

Informatik für Nichtinformatiker: ein kontext- und praxisorientiertes Konzept

Marc Berges, Andreas Mühling,
Peter Hubwieser

Horst Steuer

TUM School of Education
Technische Universität München

Fakultät für Vermessungswesen
Technische Universität München

{berges | andreas.muehling | peter.hubwieser | steuer}@tum.de

Abstract: Wir stellen die Konzeption und erste Ergebnisse einer neuartigen Informatik-Lehrveranstaltung für Studierende der Geodäsie vor. Das Konzept verbindet drei didaktische Ideen: Kontextorientierung, Peer-Tutoring und Praxisbezug („Course“). Die Studierenden sollen dabei in zwei Semestern wichtige Grundlagen der Informatik verstehen und anzuwenden lernen. Durch enge Verzahnung der Aufgaben mit einem für Nichtinformatiker relevanten Kontext, sowie einem sehr hohen Anteil von Selbsttätigkeit der Studierenden soll die Motivation für „fachfremde“ Themen gesteigert werden. Die Ergebnisse zeigen, dass die Veranstaltung sehr erfolgreich war.

1 Einleitung

Der Informatik fällt, genauso wie diversen anderen Studienrichtungen, an der Universität die Aufgabe zu, nicht nur die „eigenen“ Studierenden auszubilden, sondern auch Grundlagen-Vorlesungen für andere Studiengänge anzubieten. Diese Vorlesungen sind didaktisch oft problematisch, da sie viel Inhalt in wenig Zeit an Studierende vermitteln müssen, die dafür aber oftmals kaum eigene Motivation mitbringen – schließlich haben sich diese Personen ja nicht für ein Informatikstudium entschieden. In diesem Umfeld wurde an der TU München im Herbst 2007 eine Kooperation zwischen dem Fachgebiet *Didaktik der Informatik* und dem Fachgebiet *Geoinformationssysteme* vereinbart, um eine zweisemestrige Einführungsveranstaltung für Studierende der Geodäsie und Geoinformation zu konzipieren und durchzuführen.

2 Ein neues Lehrkonzept für die Informatik

Für das Studienjahr 2011/2012 wurde die Veranstaltung auf Grundlage der Erfahrungen aus den ersten vier Jahren nochmals völlig neu konzipiert. Wir präsentieren hier die Konzeption und anschließend erste Ergebnisse der Evaluierung. Der positive Einfluss von Kon-

textbezug auf die Motivation und den Lernerfolg sollte dabei der größte Leitfaktor sein. Die vorgestellte Konzeption, die in diesem Abschnitt beschrieben wird, funktioniert besonders gut bei Nicht-Hauptfach-Veranstaltungen, lässt sich aber auf jede Informatikveranstaltung übertragen.

Das wichtigste Ziel unserer Einführungsveranstaltung in die Informatik für die Studierenden der Geodäsie ist das Erlernen von Objektorientierter Programmierung (OOP). Daneben soll aber auch Verständnis und zum Teil Anwendungsfähigkeit zu weiteren grundlegenden Themen der Informatik erzielt werden. Um diese Ziele in der relativ knappen Zeit erreichen zu können, haben wir die ursprünglich vorgesehenen „klassischen“ zwei Vorlesungsstunden zusammen mit den beiden Übungsstunden zu vierstündigen Arbeitseinheiten mit möglichst kurzen Instruktionsphasen und möglichst hohem Übungsanteil zusammengefasst. Diese Unterrichtseinheiten weisen eine sehr enge Verzahnung von Theorie und Praxis auf. Nach kurzen Einführungen in die theoretischen Konzepte folgt sofort eine praktische Übung. Neben diesem „parallelen“ Verzahnen wurden auch immer wieder vorhergehende Konzepte in die Unterrichtseinheiten mit eingebunden.

Die 62 Studierenden der Veranstaltung im Wintersemester 2011/2012 kamen alle aus dem Studiengang Geodäsie und Geoinformation. 45 davon waren männlich, 17 weiblich. 44 Studierende kamen aus Bayern, vier aus anderen Bundesländern und 14 aus anderen Staaten.

2.1 Informatik im Kontext

Die Verbindung von Lehre mit relevanten Anwendungskontexten, insbesondere an Schulen, ist bereits seit vielen Jahren Thema der didaktischen Forschung. In diesem Umfeld sei insbesondere auf die Initiativen *Chemie im Kontext*¹ [PGFP06], bzw. *Physik im Kontext* [RMS10] verwiesen. An diese anlehnend gibt es seit einiger Zeit nun auch *Informatik im Kontext*² [DKW11]: „Bei *Informatik im Kontext* geht es primär um die Bewältigung lebensweltlicher Herausforderungen in Verbindung mit Informatiksystemen; die Kontexte beziehen sich immer auf Kontexte der Lebenswelt der Schülerinnen und Schüler“ [DKW11]. Bisher ist der Fokus all dieser Bemühungen jedoch stets auf den schulischen Bereich ausgelegt gewesen. Für die universitäre Ausbildung gibt es in den USA bereits vorhandene Ansätze. Mark Guzdial vertritt mit seinem „Media Computing“ die Idee der kontextbasierten Informatiklehre auch auf Hochschulebene [FM04]. Dabei geht es immer um dieselbe zentrale Idee: Durch den Bezug der Lerninhalte zu einem für die Lernenden relevanten Anwendungskontext soll, ganz im konstruktivistischen Sinne, die Neugier und darauf aufbauend das Interesse geweckt bzw. verstärkt werden. Insbesondere verweist Guzdial auch auf die Problematik, die sich in der universitären Lehre für Pflichtveranstaltungen außerhalb des gewählten Fachs ergibt [Guz10]. Die Motivation, sich „über den Tellerrand des eigenen Hauptfachs“ hinaus mit Themen zu beschäftigen, ist dabei meist nur schwach bis gar nicht ausgeprägt. Die Informatik hat es hierbei weiterhin besonders schwer: „*Studies of the problem point to the overemphasis in computer science classes on abstraction over ap-*

¹ <http://www.chik.de/>

² <http://www.informatik-im-kontext.de/>

plication, technical details instead of usability, and the stereotypical view of programmers as loners lacking creativity“ [FM04].

Unser Ansatz verbindet die informatischen Themen der Veranstaltung mit dem Kontext der Geodäsie und schlägt so eine Brücke zum Hauptfach der Studierenden. *„If we can introduce computational skills in a personally meaningful and relevant context, we believe students will be more likely to ‘stick with it’ and, as a result, achieve more“ [FM04].*

Das bedeutet, dass die Studierenden immer Probleme aus dem Umfeld der Geodäsie (bzw. Geodaten) mithilfe der Informatik lösen sollen. Die Daten sind dabei bewusst aus realen Datensammlungen gewählt (beispielsweise aus *OpenStreetMap* mit allen dabei auftretenden Problemen (Formatierung, Inkonsistenzen, . . .)). Insbesondere sind die Daten von uns auch immer so umfangreich gewählt, dass die manuelle Bearbeitung von Anfang an als völlig aussichtslos erscheinen muss. Hierüber soll den Studierenden also der Nutzen, sich mithilfe der Informatik dem Problem anzunähern von Anfang an verdeutlicht werden.

2.2 Themenaufteilung/-auswahl

Die erste Überlegung bezüglich der Themenaufteilung ist die Reihenfolge der Programmierthemen. Hierbei geht es in erster Linie um die beiden momentan vorherrschenden Hauptströmungen „*OOP first*“ und „*OOP later*“. Diese Begriffe beschreiben die Reihenfolge der Einführung von objektorientierten bzw. algorithmischen Programmierkonzepten. In [Ehl12] werden keine gravierenden Unterschiede bzgl. des Lernfortschritts zwischen den beiden Strömungen gefunden. Die TU München rekrutiert üblicherweise viele Studierende aus Bayern. In Bayern gibt es ein Pflichtfach Informatik, in dem nach dem „*Objects-first*“-Ansatz zunächst Objekte und Klassen eingeführt und später objektorientiert programmiert wird [Hub06]. Da wir eine möglichst homogene Gruppe zu Beginn bilden wollen, haben wir uns entschieden, mit Themen zu beginnen, die nicht direkt mit Objektorientierung in Verbindung stehen. Wir folgen einem „*OOP-later*“-Ansatz und verschieben das objektorientierte Programmieren zum Teil bis in das zweite Semester. Die Themenauswahl und -reihenfolge sieht wie folgt aus (in Klammern die Stunden, die pro Thema aufgewendet wurden): Information und Repräsentation – Reguläre Ausdrücke und Grammatiken (3), Einführung in die Tabellenkalkulation und bedingte Anweisung (2), Einführung in Datenbanken mit relationalem Modell und SQL (3), Einführung in die imperative Programmierung mit GNU R, Funktionen, Lebensbereiche von Variablen, Rekursion und imperative Kontrollstrukturen (8), Objektorientierte Modellierung (2), Objektorientierte Programmierung mit Referenzen und Feldern (7).

Die letzten drei Stunden werden für ein kleines Abschlussprojekt genutzt. Die Studierenden sollen einen Routenplaner vervollständigen, der sich aus angegebenen Adressen die Route berechnet und diese auf einer Karte darstellt. Dazu wird der Service *OpenRouteService*³ der Uni Heidelberg genutzt. Die Kartendaten kommen von *OpenStreetMap*.

³ <http://www.openrouteservice.org/>

2.3 Die passende Programmiersprache für jeden Zweck

Der zentrale Punkt der Planung ist die Wahl der richtigen Programmiersprache für den Einstieg. Am Ende des Semesters sollen alle Studierenden grundlegende Kenntnisse in OOP sowie in Java vorweisen können. Allerdings ist Java als erste imperative Sprache wegen des Overheads durch OO-Mechanismen nicht besonders gut geeignet. Um den Einstieg in die Programmierung einfacher zu gestalten, haben wir uns für die Skript-Sprache GNU-R [R D11] als erste Sprache entschieden. Matt Bower hat in seinem Beitrag [BM11] beschrieben, dass sich beim (parallelen) Lernen von zwei Programmiersprachen positive Effekte einstellen können. Die Erfolgsmomente beim Anwenden von bereits bekannten Konzepten in der neuen Sprache motivieren offensichtlich stark. GNU-R bietet sich insbesondere auch deswegen an, weil man mithilfe einer großen Auswahl an vorprogrammierten Packages relativ einfach komplexe Probleme bewältigen kann. So ist es für die Studierenden beispielsweise sehr einfach, sich per SQL-Anfrage Daten von einem Datenbankserver zu holen. Auch die graphische Darstellung von Ergebnissen fällt damit sehr leicht. Zudem ist die Syntax der imperativen Kontrollstrukturen sehr ähnlich zu Java.

2.4 Kognitive Belastung

Die extrinsische kognitive Belastung (Cognitive Load [Swe88]) ist eines der größten Probleme, die sich aus dem „Experimentieren“ mit dem Programmieren und dem realen Kontext ergeben. Die natürliche Komplexität von realen Daten, die in den Übungsaufgaben während des gesamten Semesters verwendet wurden, erfordert einige kognitive Prozesse alleine um die Daten verstehen und einordnen zu können. Der mathematische Hintergrund von Berechnungen aus der Geodäsie ist isoliert betrachtet ebenfalls für viele schon eine große Herausforderung. Hinsichtlich der Programmierung ist die Syntax für die meisten Programmieranfänger das schwierigste Element. Um diesem entgegenzuwirken, haben wir viele Aufgaben durch sog. „worked examples“ ergänzt, wie es in [CS91] beschrieben wird. Bei „worked examples“ werden Teile der Lösung vorgegeben, so dass sich auf die wirklich relevanten Inhalte der Aufgabe konzentriert werden kann.

2.5 Konstruktivismus und Programmieren

In [BH10] und [HB11] haben wir bereits darauf hingewiesen, dass Programmieren als Werkzeug in der Regel nicht in einer üblichen Einführungsveranstaltung gelehrt werden kann. Die meisten Veranstaltungen dieser Art setzen zwar Programmieren nicht direkt voraus, der Einstieg geschieht aber so schnell, dass es für einen Anfänger kaum möglich ist mitzuhalten. Neben den vielen theoretischen Inhalten einer solchen Einführungsveranstaltung ist es aber unbedingt notwendig, auch den Umgang mit geeigneten Werkzeugen zu erlernen. Aus der Notwendigkeit, Programmieren durch Selbsterfahrung zu lernen, und der Tatsache, dass noch kein hinreichendes Informatikbild vorhanden ist (vgl. [BA98]), ergibt sich das Design unserer Veranstaltung. Es wechseln sich immer wieder theoretische und praktische Inhalte in kurzen Abständen ab (max. 30 Min. pro Block). Dabei kann es durchaus sein, dass die Aufgaben so gestellt sind, dass die Studierenden erst die prakti-

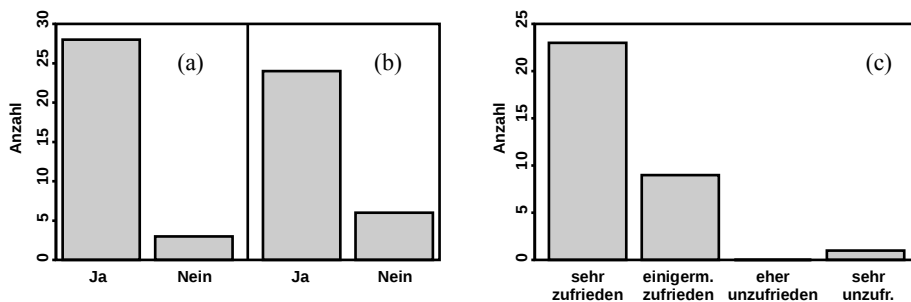


Abbildung 1: Ergebnisse der ersten Umfrage: (a) „Ich gehe gerne in die Veranstaltung ...“; (b) „Ich fände es gut, wenn andere Veranstaltungen ähnlich organisiert wären wie GEO-INF ...“; (c) „Wie zufrieden sind Sie mit der Betreuung?“

sche Erfahrung machen und danach die theoretischen Inhalte behandelt werden. Um die Studierenden zur aktiven Mitarbeit anzuhalten, müssen sie sich sog. „Aktivitätspunkte“ erarbeiten. Ein Teil dieser verpflichtenden Zusatzleistung besteht aus der aktiven Mitarbeit in den Präsenzstunden, zum anderen Teil aus speziell gekennzeichneten Hausaufgaben.

3 Ergebnisse und Erfahrungen, Ausblick

Wir stützen unsere Auswertung neben dem persönlichen Eindruck vor allem auf die Ergebnisse von zwei Umfragen und der Klausur sowie auf die erzielten Aktivitätspunkte. Die erste Umfrage wurde nach dem ersten Drittel der Vorlesungszeit erhoben, die zweite nach etwa zwei Dritteln der Vorlesungszeit im Rahmen einer zentralen Evaluierungsmaßnahme der TU München. In der ersten Umfrage ging es hauptsächlich darum, wie die Studierenden unser Konzept im Vergleich zu ihren anderen Vorlesungen erleben und ob sie prinzipiell damit zurechtkommen. Aus den in Abb. 1 dargestellten Ergebnissen lässt sich schließen, dass die Herangehensweise an die Veranstaltung von den Studierenden akzeptiert und auch für gut befunden wurde. Auch die enge Betreuung wurde im allgemeinen gut aufgenommen.

Die zweite Umfrage bestand aus einem Standard-Fragebogen, den die Studierenden in beinahe allen Veranstaltungen der TUM ausfüllen. Sie wird zentral organisiert und ausgewertet. Die Rücklaufquote liegt bei hohen 85 %. Auch hier zeigt sich: Die Studierenden sind mit der Veranstaltung insgesamt zufrieden bis sehr zufrieden.

Insgesamt lässt sich also ein positives Fazit für die neue Veranstaltung ziehen. Die Anwesenheit und die Mitarbeit der Studierenden lag durchweg auf einem hohen Niveau. Die enge Verzahnung von Vorlesung und Übung hat zu einem sehr engen Kontakt von Dozenten zu Studierenden geführt. Diese Nähe war insbesondere sehr nützlich, um auf Schwierigkeiten im Lernverlauf schnell reagieren zu können. Es wird in Zukunft auch darum gehen, den hier vorgestellten Weg in regulären Informatik-Vorlesungen bzw. Vor-

lesungen mit einer größeren Hörerzahl zu erproben. Als Verbesserungsmöglichkeit bietet sich eine Reduktion der extrinsischen kognitiven Belastung der Aufgaben an. Wie kann man es schaffen, relevante Aufgaben im Kontext der Geodäsie so zu stellen, dass die wesentlichen (informatischen) Inhalte bei der Bearbeitung nicht in den Hintergrund geraten? Im Sommersemester 2012 werden die hier beschriebenen Konzepte z. T. noch vertieft und in die Nachfolgevorlesung mit aufgenommen. Insbesondere war die enge Zusammenarbeit von Informatikern und Geodäten sehr erfolgreich und soll daher auch in den kommenden Jahren fortgesetzt und ausgebaut werden.

Literatur

- [BA98] Ben-Ari, M.: Constructivism in computer science education. In: *Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education (SIGCSE'98)*. ACM, New York, 1998; S. 257–261. – DOI: 10.1145/273133.274308
- [BH10] Berges, M., Hubwieser, P.: Vorkurse in objektorientierter Programmierung: Lösungsansatz für einen leichteren Einstieg in die Informatik. In (Engbring, E., Keil, R., Magenheimer, J., Selke, H., Hrsg.): *HDI2010 - Tagungsband der 4. Fachtagung zur „Hochschuldidaktik Informatik“*. Universitätsverlag Potsdam, 2010; S. 13–22.
- [BM11] Bower, M., McIver, A.: Continual and explicit comparison to promote proactive facilitation during second computer language learning. In: *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education (ITiCSE '11)*. ACM, New York, 2011; S. 218–222. – DOI: 10.1145/1999747.1999809
- [CS91] Chandler, P., Sweller, J.: Cognitive Load Theory and the Format of Instruction. In: *Cognition and Instruction* 8(4), 1991; S. 293–332.
- [DKW11] Diethelm, I., Koubek, J., Witten, H.: IniK – Informatik im Kontext: Entwicklungen, Merkmale und Perspektiven. In: *Log In* 31, Heft Nr. 169/170, 2011.
- [Ehl12] Ehlert, A.: *Empirische Studie: Unterschiede im Lernerfolg und Unterschiede im subjektiven Erleben des Unterrichts von Schülerinnen und Schülern im Informatik-Anfangsunterricht (11. Klasse Berufliches Gymnasium) in Abhängigkeit von der zeitlichen Reihenfolge der Themen (OOP-First und OOP-Later)*. Dissertation. FU Berlin, 2012.
- [FM04] Forte, A., Guzdial, M.: Computers for Communication, Not Calculation: Media as a Motivation and Context for Learning. In: *Proceedings of the 37th Hawaii International Conference on System Sciences (HICSS-37 2004)*. IEEE, 2004.
- [Guz10] Guzdial, M.: Does contextualized computing education help? In: *ACM Inroads* 1(4), 2010; S. 4–6. – DOI: 10.1145/1869746.1869747
- [HB11] Hubwieser, P., Berges, M.: Minimally invasive programming courses: learning OOP with(out) instruction. In: *Proceedings of the 42nd ACM technical symposium on Computer science education (SIGCSE'11)*. ACM, New York, 2011; S. 87–92.
- [Hub06] Hubwieser, P.: Functions, objects and states: Teaching informatics in secondary schools. In (Mittermeir, R. T.): *Informatics Education—The Bridge between Using and Understanding Computers (ISSEP 2006)*. Springer, Berlin, 2006; S. 104–116.
- [PGFP06] Parchmann, I., Gräsel, C., Fussangel, K., Pröbstel, C.: Chemie im Kontext – a symbiotic implementation of a context-based teaching and learning approach. In: *International Journal of Science Education* 28(9), 2006; S. 1041–1062.
- [R D11] R Development Core Team: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Wien, 2011.
- [RMS10] Reinders, D., Mikelskis-Seifert, S.: *Physik im Kontext: Konzepte, Ideen, Materialien für effizienten Physikunterricht*. Friedrich, 2010.
- [Swe88] Sweller, J.: Cognitive Load During Problem Solving: Effects on Learning. In: *Cognitive Science* 12(2), 1988; S. 257–285. – DOI: 10.1207/s15516709cog1202_4