Ingenieursdisziplin, als Kunst. Die Studierenden erhalten in Kleingruppen einen Text zu einer der Sichtweisen und stellen diese nach kurzer Vorbereitung im Plenum zur Diskussion. Zudem werden mögliche Themenbereiche für die Forschungsvorhaben vorgestellt und Rahmenbedingungen besprochen.

Schreibprozess

Der Forschungs- bzw. Schreibprozess als Ganzes steht im Vordergrund: von Themeneingrenzung über Strukturierung und Arbeit an konkretem Material bis hin zu Textentwürfen und -überarbeitung. Insbesondere werden Schreibtypen vorgestellt [ACL15]. Dieser Workshop findet in Kooperation mit dem Schreibzentrum der Universität statt.

Forschung

Typische Forschungsfragen des Software Engineering [Sh03], der Methodenreichtum der Disziplin [Ra20; SF18; Wi14] sowie Veröffentlichungsarten für Ergebnisse (insbesondere Konferenz, Journal, Workshop) werden diskutiert. Die Studierenden erhalten Beispielartikel, in denen sie Forschungsfragen und -methoden identifizieren und diskutieren [FMP19; LR19; Mu18; MW15; Po16; Sa19].

Literatur & Recherche

Die Studierenden erproben eine Lesemethode (SQ3R) an einem weiteren, bisher unbekannten Beispielartikel [Ts18], der Begriffe wie „Precision“, „Recall“, und „Oracle“ enthält, die den Studierenden typischerweise unbekannt sind. Anschließend werden Bibliotheksangebote, Suchmaschinen und Datenbanken sowie Vorgehensweisen (z. B. Snowballing, Bewertung von Quellen) für die Literaturrecherche vorgestellt. Die Studierenden führen eine kurze Recherche zu ihrem Forschungsthema durch.

Schriftliche Arbeiten

Dieser Workshop ist zweigeteilt: Im ersten Teil geht es um wissenschaftlichen Schreibstil, Methoden zur Textüberarbeitung und Schreibtypen. Im zweiten Teil analysieren die Studierenden den strukturellen Aufbau der Beispielartikel und identifizieren gemeinsam eine typische Artikelstruktur für das Software Engineering. Zudem werden Zitieren und Plagiate thematisiert. Der erste Teil findet in Kooperation mit dem Schreibzentrum der Universität statt.

Vorträge & Präsentationen

Die Studierenden sehen sich in Vorbereitung auf diesen Workshop Videos von verschiedenen Vorträgen an: mehrere wissenschaftliche Vorträge, aber auch einen TED-Talk, einen Lightning Talk auf einer Entwicklerkonferenz sowie eine politische Rede. Die Studierenden diskutieren die grundsätzlichen Elemente eines guten Vortrags und grenzen wissenschaftliche Vorträge gegenüber anderen Formen ab.

Teaser & Abstracts

Mit der „five-element organization“ [Zo15] verfassen die Studierenden erste Entwürfe für einen Abstract ihrer schriftlichen Ausarbeitung. Zudem stellen Sie den aktuellen Stand ihres Forschungsvorhabens in einem 1–2-minütigen Teaser-Vortrag vor.

4.2 Forschungsphase

In den ersten Terminen der Lehrveranstaltung werden mögliche Themenbereiche für Forschungsvorhaben vorgestellt und von den Studierenden ausgewählt. Die Studierenden können weitere Themenbereiche vorschlagen.

Die Studierenden schränken ihren Themenbereich weiter ein, recherchieren geeignete Literatur und formulieren eine Fragestellung für ihr Forschungsvorhaben. Sie wählen eine geeignete Forschungsmethode und führen das Vorhaben allein oder zu zweit durch. Die Ergebnisse bereiten sie in Form eines Vortrags sowie als wissenschaftlichen Artikel im Format der *Lecture Notes in Informatics* [Ge] der Gesellschaft für Informatik auf.

Spätestens nach dem vierten Seminartermin (Thema *Literatur & Recherche*) sind die Studierenden in der Lage, zu ihrem Thema zu recherchieren, erste Kandidaten für Forschungsfragen zu formulieren und geeignete Forschungsmethoden zu diskutieren. Spätestens nach dem Thema *Schriftliche Arbeiten* können die Studierenden die schriftliche Ausarbeitung zu ihrem Thema verfassen.

Lehrendenfeedback und weitere Absprachen erfolgen auf Studierendeninitiative. Feste Zwischenpräsentationen oder andere Rückkopplungen sind nicht vorgesehen.

4.3 Abschluss und Nachbereitung

Zum Abschluss findet eine Abschlussveranstaltung mit Vorträgen und Diskussionen statt – angelehnt an die Form einer wissenschaftlichen Tagung. Diese wird als Blockveranstaltung in der vorlesungsfreien Zeit veranstaltet. Die schriftlichen Forschungsergebnisse werden in Form von Tagungsproceedings zusammengefasst.

Im Rahmen der Abschlussveranstaltung wird auch die Lehrveranstaltung selbst mithilfe eines Evaluationsfragebogens und im offenen Gespräch bewertet. Die Studierenden werden zudem aufgefordert, ihren individuellen Forschungs- und Lernprozess schriftlich zu reflektieren, insbesondere erfolgreiche und kritische Aspekte ihrer eigenen Forschung zu benennen sowie offene Fragen zu identifizieren. Auf Wunsch kann die schriftliche Reflexion eingereicht werden und erhält zusätzliches Lehrendenfeedback.

5 Durchführung der Lehrveranstaltung

Die Lehrveranstaltung wurde im Sommersemester 2020 mit 9 Studierenden sowie im Wintersemester 2020/2021 mit 20 Studierenden durchgeführt. Die Studierenden befanden sich mindestens im 4. Fachsemester, viele kurz vor Abschluss ihres Bachelorstudiums. Im Kontext der Corona-Pandemie fanden alle synchronen Termine via Zoom statt, weitere Elemente für die gemeinsame Arbeit waren in der Lernplattform Moodle eingebunden. Die Abschlussveranstaltung fand nicht als Blockveranstaltung, sondern über mehrere Wochen am Ende der Vorlesungszeit statt.

Im Sommersemester 2020 wurden 2 Themen zu zweit bearbeitet, 5 Themen allein. Die Ausarbeitung verfassten 7 Studierende in deutscher Sprache, 2 in englischer. Die Leistungen waren – bis auf eine Ausnahme – im guten bis sehr guten Bereich.

Im Wintersemester 2020/2021 wurden 7 Themen zu zweit bearbeitet, 6 Themen allein. Die Ausarbeitung verfassten 14 Studierende in deutscher Sprache, 5 in englischer; zu einem Thema erfolgte keine Abgabe. Die Leistungen waren im guten bis sehr guten Bereich.

5.1 Themenbereiche

Die vorgeschlagenen Themenbereiche kamen in beiden Semestern aus drei Bereichen:

Klassische Untersuchungen der Softwaretechnik

Themen, die seit mindestens 20 Jahren erforscht und diskutiert werden, z. B. Modularisierung von Softwaresystemen. Als Einstieg war jeweils eine klassische Veröffentlichung angegeben.

Aktuelle Forschung der Softwaretechnik

Themen, zu denen in den vergangenen zwei Jahren auf großen Konferenzen Beiträge erschienen sind, z. B. Commits in Versionskontrollsystemen. Als Einstieg war jeweils die möglichst aktuelle Veröffentlichung angegeben.

Aktuelle Themen aus der Praxis

Themen, die im *Technology Radar* [Th] des Unternehmens ThoughtWorks aufgeführt sind, z. B. Polyglot Programming. Als Einstieg war jeweils der Eintrag im Technology Radar angegeben.

5.2 Beispiele für studentische Arbeiten

Eine Studentin hat die Verwendung von Product Backlogs in Open-Source-Software untersucht. Zunächst wurden verwandte Arbeiten präsentiert und wichtige Ergebnisse zusammengefasst. Anschließend wurde in einer Studie von vier Softwareprojekten untersucht, ob Eigenschaften, die Product Backlogs in der Literatur zugeschrieben werden, auch in Open-Source-Projekten zutreffen. Während die meisten Eigenschaften – u. a. keine Design-Dokumente oder Anforderungen; informelles Modell für Kommunikation und Koordination – bestätigt werden konnten, trafen andere – insbesondere bezüglich der Priorisierung von Einträgen – nicht auf alle Projekte zu.

Zwei Studierende haben Performance-Unterschiede zwischen GraphQL- und REST-Schnittstellen in einem konkreten Setting des Frameworks *Spring Boot* untersucht. Auf Grundlage von Literatur zum Thema wurden geeignete Metriken ausgewählt und Hypothesen für die konkrete Situation aufgestellt, u. a. wurde eine geringere CPU-Zeit bei der Nutzung von GraphQL sowohl für lesende als auch für schreibende Zugriffe erwartet. In einem Experiment konnte dies für lesenden Zugriff bestätigt werden, musste aber für schreibenden Zugriff verworfen werden.

6 Evaluation

6.1 Studierendenperspektive

Im Sommersemester 2020 erfolgte die Evaluation durch Studierende nur durch ein offenes Gespräch, im Wintersemester 2020/2021 wurden sowohl das offene Gespräch als auch ein anonymer Evaluationsfragebogen eingesetzt.

Die 9 Studierenden im Sommersemester 2020 hoben besonders positiv hervor, dass sie ein Verständnis von wissenschaftlichem Arbeiten gewonnen

hätten und sich nun gut auf die Bachelorarbeit vorbereitet sähen (7 Nennungen). Zudem betonten sie den Gewinn durch die Termine in Kooperation mit dem Schreibzentrum (4 Nennungen). Die Studierenden äußerten als Verbesserungsvorschlag für zukünftige Durchführungen der Lehrveranstaltung vor allem, noch stärker auf Arbeit in Kleingruppen zu setzen (3 Nennungen).

Im Wintersemester 2020/2021 nahmen 18 Studierende am offenen Gespräch teil und hoben besonders die Kooperation mit dem Schreibzentrum (8 Nennungen), die allgemeine Form der Lehrveranstaltung (6 Nennungen), die Vorbereitung auf die Bachelorarbeit/wissenschaftliches Arbeiten (6 Nennungen) sowie die Auswahl interessanter Themen für die Seminararbeiten (5 Nennungen) positiv hervor. Dem gegenüber stand aber auch der Wunsch nach mehr Feedback, vor allem um zu aufwändige Forschungsvorhaben frühzeitig zu erkennen (3 Nennungen).

Über den anonymen Evaluationsfragebogen bewerteten 11 Studierende die Lehrveranstaltung im Wintersemester 2020/2021. Sie würdigten in den offenen Fragen erneut insbesondere die allgemeine Form der Lehrveranstaltung (9 Nennungen), die Kooperation mit dem Schreibzentrum (5 Nennungen) sowie die Themen für die Seminararbeiten (5 Nennungen). Bei den geschlossenen Fragen stimmten alle Studierenden den Aussagen „Die Organisation der Veranstaltung war sehr gut.“, „Die Auswahl der Vortragsthemen passte sehr gut zum Seminarthema.“, „Das Seminar war inhaltlich sehr interessant.“ sowie „Ich habe sehr viel über Techniken des wissenschaftlichen Arbeits gelernt.“ zu. Lediglich bei der Zustimmung zur Aussage „In der Veranstaltung spielte individuelles Feedback zu den Präsentationen eine wichtige Rolle.“ waren die Antworten gleichmäßig von *stimmt genau* bis *stimmt nicht* verteilt.

In beiden Semestern merkten einige Studierende zudem einen hohen Arbeitsaufwand an. Der im Evaluationsfragebogen erfragte eigene Arbeitsaufwand überstieg jedoch bei keinem Studierenden den in der Modulbeschreibung vorgesehenen Arbeitsaufwand von durchschnittlich 5 Stunden pro Woche in der Vorlesungszeit.

6.2 Lehrendenperspektive

Aus Lehrendenperspektive wurde das Seminar ebenfalls überaus positiv bewertet. Ein initial hoher Aufwand für die Konzeption der Lehrveranstaltung ermöglichte erfreuliche, gewinnbringende Seminararbeit und gute Arbeitsergeb-

nisse. Der individuelle Betreuungsaufwand außerhalb der synchronen Termine blieb gering. Ein direkter Vergleich zu anderen Seminarangeboten erschien nicht sinnvoll, aber im subjektiven Eindruck erlangten die Studierenden ein besseres Verständnis der Disziplin *Software Engineering* als in der Vorgängerveranstaltung in anderer Lehr-Lernform. Insbesondere bei anschließenden Gesprächen über mögliche Bachelorarbeiten zeigten Teilnehmende des Seminars ein besseres Verständnis für Forschung, Fragestellungen und Themen des Software Engineering als Studierende, die nicht an diesem Seminar teilgenommen hatten. Alle Bachelorarbeiten der Teilnehmenden, die am gleichen Arbeitsbereich durchgeführt wurden, wurden mindestens mit *gut*, überwiegend mit *sehr gut* bewertet.

Die digitale Durchführung mit Zoom und Moodle verlief insgesamt gut. Ohne Präsenzmöglichkeit musste die Abschlussveranstaltung zur Vermeidung sehr langer Videokonferenzen über mehrere Wochen verteilt stattfinden. Dies hat sich als akzeptable Lösung herausgestellt, auch wenn die Diskussionsbeteiligung im Online-Format erwartungsgemäß gering war.

6.3 Diskussion

Im Rahmen des Seminars durchliefen die Studierenden alle in Abschnitt 2.1 genannten Phasen des Forschungsprozesses. Für die erste (Wahrnehmen eines Ausgangsproblems oder Rahmenthemas) sowie die letzte Phase (Reflexion des gesamten Prozesses) waren Vorgaben und Steuerung durch Lehrende stärker als für die übrigen Phasen. Die in Abschnitt 2.2 genannten (Detail-)Ziele des Forschenden Lernens wurden erreicht.

Das Seminarkonzept lässt sich insbesondere entlang von drei Gestaltungsfeldern analysieren [HR19; LRH19]. **Forschungscharakter:** Die studentische Forschung war disziplinär verankert und erfolgte unabhängig von Forschungsaktivitäten der Lehrenden und der Institution. Die Ergebnisse wurden primär intern präsentiert und diskutiert. **Autonomie:** Die Studierenden wählten Forschungsthema und Forschungsmethode innerhalb eines Rahmens selbst und wurden bei Planung und Durchführung nur bedarfsorientiert begleitet. **Soziale Eingebundenheit:** Die Studierenden führten ihre Forschungsprojekte allein oder zu zweit durch. Feedback der Lehrenden konnte selbstständig bei Bedarf eingeholt werden, eine abschließende Reflexion wurde angeleitet.

Die Gestaltung hinsichtlich des Forschungscharakters – Verzicht auf Interdisziplinarität, Unabhängigkeit von Forschung anderer, Fokus auf interne Ergebnisvorstellung – schien für die Situation der Studierenden im Bachelorstudium hilfreich zu sein. Die umfangreiche Einstiegsphase ermöglichte den Studierenden einen guten Umgang mit der weitgehenden Autonomie. In Hinsicht auf die soziale Eingebundenheit konnten die Studierenden nicht immer mit den Selbstorganisationserfordernissen umgehen; angeleitete Feedback- und Austauschprozesse könnten für diese Lehrveranstaltung daher besser geeignet sein.

Bezüglich der inhaltlichen Gestaltung könnte in zukünftigen Durchführungen das relativ neue *Who-What-How framework* [St20] die bisherigen Materialien zu Forschungsmethoden ergänzen oder ersetzen, da es einen einsteigerfreundlicheren Zugang bietet. Auch die Auswahl von Beispielartikel sollte entsprechend angepasst werden: Die bisherige Auswahl wurde primär so getroffen, dass möglichst verschiedene interessante Themen behandelt werden, allerdings sind nicht genügend verschiedene Forschungsmethoden vertreten.

Insgesamt hat die Evaluation der Lehrveranstaltung gezeigt, dass die Lehr-Lernform des Forschenden Lernens in der Informatik bereits im Bachelorstudium gut eingesetzt werden kann und insbesondere auch aus Sicht der Studierenden positiv bewertet wird.

7 Zusammenfassung und Ausblick

Ein Seminar *Software Engineering* im Bachelorstudium wurde in der Lehr-Lernform des Forschenden Lernens konzipiert und zweimal durchgeführt. Dabei wurde der Einstiegsphase die Hälfte der Vorlesungszeit zugesprochen, wöchentliche Termine wurden als thematisch abgeschlossene Workshops gestaltet und das Schreibzentrum der Universität wurde eingebunden. Auswahl und Bearbeitung der Seminarthemen erfolgten weitgehend autonom durch die Studierenden.

Sowohl aus Sicht der Studierenden als auch aus Lehrendenperspektive war das Seminar erfolgreich und wurde positiv evaluiert. Auch bei unterschiedlichen Vorkenntnissen konnten geeignete Grundlagen für eigene Forschungstätigkeit – insbesondere eine Abschlussarbeit – geschaffen werden. Die Studierenden konnten auch im Bachelorstudium gut mit der Autonomie umgehen.

Aus der Evaluation ergeben sich zudem direkte Anpassungsoptionen für zukünftige Durchführungen, insbesondere eine stärkere Unterstützung von

Studierenden bei der Abschätzung des nötigen Aufwandes für ihr Forschungsvorhaben. Auch sollten weitere Feedback- und Austauschprozesse zwischen Peers oder mit Lehrenden angeleitet werden.

Literaturverzeichnis

[ACL15] Arnold, S.; Chirico, R.; Liebscher, D.: Goldgräber oder Eichhörnchen – welcher Schreibertyp sind Sie? Journal der Schreibberatung 4/, S. 82–97, 2015.

[Ba20] Bačić, I.; Rodenhauser, A.; Kuhfuss, B.; Colombi Ciacchi, L.: Forschendes Lernen im Masterstudiengang Systems Engineering – Bausteine erhalten, Bausteine zusammensetzen, Ergebnisse reflektieren. In (Hoffmeister, T.; Koch, H.; Tremp, P., Hrsg.): Forschendes Lernen als Studiengangsprofil: Zum Lehrprofil einer Universität. Springer VS, Wiesbaden, S. 285–300, 2020.

[Bu70] Bundesassistentenkonferenz, Hrsg.: Forschendes Lernen – Wissenschaftliches Prüfen: Ergebnisse des Ausschusses für Hochschuldidaktik. Bonn, 1970.

[Eu05] Euler, D.: Forschendes Lernen. In (Spoun, S.; Wunderlich, W., Hrsg.): Studienziel Persönlichkeit: Beiträge zum Bildungsauftrag der Universität heute. Campus, Frankfurt a. M., S. 253–271, 2005.

[Fa04] Fakultätentag Informatik, Hrsg.: Empfehlungen zur Einrichtung von konsekutiven Bachelor- und Masterstudiengängen in Informatik an Universitäten, Cottbus, 2004.

[Fa08] Fakultätentage der Ingenieurwissenschaften und der Informatik an Universitäten, Hrsg.: Qualifikationsrahmen für Absolventen „stärker forschungsorientierter“ Studiengänge und Promovierte in den Ingenieurwissenschaften und der Informatik, 2008.

[Fa20] Fachbereich Informatik, Universität Hamburg: InfB-Sem – Seminar, 2020, URL: https://www.inf.uni-hamburg.de/studies/orga/mhb/mhb-2020-stand2020.pdf, Stand: 25. 01. 2023.

[FMP19] Fleck, G.; Moser, M.; Pichler, J.: Improving Quality of Data Exchange Files: An Industrial Case Study. In (Franch, X.; Männistö, T.; Martínez-Fernández, S., eds.): Product-Focused Software Process Improvement (PROFES 2019). Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 161–175, 2019.

[Ga07] Gast, H.: Forschendes Lernen am Beispiel der Vorlesung „Software-Architektur“. In (Reiber, K., Hrsg.): Forschendes Lernen als hochschuldidaktisches Prinzip – Grundlegung und Beispiele. Tübinger Beiträge zur Hochschuldidaktik (Band 3/1), Eberhard Karls Universität Tübingen – Arbeitsstelle Hochschuldidaktik, Tübingen, S. 13–18, 2007.

[Ge] Gesellschaft für Informatik: GI-Edition Lecture Notes in Informatics (LNI), URL: https://gi.de/service/publikationen/lni, Stand: 25. 01. 2023.

[HH13] Hesenius, M.; Herzberg, D.: Forschung mit Master-Studierenden im Software Engineering. In (Spillner, A.; Lichter, H., Hrsg.): Tagungsband des 13. Workshops „Software Engineering im Unterricht der Hochschulen“ (SEUH 2013). CEUR Workshop Proceedings, 956, Aachen, S. 115–116, 2013.

[HR19] Huber, L.; Reinmann, G.: Vom forschungsnahen zum forschenden Lernen an Hochschulen: Wege der Bildung durch Wissenschaft. Springer VS, Wiesbaden, 2019.

[Hu04] Huber, L.: Forschendes Lernen. 10 Thesen zum Verhältnis von Forschung und Lehre aus der Perspektive des Studiums. Die Hochschule: Journal für Wissenschaft und Bildung 13/2, S. 29–49, 2004.

[Hu09] Huber, L.: Warum Forschendes Lernen nötig und möglich ist. In (Huber, L.; Hellmer, J.; Schneider, F., Hrsg.): Forschendes Lernen im Studium. UniversitätsVerlagWebler, Bielefeld, S. 9–35, 2009.

[JKW15] Jakoblew, M.; Keil, R.; Winkelnkemper, F.: Forschendes Lernen durch Semantisches Positionieren. In (Forbrig, P.; Magenheim, J., Hrsg.): HDI 2014 – Gestalten von Übergängen. 6. Fachtagung Hochschuldidaktik der Informatik. Commentarii informaticae didacticae, 9, Universitätsverlag Potsdam, Potsdam, S. 109–124, 2015.

[KLZ15] Knoth, A.; Lucke, U.; Zifonun, D.: Lehre im Format der Forschung: ein interdisziplinäres Seminarkonzept. In (Nistor, N.; Schirlitz, S., Hrsg.): Digitale Medien und Interdisziplinarität. Medien in der Wissenschaft, 68, Waxmann, Münster, S. 217–227, 2015.

[LH19] Lübcke, E.; Heudorfer, A.: Die Ziele forschenden Lernens: Eine empirische Analyse im Rahmen der QPL-Begleitforschung. In (Reinmann, G.; Lübcke, E.; Heudorfer, A., Hrsg.): Forschendes Lernen in der Studieneingangsphase. Empirische Befunde, Fallbeispiele und individuelle Perspektiven. Springer VS, Wiesbaden, S. 17–58, 2019.

[LR19] Licker, N.; Rice, A.: Detecting Incorrect Build Rules. In: 41st International Conference on Software Engineering (ICSE 2019). IEEE, New York, S. 1234–1244, 2019.

[LRH19] Lübcke, E.; Reinmann, G.; Heudorfer, A.: Entwicklung eines Instruments zur Analyse forschenden Lernens. In (Reinmann, G.; Lübcke, E.; Heudorfer, A., Hrsg.): Forschendes Lernen in der Studieneingangsphase. Empirische Befunde, Fallbeispiele und individuelle Perspektiven. Springer VS, Wiesbaden, S. 127–147, 2019.

[Mo12] Mottok, J.; Schroll-Decker, I.; Niemetz, M.; Scharfenberg, G.; Hagel, B. G.: Internal Conferences as a Constructivism Based Learning Arrangement for Research Master Students in Software Engineering. In (Arabnia, H. R.; Clincy, V. A.; Deligiannidis, L., Hrsg.): Proceedings of the 2012 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS 2012). CSREA Press, Las Vegas, S. 292–299, 2012.

[Mu18] Murphy, L.; Kery, M. B.; Alliyu, O.; Macvean, A.; Myers, B. A.: API Designers in the Field: Design Practices and Challenges for Creating Usable APIs. In: Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2018). IEEE, New York, S. 249–258, 2018.

[MW15] Méndez Fernández, D.; Wagner, S.: Naming the pain in requirements engineering: A design for a global family of surveys and first results from Germany. Information and Software Technology 57/, pp. 616–643, 2015.

[Po16] Politowski, C.; Fontoura, L.; Petrillo, F.; Guéhéneuc, Y.-G.: Are the Old Days Gone? A Survey on Actual Software Engineering Processes in Video Game Industry. In: Proceedings of the 5th International Workshop on Games and Software Engineering (GAS '16). ACM Press, New York, pp. 22–28, 2016.

[Ra20] Ralph, P.; Baltes, S.; Bianculli, D.; Dittrich, Y.; Felderer, M.; Feldt, R.; Filieri, A.; Furia, C. A.; Graziotin, D.; He, P.; Hoda, R.; Juristo, N.; Kitchenham, B.; Robbes, R.; Mendez, D.; Molleri, J.; Spinellis, D.; Staron, M.; Stol, K.; Tamburri, D.; Torchiano, M.; Treude, C.; Turhan, B.; Vegas, S.: Empirical Standards for Software Engineering Research, 2020, arXiv: 2010.03525v1 [cs].

[RE12] Romeike, R.; Eichler, K.-P.: Forschendes Lernen mit Apps für Smartphones und Tablets – Studentische Forschungspartnerschaften im Lehramtsstudium Informatik/Mathematik. In (Desel, J.; Haake, J. M.; Spannagel, C., Hrsg.): DeLFI 2012: Die 10. e-Learning Fachtagung Informatik der Gesellschaft für Informatik e.V. Gesellschaft für Informatik e.V., Bonn, S. 279–290, 2012.

[Sa19] dos Santos, H. M.; Durelli, V. H. S.; Souza, M.; Figueiredo, E.; da Silva, L. T.; Durelli, R. S.: CleanGame: Gamifying the Identification of Code Smells. In: SBES '19: Proceedings of the XXXIII Brazilian Symposium on Software Engineering. ACM, New York, pp. 437–446, 2019.

[SF18] Stol, K.-J.; Fitzgerald, B.: The ABC of Software Engineering Research. ACM Transactions on Software Engineering and Methodology 27/3, pp. 1–51, 2018.

[Sh03] Shaw, M.: Writing Good Software Engineering Research Papers. In: Proceedings of the 25th International Conference on Software Engineering (ICSE 2003). IEEE, New York, S. 726–736, 2003.

[So17] Sonntag, M.; Rueß, J.; Ebert, C.; Friederici, K.; Schilow, L.; Deicke, W.: Forschendes Lernen im Seminar: Ein Leitfaden für Lehrende. 2., überarbeitete Auflage, Humboldt-Universität zu Berlin, Berlin, 2017.

[SOF18] Senft, B.; Oberthür, S.; Fischer, H.: Forschendes Lernen in der Informatik – In praxisnaher Projektgruppe einen Softwareentwicklungsprozess erforschen. In (Neuber, N.; Paravicini, W.; Stein, M., Hrsg.): Forschendes Lernen – The wider view. Schriften zur Allgemeinen Hochschuldidaktik (3), WTM-Verlag, Münster, 2018.

[St20] Storey, M.-A.; Ernst, N. A.; Williams, C.; Kalliamvakou, E.: The Who, What, How of Software Engineering Research: A Socio-Technical Framework. Empirical Software Engineering 25/, S. 4097–4129, 2020.

[Th] Thoughtworks: Technology Radar | An opinionated guide to technology frontiers, URL: https://www.thoughtworks.com/radar/, Stand: 25. 01. 2023.

[Ts18] Tsantalis, N.; Mansouri, M.; Eshkevari, L. M.; Mazinanian, D.; Dig, D.: Accurate and Efficient Refactoring Detection in Commit History. In: ICSE '18: Proceedings of the 40th International Conference on Software Engineering. ACM, New York, pp. 483–494, 2018.

[Wi14] Wieringa, R. J.: Design Science Methodology for Information Systems and Software Engineering. Springer, Berlin u. Heidelberg, 2014.

[Zo15] Zobel, J.: Writing for Computer Science. Springer, London, 2015.