Commentarii
informaticae didacticae | 10

Andreas Schwill | Ulrike Lucke (Hrsg.)

Hochschuldidaktik der Informatik

HDI 2016 – 7. Fachtagung des GI-Fachbereichs Informatik und Ausbildung/Didaktik der Informatik

13.–14. September 2016 an der Universität Potsdam

Universitätsverlag Potsdam

Commentarii informaticae didacticae (CID)

Commentarii informaticae didacticae (CID) | 10

Andreas Schwill | Ulrike Lucke (Hrsg.)

Hochschuldidaktik der Informatik

HDI2016 – 7. Fachtagung des GI-Fachbereichs
Informatik und Ausbildung/Didaktik der Informatik

13.–14. September 2016 an der Universität Potsdam

Universitätsverlag Potsdam

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de/ abrufbar.

Universitätsverlag Potsdam 2016
http://verlag.ub.uni-potsdam.de/

Am Neuen Palais 10, 14469 Potsdam
Tel.: +49 (0)331 977 2533 / Fax: 2292
E-Mail: verlag@uni-potsdam.de

ISSN (print) 1868-0844
ISSN (online) 2191-1940

Die Schriftenreihe Commentarii informaticae didacticae (CID)
wird herausgegeben von:
Sigrid Schubert, Universität Siegen
Andreas Schwill, Universität Potsdam

Druck: docupoint GmbH Magdeburg

Online veröffentlicht auf dem Publikationsserver der
Universität Potsdam:
URN urn:nbn:de:kobv:517-opus4-93511
http://nbn-resolving.de/urn:nbn:de:kobv:517-opus4-93511

Zugleich gedruckt veröffentlicht im Universitätsverlag Potsdam
ISBN 978-3-86956-376-3

Vorwort

Die 7. Fachtagung für Hochschuldidaktik, die 2016 erneut mit der DeLFI E-Learning Fachtagung Informatik stattfand, setzte das erfolgreiche Modell einer Tagung fort, die sich mit hochschuldidaktischen Fragen und der Gestaltung von Studiengängen der Informatik beschäftigt.

Thema der Tagung waren alle Fragen, die sich der Vermittlung von Informatikgegenständen im Hochschulbereich widmen. Dazu gehörten u. a.:

- fachdidaktische Konzepte der Vermittlung einzelner Informatikgegenstände,
- methodische Lösungen, wie spezielle Lehr- und Lernformen, Durchführungskonzepte,
- empirische Ergebnisse und Vergleichsstudien,
- E-Learning-Ansätze, wenn sie ein erkennbares didaktisches Konzept verfolgen,
- Studienkonzepte und Curricula, organisatorische Fragen, wie Gewinnung von Studierenden, Studieneingangsphase, Abbrecher.

Die Fachtagung widmete sich ausgewählten Fragestellungen dieses Themenkomplexes, die durch Vorträge ausgewiesener Experten und durch eingereichte Beiträge intensiv behandelt wurden.

Unser besonderer Dank gilt dem Programmkomitee und den hier nicht genannten Helfern für ihren Einsatz bei der Vorbereitung und Durchführung der Tagung.

Potsdam, im Juli 2016

Andreas Schwill und Ulrike Lucke

Programmkomitee

- Torsten Brinda, Universität Duisburg Essen
- Jörg Desel, FernUniversität Hagen
- Peter Forbrig, Universität Rostock
- Jörg Haake, FernUniversität Hagen
- Reinhard Keil, Universität Paderborn
- Ulrike Lucke, Universität Potsdam (2. Vorsitz)
- Johannes Magenheim, Universität Paderborn
- Ralf Romeike, Universität Erlangen
- Axel Schmolitzky, HAW Hamburg
- Ulrik Schroeder, RWTH Aachen
- Carsten Schulte, FU Berlin
- Andreas Schwill, Universität Potsdam (Vorsitz)
- Karsten Weicker, HTWK Leipzig

Inhaltsverzeichnis

Full Papers zum Thema „Kompetenzen“

Full Papers zum Thema „Werkzeuge“

Short Papers

Full Papers
zum Thema „Kompetenzen“

Das „Startprojekt“ – Entwicklung überfachlicher Kompetenzen von Anfang an

Elisabeth Dennert-Möller, Robert Garmann

Fakultät IV Wirtschaft und Informatik, Hochschule Hannover
30459 Hannover
Email: {elisabeth.dennert-moeller|robert.garmann}@hs-hannover.de

Zusammenfassung: Absolventinnen und Absolventen unserer Informatik-Bachelorstudiengänge benötigen für kompetentes berufliches Handeln sowohl fachliche als auch überfachliche Kompetenzen. Vielfach verlangen wir von Erstsemestern in Grundlagen-Lehrveranstaltungen fast ausschließlich den Aufbau von Fachkompetenz und vernachlässigen dabei häufig Selbstkompetenz, Methodenkompetenz und Sozialkompetenz. Gerade die drei letztgenannten sind für ein erfolgreiches Studium unabdingbar und sollten von Anfang an entwickelt werden. Wir stellen unser „Startprojekt“ als einen Beitrag vor, im ersten Semester die eigenverantwortliche, überfachliche Kompetenzentwicklung in einem fachlichen Kontext zu fördern.

1 Einleitung

In unseren Bachelor-Studiengängen „Angewandte Informatik“ und „Mediendesigninformatik“ bilden wir Menschen aus, die später in häufig interdisziplinären Projekten domänenspezifische Probleme analysieren und einer IT-Lösung zuführen. Dazu müssen sie Informatik-fachlich kompetent sein. Darüber hinaus werden sie im Beruf im Team agieren, müssen ihre eigenen Fähigkeiten einschätzen und weiterentwickeln, müssen Arbeitsprozesse planen und steuern und ggf. Mitarbeiter führen. Vor diesem Hintergrund ist es nicht nur gerechtfertigt, sondern geboten, neben der wichtigen Fachkompetenz auch überfachliche Kompetenzen aufzubauen.

Für die Lernenden besteht der Schlüssel zu einem erfolgreichen Studium im Übernehmen von Verantwortung für den eigenen Lernprozess. Diese

Verantwortung kann ihnen niemand abnehmen [B00]. Das „Startprojekt“, ein Projekt im ersten Semester, will einen Beitrag zur eigenverantwortlichen, überfachlichen „Kompetenzentwicklung von Anfang an“ leisten.

2 Kompetenzentwicklung – Herausforderung für Lernende und Lehrende

Die Gesellschaft für Informatik [GI05] beschreibt überfachliche Kompetenzen als Methodenkompetenz, soziale Kompetenz und Selbstkompetenz. Studierende sollten vom ersten Studientag an die Möglichkeit und den Anreiz haben, ihre Kompetenzen in allen vier Kompetenzfeldern weiter zu entwickeln. Wenn man sich bewusst macht, dass etwa die Entwicklung von Selbstkompetenz ein lange vor Studienbeginn begonnener, mitunter von zeitraubenden Rückschlägen gesäumter Prozess ist, erscheint es sinnvoll, diese Entwicklung kontinuierlich über alle Semester hinweg zu fördern.

2.1 Lehr- und Lernformen

In traditionellen Lehrveranstaltungen wird häufig versucht, die mit der Fachkompetenz zusammen wirkenden überfachlichen Kompetenzen in begleitenden Übungen zu entwickeln: Methodenkompetenz kann etwa anhand konkreter Beispielproblemstellungen erworben werden, die in Übungsaufgaben unter Einsatz gelernter Methoden gelöst werden müssen, Sozialkompetenz wird bspw. durch Bearbeitung von Übungsaufgaben im Team gefördert. Entwicklung von Selbstkompetenz kann unterstützt werden, indem motivationale Aspekte berücksichtigt werden und indem Anreize wie etwa Klausurboni für die Annahme von Herausforderungen geschaffen werden.

Diese Versuche, überfachliche Kompetenzentwicklung lose mit traditionellen Lehrformen zu verbinden, sind aus unserer Sicht jedoch mehr „Feigenblatt“ als von großer Wirkung. Eine wirkungsvolle Entwicklung überfachlicher Kompetenzen ist vor allem bei integrativen Ansätzen möglich, „nicht zuletzt auch deshalb, weil sie bessere Lernergebnisse bei den Fachkompetenzen zeitigen, einen engen Bezug zu den fachlichen Inhalten aufweisen und sich somit die Wahrscheinlichkeit zur Anwendung von Erlerntem in der späteren beruflichen Praxis erhöhen“ [SB04]. Hier wird anhand nichttraditioneller Lernformen wie selbstorganisiertes, problemorientiertes oder projektorientiertes Lernen die Förderung von Schlüsselqualifikationen mit dem Erwerb von Fachwissen verknüpft. Additive Ansätze, also die Vermitt-

lung von Schlüsselqualifikationen ausschließlich in separaten Veranstaltungen, sind kritisch zu sehen.

2.2 Herausforderung für Lernende und Lehrende

Studierende, die sich im Team intensiv am konkreten Problem mit der Fachmaterie auseinandersetzen, lernen in allen vier Kompetenzfeldern dazu. Sie tun das aber nur dann, wenn sie Lernmotivation mitbringen und ein tieferes Verständnis der Materie anstreben, also „deep-level learners“ [BA06] repräsentieren und nicht ausschließlich auf bestandene Prüfungen abzielen.

Nicht-traditionelle Lehrformen werden gerne auch als Lernformen bezeichnet [FW06], weil sie für den Übergang von der Lehrenden-zentrierten Wissensvermittlung zur Lernenden-zentrierten Unterstützung des Wissenserwerbs stehen. Sie stellen vor allem auch an die Lehrenden besondere Anforderungen.

Mehr als jede Lehrform erfordert die Lernform „Projekt“ die Bereitschaft der Betreuenden als Coach der Studierenden bzw. der Projektteams zu arbeiten. Die Lehrenden müssen von dieser Art „Steuerung auf Distanz“ mit dem Vertrauen in den Willen und die Fähigkeit der Lernenden, das eigene Verhalten in selbstgewählte Bahnen zu lenken, überzeugt und bereit sein, auf direkte Steuerung zu verzichten [B00].

Das bedeutet auch, dass sie Vertrauen in die Lernenden haben und dieses auch zeigen. Ihr Menschenbild sollte dem der Y-Theorie nach [Mc60] folgen, also der Führungsphilosophie moderner Unternehmen entsprechend davon ausgehen, dass auch Studierende von Natur aus leistungsbereit und aus sich selbst heraus motiviert sind. Die zum Teil noch sehr jungen Studierenden der ersten Semester stellen dieses Vertrauen gelegentlich durch ihr Verhalten auf die Probe.

Nach [B00] sollten Lehrende beim Coaching von Studierendengruppen verschiedene Rollen wahrnehmen können, nämlich gleichzeitig Auftraggebende, Vorbilder, Controller, Expert_innen und Evaluierende sein:

Auch wenn in studentischen Projekten die Aufgaben gerne relativ offen vorgegeben und letztendlich von den Gruppen konkretisiert werden, so werden doch Aufgabe und Vorgehensweise mit Meilensteinen nach Absprache mit den Betreuenden festgelegt, die damit zu Auftraggebenden werden.

Zur Wahrnehmung der Vorbildfunktion der Betreuenden gehören vor allem die Tätigkeiten Fragen und Zuhören. Beide signalisieren konsequente Neugier für das Projektthema und den Projektverlauf.

Das Controlling besteht hier in der Überwachung der Zusammenarbeit in den Projektgruppen, die gelegentlich auch die Expertise der Betreuenden für die fachlichen Aspekte der Projekte in Anspruch nehmen.

Für die Evaluation werden regelmäßig in Gruppen- und in Einzelgesprächen positive und negative Beobachtungen der Betreuenden und der Gruppenmitglieder zur Sprache gebracht und Verbesserungsvorschläge diskutiert.

Damit helfen Coaches den Studierenden bei der Entdeckung und Entwicklung ihrer Selbstlernfähigkeiten. Sie unterstützen beim Erwerb von Fachkenntnissen und leiten an, das eigene Lern- und Arbeitsverhalten zu reflektieren. Darüber hinaus leisten sie Beiträge, die Kommunikation und die Arbeit in Gruppen zu organisieren, bspw. zu planen, Besprechungen durchzuführen, Arbeitspakete zu verteilen oder Arbeitsergebnisse zu präsentieren.

3 Idee des Startprojekts

Das im Folgenden vorgestellte Projekt stellt ein Beispiel des in 2.1 beschriebenen integrativen Ansatzes dar, bei dem überfachliche Kompetenzen gemeinsam mit Fachkompetenz aufgebaut werden. Wir stellen unsere Idee eines Erstsemester-„Startprojekts“ vor und berichten über die in bisher vier aufeinanderfolgenden Wintersemestern seit 2012 gewonnenen Erfahrungen.

Das Startprojekt soll einen Beitrag zur Entwicklung fachlicher und überfachlicher Kompetenzen liefern. Ein besonderer Akzent liegt auf der Selbstkompetenz, speziell der Fähigkeit, Probleme mit Verständnis, Neugier und Ausdauer zu lösen, also ein deep-level learner sein zu wollen.

Darüberhinaus versprachen wir uns von einem Projekt eine gute Beziehung zwischen Lehrenden und Lernenden, eine bessere Vernetzung der Studierenden, Impulse zur Bildung von Lerngruppen, Einblicke in Methoden und Vorgehensweisen des Software-Engineering und Projektmanagements sowie eine Demonstration der Vielfalt von Informatikanwendungen. Nicht zuletzt wollten wir Lust auf Informatik machen und mit reizvollen Themen motivieren. Das Potential der Überforderung, welches einem Projekt gerade für Erstsemester innewohnt, wollten wir durch geringe softwaretechnische oder Projektmanagement-Anforderungen und einen hohen Präsenz-Anteil abfedern.

Wichtiger Auslöser, die Idee in die Tat umzusetzen, war zudem der Start des Niedersachsen-Technikums[1], in dem wir Abiturientinnen und Fachabi-

[1] http://www.niedersachsen-technikum.de.

turientinnen an ihrem Hochschultag eine möglichst praxisnahe Lehrveranstaltung bieten wollten, an der sie gemeinsam mit unseren „normalen“ Studierenden teilnehmen konnten. Die Lehrform Projekt erschien hier besonders geeignet, da, wie in einschlägigen Studien beschrieben, „die Kreativität und Kommunikation fördernde ... Projekte offenbar Lerninteressen von Frauen eher entgegen kommen.“ [S03]

Das Startprojekt wird von etwa 120 Teilnehmenden absolviert, die in ca. 24 Teams von 3 bis 7 Lehrkräften, überwiegend durch Professor_innen, betreut werden. Für 4 Kreditpunkte wird ein sechsstündiger Projekttag im Stundenplan der ersten Semester für das Startprojekt als Präsenzzeit reserviert. Die Lernziele des Startprojekts laut Modulbeschreibung sind:

- Selbstkompetenz: Die Studierenden identifizieren erfolgreiche Strategien der Selbstorganisation, Eigeninitiative, Recherche und Wissensaneignung.
- Soziale Kompetenz: Die Studierenden haben Teamarbeit ausprobiert und kennen die Bedeutung der Kommunikations- und Präsentationsfähigkeiten für den Projekterfolg.
- Projektmanagementkompetenz: Die Studierenden kennen einfache Methoden zur Projektplanung und Projektkontrolle und können diese in einem kleinen Projekt anwenden.
- Fachkompetenz: Die Studierenden kennen die hohe Anwendungsbandbreite der Disziplin Informatik. Sie sind in der Lage, über ein fachspezifisches Problem zielgerichtet zu debattieren, es zu analysieren und über einen mehrwöchigen Zeitraum eine Lösung zu entwickeln.

Andernorts durchgeführte Einstiegsprojekte weisen Ähnlichkeiten mit unserem Projekt auf. Unterschiede bestehen bspw. in der Gestaltung als wöchentlich stattfindendes Modul, im Gegensatz zum Blockcharakter des STEP in Bremerhaven [V^{+}14]. Verglichen mit dem EPRO der HS Bonn-Rhein-Sieg [E12] betont unser Projekt die Selbstkompetenz stärker: wir delegieren die Verantwortung für die Zieldefinition zu einem guten Teil in das Team und meiden die Prüfungsform Klausur.

Im folgenden Abschnitt stellen wir zu vier wichtigen Teilbereichen Merkmale der Gestaltung unseres Startprojekts vor: Anwesenheit und Betreuung, Themenstellung, Teamzusammensetzung, sowie Prüfungsleistung. Zu jedem Teilbereich berichten wir über unsere Erfahrungen in der Durchführung.

4 Gestaltungsmerkmale des Startprojekts und Erfahrungen

4.1 Anwesenheit und Betreuung

Die Projektarbeit findet überwiegend im Präsenzstudium mit Anwesenheitspflicht in von einigen Teams gemeinsam genutzten Räumen statt. Die Teams treffen sich zu Beginn und am Ende des wöchentlichen, sechsstündigen Projekttages mit ihrer Tutor_in zur Planung und Beratung. Etwa alle zwei Wochen wird eines der Treffen von der betreuenden Lehrkraft begleitet. Der regelmäßige Kontakt zwischen Professor_innen und Erstsemestern ist uns wichtig, um Wertschätzung zu zeigen und um einen direkteren Draht zu unseren Studierenden zu haben. Tutor_innen wirken motivierend, indem sie sich für die erreichten Ergebnisse interessieren und methodische und fachliche Tipps geben. Zudem halten sie die Anwesenheit fest und unterstützen die ausgewogene Mitarbeit aller Teammitglieder. Lehrkraft und Tutor_in nehmen also gemeinsam die Aufgaben als Vorbilder und Controller wahr (vgl. 2.2).

4.1.1 Erfahrungen

Die Anwesenheitspflicht führte zu geschäftigem Treiben in unseren Projekträumen, was von vielen Angehörigen der Abteilung als äußerst positiv wahrgenommen wurde. Die gemeinsame Arbeit vor Ort erleichterte kurzfristige Abstimmungen im Team. Das Frustrationspotential bei mangelhaften Schnittstellenabsprachen, welche bei oft unerfahrenen Erstsemestern nicht auszuschließen sind, konnte so einigermaßen in Grenzen gehalten werden.

Unser Startprojekt verbrauchte etwa 50 Prozent mehr Lehrkapazität als eine klassische 2V+2Ü-Veranstaltung. Zu Beginn bzw. am Ende des Projekts erforderten Vorbereitung und fachliche Einweisung bzw. Ergebnissicherung höhere Aufwände. In der Mitte des Semesters beschränkte sich Zusatzaufwand auf punktuelle Konfliktlösungsgespräche in „Problemteams“. Aufwände für Tutor_innen lagen je Semester bei etwa 430 Personenstunden.

4.2 Themenstellung

Jede Lehrkraft bietet „ihr“ Projektthema an. Die Themen sind an die Lehrveranstaltungen des ersten Semesters angelehnt, also Grundlagen der Informatik, Mathematik, Theoretische Informatik und Programmierung.

Wir versuchen, viele Projektthemen so zu gestalten, dass sich das Team im Rahmen einer vorgegebenen Technologie ein eigenes Ziel setzen kann. Die Rolle als Auftraggebende nach 2.2 füllen Lehrkräfte daher ganz bewusst nur unvollständig aus. Fachliche Einweisung in das Projektthema besteht i. d. R. aus einem von der Lehrkraft durchgeführten 2- bis 4-stündigen Workshop, an dem alle von ihr betreuten Teams teilnehmen, und der Aushändigung einer Literaturliste. Von da an sind die Studierenden intentionsgemäß inhaltlich auf sich allein gestellt. Es wird offen kommuniziert, dass alle lösbaren inhaltlichen Probleme vom Team selbst gelöst bzw. recherchiert werden sollten. Tutor_innen und Lehrkräfte sollen nur bei schweren fachlichen Problemen als Expert_innen (vgl. 2.2) hinzugezogen werden.

4.2.1 Erfahrungen

Als gut geeignet im Sinne der Zufriedenheit und Motivation der Studierenden haben sich Themen mit einem spielerischen Zugang zur Softwareentwicklung erwiesen: NAO[2]-Roboter durch ein Labyrinth steuern, Greenfoot[2]-Spiele programmieren, Apps mit dem App Inventor[2] entwickeln. Aber auch andere Themen wurden positiv bewertet: Strategien für ein Vier-Gewinnt-Spiel realisieren, einen Sudoku-Löser programmieren, einen Umfrageserver (Hardware, Betriebssystem und Software) konzipieren und realisieren, 3D-Spiele mit Alice[2] implementieren, Duell-Programme in der Core War[2] Programmiersprache Redcode gegeneinander antreten lassen, eine Propelleruhr bauen und programmieren, Minecraft[2] zur Modellierung komplexer Algorithmen nutzen. Die Themenstellungen sind aus Lehrendensicht wiederverwendbar, was den Aufwand für eine erneute Durchführung im Folgejahr mindern kann. Wichtig bei der Themenwahl ist selbstverständlich, dass die zur Herstellung von Projektergebnissen erforderlichen Fachinhalte von den Studierenden in kurzer Zeit selbstständig erlernt werden können.

4.3 Teamzusammensetzung

Eine Möglichkeit zur Bildung von Teams sieht die zufällige Zuordnung der Studierenden zu Teams und Themenstellungen vor. Dieses Verfahren entspricht am ehesten der beruflichen Realität, in der man sich Teammitglieder oder Thema in der Regel auch nicht frei aussuchen kann.

[2] http://www.aldebaran.com, http://www.greenfoot.org, http://appinventor.mit.edu, http://www.alice.org, http://www.corewar.info, https://minecraft.net.

Eine andere Möglichkeit besteht in der Bildung von Teams von Studierenden, die in etwa über ähnlich hohes Vorwissen verfügen. Solche „vorwissenhomogenen Teams“ haben vermutlich weniger Probleme mit der gleichmäßigen Arbeitsverteilung und der gleichberechtigten Zusammenarbeit.

4.3.1 Erfahrungen

In den zufällig gebildeten Teams, deren Mitglieder sich in der Regel vorher nicht kennen, beobachten wir wie erwartet Kommunikationsprobleme, Meinungsverschiedenheiten und Konflikte. Der Bearbeitung dieser Probleme können die teilnehmenden Studierenden kaum ausweichen. Die oben skizzierte engmaschige Betreuung durch Lehrkräfte hat es uns ermöglicht, unerwünschte soziale Prozesse, wie Ausgrenzung oder Verweigerung, frühzeitig zu erkennen und in Gesprächen zu korrigieren.

Seit dem Wintersemester 2015 versuchen wir, annähernd vorwissenshomogene Teams zu bilden. Das Vorwissen schätzen wir mit Hilfe der Kombination eines dreiseitigen Fragebogens bezüglich der bisherigen Ausbildung in der Informatik und eines kleinen Tests mit Aufgaben aus dem Grundlagenbereich. Außerdem gehören die Mitglieder jedes Teams jeweils demselben der zwei Informatikstudiengänge unserer Abteilung an.

Seit dem Wintersemester 2014 sorgen wir mit dem Ziel einer besseren Integration dafür, dass Angehörige von „Minderheiten“ in unserem Studiengang Angewandte Informatik, nämlich Frauen, speziell auch Technikantinnen, und Austauschstudierende, jeweils zu zweit im Team vertreten sind.

Die Befürchtung, dass Teams ohne Vorkenntnisse sich gegenüber Teams mit Vorkenntnissen unterlegen fühlten, bewahrheitete sich eher nicht. Wohl aber zeigte sich in trotz unserer Bemühungen entstandenen weniger homogenen Teams, dass gleichberechtigtes Arbeiten der Mitglieder sich schwer realisieren ließ, vor allem, wenn sie sich unter Zeitdruck fühlten.

Die Vernetzung der Studierenden untereinander ist nach unserer Wahrnehmung besser als früher. Diese Aussage umfasst auch die Integration ausländischer Studierender. Mangels belastbarer Zahlen aus Vorjahren ohne Startprojekt nennen wir die folgenden Zahlen als Indizienbelege für unsere Wahrnehmung: Auf einem Evaluationsbogen gab etwa ein Drittel der Studierenden an, durch das Startprojekt eine Lerngruppe gefunden zu haben. Etwas mehr als ein Drittel gab an, durch das Startprojekt Kontakte für Freizeitaktivitäten gefunden zu haben.

Die Anwesenheit von Technikantinnen in den Projektgruppen zeigte, wie es aussehen könnte, wenn es mehr Frauen in den Studiengängen und im „Informatik-Alltag“ gäbe. Die eher zurückhaltende, manchmal abwartende Haltung einiger Technikantinnen wurde allerdings mitunter von den männlichen Projektkollegen fälschlich als mangelndes Interesse interpretiert.

Im ersten Durchlauf waren etliche Studierende sehr unzufrieden mit ihrer Themenstellung. Wir haben daher im zweiten Durchlauf zu Semesterbeginn eine mehrtägige Online-Tauschbörse ermöglicht, bei der die Studierenden im Rahmen der Kapazitäten ihr Thema und damit auch ihr Team wechseln konnten. Diese wurde intensiv angenommen, wir konnten jedoch bei weitem nicht alle Tauschwünsche erfüllen.

Die Motivation, ein Projekt zu bearbeiten, ist bei einem selbstgewählten Thema sicherlich in der Anfangsphase höher. Die Erfahrung aus anderen Projekten unserer Studiengänge stimmt aber mit unseren eigenen Erfahrungen überein, nämlich, dass fast jedes Thema reizvolle Herausforderungen stellt und Spaß machen kann. Da die konkrete Ausgestaltung der offen formulierten Projektthemen bei den Projektgruppen liegt, haben wir in weiteren Durchgängen auf die Tauschbörse verzichtet.

4.4 Prüfungsleistungen

Als Evaluierende (vgl. 2.2) definieren wir die summative Prüfungsleistung im Startprojekt wie folgt: Eine Leistung wird unbenotet dann bescheinigt, wenn die individuelle Anwesenheitspflicht erfüllt ist, wenn das Projektergebnis, also Produkt und Dokumente, vorliegt und wenn die Planungs- und Verlaufsdokumente kontinuierlich durch das Team erstellt wurden. Es wird keine perfekte Planung des Vorgehens erwartet und auch keine Dokumentation vor der Implementierung. Das Projekt soll Charakterzüge einer „Spielwiese“ besitzen, auf der sich die Studierenden in einem gegebenen Themenrahmen nach Neigung inhaltlich „austoben“ können.

Während der Projektlaufzeit sind regelmäßige Treffen mit der Tutor_in oder der Lehrkraft anberaumt. Bei diesen Treffen wird von den Studierenden erwartet, dass sie ihre Fortschritte präsentieren. Tutor_innen und Lehrkräfte nutzen diese Treffen als formative Prüfungsgelegenheit, bei der der Kompetenzaufbau durch gezieltes Feedback bezüglich Fachlichem und Überfachlichem gestützt wird.

Individuelle Prüfungs- und Beratungsgespräche der Lehrkräfte mit jeder einzelnen Teilnehmerin und jedem einzelnen Teilnehmer mit einer Dauer von

ca. 15 Minuten in der Mitte oder am Ende des Semesters ermöglichen deren Reflexion der Mitarbeit im Projekt und ihrer Zufriedenheit mit der Studienfachwahl. Damit nimmt die Lehrkraft Prüfungsaufgaben, Coachingaufgaben und die Rolle einer Mentorin oder eines Mentors wahr.

Am Ende des Semesters ist von jedem Team ein Poster über die Projektergebnisse zu erstellen und auf einer Postermesse der Abteilung, häufig auch mit Demonstration am Rechner, zu präsentieren. Bei der Postermesse werden alle Projekte der Abteilung, auch die der höheren Semester und der Masterprojekte, also insgesamt etwa 50 bis 60 Projekte, präsentiert.

Die o. g. summative Prüfungsleistung des Startprojekts ist überwiegend als Teamleistung formuliert und wird nicht auf einer Notenskala benotet. Von einer individuellen Notenvergabe haben wir aus verschiedenen Gründen Abstand genommen. Ein wichtiger Grund ist die Hoffnung, das herausfordernde Thema und der Zusammenhalt des Teams reiche zusammen mit einer nicht-individuellen Teamleistungsanforderung aus, das individuelle Lernverhalten der Teammitglieder in die gewünschten Bahnen von deep-level learners zu lenken, die ihre Motivation eben nicht nur aus dem Wunsch nach einer guten Prüfungsnote ziehen. Außerdem gibt es den pragmatischen Grund, dass die o. g. Betreuungsdichte durch die Lehrkraft nicht ausreicht, um individuelle Leistungen in einem Team ausreichend differenziert zu beobachten.

4.4.1 Erfahrungen

Alle bisherigen Projektteams haben das Projektziel erreicht, wovon sich auf der Postermesse alle an Projekten beteiligten Lehrkräfte und Studierenden sowie zahlreiche Gäste selbst überzeugen konnten.

Die Auswahl geeigneter Instrumente zur Prozessdokumentation machte uns Schwierigkeiten: Die in den zwei ersten Durchläufen erstellten Sitzungsprotokolle sollten einerseits der studentischen Reflexion des Erreichten dienen und andererseits den Betreuenden die Möglichkeit bieten, sich zeit- und ortsunabhängig über den Projektfortschritt zu informieren. Beide Funktionen wurden durch eher unwillig erstellte Protokolle häufig nicht erreicht. Im dritten Durchlauf versuchten wir, die Funktion der studentischen Reflexion zu erreichen, indem wir von jeder Teilnehmerin und jedem Teilnehmer das Führen eines persönlichen Lerntagebuchs verpflichtend einforderten, dessen wöchentliche Erweiterung durch themenbezogene Einträge durch die Tutor_innen geprüft wurde. Diese Art der Dokumentation stieß bei den

Studierenden auf großen Widerstand und schien zu eher ablehnender Haltung zur Projektarbeit zu führen.

Im letzten Semester forderten wir von jedem Team einen wöchentlichen Statusbericht mit einer „Ampel“, die mit ihrer Farbe eine Zusammenfassung des Status anzeigte, dem in Stichpunkten beschriebenen aktuellen Status, den Meilensteinen der kommenden vier Wochen, aktuellen Problempunkten und den Maßnahmen zu ihrer Bewältigung sowie Anforderungen an die Lehrkraft zum Fortschreiten des Projekts. Die Bedeutung jedes dieser Punkte wurde immer wieder erklärt. Diese Art der Dokumentation funktionierte bisher am besten, obwohl immer wieder Erklärungen zum Zusammenhang zwischen der Ampelfarbe und den Problempunkten eingefordert werden mussten. In sich konsistente Statusberichte lieferten den Betreuenden kompakte Information und die Grundlage für Diskussionen und Beratungen bei den Teamsitzungen.

Eine Neuerung des letzten Semesters, der Abschluss eines Projektauftrages zwischen Betreuenden und jedem Team, wurde von allen Beteiligten unterzeichnet und bewirkte, dass die Verantwortung jedes einzelnen Teammitgliedes für das angestrebte Ergebnis noch deutlicher wurde.

Die Postermesse aller Projekte aller Studierenden der 1., 3., 5. Bachelorsemester und des 3. Mastersemesters erleben wir als großen Gewinn. Studierende bringen Freund_innen und Verwandte mit, um ihnen durchaus mit Stolz ihre Ergebnisse zu präsentieren. Sie vertreten ihr Projektergebnis im Gespräch mit dem Messepublikum mitunter leidenschaftlich und selbstbewusst. Die Messe bietet die Gelegenheit, die eigene Projektarbeit wiederholt den verschiedenen vorbeikommenden Interessierten vorzustellen.

Aber auch über den Kompetenzerwerb der Teilnehmerinnen und Teilnehmer hinaus hat die Postermesse positive Effekte. Sie schafft ein Abteilungs-„Wir“-Gefühl und bewirkt u. a., dass die Informatik und unsere Studiengänge über Abteilungs- und Hochschulgrenzen hinweg sichtbar werden. Den durchaus vorhandenen Organisationsaufwand für die Postermesse werden wir aus diesen Gründen auch bei den künftigen Durchläufen nicht scheuen.

Dem beim ersten Durchlauf zu beobachtenden Effekt, dass einige Studierende vor ihrem eigenen Poster stehen blieben und die Gelegenheit, sich abteilungsweit über Informatikthemen zu informieren, nicht nutzten, haben wir seit dem zweiten Durchlauf mit einer Abstimmung und einer öffentlichen Preisverleihung erfolgreich entgegen gewirkt. Alle Studierenden waren aufgefordert, eine Stimme für das beste Poster abzugeben. Die Möglichkeit,

einen Preis zu erhalten, hat zudem sicher zusätzlich motivierend auf viele Studierende bei der Postererstellung gewirkt.

5 Zusammenfassung

In diesem Beitrag stellen wir die Annahme auf, dass es möglich und sinnvoll ist, überfachliche Kompetenzen im Informatikstudium von Beginn an zu fördern und geeignet zu prüfen. Die überwiegend im Grundstudium eingesetzte Prüfungsform Klausur steht diesem Ziel entgegen, weshalb wir im Startprojekt ein Team mit der Aufgabe betrauen, ein Ziel zu formulieren und dieses nach einem selbst aufgestellten Plan zu erreichen. Die von den Studierenden abverlangten formalen Elemente zur Planung, die Statusberichte, Postererstellung und Präsentation unterstützen die Entwicklung von Methodenkompetenz. Ganz überwiegend jedoch dient das Projekt dem Aufbau von Sozial-, Selbst- und Fachkompetenz. Von den Lehrenden erfordert die Begleitung eines solchen Projektes die Rolle eines Coaches einzunehmen.

Wird das Startprojekt seinem Ziel gerecht? Kompetenz können wir nicht als solche messen [SH13], sondern es wird von einer beobachtbaren Handlung auf die zugrundeliegende Kompetenz geschlossen. In diesem Sinne können wir aus den beobachteten Handlungen unserer Studierenden einen quantitativ schwer messbaren, jedoch qualitativ beobachtbaren Kompetenzzuwachs konstatieren. Wir halten das Startprojekt für einen wertvollen Baustein im ersten Bachelorsemester unserer Informatik-Studiengänge, den wir für andere Studiengänge mit ähnlichen Lernzielen wärmstens weiter empfehlen wollen.

Literatur

[B00] H. Blom: Der Dozent als Coach, Hochschulwesen, Wissenschaft und Praxis, Luchterhand, 2000.

[GI05] Gesellschaft für Informatik: Bachelor- und Masterprogramme im Studienfach Informatik an Hochschulen, 2005. https://www.gi.de/fileadmin/redaktion/empfehlungen/GI-Empfehlung_BaMa2005.pdf.

[SB04] H. Schaepner, K. Briedis: Kompetenzen von Hochschulabsolventinnen und Hochschulabsolventen, berufliche Anforderungen und Folgerungen für die Hochschulreform. HISKurzinformation A 6 / 2004, HIS, Hannover. http://www.his.de/pdf/pub_kia/kia200406.pdf.

[BA06] C. Brabrand, J. Andersen: Teaching Teaching & Understanding Understanding. Kurzfilm, Aarhus University Press, University of Aarhus, Dänemark, 2006. http://www.daimi.au.dk/~brabrand/short-film/.

[FW06] P. Forster, A. Winteler: Vom Lehren zum Lernen: Ein neues Paradigma für die Hochschullehre. In: Ausbildung in der Logistik, DUV, 2006.

[Mc60] D. McGregor: The Human Side of Enterprise, McGraw-Hill, 1960.

[S03] B. Schinzel: Curriculare Vorschläge zur Erhöhung des Frauenanteils in der Informatik, 2003. http://mod.iig.uni-freiburg.de/cms/fileadmin/publikationen/curriculuminf.pdf.

[E12] Modul Einsteigerprojekt (EPRO), Hochschule Bonn-Rhein-Sieg, 2012. https://eva2.inf.h-brs.de/akkr/public/studiengangtabelle/6/243/.

[V+14] K. Vosseberg et al.: Projektorientierte Studieneingangsphase: Das Berufsbild der Informatik und Wirtschaftsinformatik schärfen. HDI 2014.

[SH13] N. Schaper, F. Hilkenmeier: HRK-Zusatzgutachten Umsetzungshilfen für kompetenzorientiertes Prüfen, 2013. http://www.hrk-nexus.de/fileadmin/redaktion/hrk-nexus/07-Downloads/07-03-Material/zusatzgutachten.pdf.

Lernwirksamkeits- und Zielgruppenanalyse für ein Lehrvideo zum informatischen Problemlösen

Bertold Kujath

Didaktik der Informatik, Universität Potsdam
14482 Potsdam
Email: kujath@cs.uni-potsdam.de

Zusammenfassung: Aus einer Vergleichsstudie mit starken und schwachen Problemlösern konnten Erkenntnisse über die effizienten Herangehensweisen von Hochleistern an Informatikprobleme gewonnen werden. Diese Erkenntnisse wurden in einem Lehrvideo zum informatischen Problemlösen didaktisch aufgearbeitet, sodass Lernenden der Einsatz von Baumstrukturen und Rekursion im konkreten Kontext gezeigt werden kann. Nun wurde die tatsächliche Lernwirksamkeit des Videos sowie die Definition der Zielgruppe in einer Vergleichsstudie mit 66 Studienanfängern überprüft.

1 Einleitung

Das Medium „Lehrvideo“ ist in der letzten Zeit durch die Verbreitungsmöglichkeiten im Internet wieder in das Interesse der Didaktik gerückt. Trotz der bekannten Schwachpunkte dieses Mediums bedingt durch seine Eindimensionalität, bietet das Lehrvideo einen didaktischen Mehrwert durch hohe Anschaulichkeit und die Möglichkeit, auch komplexe Abläufe dynamisch darzustellen [KF94]. Auch [Pa86] und [Be09] sehen aus lernpsychologischer Sicht Vorteile audiovisueller Medien, da hier Lehrinhalte gleichzeitig durch unterschiedliche Kodierungsformen (Bild/Text) präsentiert werden können, wodurch die Aufmerksamkeits- und Behaltensleistung erhöht wird. Einen noch stärkeren Lerneffekt kann ein Lehrvideo dann bieten, wenn es nicht isoliert eingesetzt, sondern im Rahmen eines abgestimmten Unter-

richtskonzepts verwendet wird [SF15], etwa als einführende Lernhilfe für ein neues Thema.

In der vorliegenden Arbeit wird die Lernwirksamkeit eines Videos zum informatischen Problemlösen analysiert. Aus den didaktisch aufgearbeiteten Ergebnissen einer Laut-Denken-Studie mit starken und schwachen Problemlösern wurde ein Lehrvideo produziert, das einen fiktiven Hochleister bei der Bearbeitung eines vergleichsweise schwierigen Färbeproblems zeigt [Ku11a]. Dieser Hochleister spricht dabei seine Überlegungen laut aus, die von ihm fortlaufend angefertigten Lösungsskizzen werden von oben gezeigt und abschnittsweise durch animierte Filmsequenzen und verbale Erläuterungen verdeutlicht. Das Video behandelt die informatischen Problemlösemethoden „Bäume“ und „Rekursion“ und ist in erster Linie für diejenigen Informatiklernenden gedacht, die diese Inhalte zwar kennen, aber nicht im konkreten Problemlösekontext anwenden können. Eine detaillierte Erläuterung des dem Video zugrunde liegenden Konzepts sowie eine Beschreibung der Zielgruppe finden sich in [KS09] und [Ku11b].

1.1 Bisherige Evaluationen

1.1.1 Befragung von Erstsemesterstudenten

In einem ersten Evaluationsschritt wurde das fertige Video durch eine Befragung beurteilt. Insgesamt 60 Erstsemesterstudenten der Informatik und Wirtschaftsinformatik wurde das Video präsentiert und anschließend ein Fragebogen vorgelegt. Der Fragebogen enthielt Fragen zur Informatik-Vorerfahrung, zur Selbsteinschätzung der eigenen Problemlösefähigkeit, Verständnisfragen zum Inhalt und Fragen zur persönlichen Bewertung des Videos hinsichtlich Verständlichkeit und der Eignung als Lehrmittel. 85 % bezeichneten ihre Problemlösefähigkeit als „schwach“ oder „eher schwach“. Diese Teilnehmer gehörten somit zur Zielgruppe des Videos. Die Auswertung des Fragebogens ergab eine hohe Akzeptanz des im Video umgesetzten Konzepts und signalisierte einen Bedarf an gezielter Vermittlung konkreter Problemlösetechniken. Mehrheitlich erklärten die Befragten, das Video hätte ihnen weiterführende Erkenntnisse beim Bearbeiten von Informatikproblemen gebracht. 58 % der Teilnehmer äußerten, durch das Video eine höhere Motivation zur Beschäftigung mit Informatikproblemen zu haben. Mehr als 70 % der befragten Studenten würden es befürworten, auch andere informatische Methoden nach einem solchen Konzept zu erlernen.

1.1.2 Erste Lernwirksamkeitsstudie

Während die Befragung auf die subjektive Bewertung durch die Studenten abzielte, untersucht [Sc13] in seiner Studie, ob tatsächlich eine Lernwirksamkeit des Videos festgestellt werden kann, näher beschrieben auch in [KS14]. Die Auswertung der Studie erfolgte gestützt durch zwei Hypothesen. Nach der ersten Hypothese finden Probanden aus der Gruppe „mit Video" häufiger die richtige Lösung als Probanden der Gruppe „ohne Video". In der Gruppe „mit Video" präsentierten von zufälligen Fehlern bereinigt 25 % der Probanden die richtige Lösung, in der Gruppe „ohne Video" dagegen nur 18 %, womit [Sc13] Hypothese 1 bestätigt sieht, wenn auch nicht mit einem ausreichenden Signifikanzniveau. Hypothese 2 besagt, dass Probanden aus der Gruppe „mit Video" durch Anwendung wenigstens einer der im Lehrvideo gezeigten Methoden den qualitativ besseren Lösungsweg nutzen. Die Analyse der Lösungen ergab, dass 17 % der Teilnehmer der Gruppe „mit Video", aber kein einziger der Teilnehmer aus der Gruppe „ohne Video" wenigstens eine der im Video gezeigten Methoden bei der Lösungsbearbeitung angewendet hat. Hypothese 2 ist anhand dieses Ergebnisses nicht signifikant bestätigt, zumindest aber nicht widerlegt.

1.2 Zusammenfassung der Ergebnisse bisheriger Evaluationen

Bisherige Auswertungen belegen eine didaktische Relevanz des Videos. Die persönlichen Einschätzungen der Teilnehmer aus den Befragungen sind zwar subjektiv, aber der Autor sieht das belegte hohe Maß an Akzeptanz seitens der Zielgruppe gegenüber einem bestimmten Lehrmedium als wichtige Voraussetzung für eine Lernmotivation an. Die Studie nach [Sc13] deutet ebenfalls auf eine Lernwirksamkeit des Videos hin, sollte aber durch weitere Forschungsergebnisse bestätigt werden.

2 Erweiterte Lernwirksamkeitsstudie und Zielgruppenanalyse

2.1 Studiendesign

Aufgrund der geringen Teilnehmerzahl und der fehlenden Signifikanz der Ergebnisse der in Kapitel 1.1.2 beschriebenen Lernwirksamkeitsstudie wurde eine zweite Studie mit deutlich mehr Teilnehmern konzipiert. Teilnehmer für

diese Studie wurden im ersten Semester angeworben, die Bewerbung erfolgte über einen Online-Fragebogen. Hier sollten zunächst, ähnlich wie in den vorangegangenen Evaluationen, u. a. Fragen zur informatischen Vorerfahrung sowie Fragen zur Selbsteinschätzung der eigenen Problemlösefähigkeit beantwortet werden. Um das Ausmaß der Vorerfahrung genauer quantifizieren zu können, wurden im Fragebogen verschiedene informatische Inhalte (Programmierung, Algorithmen und Datenstrukturen, Automatentheorie, Baumstrukturen, Graphentheorie und Rekursion) aufgeführt und nach dem Grad des Verständnisses der jeweiligen Inhalte auf einer Skala von 1 (gar nicht) bis 5 (vollständig) gefragt. Relevant für die vorliegende Studie waren die Inhalte „Baumstrukturen" und „Rekursion". Die Selbsteinschätzung der eigenen Problemlösefähigkeit erfolgte durch die Antwortmöglichkeiten „stark", „eher stark", „eher schwach" und „schwach". Um bei denjenigen Teilnehmern, die bei der Aufgabenbearbeitung während der Studie unspezifisch und ohne informatische Problemlösemethoden vorgehen, die möglichen Ursachen besser identifizieren zu können, wurden die Studenten zusätzlich gebeten, sich als Problemlöser einer der vier folgenden Kategorien zuzuordnen:

A = „Informatische Problemlösemethoden beherrsche ich sicher und kann sie auch einsetzen.", im folgenden Kategorie „$\mathbf{A}_{\mathbf{alles}}$" genannt,

B = „Ich kenne informatische Problemlösemethoden und weiß auch, wie ich sie anwenden kann. Ich erkenne aber nicht, wann ich sie einsetzen kann.", im folgenden Kategorie „$\mathbf{B}_{\mathbf{wann}}$" genannt,

C = „Ich kenne informatische Problemlösemethoden zwar, weiß aber nicht, wie ich diese konkret anwenden kann.", im folgenden Kategorie „$\mathbf{C}_{\mathbf{wie}}$" genannt, und

D = „informatische Problemlösemethoden sind mir unbekannt.", im folgenden Kategorie „$\mathbf{D}_{\mathbf{nichts}}$" genannt.

2.1.1 Zusammensetzung der Teilnehmer

Nach Eliminierung ungültig beantworteter Online-Fragebögen wurden insgesamt 66 Teilnehmer zur Studie eingeladen. Die Teilnehmer wurden in zwei Gruppen zu je 33 Studenten aufgeteilt. Anhand der Kategorien „informatische Vorerfahrung", „Problemlösefähigkeit" und „Problemlösekategorie" und der dazu von den Teilnehmern ausgewählten Antwortmöglichkeiten wurden präselektiv die Teilnehmer derart aufgeteilt, dass eine möglichst paritätische Zusammensetzung der Teilgruppen hinsichtlich der (vermuteten) Problemlösekompetenz erzielt werden konnte.

In Bezug auf die oben geschilderten Fragenauswahl ergaben sich wie in den Tab. 1–3 dargestellt für die einzelnen Teilgruppen und die Gesamtgruppe nachfolgend aufgeführte Verteilung der Antworten:

	Grad der Vorerfahrung Baumstrukturen				
Häufigkeit in der…	**5**	**4**	**3**	**2**	**1**
… Gruppe „mit Video“[1]	0	11	14	5	2
… Gruppe „ohne Video“	4	9	12	5	3
… Gesamtstichprobe	**4**	**20**	**26**	**10**	**5**
	Grad der Vorerfahrung Rekursion				
Häufigkeit in der…	**5**	**4**	**3**	**2**	**1**
… Gruppe „mit Video“	1	10	13	7	1
… Gruppe „ohne Video“	2	7	9	8	7
… Gesamtstichprobe	**3**	**17**	**22**	**15**	**8**

Tab. 1: Informatische Vorerfahrung der Teilnehmer mit Bäumen und Rekursion

	Selbsteinschätzung Problemlösefähigkeit			
Häufigkeit in der…	**stark**	**eher stark**	**eher schwach**	**schwach**
… Gruppe „mit Video“	0	15	17	0
… Gruppe „ohne Video“	3	14	16	0
… Gesamtstichprobe	**3**	**29**	**33**	**0**

Tab. 2: Einschätzung der eigenen Problemlösefähigkeit

	Problemlösetyp			
Häufigkeit in der…	$\mathbf{A_{alles}}$	$\mathbf{B_{wann}}$	$\mathbf{C_{wie}}$	$\mathbf{D_{nichts}}$
… Gruppe „mit Video“	1	10	16	5
… Gruppe „ohne Video“	2	9	18	4
… Gesamtstichprobe	**3**	**19**	**34**	**9**

Tab. 3: Verteilung der Problemlösetypen

2.1.2 Diskussion der Aufgaben

Die erste Aufgabe bestand in einem vergleichsweise anspruchsvollen 3-Färbeproblem, bei dem eine Baumstruktur zur Lösung konstruiert werden

[1] Bei den tabellarischen Darstellungen, die auf die Ergebnisse des Fragebogens zurückgreifen, beträgt die Gesamtsumme in der Gruppe „mit Video“ nur 32 und nicht 33, da der Teilnehmer T133 aus dieser Gruppe den Fragebogen nicht beantwortet hat.

konnte und aus dieser Baumstruktur dann durch Rekursion die Frage nach der Formel für die Anzahl der Färbemöglichkeiten für beliebige *n* hergeleitet werden konnte. Dieser Teil des Tests sollte zeigen, wie die einzelnen Teilnehmer einen solchen Aufgabentyp ohne weitere Hinweise in der Regel bearbeiten würden. Die zweite Aufgabe, auch ein Färbeproblem, war ebenfalls mit den im Video gezeigten Methoden lösbar, allerdings mit einem gewissen Maß an Transferleistung. So war die in der Aufgabe gesuchte Zahl nicht direkt aus der gesamten Baumstruktur rekursiv ableitbar, sondern nur unter Hinzunahme eines Teilbaums, wodurch die direkte Anwendung des im Video Gezeigten nicht genügte und somit ein bloßes Memorieren des Lösungsbeispiels aus dem Video nicht zur Lösung führte. Zusätzlich gab es für beide Gruppen beim zweiten Färbeproblem die Zusatzbedingung: „Lösen Sie die Aufgabe mithilfe von Baumstrukturen und Rekursion." Dies sollte zum einen eine gewisse Chancengleichheit zwischen den Problemlösern der Kategorie B_{wann} („weiß nicht wann") aus der Gruppe „ohne Video" zu denen der Gruppe „mit Video" herstellen, da diese durch das Video einen Hinweis über mögliche Problemlösemethoden erhalten haben. Zudem sollten die Teilnehmer der Gruppe „mit Video" forciert werden, das im Video Gezeigte auch tatsächlich anzuwenden.

2.2 Ablauf des Tests

Beide Gruppen wurden zeitgleich in getrennte Hörsäle eingeladen. Ziel war es, in einer Vergleichsstudie die Veränderung des Problemlöseverhaltens nach Präsentation des Lehrvideos zu untersuchen. Präsentiert wurden jeder Gruppe insgesamt zwei verschiedene Färbeprobleme, beide Gruppen bekamen zunächst dasjenige Färbeproblem zur Bearbeitung vorgelegt, dass auch im Video vom fiktiven Hochleister „Tom" beispielhaft gelöst wird. Die Gruppe, die das Lehrvideo nicht zu sehen bekam, löste direkt im Anschluss daran auch das zweite Färbeproblem. Beide Aufgaben aus der Studie sowie eine kurze Diskussion der Lösungen sind im Anhang aufgeführt. Die zweite Gruppe, die Gruppe „mit Video", sah direkt nach dem Lösen der ersten Aufgabe das Lehrvideo, das ohne ergänzende didaktische Kommentare präsentiert wurde. Da es sich im Lehrvideo um dieselbe Aufgabe handelte, die unmittelbar zuvor von den Probanden bearbeitet wurde, erhoffte sich der Autor eine höhere Motivation und Aufmerksamkeitsleistung seitens der Teilnehmer bei der Betrachtung des Videos. Direkt im Anschluss an die Videovorführung bekam auch die Gruppe „mit Video" die zweite Aufgabe

zur Bearbeitung vorgelegt.[2] Zusätzlich wurden die Probanden bei jeder Aufgabe gebeten, kurz ihre Vorgehensweise bei der Aufgabenbearbeitung zu kommentieren. Für jede Aufgabe hatten die Teilnehmer 35 Minuten für die Lösungsbearbeitung zur Verfügung.

2.3 Auswertung der Lösungsbearbeitungen

Die Auswertung der insgesamt 132 Lösungsbearbeitungen erfolgte unabhängig voneinander durch zwei Gutachter[3], die während dieser Tätigkeit keine Kenntnis über die tatsächliche Gruppenzugehörigkeit der einzelnen Teilnehmer hatten. Zur Analyse standen die Blätter mit den angefertigten Lösungsskizzen und die Kurzkommentare der Teilnehmer zur Verfügung. Als problematisch bei der Analyse stellte sich heraus, dass es nur knappe Kommentare zu den Skizzen seitens der Probanden gab und die Lösungswege teilweise nicht in chronologischer bzw. inhaltlicher Reihenfolge auf die Skizzenblätter geschrieben wurden. Um eine aussagekräftige und zugleich auch möglichst gesicherte Bewertung zu ermöglichen, wurde die Analyse auf die Identifizierung eindeutiger und offensichtlicher Strukturen beschränkt.

2.3.1 Bewertungsschema und Diskussion

Die vorhandenen Skizzenblätter wurden von den Gutachtern getrennt gesichtet und für jedes der beiden Probleme in das folgende Bewertungsschema eingeordnet:

1: Baumstruktur und Rekursion vollständig, korrekt und lösungsführend angewendet.
2: Baumstruktur und Rekursion nur teilweise, im Ansatz aber erkennbar angewendet.
3: Jede andere Lösungsweg einschließlich „leeres Blatt".

Mit dieser Methode konnte die Bewertung vergleichsweise schnell und verlässlich anhand der grafischen Beurteilung der Skizzen vorgenommen werden. Die Bewertungen beider Gutachter waren überwiegend identisch, in einigen Fällen wurde eine unterschiedliche Bewertung im Dialog der beiden Gutachter nachträglich vereinheitlicht. Lösungen, die nicht mit den im Video gezeigten Inhalten in Verbindung standen, konnten durchaus auch zum

[2] Der Vollständigkeit halber sei erwähnt, dass auch die Gruppe „ohne Video" das Video gesehen hat, allerdings erst am Ende des Tests, nachdem die Aufgabenzettel bereits eingesammelt waren.
[3] Der eine Gutachter war Dozent für Algorithmen und Datenstrukturen, der andere der Autor selbst.

Erfolg führen, sollten hier aber nicht näher betrachtet werden. Nach Abschluss der Bewertung ergaben sich für jeden Teilnehmer zwei Zahlenwerte, jeweils ein Wert zwischen 1 und 3 für die Bearbeitung des ersten Färbeproblems und jeweils ein Wert zwischen 1 und 3 für die Bearbeitung des zweiten Färbeproblems. Eine qualitative Verbesserung des Lösungsweges im Sinne der beim zweiten Problem gestellten Zusatzbedingung liegt demnach vor, wenn bei einem Teilnehmer die Differenz zwischen erstem und zweitem Zahlenwert entweder +1 oder +2 beträgt. Liegt eine solche Verbesserung bei einem Teilnehmer vor, der zugleich der Gruppe „mit Video“ zugeordnet war, kann bei diesem Teilnehmer ein Lerneffekt durch Betrachtung des Videos vermutet werden. Da auch Verbesserungen in der Gruppe „ohne Video“ auftreten können (insbesondere durch den Zusatzhinweis), ist eine Lernwirksamkeit des Videos dann gegeben, wenn Verbesserungen des Lösungsweges beim zweiten Problem signifikant häufiger in der Gruppe „mit Video“ auftreten.

2.4 Ergebnisse

Lediglich 12 von 66 Teilnehmer (18,2 %) haben bereits beim ersten Mal, also ohne jegliche Hinweise zur Lösungsmethodik, mit Baumstrukturen und Rekursion gearbeitet und davon nur 5 Teilnehmer (7,6 %) vollständig und korrekt. Bis auf Ausnahmen waren die restlichen 54 Lösungsbearbeitungen ohne spezifische informatische Methoden, überwiegend durch bloßes Ausprobieren der Farbkombinationen angefertigt worden und ohne quantifizierbaren Lösungserfolg. Diese Problemlöser entsprechen damit der einführend beschriebenen Zielgruppe des Videos.

Beim zweiten Problem ergaben sich bei einer Vielzahl von Teilnehmern nach dem zuvor geschilderten Bewertungsschema Verbesserungen des Lösungsweges. In der Gruppe „mit Video“ zeigte sich bei insgesamt 19 von 33 Teilnehmern (57,6 %) eine Verbesserung des Lösungsweges, bei der Gruppe „ohne Video“ bei 11 von 33 Teilnehmern (33,3 %), siehe hierzu auch Tab. 4.

Gruppe	Teilnehmer	Verbesserungen	Prozentsatz
„mit Video“	33	19	57,6 %
„ohne Video“	33	11	33,3 %

Tab. 4: Verbesserung des Lösungsweges nach Gruppen

Diese Ergebnisse entsprechen einem Signifikanzniveau von 5 %. Berücksichtigt man zusätzlich den Grad der Lösungsverbesserung (GdLv), also die Frage, in welchem Ausmaß sich der Lösungsweg verbessert hat, findet man in der Gruppe „mit Video“ bei 4 von 33 Teilnehmern (12,1 %) eine Verbesserung des Lösungswegs um den Wert 2, in der Gruppe „ohne“ trat dies dagegen kein einziges Mal auf. In beiden Gruppen gab es auch eine geringe Anzahl von Verschlechterungen des Lösungswegs, die hier aber nicht weiter kommentiert werden sollen. In Tab. 5 wird dieser Zusammenhang zusammengefasst.

Häufigkeit in der…	**Grad der Lösungsverbesserung (GdLv)**				
	-2	**-1**	**0**	**1**	**2**
… Gruppe „mit Video“	1	2	11	15	4
… Gruppe „ohne Video“	0	1	21	11	0

Tab. 5: Grad der Lösungsverbesserung über Teilnehmer und Gruppen

2.5 Wem nützt das Video am meisten?

Im folgenden Abschnitt untersuchen wir, wie sich diejenigen Teilnehmer, die ihren Lösungsweg beim zweiten Färbeproblem verbessert haben, auf die in Abschnitt 2.1.1 geschilderten Selektionskriterien „informatische Vorerfahrung“, „Selbsteinschätzung“ und Problemlösetyp“ verteilen. Die folgenden Tabellen enthalten die Prozentsätze derer, die sich in jeder Antwortkategorie verbessert haben. Damit soll die bisherige Zielgruppendefinition für das Video überprüft und präzisiert werden.

2.5.1 Informatische Vorerfahrung

In welchem Maße sich die informatische Vorerfahrung auf den Lernerfolg des Einzelnen auswirkt, soll hier beispielhaft anhand der Vorerfahrung mit Baumstrukturen demonstriert werden. In Tab. 6 ist aufgeführt, bei wie vielen Teilnehmern sich anteilig an der Gesamtanzahl des jeweiligen Kriteriums pro Gruppe der Lösungsweg bei der Bearbeitung des zweiten Färbeproblems verbessert hat. Der Übersichtlichkeit halber wurden in der Tabelle die in Kapitel 2.1 beschriebenen Werte für die Vorerfahrung zusammengefasst, und zwar die Werte 5 und 4 zu „Hoch“ und die Werte 2 und 1 zu „Niedrig“. Der Vorerfahrungswert 3 entspricht dem Attribut „Mittel“.

Die Gruppe „mit Video“ beinhaltet die meisten Teilnehmer mit einem mittleren Maß an Vorerfahrung mit Baumstrukturen. Insgesamt 64,3 % dieser Teilnehmer (neun von 14) zeigte nach Betrachten des Videos eine Verbesserung des Lösungsweges. In der Gruppe „ohne Video“ waren dies nur 25,0 % (drei von zwölf). Auch die anderen Teilnehmer schnitten in der Gruppe „mit Video“ besser ab. 63,4 % der Teilnehmer dieser Gruppe mit einem hohen Maß an Vorerfahrung verbesserten ihren Lösungsweg bei der zweiten Aufgabe, in der Gruppe „ohne Video“ waren dies nur 46,2 %. Mit einem niedrigen Maß an Vorerfahrung konnten sich in der Gruppe „mit Video“ noch 42,9 % der Teilnehmer und in der Gruppe „ohne Video“ nur 25,0 % bei der zweiten Aufgabe verbessern.

In der Gruppe „mit Video“ profitierten nach diesen Ergebnissen Teilnehmer mit hoher und mittlerer Vorerfahrung am meisten, Teilnehmer mit niedriger Vorerfahrung fallen dagegen deutlich ab. In der Gruppe „ohne Video“ war offenbar ein hohes Maß an Vorerfahrung mit Baumstrukturen notwendig, um die Aufforderung, mit Bäumen zu arbeiten, umzusetzen.

Wer hat sich verbessert? Aufteilung nach der Vorerfahrung mit Bäumen			
Gruppe …	**„Hoch“**	**„Mittel“**	**„Niedrig“**
… „mit Video“	63,4 % (7/11)	64,3 % (9/14)	42,9 % (3/7)
… „ohne Video“	46,2 % (6/13)	25,0 % (3/12)	25,0 % (2/8)

Tab. 6: Prozentuale Aufteilung der Lösungsverbesserung nach der Vorerfahrung mit Baumstrukturen

2.5.2 Selbsteinschätzung

Bei der Frage nach der Selbsteinschätzung der eigenen Problemlösestärke hat sich im Gegensatz zu vorangegangenen Befragungen kein einziger Teilnehmer als „schwach“ eingeschätzt. In Tab. 7 ist aufgeführt, bei wie vielen Teilnehmern des jeweiligen Kriteriums jeder Gruppe sich bei der Bearbeitung des zweiten Färbeproblems die Lösung verbessert hat. Die meisten Teilnehmer aus beiden Gruppen finden sich in der Kategorie „eher stark“ und „eher schwach“. Da sich in der Gruppe „mit Video“ keine Teilnehmer befinden, die sich als starke bzw. schwache Problemlöser eingeschätzt haben, können hierzu keine Aussagen gemacht werden.

60,0 % der „eher starken“ und 58,8 % der „eher schwachen“ Teilnehmer aus der Gruppe „mit Video“ verbesserten ihren Lösungsweg nach Betrachten des Videos, nach dem vorliegenden Datenmaterial die „eher starken“

Problemlöser etwas mehr. In der Gruppe „ohne Video“ schnitten die nach eigener Aussage „eher schwachen“ Problemlöser bei der Umsetzung des Lösungshinweises bei der zweiten Aufgabe am besten ab. Das sind mit 43,8 % deutlich mehr als in der Rubrik der starken Problemlöser.

Wer hat sich verbessert? Aufteilung nach Selbsteinschätzung				
Gruppe …	**stark**	**eher stark**	**eher schwach**	**schwach**
… „mit Video“	0,0 % (0/0)	60,0 % (9/15)	58,8 % (10 /17)	0,0 % (0/0)
… „ohne Video“	33,3 % (1/3)	21,4 % (3/14)	43,8 % (7/16)	0,0 % (0/0)

Tab. 7: Prozentuale Aufteilung der Lösungsverbesserung nach der Selbsteinschätzung der Problemlösestärke

2.5.3 Problemlösekategorie

Die Frage, zu welcher Kategorie im Sinne der weiter oben beschriebenen Problemlösetypen ein Problemlöser zählt, ist für die didaktischen Implikationen von erheblicher Bedeutung. Wir wollen nun untersuchen, wie sich diese Problemlösetypen innerhalb derjenigen Teilnehmer verteilen, deren Lösungsweg sich bei der zweiten Aufgabe verbessert hat.

Wer hat sich verbessert? Aufteilung nach PL-Kategorie				
Gruppe …	**A_{alles}**	**B_{wann}**	**C_{wie}**	**D_{nichts}**
… „mit Video“	0,0 % (0/1)	80,0 % (8/10)	56,3 % (9/16)	40,0 % (2/5)
… „ohne Video“	50,0 % (1/2)	44,4 % (4/9)	27,8 % (5/18)	25,0 % (1/4)

Tab. 8: Prozentuale Aufteilung der Lösungsverbesserungen anteilig nach der Problemlösekategorie

In der Gruppe „mit Video“ waren insgesamt zehn Teilnehmer des Problemlösetyps „B_{wann}“ (weiß nicht wann), d. h., nach eigenen Angaben können diese Teilnehmer informatische Problemlösemethoden anwenden, erkennen aber nicht, wann dies konkret möglich ist. 80,0 % von diesen Teilnehmern haben nach Präsentation des Videos einen verbesserten Lösungsweg gezeigt. In der Gruppe „ohne Video“ waren es hingegen nur 44,4 %, also weniger als die Hälfte.

Ein ähnliches Verhältnis findet sich in der Kategorie „C_{wie}“, zu der diejenigen Problemlöser zählen, die nach eigener Aussage informatische

Methoden zwar kennen, aber nicht konkret anwenden können. Unter denen, die das Video gesehen haben, waren 16 Teilnehmer dieser Kategorie, 56,3 % davon haben sich bei der Bearbeitung des zweiten Färbeproblems verbessert. In der Gruppe „ohne Video“ waren 18 Teilnehmer dieser Kategorie, bei 27,8 % war der Lösungsweg bei der zweiten Aufgabe verbessert. Die vorliegenden Daten legen nahe, dass Teilnehmer des Problemlösetyps B_{wann} mit deutlichem Vorsprung am meisten von dem Lehrvideo profitiert haben.

3 Zusammenfassung und Ausblick

Die Ergebnisse der vorliegenden Lernwirksamkeitsanalyse sind mit einem Signifikanzniveau von 5 % mit hoher Wahrscheinlichkeit nicht zufällig und belegen eine Lernwirksamkeit des getesteten Lehrvideos. Zu bedenken ist, dass das Video den Teilnehmern ohne jegliche erläuternde Kommentare präsentiert wurde und diese Teilnehmer direkt im Anschluss daran die Inhalte aus dem Video bei der Bearbeitung eines weiteren Färbeproblems umsetzen sollten. Zu erwarten ist, dass bei entsprechender didaktischer Aufarbeitung in der Zeit unmittelbar vor und nach dem Video die Lernwirksamkeit des Videos noch erheblich gesteigert werden kann. Deshalb soll nun eine Unterrichtseinheit zum Thema Bäume und Rekursion im konkreten Problemlösekontext entworfen werden, bei der am Anfang als einführendes Beispiel das Lehrvideo gezeigt wird und in deren Verlauf die Inhalte des Videos weiter vertieft werden.

Auch der Frage, welcher Typ von Problembearbeiter am ehesten von einem solchen Video profitiert, und damit der Frage nach der Präzisierung der Zielgruppenbeschreibung wurde in dieser Studie nachgegangen. Bei der Interpretation der Analysedaten wurde davon ausgegangen, dass die auf reiner Selbsteinschätzung der Befragten beruhenden Angaben in einem ausreichenden Maße zutreffend sind. Die Gruppe derer, die sich ein hohes und mittleres Maß an Vorerfahrung mit Baumstrukturen bescheinigten, schnitt in dieser Kategorie nach der Betrachtung des Videos erwartungsgemäß am besten ab, diejenigen mit mittlerer Vorerfahrung sogar noch etwas mehr. Das Lehrvideo ist also für Lernende mit wenig oder gar keinen Vorkenntnissen über die gezeigten Inhalte weniger geeignet. In der Kategorie der Selbsteinschätzung der eigenen Problemlösestärke sind die „eher schwachen“ und „eher starken“ Problemlöser nahezu gleich auf, allerdings kann keine Aussage zu rein „starken“ und rein „schwachen“ Problemlösern gemacht werden, da diese in der Gruppe „mit Video“ nicht vorhanden waren.

Es kann hier aber davon ausgegangen werden, dass Lernende mit einem gewissen Maß an Problemlösefähigkeit einen größeren Nutzen aus dem Lehrvideo ziehen. Etwas erwartungswidrig ist hingegen das hohe Maß an Verbesserungen bei dem Problemlösetyp der Kategorie B_{wann} aus der Gruppe „mit Video“, siehe Tab. 8. In dieser Kategorie ist der Prozentsatz der verbesserten Lösungswege nahezu doppelt so hoch wie bei dem gleichen Problemlösetyp in der Gruppe „ohne Video“, obwohl diese Teilnehmer ebenfalls einen Lösungshinweis auf Bäume und Rekursion erhielten. Dies lässt vermuten, dass die Selbsteinschätzung in diesem Fall nicht ausreichend zutrifft, dass also viele Teilnehmer dieser Kategorie eben nicht genau wussten, wie letztlich solche informatischen Problemlösemethoden angewendet werden können und dies erst durch das Video vermittelt bekommen haben. Deshalb sollte bei zukünftigen Analysen dieser Art die Selbsteinschätzungen der Teilnehmer, insbesondere bei der Frage des Problemlösetyps, durch harte Testdaten ersetzt werden.

4 Literatur

[Be09] Berk, R. A.: Multimedia teaching with video clips: TV, movies, YouTube and mtvU in the college classroom. International Journal of Technology in Teaching and Learning, 2009.

[KF94] Kittelberger, R.; Freisleben, I.: Lernen mit Video und Film. Beltz Verlag, Weinheim, 1994.

[KS14] Kujath, B.; Schütze, C.: Evaluation der Lernwirksamkeit eines Lehrvideos zum informatischen Problemlösen. HDI 2014: Gestalten von Übergängen, Universität Potsdam, 2015.

[KS09] Kujath, B.; Schwill, A.: Hochleistern über die Schultern geschaut – Konzeption eines Lehrvideos zur Vermittlung von Problemlösekompetenz. Aus: Informatik als Dialog zwischen Theorie und Anwendung. Festschrift für Volker Claus zum 65. Geburtstag. Springer Vieweg, 2009.

[Ku11a] Kujath, B.: Keine Angst vor Informatikproblemen. Hochleistern über die Schulter geschaut. Ein Lehrvideo zum informatischen Problemlösen. Universitätsverlag Potsdam, 2011.

[Ku11b] Kujath, B.: Kann man durch Abgucken (Beobachten) von Hochleistern lernen? Praxisbericht INFOS 2011, Münster.

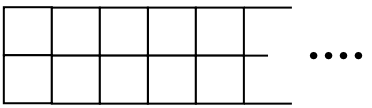
[Pa86] Paivio, A.: Mental representations: A dual coding approach. Oxford University Press, New York, 1986.

[Sc13] Schütze, C.: Analyse der Lernwirksamkeit eines Lehrfilms zum problemlösenden Denken im Informatikunterricht, Masterarbeit an der Universität Potsdam, Institut für Informatik, 2013.

[SF15] Sailer, M.; Figas, P.: Audiovisuelle Bildungsmedien in der Hochschullehre. Eine Experimentalstudie zu zwei Lernvideotypen in der Statistiklehre. http://bildungsforschung.org, Ausgabe 1, 2015.

Anhang

1. Aufgabe (35 Minuten)
Ein längs liegendes Rechteck der Länge n sei aufgeteilt in 2 Reihen und n-Spalten von 1 x 1 -Quadraten:



Jedes dieser 1x1-Quadrate soll nun mit einer der Farben *weiß, grau* und *schwarz* eingefärbt werden, an den Kanten zusammen liegende Quadrate dürfen nicht die gleiche Farbe haben.
Als weitere Bedingung gilt: Die untere Hälfte des längs liegenden Rechtecks sei bereits mit einer beliebigen Farbsequenz vorgefärbt. Wieviele unterschiedliche Möglichkeiten gibt es nun, die noch leere obere Hälfte korrekt zu färben?
Antwort: Diese Anzahl hängt von der Länge n des Rechtecks und der Beschaffenheit der Farbsequenz in der unteren Hälfte ab.

a) Wie muss die untere Farbsequenz für ein beliebiges n beschaffen sein, sodass man oben die maximale Anzahl von Färbemöglichkeiten hat?
b) Wie kann man für jedes n diese maximale Anzahl jeweils errechnen?

(Lösungshinweis: Die Vorgehensweise aus dem Video besteht darin, zunächst eine eingehende Aufgabenanalyse durchzuführen. Das führt zu der Erkenntnis, dass die maximale Anzahl von Färbemöglichkeiten genau dann realisiert werden kann, wenn in der unteren Reihe nur zwei Farben abwechselnd vorkommen. Somit kann dann ein Entscheidungsbaum konstruiert werden, der die Färbemöglichkeiten der oberen Reihe mit wachsender Länge vollständig abbildet. Aus diesem Baum ist dann die Formel für die maximale Anzahl der Färbemöglichkeiten rekursiv herleitbar: $A_{n+1} = A_n + A_{n-1}$)

2. Aufgabe (35 Minuten), nach Präsentation des Videos
Wir codieren positive Ganzzahlen 1, 2, 3, 4, ... mit den Ziffern 1 und 0. Es gelten die üblichen Regeln für Binärzahlen und es gibt keine führenden Nullen. Als zusätzliche Regel soll gelten: In den 0-1-Folgen dürfen keine zwei 1en hintereinander stehen.
Erlaubt ist also: 1, 10, 100, 101, 1000, 1001, ..., 1001010000101000101010001, ... usw. nicht aber beispielsweise 1011.
Diese Ziffernfolgen repräsentieren die natürlichen Zahlen in der Reihenfolge ihres Auftretens, also 1 codiert 1, 10 codiert 2, 100 codiert 3, 101 codiert 4, 1000 codiert 5

usw. Die Sortierreihenfolge für die Ziffernfolgen ist zuerst die Länge, dann die lexikographische Form.

Welche Zahl wird durch die Ziffernfolge 1001000001001000 codiert? Lösen Sie diese Aufgabe bitte mithilfe von Rekursion und Baumstrukturen.

(Lösungshinweis: Auch hier kann mit einer Baumstruktur gearbeitet werden, indem man die Möglichkeiten der ersten Zahlen vollständig modelliert. Anhand der Baumstruktur kann erkannt werden, dass auch diesmal Fibonaccizahlen eine Rolle spielen. Fibonaccizahlen treten immer bei der Hinzunahme einer neuen Stelle auf, also 1=1, 10=2, 100=3, 1000=5, 10000=8 usw. Ähnlich wie beim ersten Färbeproblem lässt sich aus der Baumstruktur die Formel für die gesuchte Zahl ableiten, allerdings in der untersten Ebene nur unter Hinzunahme eines Teilbaums. Wer alles richtig gemacht hat, kommt auf die gesuchte Zahl: **2000.)**

Eine Studie zum kollaborativen Modellieren in der Softwaretechnik-Ausbildung

Anke Dittmar, Gregor Buchholz, Mathias Kühn

Institut für Informatik, Universität Rostock
18059 Rostock
Email: {anke.dittmar|gregor.buchholz|mathias.kuehn}@uni-rostock.de

Zusammenfassung: Die Vermittlung von Modellierungsfähigkeiten in der Softwaretechnik-Ausbildung konzentriert sich meist auf Modellierungskonzepte, Notationen und Entwicklungswerkzeuge. Die Betrachtung der Modellierungsaktivitäten, etwa die Entwicklung und Gegenüberstellung alternativer Modellvorschläge, steht weniger im Vordergrund. Die vorliegende Studie untersucht zwei Formen des kollaborativen Modellierens am Tabletop in Bezug auf ihren Einfluss auf die Modellierungsaktivitäten in kleinen Gruppen. Die Ergebnisse zeigen, dass sowohl selbstorganisierte als auch moderierte Modellierungssitzungen das Entwickeln eines gemeinsamen Modellverständnisses fördern. In moderierten Sitzungen wurden zudem mehr alternative Lösungsideen entwickelt und in stärkerem Maße diskutiert.

1 Einleitung

Für die Softwaretechnik sind Modelle von fundamentaler Bedeutung. Wir arbeiten fast nur mit Modellen und verfolgen dabei unterschiedliche Ziele [L02]. Modelle werden in der Praxis meist von mehreren Beteiligten erstellt und genutzt, beispielsweise um ein gemeinsames Verständnis über Anforderungen an ein Software-System oder dessen Architektur aufzubauen. Die Vermittlung von Modellierungsfähigkeiten bildet deshalb einen zentralen Bestandteil der Softwaretechnik-Ausbildung. Aufgrund ihres derzeitigen Status als De-facto-Standard in der Industrie stehen die Struktur- und Verhaltensdiagramme der UML dabei überwiegend im Mittelpunkt. Übliche Lerninhalte sind folglich grundlegende Kenntnisse wie UML-Syntax, statische

und dynamische Modelle und deren Integration sowie Fertigkeiten im Umgang mit CASE-Tools. Hughes und Parkes [HP04] weisen darauf hin, dass die auf den UML-Grundkenntnissen aufbauenden Fähigkeiten in der Entwicklung, Gegenüberstellung und Diskussion alternativer Lösungen zur Steigerung der Modellqualität oft nur unzureichend ausgeprägt bleiben. Ein Grund mag darin liegen, dass Lehrende oft zu wenig Einblick in den gemeinsamen Umgang mit Modellen besitzen, der in der Regel innerhalb von studentischen Gruppenprojekten erfolgt. Daher werden eher die Ergebnisse als der Vorgang der Modellerstellung reflektiert.

Dieser Beitrag stellt eine empirische Untersuchung zum Einsatz von Modellierungssitzungen in der Softwaretechnik-Ausbildung vor. In einer solchen Sitzung geht es um ein kurzzeitiges gemeinsames Arbeiten an einem festgelegten Modellierungsziel und die Entwicklung eines gemeinsamen Modellverständnisses. Die Gruppenmitglieder sollen lernen, gleichberechtigt Ideen einzubringen, zu diskutieren, weiterzuentwickeln oder zu verwerfen. Die Modellierungssitzungen werden durch einen Tabletop unterstützt. Multitouch Tabletops stellen Nutzern ein gemeinsames Display zur Verfügung und erlauben ihnen gleichzeitiges Interagieren (s. Abb. 1). Damit weisen sie gegenüber „herkömmlichen" Umgebungen wie Papier und Stift oder Whiteboard Vorteile für lokales kollaboratives Arbeiten auf und eine Reihe von Studien untersucht ihren Einsatz für das Lernen in kleinen Gruppen (z. B. [DE11]). Burd et al. konnten zeigen, dass ein Tabletop-Editor für UML-Zustandsdiagramme gegenüber einem vergleichbaren Desktop-Editor ein gleichberechtigteres Arbeiten der Gruppenteilnehmer ermöglichte und die gegenseitige Blockierung (production blocking) verminderte [B+12, B+13].

Im Rahmen der Studie interessierte uns zum einen, inwieweit unterschiedliche Gestaltungen der Modellierungssitzungen mit dem Tabletop die inhaltliche Qualität der Diskussion und die Modellierungsaktivitäten der Gruppe beeinflussen. Die Studie untersucht und vergleicht daher zwei Sitzungsvarianten (selbstorganisiert und moderiert mit Brainwriting-Phasen), in denen Gruppen von je drei Studenten UML-Klassendiagramme erarbeiten. Zum anderen interessierte uns, wie die Teilnehmer ihre Modellierungsaktivitäten und -ergebnisse reflektieren.

Die Ergebnisse zeigen, dass in den moderierten Sitzungen mehr alternative Lösungsideen entwickelt und in stärkerem Maße einander gegenübergestellt und diskutiert wurden als in den nicht moderierten. Die Modellierungssitzungen wurden von den Teilnehmern als neuartig empfunden und die Arbeitsweise am Tabletop als nützliche Ergänzung bisheriger Praktiken angesehen.

2 UML-Modellierung in der Softwaretechnik-Ausbildung[1]

In verfügbarem Material zur modellierungsbezogenen Wissensvermittlung stellt die Behandlung technischen Wissens den weitaus überwiegenden Anteil dar. Dies umfasst die Grundlagen der Objektorientierung ebenso wie strukturelle Hilfen, z. B. in Form von Entwurfsmustern.

Balzert etwa formuliert in [B00] als Ziele der Lehreinheiten „objektorientierten Analyse“ und „objektorientiertes Design“ zwar durchaus die Beherrschung „methodischer Schritte zum Erstellen eines OOA-Modells“. Die Beschreibung der Prozesse [B00, S. 386ff] ist selbst gewissermaßen objektorientiert, gibt also geleitet von Checklisten die Kriterien der Modelle im schrittweisen Verlauf der Vorgehensweise an und richtet damit das Augenmerk weniger auf das Subjekt (den oder die Modellierenden) und *wie* die Erreichung der jeweiligen Teilziele eingeübt und praktiziert werden kann.

Sommerville [S07] und Kleuker [K11] schildern den Ablauf eines objektorientierten Entwurfs als linearen bzw. iterativen Prozess, der anhand von Richtlinien quasi zwangsläufig die erhobenen Anforderungen in *ein* (Klassen-)Modell überführt. Eine Erörterung des Erlernens dieses Prozesses, etwa durch eine ausdrückliche Gegenüberstellung von Alternativen oder ein kooperatives Abwägen unterschiedlicher Gestaltungsansätze, findet hier nicht statt. Ein Anliegen der vorliegenden Studie ist das Beitragen neuer Erkenntnisse um Methoden zum Erwerb dieser Fähigkeiten.

Neben der Vermittlung von technischen und methodischen Fertigkeiten ist auch die Anwendung von Modellierungssoftware (z. B. CASE-Tools) Gegenstand der Lehre. Obgleich das Erlernen und der Einsatz von industriellen Werkzeugen Praxisrelevanz aufweist, erschweren eine im Lehrumfeld nicht benötigte Funktionsvielfalt und schwer vermittelbare Laufzeitsemantik [PS04] sowie nicht zuletzt auch die oft hohen Beschaffungskosten die Erreichung von Ausbildungszielen. In Reaktion darauf hat die Entwicklung maßgeschneiderter Lösungen auf spezifische Lehr-Anforderungen hin eine gewisse Tradition, siehe *Ideogramic UML* [HR02], *Integrated Learning Environment (ILE)* [T$^+$11] und weitere. *ILE* erlaubt sowohl das individuelle als auch das kooperative Bearbeiten von Aufgaben. Dies schafft insbesondere bei kommentierender Begleitung durch einen Lehrenden als günstig zu bewertende Voraussetzungen für eine aktive,

[1] Dieser Abschnitt bezieht sich insbesondere auf Modellierung mit der UML, da eine umfassendere Betrachtung der Softwaretechnik-Ausbildung zur Modellierung den Rahmen des Beitrages sprengen würde.

kritische Erschließung des Lernstoffes UML-Modellierung und hilft überdies dabei, mit der in den Frontal-Unterricht oft schwierig zu integrierenden Auseinandersetzung mit alternativen Entwürfen für nicht-triviale Aufgabenstellungen umzugehen. Allerdings sind die Interaktionsmodalitäten zwischen den Lernenden und mit dem Lehrenden dabei (wie i. A. in Computer Supported Collaborative/Cooperative Learning-Umgebungen mit örtlicher Trennung der Lernenden) zwangsläufig stets nur mittelbar. In den Modellierungssitzungen dieser Studie unterstützt ein kollaborativer Tabletop-Editor den gemeinsamen Erwerb von Modellierungsfähigkeiten in synchroner, räumlich nicht getrennter Arbeitsweise.

3 Studie

Die Studie untersucht UML-Modellierungssitzungen mit dem Tabletop, in denen kleine studentische Gruppen an einem festgelegten Modellierungsziel arbeiten. Ziel des Einsatzes solcher Sitzungen ist es, den Studenten Fähigkeiten im Aufbau eines gemeinsamen Modellverständnisses und in der systematischen Entwicklung und Gegenüberstellung alternativer Modellierungsideen zu vermitteln. Dabei wird angenommen, dass ein gleichberechtigteres Arbeiten der Gruppenmitglieder, wie es durch Tabletop-Anwendungen ermöglicht werden soll, die Erreichung dieser Ziele fördert. Die Studie geht außerdem der Frage nach, ob eine zusätzliche Moderation einer Modellierungssitzung sich weiterhin positiv auf die Erreichung obiger Ausbildungsziele auswirkt. Folgende Forschungsfragen leiteten die Studie:

F1 Wie aktiv beteiligen sich die einzelnen Gruppenmitglieder an den moderierten und selbstorganisierten Modellierungssitzungen?

F2 Welche Arbeitsweisen der Gruppe sind in moderierten und selbstorganisierten Modellierungssitzungen zu beobachten?

F3 Wie werden die Modellierungsaktivitäten und -ergebnisse von den Gruppenmitgliedern reflektiert?

3.1 Teilnehmer und Methode

An der im Sommer 2015 durchgeführten Studie nahmen 30 Studenten der Informatik (5 weiblich, 25 männlich) im Alter von 20 bis 27 Jahren (Durchschnitt: 22 Jahre) teil. Keiner der Teilnehmer hatte Erfahrung mit der Arbeit am Tabletop; alle besaßen UML-Grundkenntnisse und hatten zumin-

dest aus Studentenprojekten praktische Erfahrungen in objektorientierter Modellierung. Insgesamt wurden 10 Versuche mit je 3 Teilnehmern durchgeführt, wobei 5 Gruppen in moderierten Sitzungen arbeiteten (s. unten) und 5 Gruppen selbstorganisiert (*between-subjects design*). Zwei Pilotversuche dienten zur Verbesserung der Versuchsdurchführung.

Die Gruppen nutzten einen kollaborativen Editor für Klassendiagramme auf einem Multi-touch Tabletop SUR40 (s. Abb. 1) [P14]. Vor jedem Versuch gaben die Teilnehmer ihre Einverständniserklärung zur Teilnahme an der Studie. Die Gruppen wurden kurz in die Bedienung des Editors eingeführt und konnten sich mit der gemeinsamen Nutzung des Editors vertraut machen. Danach führte der Versuchsleiter in die Aufgabenstellung ein und jedes Gruppenmitglied erhielt einen Zettel mit einer detaillierten Problembeschreibung. In moderierten Sitzungen wurde außerdem der Moderator vorgestellt. In der Aufgabe ging es um die Erstellung eines OOA-Modells. Die Komplexität wurde so gewählt, dass verschiedene Aspekte des gegebenen Problembereiches betrachtet werden mussten, aber eine Bearbeitung innerhalb einer Sitzung möglich war. Die Gruppen bestimmten das Ende ihrer Modellierungssitzung selbst. Danach füllte jeder Teilnehmer einen Fragebogen mit geschlossenen und offenen Fragen zur Person und zur Modellierungssitzung (z. B. zur Zufriedenheit mit dem Modell und zur Zusammenarbeit) aus. Der Versuch wurde durch ein semi-strukturiertes Gruppeninterview abgeschlossen.

In den **moderierten Modellierungssitzungen** übernahm ein Mitglied des Forschungsteams die Rolle des Moderators, um die systematische Betrachtung von Teilthemen und Modellierungsideen zu fördern. Dazu wurden in der Moderation Techniken aus dem Brainwriting [PY00] verwendet. Zunächst bat der Moderator die Gruppe, relevante *Themenbereiche* in der gegebenen Aufgabenstellung zu identifizieren. Für jeden dieser Bereiche eröffnete er eine Brainwriting-Phase, in der separat Modellierungsvorschläge erarbeitet wurden (s. Abb. 1). Anschließend bat der Moderator die Teilnehmer, selbstständig ihre Ideen vorzustellen und zu diskutieren, um schließlich eine gemeinsame Teillösung für den aktuell betrachteten Themenbereich in das Gesamtmodell (Mitte des Tabletops) zu integrieren.

3.2 Datenerhebung und Analyse

Die Audioaufzeichnungen der Gruppeninterviews wurden transkribiert und die Fragebögen digital erfasst. Die Modellierungssitzungen wurden mit einer

digitalen Videokamera und die Interaktionen mit dem Tabletop mit CamStudio™ [2] aufgezeichnet. Für die parallele Analyse beider Videos wurde das ELAN-Tool[3] verwendet [W+06]. Nach der Transkription der sprachlichen Äußerungen erfolgte eine schichtenweise Codierung:

- Level 1: Erfassen der Aktivitäten der Teilnehmer P1, P2 und P3 in den Modalitäten „sprachliche Äußerung“ und „Interaktion mit Tabletop“.
- Level 2: Erfassen inhaltlicher Bezüge: Identifikation von Themenbereichen und ihren Teilthemen sowie von koordinierenden Aktivitäten.
- Level 3: Identifikation von Modellideen (Optionen) für Teilthemen.

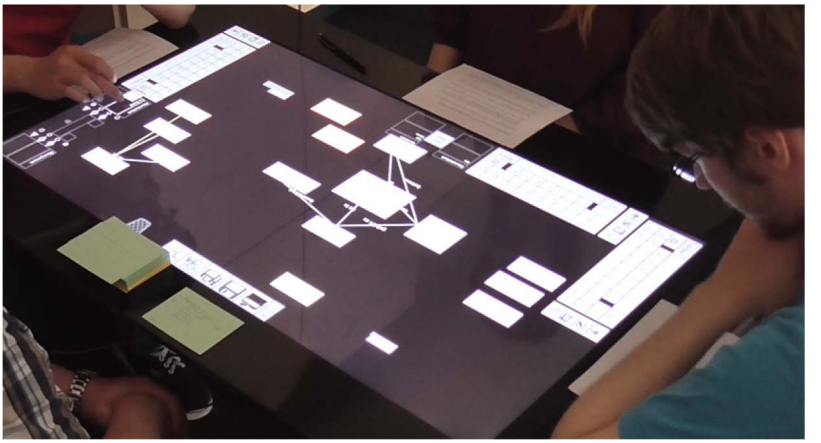


Abb. 1: Brainwriting-Phase in einer moderierten Modellierungssitzung

Moderierte Gruppen bestimmten die Themenbereiche am Anfang explizit. Teilthemen kristallisierten sich meist während der Vorstellung der Modellvorschläge nach der Brainwriting-Phase zu einem Bereich heraus. Hinweise für Themenbereiche in nicht moderierten Gruppen ergaben sich aus den sprachlichen Äußerungen (z. B. „Machst du Bereich XYZ?“), aus der Verwendung von Ordnern, die das Ausblenden und Benennen von Teildiagrammen ermöglichten, und aus Aussagen in Fragebögen und Interviews.

Die Codierungsergebnisse wurden in Excel ausgewertet und mit einem eigenen Tool visualisiert. Die folgende Darstellung einiger ausgewählter Resultate orientiert sich an den eingangs erwähnten Forschungsfragen.

[2] http://camstudio.org/.
[3] https://tla.mpi.nl/tools/tla-tools/elan/.

3.3 Resultate

Die Analyse der Teilnehmeraktivität entsprechend **Fragestellung F1** erfolgte auf Codierungslevel 1. Die Visualisierung des Datenmaterials beider Formen der Modellierungssitzungen zeigt eine gleiche Intensität der Beteiligung an der Gruppenarbeit zwischen den Teilnehmern (z. B. Abb. 2, 3).

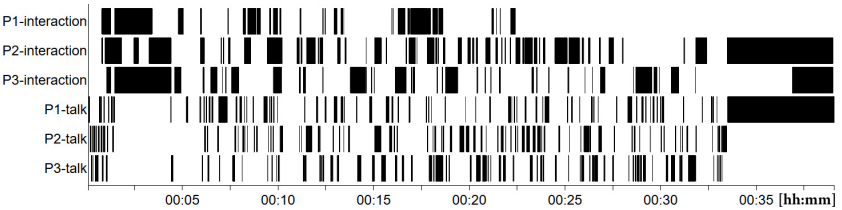


Abb. 2: Teilnehmeraktivität in V4 (nicht moderiert)

Am Beispiel der Gruppe V4 soll im Folgenden gezeigt werden, dass die Visualisierungen auch Hinweise auf spezifisches Verhalten einzelner Gruppenmitglieder geben. In Abb. 2 ist die Aktivitätsverteilung der Teilnehmer P1, P2 und P3 dargestellt: Eine rasche Folge sprachlicher Äußerungen über die ersten 1,5–2 Minuten geht für weitere 3 Minuten in paralleles, weitestgehend unkommentiertes Arbeiten über. Bis Ablauf der ersten 22 Minuten ist das gleichmäßige Einbringen der Teilnehmer in Äußerungen und Interaktionen zu erkennen. Ab der 23. Minute ist hier ein Ausbleiben weiterer Auftreten der Modalität „Interaktion mit Tabletop“ bei P1 festzustellen: die Beteiligung an der Gruppenarbeit findet seitens P1 in sprachlichen Äußerungen jedoch weiterhin statt. Im weiteren Verlauf zeigt sich zudem eine Reduzierung der Parallelität in der Interaktion von P2 und P3 – wurde zunächst noch gelegentlich gleichzeitig am Tisch gearbeitet, kommt es zunehmend zu einer Linearisierung der Aktivitäten.

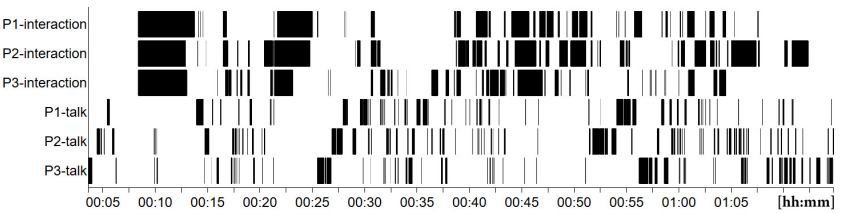


Abb. 3: Teilnehmeraktivität in V2 (moderiert)

Die Aktivitätsvisualisierungen der moderierten Sitzungen zeigen im Gesamtbild eine ebenfalls gleich starke Einbindung aller Teilnehmer in beiden hier

unterschiedenen Modalitäten. Zusätzlich ist hier der Einfluss der Moderation sichtbar, wie in Abb. 3 am Beispiel der Gruppe V2 gezeigt: Die Brainwritingphasen (8.–14., 22.–25. und 43.–52. Minute) sind durch hohe Interaktionsintensität und nur gelegentliche Äußerungen gekennzeichnet. Nach Abschluss dieser Phasen ist in unterschiedlicher Reihenfolge (00:14 P1–P2–P3, 00:23 P3–P2–P1, 00:52 P2–P1–P3) das (verbale) Vorstellen der individuellen Lösungsansätze erkennbar.

		Dauer (min)	Themen-bereiche	Teil-themen	Optionen	Optionen/ Teilthema	Frage 1	Frage 2
moderiert	v1	80	4	15	30	2,0	1/ 2/ 1	1/ 1/ 1
	v2	75	4	10	21	2,1	2/ 2/ 2	1/ 1/ 1
	v3	80	5	12	32	2,7	2/ 1/ 4	1/ 1/ 1
	v6	75	4	14	32	2,3	1/ 2/ 2	1/ 1/ 1
	v10	80	4	18	47	2,6	4/ 2/ 2	4/ 3/ 1
nicht moderiert	v4	40	4	16	22	1,4	1/ 1/ 1	2/ 2/ 1
	v5	65	4	11	20	1,8	1/ 1/ 1	1/ 1/ 3
	v7	50	4	16	35	2,2	2/ 1/ 1	2/ 2/ 2
	v8	40	6	23	27	1,2	2/ 1/ 1	1/ 2/ 1
	v9	30	0	12	23	1,9	2/ 1/ 1	1/ 3/ 2.5

Frage 1: Wie stark wurden Ihre Vorschläge und die anderer Gruppenmitglieder in der Diskussion beachtet und bearbeitet?

1 (sehr stark) bis 5 (gar nicht)

Frage 2: Wie gut konnten Sie die Modellierungs-aktivitäten der gesamten Gruppe verfolgen?

1 (sehr gut) bis 5 (gar nicht)

Videoanalyse — aus Fragebögen der 3 Gruppenmitglieder

Abb. 4: Einige Ergebnisse aus der Video- und Fragebogenanalyse

Fragestellung F2 bezog sich auf die Arbeitsweisen in beiden Sitzungsformen. Die Auswertung des Datenmaterials (Level 2 und 3) ergab, dass in den moderierten Sitzungen durch alle Gruppenmitglieder mehr Optionen entwickelt wurden als in den nicht moderierten, wobei sich die Anzahl der besprochenen Teilthemen nicht wesentlich unterschied (s. Abb. 4). Die Ursachen für die längere Dauer moderierter Sitzungen ist nicht nur in den Brainwriting-Phasen und der Moderation zu sehen, sondern auch in den sprachlichen Äußerungen der Teilnehmer. Während die Anzahl der Äußerungen bezogen auf Themenbereiche und Teilthemen zwischen beiden Sitzungsformen nur wenig differiert (moderiert: 236, nicht moderiert: 217), betrug deren durchschnittliche Länge in moderierten Gruppen 8 Sekunden gegenüber 5 in den nicht moderierten Gruppen. In letzteren gab es jedoch mehr koordinierende Äußerungen. Beispielsweise übernahmen Teilnehmer die Rolle, wiederholt aus dem Aufgabentext vorzulesen, um die Gruppe zur Überprüfung des bisher entwickelten Modells aufzufordern.

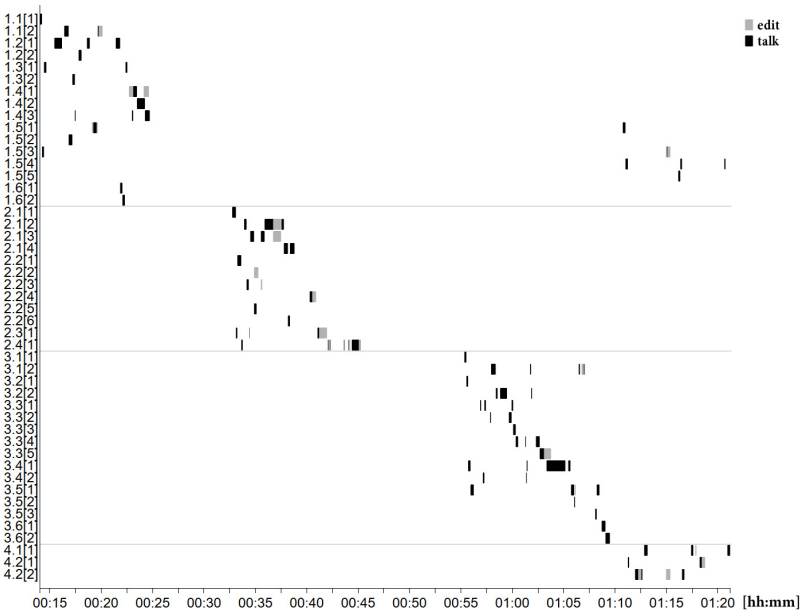


Abb. 5: Aktivitäten zu den Optionen in V12 (moderiert), ohne Brainwriting

In den moderierten Sitzungen wurden die entwickelten Optionen auch in stärkerem Maße von der gesamten Gruppe gegenübergestellt. Abb. 5 und 6 zeigen exemplarisch die Zuordnung der Gruppenaktivitäten (sprachliche Äußerungen und Interaktionen mit dem Tabletop, teilweise überlappend) zu den Optionen (Notation: *Themenbereich.Teilthema*[*Option*]). Es wird deutlich, dass V12 die Optionen zu einzelnen Themenbereichen relativ geschlossen besprochen hat. Die Gruppenmitglieder von V5 „springen" dagegen viel stärker zwischen Themenbereichen und bearbeiten sie besonders am Anfang parallel, was teilweise zu Schwierigkeiten im gemeinsamen Verständnis bzw. in der Akzeptanz (P3: *„das würde mich so stören an deiner Modellierung da hinten"*) und zu wiederholten Besprechungen der Optionen führte. Die Diagramme der anderen moderierten bzw. nicht moderierten Gruppen weisen eine ähnliche Struktur auf wie die in Abb. 5 bzw. 6 dargestellten.

In der **Fragestellung F3** ging es um die subjektive Widerspiegelung der erlebten Modellierungssitzung. Die Interviews zeigten, dass diese von der Mehrzahl der Teilnehmer als neuartig empfunden und die Arbeitsweise am Tabletop als nützliche Ergänzung bisheriger kollaborativer Praktiken angesehen wurden (*„wenn man das z. B. mit Stift und Zettel gemacht hätte… man hätt' halt nicht die ganzen Klassen durch die Gegend schieben können"*

(V1,P2); *„Arbeiten auf Augenhöhe“* (V4,P3); *„Ich fand das auch ’ne angenehme Arbeitsweise… hab’ uns für relativ produktiv empfunden“* (V5,P3)).

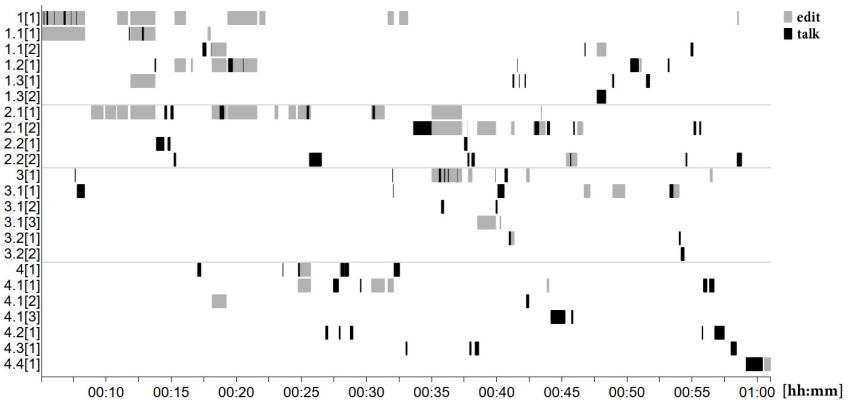


Abb. 6: Aktivitäten zu den Optionen in V5 (nicht moderiert)

Gruppen mit Moderation schätzten diese als hilfreich ein *(„Das war vorher eher so, dass gleich alle an einem gearbeitet haben, irgendwie. Und man wollte halt das Beste gleich hinkriegen. Aber so war’s eigentlich schon besser. Weil da hat jeder erst mal für sich gemacht… und so konnte man dann halt aus jeder Version quasi die Vorteile übernehmen“* (V1,P3)). In allen 5 Gruppen wurden selten Modellvorschläge aus einer Brainwriting-Phase vollständig gelöscht. Stattdessen fand ein „Mischen“ und Weiterentwickeln der verschiedenen Vorschläge statt (*„Ich fand’s auch positiv, dass man nicht alles wegschmeißen musste von einem, sondern dass man halt gemischt hat.“* (V3,P3)). Die Gruppen ohne Moderation berichteten oft von Rollen, die die einzelnen Teilnehmer zeitweise eingenommen haben (*„dass wir am Anfang recht schnell so uns grob aufgeteilt haben… und da hat P1 ja dann hauptsächlich den größeren Überblick behalten“* (V5,P3)).

In den Fragebögen überdeckten sich die freien Äußerungen zu guten und ungeeigneten Modellierungsideen oft bei den Gruppenmitgliedern. Abb. 4 enthält Antworten auf die Fragen zur Beachtung der Vorschläge und der Verfolgung der Modellierungsaktivitäten.

4 Diskussion und Fazit

Die Analyse der Modellierungssitzungen und Befragungen deutet darauf hin, dass in allen Gruppen ein gemeinsames Modellverständnis entwickelt wurde, wobei in den nicht moderierten Gruppen infolge von Aufgabenverteilung und paralleler Arbeit wiederholende Besprechungen von Teilthemen und entsprechenden Optionen notwendig waren. Die Ergebnisse bestätigen auch Erkenntnisse früherer Studien zur Benutzung von Tabletops, die ein gleichberechtigtes Arbeiten in der Gruppe feststellen [B+12,B+13].

Obwohl es in den nicht moderierten Sitzungen zu Rollenbildungen kam und Gruppenmitglieder zumindest zeitweise eine moderierende Funktion einnahmen, führten die moderierten Sitzungen insgesamt zu einer systematischeren Entwicklung und Gegenüberstellung von Optionen durch die Teilnehmer. Die Interviews und das Verhalten in den moderierten Sitzungen legen nahe, dass die Teilnehmer die Moderation sinnvoll fanden. Interessant ist in diesem Zusammenhang, dass die Teilnehmer der nicht moderierten Gruppen die Beachtung der Modellvorschläge durch die Gruppe positiver bewerteten als die der moderierten (s. Abb. 4). Eine Erklärung könnte sein, dass in moderierten Gruppen Lösungen aus dem Brainwriting explizit gelöscht werden mussten und dies von den Erstellern als „Nichtbeachtung" interpretiert wurde. In nicht moderierten Sitzungen wurden einige Optionen nur vorgeschlagen, aber weder editiert noch weiter diskutiert. Die Teilnehmer haben das offensichtlich nicht so wahrgenommen.

Die Studie beschränkt sich auf die kollaborative Erstellung von Klassendiagrammen. Zukünftige Forschung sollte auch die Kombination von statischer und dynamischer Modellierung betrachten. Die gewählte Form der Moderation nutzte Ideen aus dem Brainstorming [PY00]. Die Probleme und Lösungen der Softwaremodellierung sind jedoch wesentlich komplexer als typische Brainstorming-Probleme. Für die durch Themenbereiche geleitete Moderation spricht, dass sich nur in einer nicht moderierten Sitzung keine Themenbereiche herausbildeten, die die Organisation der Gruppenarbeit lenkten. Hier wurde ausschließlich auf der Ebene von Klassen und Assoziationen diskutiert und die entstandenen Modellideen wurden wenig reflektiert (s. Abb. 4, V9). In der zukünftigen Arbeit sollten alternative Formen der Moderation entwickelt und erprobt werden. Weiterhin ist eine bessere technische Unterstützung von Brainwriting-Phasen denkbar, etwa durch eine kombinierte Verwendung von Tablets und Tabletop, wie wir sie gegenwärtig untersuchen.

Der Einsatz von kollaborativen UML-Modellierungssitzungen mit Tabletop-Editoren erfordert einen hohen Einsatz an technischen und personellen Ressourcen. Doch zusammenfassend lässt sich sagen, dass die Ergebnisse der Studie zeigen, dass bereits das einmalige Erleben einer solchen Sitzung zu einer positiven Lernerfahrung führen kann.

Danksagung

Wir bedanken uns bei Björn Pullwer, Peter Forbrig, Anne Gutschmidt und bei den Teilnehmern an der Studie für ihre Unterstützung dieser Arbeit.

Literatur

[B00] H. Balzert: Lehrbuch der Softwaretechnik Bd. 1, 2. Aufl. 2000.

[B+12] M. Basheri, L. Burd, N. Baghaei: A Multi-touch Interface for Enhancing Collaborative UML Diagramming. OzCHI'12, ACM (2012), 30–33.

[B+13] L. Burd, N. Baghaei, M. Basheri, M. Munro: Collaborative Learning Skills in Multi-touch Tables for UML Software Design. Int. Journal of Advanced Computer Science and Applications IJACSA 4(3), 2013, 60–66.

[DE11] P. Dillenbourg, M. Evans: Interactive tabletops in education. Int. Journal of Computer-Supported Collaborative Learning 6(4), 2011, 491–514.

[HP04] J. Hughes, S. Parkes: Does Requiring Students to Produce Alternative Solutions Promote a High Quality of Software Design? H. R. Arabnia, H. Reza (Hrsg.): SERP '04, 2004, 431–437.

[HR02] K. M. Hansen, A. V. Ratzer: Tool support for collaborative teaching and learning of object-oriented modeling. ITiCSE 2002, ACM, 146–150.

[K11] S. Kleuker: Grundkurs Software-Engineering mit UML: der pragmatische Weg zu erfolgreichen Softwareprojekten. Wiesbaden, 2011.

[L02] J. Ludewig: Modelle im Software Engineering – eine Einführung und Kritik. M. Glinz, G. Müller-Luschnat (Hrsg.): Modellierung 2002, 7–22.

[P14] B. Pullwer: Entwicklung eines kollaborativen Klasseneditors für Multi-Touch. Masterarbeit, Univ. Rostock, 2014.

[PS04] J. Pleumann, J. Schröder: Didaktische Modellierungswerkzeuge für die Präsenzlehre in der Softwaretechnik. INFORMATIK 2004 – Informatik verbindet, Bd. 1, Beiträge der 34. Jahrestagung der GI, 2004, 409–413.

[PY00] P. B. Paulus, H. Yang: Idea Generation in Groups: A Basis for Creativity in Organizations. Organizational Behavior and Human Decision Processes 82(1):76–87, 2000.

[S07] I. Sommerville: Software Engineering, Pearson Studium, 8. Aufl., 2007.

[T+11] S. Trapp, E. Ramollari, M. Heintz, S. Weber, D. Dranidis, J. Börstler: Collaborative learning of UML and SysML. International Journal of Engineering Pedagogy: iJEP 1 (2011), Nr. 2, 6–12.

[W+06] P. Wittenburg, H. Brugman, A. Russel, A. Klassmann, H. Sloetjes: ELAN: a professional framework for multimodality research. Proc. of the 5th Conf. on Language Resources and Evaluation, 2006, 1556–1559.

.

Full Papers
zum Thema „Werkzeuge“

Werkzeugunterstützung bei der Vermittlung der Grundlagen wissenschaftlichen Schreibens

Oliver Zscheyge, Karsten Weicker

Fakultät Informatik, Mathematik und Naturwissenschaften
Hochschule für Technik, Wirtschaft und Kultur Leipzig
Leipzig
Web: portal.imn.htwk-leipzig.de/fakultaet/personen/weicker
Email: karsten.weicker@htwk-leipzig.de

Zusammenfassung: Der Unterricht großer Studierendengruppen im wissenschaftlichen Schreiben birgt vielfältige organisatorische Herausforderungen und eine zeitintensive Betreuung durch die Dozenten. Diese Arbeit stellt ein Lehrkonzept mit Peer-Reviews vor, in dem das Feedback der Peers durch eine automatisierte Analyse ergänzt wird. Die Software Confopy liefert metrik- und strukturbasierte Hinweise für die Verbesserung des wissenschaftlichen Schreibstils. Der Nutzen von Confopy wird an 47 studentischen Arbeiten in Draft- und Final-Version illustriert.

1 Motivation

Das Studium soll die Absolventen dazu befähigen, wissenschaftlich arbeiten zu können. Neben dem zwingend notwendigen fachlichen Wissen nimmt dabei die Fähigkeit, seine Ergebnisse adäquat und entsprechend der fachlichen Standards zu dokumentieren und zu publizieren, eine ebenso bedeutende Rolle ein. Wird dies im Studium erst im Rahmen von Studien-, Bachelor- oder Diplomarbeiten thematisiert, resultiert oft ein hoher individueller Betreuungsaufwand, wobei sich das Feedback vom einen zum nächsten Kandidaten gleicht. Daher wurde an der HTWK Leipzig ein Modul „Grundlagen wissenschaftlichen Arbeitens“ eingerichtet, in welchem zu einem frühen Zeitpunkt die Grundlagen und Gepflogenheiten präsentiert und die Nota-

tionen und Vorgehensweisen an kürzeren Texten geübt werden. Wie in allen Grundlagenveranstaltungen bedeutet dies einen hohen Betreuungsaufwand seitens der Lehrenden. Daher ist die Unterstützung durch den Einsatz von Werkzeugen sinnvoll.

Nach einem kurzen Abriss der Literatur in Abschnitt 2, behandelt Abschnitt 3 die Form der Lehrveranstaltung. Anschließend wird detaillierter die mögliche Unterstützung durch die Analysesoftware Confopy in Abschnitt 4 sowie konkrete Ergebnisse anhand der studentischen Texte der letzten beiden Jahre in Abschnitt 5 präsentiert.

2 Verwandte Arbeiten

In der Literatur gibt es zahlreiche Arbeiten, die sich mit der automatisierten Bewertung der sprachlichen und argumentativen Qualitäten von Schülern und Bachelorstudenten auseinandersetzen. [VNC03] liefern einen Überblick über die verschiedenen Ansätze der automatisierten Bewertung. [BB05] vergleicht in einer umfassenden Analyse die Techniken mit den Bewertungen durch Experten. Ein konkretes System, e-rater V.2, wird in [AB06] vorgestellt. Die Kriterien und Metriken dieser Arbeiten wurden weitestgehend im hier präsentierten System Confopy berücksichtigt, auch wenn sich die vorgenannten Arbeiten mehr auf allgemeine Essays als auf wissenschaftliche Schriften konzentrieren.

Unabhängig von der Automatisierbarkeit wird in [Tim11] ein allgemeiner Katalog an Kriterien erarbeitet, mit dem die Kompetenz des wissenschaftlichen Schreibens beurteilt werden kann.

Für eine tiefere inhaltliche Analyse, z. B. ob stichhaltig argumentiert wird, ist eine genauere semantische Analyse von Texten erforderlich. Anhaltspunkte können hier Forschungsergebnisse zur Textzusammenfassung und -extraktion liefern. Ein frühes Beispiel ist die Arbeit [TM02]. Diese Ansätze sind sehr aufwändig und werden aktuell in Confopy nicht benutzt.

3 Modul „Grundlagen wissenschaftlichen Arbeitens“

Das Modul „Grundlagen wissenschaftlichen Arbeitens“ findet im Bachelorstudium Informatik im 2. Fachsemester und im Bachelorstudium Medieninformatik im 3. Fachsemester statt. Dabei orientiert sich der Ablauf des praktischen Anteils an dem einer wissenschaftlichen Tagung: Die Studieren-

den verfassen ein Schriftstück und reichen dieses ein. Es folgt eine Review-Phase, in der die Studierenden drei Arbeiten von Kommilitonen begutachten und die Gutachten entsprechend abgeben. Jeder Autor erhält das Feedback der Peers und überarbeitet seinen Text, den er dann in der endgültigen Fassung abgibt. Anschließend halten die Studierenden einen Vortrag zu ihrem Thema. Diese Vorgehensweise ist verbreitet [Gui01,Wei06]. Für die Einführung in das wissenschaftliche Arbeiten finden begleitend Vorlesungen zu verschiedenen Themen statt. Abb. 1 zeigt den zeitlichen Ablauf des Moduls.

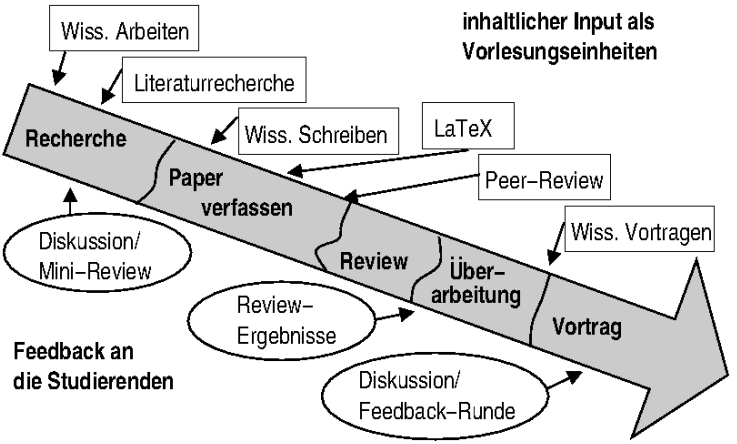


Abb. 1: Grundsätzlicher Ablauf der Lehrveranstaltung (nach [Wei06])

Die Themen umfassen einfache, für das erste Studienjahr geeignete Teilbereiche der Informatik, die zufällig verteilt werden. Die Benotung des Moduls erfolgt gemäß eines Punkteschemas, in welchem 60 Punkte erreichbar sind. Ab 24 Punkten ist der Kurs bestanden. Die Punkte werden für die folgenden Aspekte vergeben:

- Vortrag: 15 Punkte
- Ausarbeitung (Final-Version): 15 Punkte
- Überarbeitung der eigenen Ausarbeitung: 5 Punkte
- Gutachten zu den anderen Arbeiten: 15 Punkte
- aktive Beteiligung in den Seminaren: 10 Punkte

Die Lehrveranstaltung wird seit dem Sommersemester 2014 durchgeführt.

Die Studierenden bewerten insbesondere den praktischen Teil der Lehrveranstaltung als arbeitsintensiv. Durch das Konzept der Peer-Bewertung bleibt der Betreuungs- und Benotungsaufwand gering, da ein großer Teil des Feedbacks im Lernprozess zwischen den Studierenden stattfindet. Allerdings stellt dies auch eine große Schwachstelle dar, da von einigen studentischen Gutachtern wichtige Kriterien des wissenschaftlichen Schreibens ignoriert bzw. pauschale Bewertungen wie „Alles toll! Interessantes Thema!“ abgegeben werden.

4 Automatische Analyse wissenschaftlicher Texte

Um sicherzustellen, dass jeder Autor ein hilfreiches Feedback erhält, sollten die Gutachten um eine zusätzliche „unbestechliche“ Meinung ergänzt werden. Aufgrund der großen Menge der Arbeiten und der sehr kurzen Zeitspanne, die für die Begutachtung zur Verfügung steht, kann dies nicht durch die Lehrkräfte realisiert werden. Daher war die Idee entstanden, durch das neu entwickelte Werkzeug Confopy ein automatisierbares Feedback mit sinnvollen Hinweisen zur Wissenschaftlichkeit des Schreibstils zu liefern. Perspektivisch sollen Fachspezifika stärker berücksichtigt werden – insbesondere die mathematisch-formale Sprache der Informatik im Gegensatz zu Normen anderer Domänen wie Betriebswirtschaft oder Rechtswissenschaften. Der hier präsentierte Ansatz wurde erstmals in [Zsc15] vorgestellt.

4.1 Messbare Sprachmerkmale

Ein Text wird unter Zuhilfenahme des TIGER-Corpus [Bra04] als deutsche Sprachreferenz in Wort- und Satztoken zerlegt. Über das POS-Tagging (*Part-Of-Speech*) werden jedem Wort seine Art (Artikel, Substantiv, Verb etc.) sowie weitere Attribute zugeordnet. Auf dieser Basis können nun verschiedene Sprachmerkmale eines Textes bestimmt werden. Derzeit werden die folgenden allgemeinen Sprachmetriken erhoben:

- die *durchschnittliche Wortlänge* wordlength,
- die *Rechtschreibfehler* spellcheck als Anteil der Wörter im Text, die nicht als deutsche Wörter erkannt werden,
- der *relative Umfang des Wortschatzes* lexicon als das Verhältnis der im Text vorkommenden Wörter (in ihrer Grundform) zur Länge des Textes

als Anzahl aller Wörter (z. B. würde „Buch der Bücher“ lexicon=2/3 aufweisen, da „Buch“ und „Bücher“ dieselbe Grundform besitzen) und
- die *Lesbarkeit* ari (*automated readability index*) nach [SS67] als 1:9-gewichtete Summe der durchschnittlichen Satz- und Wortlänge – ari=50 entspricht dem Niveau eines Viertklässlers, ari=70 ist Zeitungssprache.

Darüber hinaus werden die folgenden Metriken für Sprachmerkmale wissenschaftlicher Texte erhoben:

- drei Metriken bezüglich des *Schreibstils* als relative Häufigkeit verschiedener Wörter: persönlicher Stil („ich“, „wir“, „Sie“) personalstyle, unpersönlicher Stil („man“) impersonalstyle und Passiv-Konstrukte („wird“, „werden“) – der Wert sollte jeweils möglichst niedrig sein –,
- die *passende Zeitform* simplepres als Anteil der Verben, die im Präsens benutzt werden – der Wert sollte möglichst hoch sein –,
- *verstärkende bzw. unpräzise Adverbien* („leicht“, „sehr“, „viel“) adverbmodifier relativ zur Gesamtwortzahl – der Wert sollte möglichst niedrig sein –,
- *inaktive Verben* („gehören“, „liegen“, „beinhalten“) deadverbs relativ zur Anzahl der Sätze – je niedriger, desto besser –,
- *Füllwörter* („also“, „offenbar“, „bereits“) fillers relativ zur Gesamtwortzahl – je niedriger, desto besser –,
- *Beispielgehalt* examplecount als absolutes Vorkommen von Wörtern wie („Beispiel“, „beispielsweise“, „z. B.“) – je höher, desto besser –
- die *Variation der Satzlänge* sentlengthvar als eine der wenigen Möglichkeiten, wissenschaftliche Texte abwechslungsreich zu gestalten (neben aktiven, anschaulichen Sätzen mit hohem Beispielgehalt), gemessen als durchschnittliche Differenz der Länge direkt aufeinander folgender Sätze im Text – je höher, desto besser.

Für all diese Merkmale wurden für Texte in der Größenordnung, wie wir sie in unserer Lehrveranstaltung zu erwarten haben, typische Wertebereiche ermittelt. Dies ermöglicht, für beliebige Texte die Metriken zu ermitteln und die Autoren auf Abweichungen hinzuweisen.

4.2 Strukturbezogene Regeln

Neben der rein metrikbasierten Auswertung eines Textes erlaubt die vorliegende Tokenisierung auch Aussagen über die Struktur eines Textes. In Confopy wurden zunächst beispielhaft drei Regeln formuliert:

- Kapitel sollen eine Einleitung haben und nicht direkt mit einem Unterkapitel starten.

  is_chapter(x) → *has_introduction(x)*
- Weist ein Abschnitt Unterabschnitte auf, so sollten dies mindestens zwei sein.

  (*is_chapter(x)* or *is_section(x)*) → |*subsections(x)*| <> *1*
- Gleitobjekte sind beschriftet und werden im Text vor der Abbildung referenziert.

  is_float(x) → *(was_referenced_before(x) and has_caption(x))*

Hier sind noch weitere Regeln denkbar. Wenn man sich auf ein zahlbasiertes Zitierschema beschränkt, könnten auch Literaturquellen erkannt und deren Referenz im Text geprüft werden. Auch sind Aussagen über die in der Informatik oft unbeliebten Fußnoten denkbar – so könnten Fußnoten nur erlaubt werden, wenn sie eine URL enthalten, um beispielsweise auf Software-Werkzeuge hinzuweisen. In eher formalen Arbeiten können Definitionen erkannt und die Verwendung der definierten Begriffe geprüft werden. Auch Aussagen über die Verwendung von mathematischen Notationen und Formeln sind denkbar, bedürfen allerdings mehr Arbeit bzgl. der Tokenisierung über weitere Heuristiken.

4.3 Confopy

Die Software Confopy setzt alle in Abschnitt 4.2 aufgeführten Metriken und Regeln um. Sie kann einen Text in einem XML-Beschreibungsformat einlesen und analysieren. Es steht ebenfalls ein Prototyp zur Verfügung, der aus PDF-Dokumenten die XML-Beschreibung extrahiert – allerdings mit Einschränkungen, die dem Format PDF als Seitenlayoutbeschreibung und nicht als syntax- und semantikerhaltende Beschreibung geschuldet sind. Viele gängige insbesondere mit LaTeX erzeugte Dokumente werden im ein- und zweispaltigen Layout erkannt. Textblöcke wie Paragraphen, Überschriften, Fußnoten und Gleitobjektbeschriftungen sowie der Textfluss werden über Heuristiken ermittelt.

Confopy ist mit Python realisiert und es werden die Bibliotheken pdfminer, lxml, nltk (Natural Language Toolkit), pattern und PyEnchant benutzt (vgl. https://pypi.python.org/pypi).

Abb. 2 zeigt beispielhaft die Ausgabe von Confopy auf einer studentischen Arbeit, die dem Studierenden als Feedback neben Peer-Gutachten zur Verfügung gestellt werden kann.

Metriken
 * adverbmodifier 0.03 (erwartet: max. 0.04)
    OK!
 * ari 62.66 (erwartet: max. 71.96)
    OK!
 * deadverbs 0.09 (erwartet: max. 0.06)
    Versuche weniger tote Verben wie gehören, liegen, befinden, beinhalten, geben, bewirken etc. zu verwenden!
 * examplecount 5.0 (erwartet: min. 1.00)
    OK!
 * fillers 0.04 (erwartet: max. 0.03)
    Zu viele Füllwörter!
 * impersonalstyle 0.15 (erwartet: max. 0.05)
    Zu viele Sätze mit ‚man'.
 * lexicon 0.44 (erwartet: zw. 0.46 und 0.56)
    Zu geringer Wortschatz
 * passiveconstructs 0.3 (erwartet: max. 0.37)
    OK!
 * personalstyle 0.09 (erwartet: max. 0.06)
    Zu persönlicher Schreibstil! Sätze mit ‚ich', ‚wir', ‚sie' umschreiben!
 * sentlength 18.44 (erwartet: max. 17.38)
    Zu viele lange Sätze!
 * sentlengthvar 11.19 (erwartet: min. 5.27)
    OK!
 * simplepres 0.83 (erwartet: min. 0.74)
    OK!
 * spellcheck 0.21 (erwartet: zw. 0.10 und 0.20)
    Entweder zu viele Rechtschreibfehler oder zu viele Fremdwörter!
 * wordlength 4.41 (erwartet: max. 6.28)
    OK!

Regeln
Kapitel "2. Aufbau der Programmiersprache" hat keine Einleitung!
Gleitobjekt "Tabelle 1: Die wichtigsten arithmetischen und logischen Befehle in Forth" wird nicht im Text referenziert!
Gleitobjekt "Tabelle 1: Die wichtigsten arithmetischen und logischen Befehle in Forth" wird nicht im vorstehenden Text referenziert!

Abb. 2: Beispiel eines durch Confopy erzeugten Berichts für den Autor

Confopy kann auch zwei Versionen derselben Arbeit vergleichen (siehe Abb. 3), z. B. im Rahmen der Notenfindung als Indikator für die Überarbeitung der Arbeit durch die Studentin.

```
PROGRESS: Vorher- --> Nachher-Wert.
            (+) ... Erhöhung
            (-) ... Verringerung
            (=) ... gleichbleibend

METRIC              | PROGRESS
--------------------+---------------------
wordlength          | 04.41 --> 04.45  (+)
spellcheck          | 00.21 --> 00.16  (-)
lexicon             | 00.44 --> 00.49  (+)
sentlength          | 18.44 --> 17.88  (-)
ari                 | 62.66 --> 62.74  (+)
personalstyle       | 00.09 --> 00.08  (-)
impersonalstyle     | 00.15 --> 00.06  (-)
passiveconstructs   | 00.30 --> 00.33  (+)
simplepres          | 00.83 --> 00.80  (-)
adverbmodifier      | 00.04 --> 00.04  (=)
deadverbs           | 00.09 --> 00.06  (-)
fillers             | 00.03 --> 00.04  (+)
examplecount        | 05.00 --> 06.00  (+)
sentlengthvar       | 11.19 --> 08.82  (-)
```

Abb. 3: Ausgabe von Confopy beim Vergleich der Metriken zweier Versionen derselben studentischen Arbeit

5 Ergebnisse

Dieser Abschnitt präsentiert die Ergebnisse von Confopy auf 47 studentischen Arbeiten, die jeweils in einer ersten eingereichten Version (Draft) und einer Version nach Begutachtung und Überarbeitung (Final) vorliegen. Da die Software in den vergangenen Lehrveranstaltungen noch nicht zur Verfügung stand, kann hier nicht der direkte Einsatz evaluiert werden. Stattdessen soll die Analyse der Ergebnisse zwei Thesen untermauern: (1) Confopy kann dem Autor sinnvolle Hinweise liefern und (2) in einer „Groß-

veranstaltung“ kann ein automatisiertes Feedback Peer-Reviews ergänzen und damit den Lernprozess gezielter beeinflussen.

Tab. 1 zeigt die zusammengefassten Messungen für die Draft- und Final-Versionen. Dies veranschaulicht einerseits, dass sich Texte aus Zeitungsartikeln (im TIGER-Corpus) von (studentischen) wissenschaftlichen Arbeiten messbar unterscheiden. An den markierten Werten liegt der Mittelwert aller Arbeiten in einem Bereich, in dem Hinweise an den Autor geliefert werden, was (1) unterstreicht. Die fehlende Entwicklung zwischen Draft- und Final-Version zeigt, dass das reine Feedback der Peers nicht ausreicht (2).

	Draft-Version		Final-Version		
Metrik	Mittelwert	Sdv	Mittelwert	Sdv	TIGER
wordlength	4,65	0,34	4,47	0,76	5,52
spellcheck	***0,21***	0,06	***0,22***	0,11	0,08
lexicon	***0,45***	0,09	***0,44***	0,11	0,11
sentlength	***18,09***	3,36	***17,98***	4,84	15,58
ari	64,78	3,66	63,08	10,45	70,51
personalstyle	0,06	0,07	0,05	0,06	0,1
impersonalstyle	***0,07***	0,09	0,05	0,07	0,02
passiveconstructs	0,07	0,09	0,3	0,15	0,1
simplepres	0,8	0,13	0,78	0,17	0,66
adverbmodifier	0,03	0,01	0,03	0,01	0,04
deadverbs	0,05	0,03	0,05	0,04	0,05
fillers	0,02	0,01	0,02	0,01	0,03
examplecount	2,96	2,55	2,81	2,61	293,0
sentlengthvar	9,85	3,58	10,17	3,8	9,09

Tab. 1: Durchschnittlich gemessene Metriken in den studentischen Arbeiten im Vergleich zum TIGER-Corpus. Werte, deren Mittelwert von der empirisch ermittelten Empfehlung abweicht, sind markiert.

Der Nutzen des Werkzeugs im Sinne von (1) wird auch in Tab. 2 deutlich, da fast jeder Autor durchschnittlich 3–4 konstruktive Hinweise erhalten hat, die aus Bereichsverletzungen abgeleitet wurden. Für etwa die Hälfte der Arbei-

ten konnten zusätzlich strukturelle Hinweise aufgrund von durchschnittlich 3–4 Regelverletzungen gegeben werden.

	Draft-Version		Final-Version	
	Anzahl	Verletzungen	Anzahl	Verletzungen
Arbeiten mit Bereichsverletzungen	46	3,5	47	3,51
Arbeiten mit Regelverletzungen	22	3,91	25	3,32

Tab. 2: Anzahl der Arbeiten mit Bereichs- und Regelverletzungen sowie die durchschnittliche Anzahl der Verletzungen pro Arbeit

Tab. 3 zeigt, wie sich die Metrikwerte in den einzelnen Arbeiten verändert haben. Dabei werden jeweils die Paper zusammengefasst, bei denen der Metrikwert größer bzw. kleiner wurde. Diese Tabelle demonstriert, dass die Metriken bei der Überarbeitung eines Papers eine Aussagekraft besitzen. Dort ist zu sehen, dass bei vielen Metriken insbesondere in der erwünschten Richtung im Durchschnitt erhebliche Veränderungen erreicht werden können. Beispielsweise bei der Lesbarkeit (ari) stellt eine durchschnittliche Verbesserung von 7,47 einen beachtlichen Qualitätsgewinn dar, wenn der obere akzeptierte Schwellwert bei 71,96 liegt – die Erwartungsbereiche können der Ausgabe in Abb. 2 entnommen werden. Auch bei der Verwendung der Passivkonstruktionen steht eine Verbesserung um 0,195 Punkte dem erlaubten Maximalwert von 0,37 gegenüber. In jeweils 12 Arbeiten wurden bei der Überarbeitung so durchschnittlich 1,833 Bereichsverletzungen und 3,5 Regelverletzungen entfernt.

6 Zusammenfassung und Ausblick

Die vorgestellte Software Confopy kann sinnvoll den Peer-Review im Rahmen der Lehre wissenschaftlichen Schreibens ergänzen, wie die präsentierten Messungen zeigen. Weitere Anpassungen für die Konventionen in Schriften der Informatik sind möglich und können in der näheren Zukunft umgesetzt werden. Eine genauere Analyse beispielsweise der Schlüssigkeit von Argumentation im Text kann über den präsentierten Ansatz nicht erfolgen.

Inwieweit das Feedback von den Studierenden als hilfreich empfunden wird und ob sich auf dieser Basis die Qualität der studentischen Arbeiten verbessern, kann nur über genauere empirische Studien in den nächsten Jahren ermittelt werden. In jedem Fall zeigen erste Rückmeldungen, dass die Metriken eine gewisse Selbstreflektion begünstigen.

	Erhöhung des Metrikwerts		Verringerung des Metrikwerts		Anzahl gleich-bleibender Werte
Metrik	Anzahl	Delta	Anzahl	Delta	
wordlength	16	0,268	***31***	***0,643***	0
spellcheck	14	0,095	***33***	***0,066***	0
lexicon	15	0,12	32	0,08	0
sentlength	28	4,525	***19***	***5,111***	0
ari	19	4,02	***28***	***7,47***	0
personalstyle	26	0,084	***20***	***0,028***	1
impersonalstyle	30	0,118	***12***	***0,056***	5
passiveconstructs	22	0,0139	***25***	***0,195***	0
simplepres	***28***	***0,061***	19	0,172	0
adverbmodifier	32	0,013	***15***	***0,014***	0
deadverbs	26	0,038	***18***	***0,035***	3
fillers	27	0,013	***20***	***0,011***	0
examplecount	***27***	***3,815***	13	9,2462	7
sentlengthvar	***28***	***4,157***	19	3,947	0
Bereichsverletzungen	15	1,733	***12***	***1,833***	20
Regelverletzungen	15	2,600	***12***	***3,500***	20

Tab. 3: Veränderung der Metrikwerte und Regelverletzungen im paarweisen Vergleich der Draft- und Final-Versionen. Die Werte der erwünschten Richtung sind fett und kursiv hervorgehoben, „Anzahl“ entspricht der Anzahl der Arbeiten und “Delta“ der durchschnittlichen Veränderung in diesen Arbeiten.

Beim Einsatz in der Lehre muss berücksichtigt werden, dass es sich um keine Software zur Bewertung/Benotung handelt, sondern lediglich gewisse kon-

struktive Verbesserungsvorschläge generiert werden. Studierende müssen sich dessen bewusst sein, dass die Textqualität nach Abschluss des Moduls durch einen menschlichen Prüfer bewertet wird.

Die Software steht zum Download unter der folgenden URL bereit: https://github.com/ooz/Confopy .

Confopy zeigt für dieses Paper nur eine Bereichsverletzung an: die Sätze sind mit durchschnittlich 17,54 Worten zu lang. Die Lesbarkeit sollte mit ari=66,03 gegeben sein. Der Wortschatz bewegt sich mit 0,46 an der unteren Grenze.

Literatur

[AB06] Y. Attali, J. Burstein: Automated Essay Scoring With e-rater V.2. Journal of Technology, Learning, and Assessment, 4:3, S. 191–205, 2006.

[BB05] R. E. Bennett, A. Ben-Simon: Toward More Substantively Meaningful Automated Essay Scoring. Journal of Technology, Learning, and Assessment, 6:1, S. 1–47, 2007.

[Bra04] S. Brants et. al.: TIGER: Linguistic Interpretation of a German Corpus. Journal of Language and Computation, 2, S. 597–620, 2004.

[Gui01] W. H. Guiford: Teaching Peer Review and the Process of Scientific Writing. Advances in Physiology Education, 25:3, S. 167–175, 2001.

[SS67] E. A. Smith, R. J. Senter: Automated Readability Index. Technischer Bericht AMRL-TR-66-220, Aerospace Medical Research Laboratories, Wright-Patterson Air Force Base, Ohio, 1967.

[TM02] S. Teufel, M. Moens: Summarizing Scientific Articles: Experiments with Relevance and Rhetorical Status. Computational Linguistics, 28:4, S. 409–445, 2002.

[Tim11] B. E. C. Timmerman et. al.: Development of a ‚universal' rubric for assessing undergraduates' scientific reasoning skills using scientific writing. Assessment & Evaluation in Higher Education, 36, S. 509–547, 2011.

[VNC03] S. Valenti, F. Neri, A. Cucchiarelli: An Overview of Current Research on Automated Essay Grading. Journal of Information Technology Education, 2, S. 319–330, 2003.

[Wei06] N. Weicker, B. Draskoczy, K. Weicker: Fachintegrierte Vermittlung von Schlüsselkompetenzen der Informatik. In: HDI 2006, Hrsg.: P. Forbrig, G. Siegel, M. Schneider, S. 51–62, Bonn, Gesellschaft für Informatik, 2006.

[Zsc15] O. Zscheyge: Automatische sprachliche und strukturelle Analyse wissenschaftlicher Texte. Masterarbeit, HTWK Leipzig, 2015.

Mathematisches Argumentieren und Beweisen mit dem Theorembeweiser Coq

Sebastian Böhne*, Maria Knobelsdorf+, Christoph Kreitz*

*Fachbereich Informatik, Universität Potsdam
14482 Potsdam

+Fachbereich Informatik, Universität Hamburg
22527 Hamburg

Zusammenfassung: Informatik-Studierende haben in der Mehrzahl Schwierigkeiten, einen Einstieg in die Theoretische Informatik zu finden und die Leistungsanforderungen in den Endklausuren der zugehörigen Lehrveranstaltungen zu erfüllen. Wir argumentieren, dass dieser Symptomatik mangelnde Kompetenzen im Umgang mit abstrakten und stark formalisierten Themeninhalten zugrunde liegen und schlagen vor, einen Beweisassistenten als interaktives Lernwerkzeug in der Eingangslehre der Theoretischen Informatik zu nutzen, um entsprechende Kompetenzen zu stärken.

1 Ausgangslage

In der Studieneingangsphase der Informatik-Bachelorstudiengänge in Deutschland stellen die mathematischen und theoretischen Fachanforderungen in den korrespondierenden Einführungsveranstaltungen nach wie vor eine große Herausforderung dar. Problematisch für die Studierenden sind dabei weniger konkrete Wissenslücken denn generelle Schwierigkeiten im Umgang mit mathematischen Formalismen, dem damit verbundenen Abstraktionsgrad, sowie der Übertragung mathematischer Methoden und Problemlösestrategien auf die Informatik (Knobelsdorf und Frede, 2016). Hier kann immer wieder durch die Lehrenden beobachtet werden, dass einige Studierende bereits große Schwierigkeiten haben, grundlegende mathematische Formalismen überhaupt zu lesen und ihren Inhalt zu verstehen. Diejenigen Studierenden, die diese Hürde überwinden, sind dann nicht

automatisch auch in der Lage, eine Lösung anhand der mathematischen Formalismen zu entwickeln. Die entwickelten Beweisideen für ein konkretes Problem formal korrekt aufzuschreiben, stellt dann noch einen weiteren notwendigen Schritt dar, den viele Studierende erst noch erlernen müssen. Insgesamt wird deutlich, dass Studierende einen großen Entwicklungsbedarf hinsichtlich ihrer formal-mathematischen Fachkompetenzen haben. Daher ist die Entwicklung entsprechender didaktischer Maßnahmen und Werkzeuge, die den Studierenden helfen, zugehörige Kompetenzen zu entwickeln, so außerordentlich wichtig.

In diesem Artikel stellen wir einen fachdidaktischen Ansatz für die Eingangslehre der Theoretischen Informatik dar, der mit einem Beweisassistenten arbeitet und den oben genannten Bedarf aufgreift. Der Ansatz verfolgt das Ziel, Grundlagen im mathematischen Argumentieren und Beweisen zu vermitteln und die formal-präzise Darstellung und Anwendung theoretischer Konzepte einzuüben. Dadurch sollen die Studierenden ein Verständnis für den Zusammenhang zwischen Mathematik und Informatik entwickeln sowie befähigt werden, Fortschritte im strukturierten und abstrakten Denken zu machen. In Abschnitt 2 werden wir unseren Ansatz theoretisch begründen und als konkretes Werkzeug den Beweisassistenten Coq vorstellen. Das Vermittlungskonzept mit Coq sowie konkrete Inhalte und Aufgaben werden in Abschnitt 3 vorgestellt. Der Artikel schließt mit einem Fazit und Ausblick auf weitere Schritte in Abschnitt 4.

2 Beweisen lernen mit einem Beweisassistenten

Computergestützte Systeme zur Durchführung mathematischer Beweise werden seit den 1960er Jahren entwickelt. Man unterscheidet zwischen Systemen, die Beweise erzeugen, sowie solchen, die Menschen beim Erzeugen von Beweisen unterstützen. Letztere werden auch Beweisassistenten oder interaktive Theorembeweiser genannt. Beweisassistenten wurden bereits des öfteren in weiterführenden Veranstaltungen in der Theoretischen Informatik und Mathematik eingesetzt (Henzand Hobor 2011.), (Delahaye, 2005), (Andrew et al., 2004). In der Eingangslehre der Theoretischen Informatik hingegen sind Beweisassistenten bisher nur sehr vereinzelt eingesetzt worden (Sakowicz and Chrząszcz, 2007). Insgesamt wird dabei überwiegend eine sehr positive Resonanz bei den Studierenden festgestellt, so dass die Entwicklung und Erprobung eines didaktischen Konzepts für die Nutzung von Beweisassistenten auch in der Studieneingangsphase als vielverspre-

chend erscheint. In weiteren Verlauf dieses Abschnitts werden wir Beweisassistenten genauer vorstellen und theoretisch herleiten, warum ihr Einsatz in der Eingangslehre sinnvoll ist.

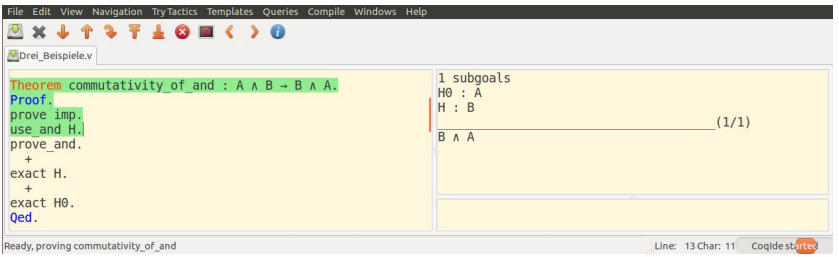
Es existiert eine ganze Reihe von Beweisassistenten. Einer von ihnen, Coq (Bertot und Casteran, 2004), ist Gegenstand dieses Artikels. Andere Beweisassistenten sind beispielsweise Isabelle (Nipkow, Paulson und Wenzel, 2002) oder NuPRL (Constable et al., 1986), (Allen et al., 2006), bei weitläufigerer Auslegung des Begriffs auch Agda (Bove, Dybjer und Norell, 2009) oder Idris (Brady, 2013). Einige der Beweisassistenten, darunter auch Coq, fußen auf der Typentheorie. Insbesondere werden Theoreme als Typen und Beweise für Theoreme als Elemente des jeweiligen Typs aufgefasst. Fast alle Theoreme beinhalten Implikationen und/oder Allquantifikationen. Zugrundeliegende Beweise solcher Theoreme sind dann Funktionen (im getypten λ-Kalkül), so dass solche Beweisassistenten eng verzahnt mit funktionaler Programmierung sind.

Bei Beweisassistenten im engeren Sinne des Begriffes kann der Nutzer Schritt für Schritt Wissen aus den Voraussetzungen generieren und das bisherige Ziel durch ein leichter zu zeigendes, aber hinreichendes Ziel ersetzen. Konkret werden Beweise durch eine Auflistung von sogenannten Taktiken innerhalb einer dem Programmieren ähnlichen Entwicklungsumgebung gewonnen, wobei den Taktiken systemintern Inferenzregeln zugrunde liegen. Das System führt die durch die Taktiken vermittelten Inferenzregeln aus. Dabei prüft es jeden Beweisschritt auf seine formale Korrektheit. Dies ist in erster Näherung mit dem Compilereinsatz beim Programmieren vergleichbar.

Für den Einsatz in der Studieneingangsphase ist es entscheidend, dass die Studierenden zum Arbeiten mit einem Beweisassistenten keinerlei typentheoretisches Vorwissen haben müssen. Weder müssen sie wissen, dass Theoreme Typen sind, noch, dass die zugehörigen Beweise Elemente solcher Typen, meist also Funktionen, sind. Für sie hat ein Beweis eines Theorems einfach einen Namen und kann durch eine Abfolge von Schritten gewonnen werden. Dadurch kommen Beweisassistenten wie Isabelle, NuPRL oder Coq überhaupt erst für eine didaktische Verwendung innerhalb einer Theorieveranstaltung in Frage.

Betrachten wir zur Verdeutlichung einen Beweis für $A \wedge B \rightarrow B \wedge A$ in einem Beweisassistenten. Um die Implikation zu beweisen, müssen wir $A \wedge B$ annehmen und zeigen, dass daraus $B \wedge A$ folgt. Der Beweisassistent muss daher eine entsprechende Taktik (*prove_imp*) zur Verfügung stellen. Aus

unserem Wissen um A ∧ B können wir dann zunächst das Wissen um die Einzelbestandteile, also A und B, generieren, wobei auch hier wieder eine entsprechende Taktik (*use_and*) vorhanden sein muss. Um schließlich B ∧ A zu beweisen, müssen wir einzeln sowohl B als auch A beweisen (*prove_and*). Dies ist nicht schwer, da uns die entsprechenden Aussagen bereits zur Verfügung stehen, und Beweisassistenten erlauben in dem Fall daher den Abschluss des jeweiligen Beweises (*exact*). Die konkrete Umsetzung des Theorems in Coq ist im folgenden Schnappschuss zu sehen:



Auf der linken Seite ist der Beweis als Abfolge von Taktiken zu sehen, wobei die bereits ausgeführten Taktiken grün markiert sind. Das jeweils aktuelle Beweisziel (hier nach Ausführung der Taktik use_and) ist rechts zu sehen.

Ein möglicher Einwand gegen einen Einsatz von Beweisassistenten in der Studieneingangsphase könnte der Umstand sein, dass keiner von ihnen ursprünglich für eine didaktische Verwendung konzipiert wurde. Stattdessen sind sie für wissenschaftliche und industrielle Anwendungen gedacht und richten sich an Theoretiker und Mathematiker (Nederpelt und Geuvers, 2014). Trotzdem besitzen sie auch aus didaktischer Sicht verschiedene Vorteile gegenüber Stift und Papier, die unserer Meinung nach für die Eingangslehre in der Theoretischen Informatik außerordentlich wichtig sind:

- Lernende erhalten unmittelbares Feedback auf ihre Beweisidee, indem sie genau angezeigt bekommen, ob ein Beweisschritt erfolgreich anwendbar ist. Dies steht im Gegensatz zur üblichen Praxis, wo Studierende in den Übungen oftmals gar kein individualisiertes Feedback bekommen und die Rückgabe der Hausaufgaben erst eine Woche später erfolgt. Es kann dann auch nur eine Rückmeldung zum Gesamtprodukt und nicht zu den einzelnen Schritten bei der Beweisfindung gegeben werden. Dies wiederum verhindert einen konstruktiven Umgang mit fehlerhaften Ansätzen.
- Bei Beweisassistenten bekommen Lernende eine direktes Feedback, welche Schritte sie noch ausführen müssen, damit ein Beweis vollständig

abgeschlossen ist. Genauer zeigt ein Beweisassistent immer an, welcher Teilschritt als nächstes zu zeigen ist und welche Voraussetzungen dabei genutzt werden können. Dies ist gerade zu Beginn sehr hilfreich, denn den Studierenden fehlt oft eine klare Vorstellung davon, wie sie konkret vorgehen müssen, um einen Beweis zu entwickeln (Knobelsdorf und Frede, 2016).

- Lernende werden an eine präzise und formal korrekte Vorgehensweise gewöhnt, da Beweisassistenten zu einer kompromisslosen Anwendung mathematischer Formalismen zwingen. Es ist bspw. nicht mehr möglich, statt dem formalen Argument eine Zeichnung aufs Blatt zu bringen und auf die Milde des Korrektors zu hoffen. Die formalen Regeln müssen auf genau die formalen Objekte angewandt werden. Hat etwa eine Eingabe in eine Funktion den falschen Typ, so geht es an dieser Stelle nicht weiter, bevor der Studierende die richtige Einsetzung macht. Dies hat den entscheidenden Vorteil, dass Fehler frühzeitig erkannt und in konstruktiver Art und Weise behoben werden können.
- Lernende können zu Beginn verschiedene Beweisideen ausprobieren, indem sie mit dem jeweiligen System spielen. Das garantiert höhere Flexibilität gegenüber Stift- und Papierbeweisen, in denen man nur sehr schlecht Dinge streichen oder verbessern kann, ohne den Beweis neu aufschreiben zu müssen. Aus dem gleichen Grund sollte ein Beweisassistent auch besser darin unterstützen, mit einem Beweis überhaupt zu beginnen.

Für die Entwicklung eines didaktischen Lehr-Lern-Konzepts haben wir den Beweisassistenten Coq ausgewählt, da er eine sehr gute Dokumentation sowie eine vergleichsweise einfach zu verstehende Schnittstellengestaltung besitzt. Auch die Installation von Coq ist deutlich einfacher umzusetzen. Ferner gehört Coq zu den Beweisassistenten, die am meisten in der Hochschullehre, insbesondere in der Studieneingangsphase, erprobt wurden.

3 Entwicklung eines didaktischen Lehr-Lern-Konzepts

3.1 Vermittlungsrahmen

Für die konkrete Vermittlung relevanter Kompetenzen in der Auseinandersetzung mit Coq und Beweisaufgaben aus der Theorie schlagen wir den Cognitive Apprenticeship Ansatz vor (Collins, 1991). Diesen didaktischen

Ansatz haben wir bereits im Rahmen der Lehrveranstaltung „Theoretische Informatik I“ im Bachelorstudiengang Informatik an der Universität Potsdam erfolgreich eingesetzt (Knobelsdorf, Kreitz und Böhne, 2014), (Knobelsdorf 2015).

Der von Collins et al. erarbeitete Ansatz plädiert dafür, die in einem Fach eingesetzten Fach- und Handlungskompetenzen stärker in den Fokus der Lehre zu setzen. In der universitären Lehre wird traditionell auf deklaratives Fachwissen und damit auf die aus den Tätigkeiten der Fachcommunity hervorgegangenen Wissens-„Produkte“ fokussiert. Das wird in der Theoretischen Informatik besonders deutlich, wo Modelle, Theoreme und korrespondierende Beweise vorgestellt werden, während die für die Entwicklung benötigten Fachkompetenzen eher implizit thematisiert werden. Letztere müssen sich die Studierenden daher aus den entsprechenden Vorlesungsinhalten selbst rekonstruieren. Ein Unterfangen, das viele Studierende vor große Herausforderungen stellt, insbesondere wenn sie keine Affinität für diesen Bereich der Informatik haben. Dies liegt nicht zuletzt an der kognitiven Natur besagter Fachkompetenzen, deren korrespondierende Handlungen nur in Teilen beobachtbar sind und damit auch nur teilweise nachvollzogen werden können.

Collins et al. argumentieren nun, dass die Vermittlung solcher Fachkompetenzen daher im Wesentlichen von der Fähigkeit der Lehrenden abhängt, ihre Handlungen bewusst zu reflektieren und sprachlich explizit darzulegen. Collins et al. schlagen daher einen didaktischen Ansatz vor, der sich an der traditionellen Handwerksausbildung orientiert und zum Ziel hat, die kognitiven Fach- und Handlungskompetenzen zu explizieren und in den Fokus der Vermittlung zu setzen. Dieser Anfang der 1990er Jahre vorgeschlagene Ansatz greift damit der aktuellen Kompetenzorientierung im Bildungsbereich vor.

Der Ansatz schlägt mehrere Formen der Vermittlung und Gestaltung eines Lehr-Lern-Prozesses vor, die sich an einer konstruktivistischen Didaktik und dem situierten Lernen orientieren (Reinmann-Rothmaier und Mandl, 2001). Dabei sind die folgenden drei Ansätze für uns besonders relevant:

1. Modeling: Die lehrende Person demonstriert relevante Fachhandlungen in der konkreten Auseinandersetzung am Beispiel vor. Lernende werden befähigt, diese beobachten und nachahmen zu können.
2. Coaching: In der aktiven Erprobung des Beobachteten werden Lernende von der lehrenden Person begleitet und erhalten direktes Feedback.

3. Scaffolding: Die Entwicklung der Fach- und Handlungskompetenzen wird durch ein helfendes Gerüst vorgegeben.

3.2 Gestaltung eines möglichen Kursablaufs

Wir wollen uns nun mit der Frage auseinandersetzen, wie ein konkreter Kurs auszusehen hat, der in der Lage ist, den Umgang mit mathematischen Formalismen mithilfe des Cognitive Apprenticeship Ansatzes erfolgreich zu schulen.

Zunächst wird der Modeling-Aspekt klassisch durch Vorlesungen realisiert, wobei im Gegensatz zu üblichen Vorlesungen das handwerkliche Rüstzeug nicht implizit bleibt, sondern den expliziten Gegenstand der Vorlesung darstellt. Genauso wie bei anderen Lehrveranstaltungen auch werden Übungen zu den Inhalten der Vorlesung angeboten, wobei auf die Übungen mindestens genauso viel Zeit verwendet wird, wie auf die Vorlesungen selbst. Die Übungen unseres Kurses unterscheiden sich dabei jedoch drastisch von der üblichen Variante: Zwar werden die Studierenden in den Übungen, wie es üblich ist, ganz normal vom Lehrteam begleitet, welches die üblichen Hilfen anbieten kann, vor allem aber übernimmt nun der Beweisassistent, hier Coq, ständiges und individualisiertes Feedback zu den Lösungsansätzen der Studierenden (Coaching). Darüberhinaus wird durch das von uns „modifizierte“ Coq-System, durch die Auswahl der Aufgaben sowie durch vorbereitete Coq-Dateien den Studierenden ein Gerüst an die Hand gegeben, das die Lernenden schrittweise in der Beweisführung unterstützt (Scaffolding). Die zu bearbeitenden Aufgaben sind so gestellt, dass sie unmittelbar das in der Vorlesung Vorgestellte aufgreifen und im Sinne des Cognitive-Apprenticeship-Ansatzes, sowie allgemeiner einer konstruktivistisch orientierten Didaktik, den Lernenden die Möglichkeit zur Erprobung und Ausgestaltung geben.

Am Ende der Übungen verbleibt ein gewisser Zeitrahmen (ca. 30 Minuten) für die Arbeitssicherung. Erst jetzt wird der Übungsleiter vor der gesamten Gruppe aktiv, indem er wesentliche Eckpunkte der Aufgabenlösung vorstellt und dabei auf die stattgefundene Bearbeitungsphase eingeht. Die Studierenden haben hier die Möglichkeit, weitere Fragen zu stellen und gemeinsam abschließend in der Gruppe zu diskutieren.

3.3 Auswahl und Begründung der Themeninhalte

Als Inhalte für einen möglichen Kurs schlagen wir eine thematische Zweiteilung vor: Logik, genauer Aussagen- und Prädikatenlogik (beliebiger Stufe), und Datentypen, genauer: Verbundsdatentypen, Listen, Natürliche Zahlen sowie Binärbäume. Der thematische Start mit Logik erfolgt im wesentlichen aus zwei Gründen:

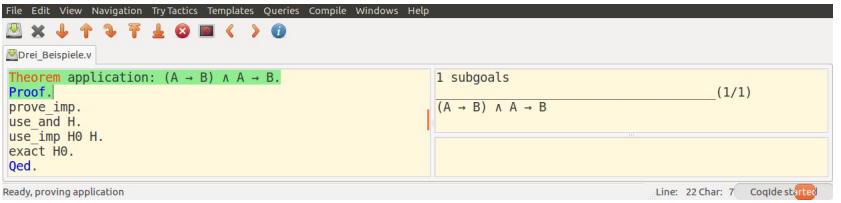
1. Die Logik stellt eine ungetrübte mathematische Argumentationsstruktur in Reinform dar. Die übliche mathematische Argumentation hingegen vermengt inhaltliche Intuition und logische Argumentation. Für die Lehre hat letzteres den entschiedenen Nachteil, dass die von den Mathematikexperten verwendeten logischen Methoden und Kompetenzen für die Studierenden unsichtbar bleiben müssen, da sie von inhaltlichen Erwägungen überdeckt werden. Durch die Konzentration auf Logik rufen wir die Existenz solcher zugrundeliegender Methoden und Kompetenzen ins Bewusstsein und machen sie explizit.
2. Die Klarheit der Logik spiegelt sich auch in der einfachen Handhabung innerhalb des Beweisassistenten wider. Während inhaltliche Theorien einiger Vorbereitung in Coq bedürften, können die Studierenden bei Aussagen zur Logik direkt mit der Arbeit beginnen. Dazu bedarf es lediglich einiger vorbereiteter Taktiken, die von den Studierenden aufgerufen werden können, so wie sie auch vorgegebene Prozeduren oder Methoden beim Programmieren nutzen.

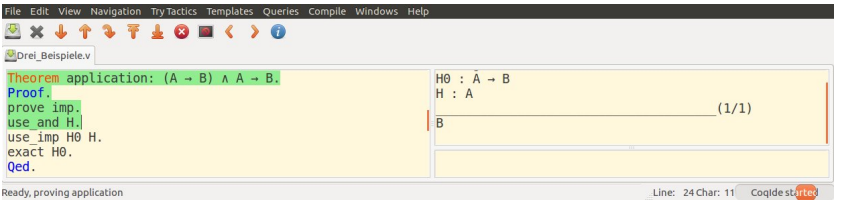
Die Auseinandersetzung mit Datentypen, als konkretem inhaltlichen Thema in der zweiten Hälfte eines möglichen Kursablaufs führt dann zurück in den mathematischen „Argumentationsalltag“. Die Behandlung von Datenstrukturen ergibt sich dabei notwendigerweise, weil die Inhalte erst konstruiert werden müssen, über die später Beweise zu führen sind. Darüberhinaus bieten sich die konkret gewählten Datentypen besonders an, da sie relativ einfach sind und zum Kernkanon in der Informatiklehre gehören.

3.4 Erläuterungen an zwei Beispielen

Im Folgenden stellen wir zwei Beispieltheoreme und ihren Beweis in Coq vor. Das erste Theorem ist Teil der Aussagenlogik, wohingegen das zweite die Kommutativität der natürlichen Zahlen behauptet.

Das erste Theorem lautet: (A → B) ∧ A → B. In Coq geben wir dem Theorem einen Namen, application, und beginnen den Beweis mit dem Schlüsselwort Proof:

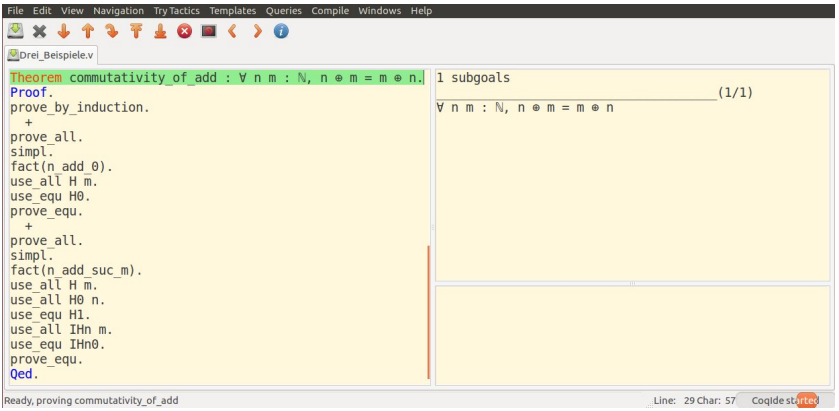




Mit dem Schlüsselwort *prove_imp* geben wir nun an, dass wir eine Implikation beweisen wollen. Daraufhin steht uns die Prämisse im rechten Fenster zur Verfügung, um die Konklusion herzuleiten. Vermittelst *use_and* können wir die uns zur Verfügung stehende Konjunktion in ihre beiden Bestandteile aufteilen:

Wir wenden nun mit *use_imp* die Implikation A → B und die Prämisse A an, um B zu erhalten. Dies ist aber genau die zu beweisende Aussage, weswegen wir unseren Beweis mit *exact* abschließen können.

Das zweite Theorem lautet: ∀ n m : ℕ , n ⊕ m = m ⊕ n. Seinen Beweis wollen wir nicht in seinen Einzelheiten besprechen. Stattdessen erläutern wir lediglich die wesentlichen Unterschiede gegenüber dem obigen Beispiel. Zunächst stellen wir den Beweis in Coq vor:



Der Beweis in diesem Beispiel ist ein Induktionsbeweis und deutlich länger als im vorhergehenden Beispiel, obwohl wir bereits auf die beiden Lemmata *n_add_0* und *n_add_suc_m* zurückgreifen. Diese Lemmata wurden mithilfe von Induktion bewiesen. Dieses Mittels bedienen wir uns auch in diesem Beweis. Es sind also zwei Teilziele zu zeigen: Der Induktionsanfang und der Induktionsschritt, die jeweils durch ein ‚+' sichtbar voneinander getrennt dargestellt sind. Der Beweis besitzt hier also mehr Struktur als im obigen Beispiel. Darüberhinaus werden in diesem Beispiel Regeln rund um die Allquantifikation und die Gleichheit genutzt, für die es im aussagenlogischen Fall keine Entsprechung gibt.

4 Fazit und Ausblick

Mit dem vorgestellten Konzept wird ein für die Eingangslehre der Theoretischen Informatik neues Lernwerkzeug vorgeschlagen. Dieses kann nicht nur innovative Impulse für eine Neugestaltung der Lehre in diesem Bereich liefern, sondern insbesondere den Lernprozess und die damit einhergehende Fachkompetenzentwicklung individuell unterstützen und die Studierenden dazu anregen, selbstbestimmtes Lernen und Arbeiten in diesem Bereich zu intensivieren.

Das hier vorgestellte Lehr-Lern-Konzept mit dem Einsatz von Coq als Lernwerkzeug ist nicht auf bestimmte Themen, Syntax oder einen speziellen Ablauf beschränkt. Es lässt sich ohne prinzipielle Probleme auf alle mathematischen und theoretischen Inhalte übertragen. Zugestandenermaßen erfordert jedes Themengebiet eine didaktische Formalisierung und die entsprechende Vorbereitung des Systems. Die technische Umsetzung sowie die

grundsätzlichen Kernüberlegungen können aber im Wesentlichen übernommen werden. Mögliche neue Einsatzgebiete sind denkbar für die Studieneingangsphase der Mathematik und auch der Physik, wo ähnlich der Informatik mathematische Grundlagen erlernt und dann auf das eigene Fach übertragen werden.

Um den Einsatz unseres Konzepts zu erproben, planen wir Anfang Oktober 2016 am Fachbereich Informatik der Universität Hamburg einen entsprechenden Kurs durchzuführen. Die Veranstaltung ist als zweiwöchige Blockveranstaltung konzipiert und richtet sich an Bachelor-Studierende der Informatik und Wirtschaftsinformatik, die im Wintersemester 2016/2017 ins dritte oder höhere Fachsemester kommen. Parallel dazu wird eine empirische Evaluation des Konzepts stattfinden. Die dabei gewonnen Erkenntnisse sollen genutzt werden, um das Konzept weiterzuentwickeln und ein für Coq passendes Schnittstellendesign zu konzipieren, welches insbesondere Studierende in der Studieneingangsphase anspricht.

Literatur

Allen, S., Bickford, M., Constable, R., Eaton, R., Kreitz, C., Lorigo, L., Moran, E. 2006. Innovations in Computational Type Theory using Nuprl. Journal of Applied Logic, 4(4):428–469.

Bertot, Y., Casteran, P. 2004. Interactive Theorem Proving and Program Development Coq'Art: The Calculus of Inductive Constructions, Springer.

Bove, A., Dybjer, P., Norell, U. 2009. A Brief Overview of Agda – A Functional Language with Dependent Types. In TPHOLs 2009, LNCS 5674, Springer, 73–78.

Brady, E. 2013. Idris, a General Purpose Dependently Typed Programming Language: Design and Implementation. Journal of Functional Programming, 23(5):552–593.

Collins, A., Brown, J. S., Holum, A. 1991. Cognitive apprenticeship: Making thinking visible. American Educator, 6(11):38–46.

Constable, R., et. al. 1986. Implementing Mathematics with the Nuprl Development System, Prentice-Hall.

Delahaye, D., Jaume, M., Prevosto, V. 2005. Coq, un outil pour l'enseignement. Technique et Science Informatiques, 24(9):1139–1160.

Henz, M., Hobor, A. 2011. Teaching Experience: Logic and Formal Methods with Coq. In Certified Programs and Proofs, Lecture Notes in Computer Science, Vol. 7086, Springer, 199–215.

Knobelsdorf, M. 2015. The Theory Behind Theory – Computer Science Education Research Through the Lenses of Situated Learning. In Proceedings of ISSEP Conference, Lecture Notes in Computer Science, Vol. 9378, Springer, 21–21.

Knobelsdorf, M., Frede, C. 2016. Analyzing Student Practices in Theory of Computation in Light of Distributed Cognition Theory. In Proceedings of the 12th ICER Conference, ACM, in press.

Knobelsdorf, M., Kreitz, C., Böhne, S. 2014. Teaching theoretical computer science using a cognitive apprenticeship approach. In Proceedings of the 45th SIGCSE Conference, ACM, 67–72.

Nederpelt, R., Geuvers, H. 2014. Type Theory and Formal Proof: An Introduction, Cambridge University Press.

Nipkow, T., Paulson, L., Wenzel, M. 2002. Isabelle/HOL A Proof Assistant for Higher-Order Logic, Lecture Notes in Computer Science, Vol. 2283, Springer.

Reinmann-Rothmeier, G., Mandl, H. 2001. Unterrichten und Lernumgebungen gestalten. In A. Krapp und B.Weidenmann (Hrsg): Pädagogische Psychologie. Weinheim Beltz/PVU, 601–646.

Sakowicz, J., Chrząszcz, J. 2007. Papuq, a Coq assistant. In Proceedings of PATE'07 conference, Elsevier, 79–96.

Einsatz von Theorembeweisern in der Lehre

Alexander Steen, Max Wisniewski, Christoph Benzmüller

Fachbereich Mathematik und Informatik, Freie Universität Berlin
14195 Berlin
Email: {a.steen|m.wisniewski|c.benzmueller}@fu-berlin.de

Zusammenfassung: Dieser Beitrag diskutiert den Einsatz von interaktiven und automatischen Theorembeweisern in der universitären Lehre. Moderne Theorembeweiser scheinen geeignet zur Implementierung des dialogischen Lernens und als E-Assessment-Werkzeug in der Logikausbilding. Exemplarisch skizzieren wir ein innovaties Lehrprojekt zum Thema „Komputationale Metaphysik", in dem die zuvor genannten Werkzeuge eingesetzt werden.

1 Einleitung

Die Formale Logik ist heutzutage gleichermaßen in der Philosophie, der Mathematik und der Informatik beheimatet. Mit leicht unterschiedlicher Ausprägung ist sie in all diesen Disziplinen sowohl auf Bachelor- als auch auf Master-Niveau in Lehrveranstaltungen der jeweiligen Studiengänge vertreten. Hierbei kommt der Logik in der wissenschaftlichen Praxis dieser drei Fächer eine Doppelrolle zu.

Dem Humboldt'schen Ideal der Einheit von Forschung und Lehre folgend, formt sie ein implizites Fundament für wissenschaftliches Arbeiten, Forschung und Lehre. Logisches Denken ist dabei „*eines der grundlegenden Instrumente des kritischen Denkens und eine Basis des Argumentierens*" [Kru10, S. 52]. Demnach sei nach Kruse die Auseinandersetzung mit dem Argumentbegriff und mit dem Ziehen von Schlüssen zwingender Kernbestandteil der Lehre kritischen Denkens. Das formale Studium genau jener Begriffe ist spätestens seit Aristoteles der Hauptinhalt der Logik als wissenschaftliche Disziplin. In der wissenschaftlichen Ausbildung ist die Fähigkeit zur kritischen Auseinandersetzung und Reflexion basierend auf logischer und strukturierter Argumentation immerzu intrinsisches Qualifikationsziel.

Die zweite, besondere Rolle der Logik in den oben genannten Disziplinen bezieht sich auf ihre explizite Erscheinung in der Lehre. Insbesondere in der universitären Informatikausbildung nimmt die Formale Logik als Grundlage und Werkzeug zur Formalisierung, Abstraktion und Analyse von Problemen und Lösungsansätzen eine zentrale Schlüsselposition mit vielen verschiedenen Ausprägungen ein. So sind formale Beweise und Argumentationen über abstrakte, prinzipielle Eigenschaften von Sprachen (z. B. unter den Aspekten der Berechenbarkeit, Entscheidbarkeit oder Komplexität) eines der Hauptaugenmerke in der Lehre der theoretischen Informatik. In der praktischen Informatik gehören Logik-Kalküle, wie der Hoare-Kalkül oder Petri-Netze, sowie formale Methoden zur Grundausbildung. Auch bei praktischen Aspekten, wie dem Programmieren als solches, liegen zahlreiche formale Systeme zu Grunde, die als Teil einer forschungsvorbereitenden Ausbildung elementarer Bestandteil der universitären Lehre sind.

Unserer Beobachtung nach sind es oftmals gerade diese formalen und theoretischen Teilaspekte der Informatik, die die Studierenden beim Studienprozess behindern, da Module mit diesen Themen in der Regel auch zum Pflichtanteil eines Studiums gehören. Die Probleme liegen nicht zuletzt an dem abstrakten Charakter, der diesen Themen anhaftet. Allerdings scheint es auch signifikant an einer sinnvollen didaktisch-methodischen Umsetzung dieser Grundlagenlehre zu mangeln, insbesondere an Übungs- bzw. Anwendungsphasen, die nicht lediglich aus theoretischen Aufgabenstellungen bestehen, welche mit Stift und Papier zu lösen sind.

Ausgehend von dem Forschungsgebiet der Formalen Logik und der künstlichen Intelligenz formte sich seit der revolutionären Entwicklung des Resolutionskalküls [Rob65] und verwandter Techniken die Disziplin der Automatischen Deduktion. Die resultierenden Systeme, allem voran sog. automatische und interaktive Theorembeweiser, sind inzwischen nicht mehr nur reines Produkt der akademischen Grundlagenforschung, sondern erleben in den vergangenen Jahren, dank immenser Fortschritte, eine zunehmende Verwendung in der Wissenschaftspraxis der Mathematik und Informatik.[1] Formale Methoden erfreuen sich zudem einer gesteigerten Nachfrage in der Wirtschaft und ebenso einer zunehmenden Aufmerksamkeit in der Gesellschaft.

[1] Beginnend mit dem Beweis des Vier-Farben-Satzes [AH89]. Ein weiterer Meilenstein war der computergestützte Beweis der Kepler'schen Vermutung im Flyspeck-Projekt [H+15].

Wir sind der Überzeugung, dass der frühe und intensive Einsatz solcher Werkzeuge nicht nur einen forschenden Charakter der universitären Informatiklehre unterstützen kann, sondern elementar zum Verständnis von theoretischen Inhalten beiträgt. Studierende können mit Hilfe von Beweisassistenzsystemen selbstständig praktische Experimente durchführen und sich damit auch bei theoretischen Inhalten computergestützt aktiv in einer Dialogsituation erproben: Zum einen können eigene Fehler in Lösungswegen durch direkte Interaktion mit dem System in Echtzeit aufgedeckt und rekonstruiert werden. Zum anderen können die dabei entstehenden Aufzeichnungen als formal verifizierte Lerntagebücher genutzt werden.

Im Folgenden stellen wir kurz einige aktuelle Entwicklungen der automatischen Deduktion vor. Anschließend wird argumentiert, wie mit Hilfe von Theorembeweisern dialogisches Lernen für die Lehre von theoretischen Grundinhalten der Informatik implementiert werden kann. Im Anschluss geben wir ein kurzes Fallbeispiel, in dem Theorembeweiser in einer konkreten Lernveranstaltung im Grenzgebiet der Informatik, Mathematik und Philosophie eingesetzt werden.

2 Logik und Theorembeweiser

Zu Beginn des letzten Jahrhunderts versuchten Bertrand Russell und Alfred Whitehead in ihrer Principia Mathematica das gesamte Gebäude der Mathematik auf ein präzises logisches Fundament zurückzuführen. Als Grundlage ihres Bestrebens setzten sie ein typisiertes System der höherstufigen Logik ein. Unter höherstufiger Prädikatenlogik verstehen wir eine Logik, in der universelle und existenzielle Aussagen über beliebige Mengen und Funktionssymbole ausdrückbar sind. Dem gegenüber steht die wesentlich bekanntere Prädikatenlogik erster Stufe, die Allaussagen nur über Individuen erlaubt. Dem Bestreben, der Mathematik eine Grundlage purer Logik zu geben, wurde allerdings durch Kurt Gödels ersten Unvollständigkeitssatz (zunächst) ein Ende gesetzt.

Heutzutage haben sich die konkreten Logik-Systeme der verschiedenen Forschungsgebiete stark voneinander wegentwickelt. In der Mathematik wird als Grundlage zumeist eine Zermelo-Fraenkel Axiomatisierung der Mengenlehre in Logik erster Stufe herangezogen, Informatiker beschränken sich nicht selten auf entscheidbare Logikfragmente, und Philosophen untersuchen neben einer Vielzahl von nicht-klassischen Logiken auch Varianten höher-

stufiger Logiken. Nicht-klassische Logiken spielen auch im Bereich der künstlichen Intelligenz eine große Rolle.

Die Logik, auf die wir uns konzentrieren, ist die einfache Typentheorie [Chu40], eine Form der höherstufigen Prädikatenlogik, die von Alonzo Church entwickelt und von Leon Henkin mit einer praktisch benutzbaren Semantik ausgestattet wurde [Hen50]. Die Wahl dieser Logik hat gegenüber den anderen Alternativen gewisse Vorteile: Erstens kann man im Gegensatz zur Prädikatenlogik erster Stufe auf natürlichere, elegantere Art und Weise (komplexe) Aussagen formulieren und beweisen. Zweitens gibt es, im Gegensatz zu den mathematischen Logiken, speziell zugeschnittene Programme, mit denen Beweise automatisch generiert oder mitverfolgt werden können. Das dritte, vielleicht stärkste Argument ist, dass in den letzten Jahren untersucht wurde, dass viele der Speziallogiken der Philosophie und KI in der einfachen Typentheorie dargestellt werden können (siehe z. B. [Ben11] und darin enthaltene Quellen).

Für Studierende bilden formale Kalküle, wie sie in der Logik und anderen formalen Disziplinen genutzt werden, erfahrungsgemäß ein großes Hindernis. Oft sind Studierende es gewohnt, intuitiv zu argumentieren, viele Schritte in einem (vermeintlichen) Beweis zusammenzufassen oder sogar an Stellen, die einem als trivial erscheinen, Lücken zu lassen. Infolgedessen ist es zumeist schwer, die formale Korrektheit eines solchen „Beweises“ zu verifizieren, da insbesondere aufgrund von Ungenauigkeiten nicht alle Einzelschritte als korrekt garantiert werden können. Computergestützte Beweisassistenzsysteme wurden zu genau diesem Zweck intensiv erforscht. Sie sollen, der Idealvorstellung nach, dem Menschen bei der Beweisführung assistieren, indem die eingegebenen Beweisschritte verifiziert oder sogar in einfachen Fällen vom System eigenständig ausgefüllt werden. Generell können Beweisassistenzsysteme in zwei Kategorien eingeordnet werden: Interaktive Theorembeweiser und Automatische Theorembeweiser.

2.1 Beweisinteraktion mit Isabelle

Interaktive Theorembeweiser gehen für die Zeit der Benutzung einen andauernden Dialog mit dem Benutzer ein, in dem dieser beim Führen eines Beweises bzw. von Beweisschritten durch das System unterstützt wird. Der Beweisprozess läuft dabei wie folgt ab: Die grundlegenden Annahmen (Axiome) und das Beweisproblem (Behauptung) werden in einer meist graphischen Oberfläche in einer speziellen Logik-Syntax eingegeben. Im

anschließenden Beweisprozess kann der Beweiser dann beispielsweise dabei helfen, das Ziel in kleine Schritte zu unterteilen, offene (Unter-)Ziele aufzuzeigen und ausführbare Schlussregeln aufzulisten. Außerdem gibt es die Möglichkeit, sogenannte Beweistaktiken auf ein Ziel auszuführen, die die Beweisführung auf höherem Abstraktionsniveau vorwärts bringen.

Der größte Vorteil des Einsatzes eines solchen Systems im Vergleich zu einem herkömmlichen Beweis ist, dass jeder einzelne Schritt des Beweises vom System verifiziert werden kann bzw. muss. So kann sich der Benutzer nach Abschluss des Beweises sicher sein, dass dieser wirklich korrekt ist. Im anderen Fall bekommt der Benutzer bereits frühzeitig (beim Führen eines Unterbeweises) die Rückmeldung, dass ein bestimmter Beweisschritt nicht korrekt ist oder nicht als korrekt erkannt werden kann.

Isabelle ist ein solcher interaktiver Theorembeweiser [NPW02], der inzwischen weit entwickelt und dank seiner graphischen Oberfläche auch für ein breites Publikum benutzbar ist. Außerdem erlaubt Isabelle den Aufruf von externen automatischen Theorembeweisern (vgl. § 2.2), um ganze Unterziele automatisch zu lösen.

Isabelle besitzt einen kleinen, übersichtlichen inneren Beweiskern (basierend auf dem Kalkül des natürlichen Schließens), der Grundlage aller Beweisverifikation ist. Darüber hinaus kann man in Isabelle aber auch weitere Beweiskalküle darstellen, indem man dessen Regeln durch den Beweiskern herleitet und als neue Regeln anbietet. Dies ermöglicht auch die Definition komplexerer Beweistaktiken. Neben den oben genannten Funktionen eines interaktiven Beweisers erlaubt Isabelle auch das Generieren von verifizierten PDF-Dokumenten. So ist es möglich, eine Publikation vollständig in Isabelle anzufertigen. Dies birgt den immensen Vorteil, dass damit alle Beweise im Dokument als korrekt nachgewiesen sind. Weitere praktische Stärken von Isabelle als Beweissystem werden in unserem Fallbeispiel in Abschnitt 4 beschrieben.

2.2 Beweisautomatisierung mit Leo

Automatische Theorembeweiser unterstützen den Benutzer nicht unmittelbar bei der Beweissuche, sondern versuchen das Problem selbstständig, d. h. ohne menschliche Interaktion, zu lösen. Es werden also ähnlich wie bei einem interaktiven Theorembeweiser das Eingabeproblem bzw. die Grundannahmen formuliert, dann jedoch übernimmt das System vollständig die Kontrolle über die Beweissuche. Als Ergebnis des Beweisprozesses geben

viele der automatischen Theorembeweiser einen vollständigen Beweis zurück, mindestens jedoch eine Meldung über Erfolg oder Fehlschlag der Beweissuche. Diese Beweise sind in der Regel sehr feinkörnig, technisch kompliziert, sehr lang und können insbesondere von Menschen meist nur sehr schwer nachvollzogen werden.

Man kann automatische Theorembeweiser gewinnbringend in eine menschliche Beweissuche integrieren. Dies ist vor allem deshalb sinnvoll, weil automatische Theorembeweiser in der Regel „stärker" sind als vergleichbare Taktiken interaktiver Theorembeweiser (d. h. sie finden mehr Beweise). So kann man einen automatischen Beweiser eigenständig einen Unterbeweis generieren lassen, falls man an einem Punkt angekommen ist, der für einen Menschen trivial erscheint. Auch mit diesem Vorgehen erhält man am Ende des Prozesses einen vollständigen, formal verifizierten Beweis der ursprünglichen Aussage.

Das System LEO-II [B+15] und das neu entstehende System Leo-III [WSB14] sind automatische Theorembeweiser für höherstufige Logik. Die maschinenorientierte Ein- und Ausgabesyntax ist jedoch für Einsteiger sehr unübersichtlich. Komfortabler ist hingegen die Benutzung von Leo aus Isabelle heraus. Aus den gut lesbaren Formeln von Isabelle können im Hintergrund Formeln in der zuvor genannten Syntax erzeugt werden, die dann an Leo übermittelt werden.

Diese Verbindung von automatischen und interaktiven Theorembeweisern bringt viele Vorteile mit sich. Die meisten Beweise, die Informatiker, Mathematiker und Philosophen führen, arbeiten in sehr großen Schritten. Damit sich der Benutzer eines solchen Systems nicht mit kleinschrittigen Beweisen aufhält, kann er das nächste Theorem oder Lemma postulieren und die entstehenden Beweislücken durch einen automatischen Theorembeweiser auffüllen lassen. So entstehen in enger Kooperation einfache, lesbare und trotzdem voll verifizierte Beweise.

3 Einsatz in der Lehre

Wie bereits in der Einleitung angedeutet, können Theorembeweiser beim „*logischen Denken*" (vgl. [Kri10, S. 52]) assistieren, und so einen reflektierten und strukturierten Forschungs- und Lernprozess unterstützen. Im Folgenden beleuchten wir zwei der möglichen Einsatzszenarien von Theorembeweisern in der universitären Lehre: Als Instrument zur Implementierung von dialogischem Lernen, und als Werkzeug zum E-Assessment.

Weitere Anwendungsfälle im Kontext des hybriden Lernens (Blended Learning) oder des E-Learnings sind denkbar.

3.1 Theorembeweiser als Werkzeug des dialogischen Lernens

Dialogisches Lernen ist ein Unterrichtsmodell, welches den Prozess des Lehrens und des Lernens als Dialog strukturiert, in dem alle Beteiligten von den Beiträgen und Ergebnissen aller profitieren. Diese werden im Sinne des von U. Ruf und P. Gallin entwickelten Konzepts [Ruf08] als neues Angebot für weitere Lernprozesse verstanden und nehmen Einfluss auf den weiteren Lernverlauf. Diese Idee basiert unter anderem auf der Grundannahme, dass hohe Unterrichtsqualität vor allem durch gegenseitiges Zuhören und sinnvolle (An-)Erkennung des Lernangebotes und -ergebnisses resultiert. Dabei wird versucht, das Lernangebot an die Nutzung und die Nutzung an das Lernangebot zurückzukoppeln.

Die folgenden vier Instrumente des dialogischen Lernens stehen in iterativer Verkettung von Produktion und Rezeption [RG99]:

1. **Kernidee.** Die Kernidee skizziert ein dem Lernenden unbekanntes Themenfeld, verzichtet bewusst auf unnötige Detailinformationen, und lenkt dabei insbesondere den Fokus auf „*den Witz der Sache*“ [RG99]. Eine Kernidee ist immer ein Einstiegspunkt zum selbstständigen Lernen und fordert den Lernenden auf, sein eigenes Verhältnis zum Inhalt zu klären.
2. **Auftrag.** Ein Auftrag im Kontext des dialogischen Lernens muss für alle erfüllbar sein, um so jedem Begabungsniveau einen persönlich zugeschnittenen Zugang zu gewähren. Dennoch ist der Auftrag so konzipiert, dass er herausfordernd ist und zu relativen Höchstleistungen anspornt. Dies wird ebenfalls durch eine offene Auftragswahl begünstigt, in der die Lernenden ihren eigenen Lösungswegen nachgehen können.
3. **Lernjournal.** In dem Lernjournal wird der chronologische Lernfortschritt des Lernenden qualitativ hochwertig dokumentiert. Das Lernjournal dient auch als Nachweis der intensiven Bearbeitung eines Auftrags.
4. **Rückmeldung.** Die Rückmeldung geht auf den individuellen Lernenden ein und setzt sich intensiv mit erfolgreichen Lernschritten, Fehlschlägen und vielversprechenden Wegen aus dem Lernjournal auseinander. Aus den interessantesten Einträgen bzw. Passagen aus den gesammelten Lernjournalen wird anschließend eine „Autografensammlung“ erstellt, die wiederum als Lehrangebot genutzt wird.

Der Einsatz von Theorembeweisern als Werkzeug der Lehrenden und Lernenden während des Lernprozesses von theoretischen Grundbegriffen der Informatik kann hierbei als gewinnbringender Multiplikator eingesetzt werden. Insbesondere interaktive Theorembeweiser unterstützen die Lernenden bei der Auftragserfüllung, der Formalisierung und Verifikation von verschiedensten Herangehensweisen. Dabei können sowohl einfache als auch anspruchsvolle Modelle von den Lernenden durch die interaktive Beweisassistenz umgesetzt werden.

Des Weiteren dient der Vorgang des Formalisierens im System direkt als entsprechender Eintrag im Lernjournal. Das Lernjournal kann also für entsprechende theoretische Aufträge vollständig innerhalb des Isabelle-Systems geführt werden. Ein großer Vorteil dieses Ansatzes ist es, dass die Ergebnisse damit automatisch formal verifiziert sind. Auch fehlgeschlagene Lösungsversuche können mit Hilfe einer speziellen Annotation in dem formal verifizierten Dokument beibehalten werden. Als Endergebnis eines solchen digitalen Lernjournals kann durch den Einsatz von Isabelle also ein verifiziertes PDF-Dokument stehen, welches einfach an die Lehrenden für eine Rückmeldung gegeben werden kann. Für eine aus der Rückmeldung resultierenden Autografensammlung können die obigen verifizierten Formalisierungen an die Lernenden weitergegeben werden, die dann für weitere Aufträge als Lernangebot wahrgenommen werden können.

Durch die inzwischen weit entwickelte und gut benutzbare Oberfläche von Isabelle gibt es kaum nennenswerte Voraussetzungen für potenzielle Lernende, die mit dem System konfrontiert werden. Ausgehend von kleinen Beispielbeweisen und -formalisierungen können ohne weitere technische Kenntnisse ähnliche und darauf aufbauende Aufträge bearbeitet werden. Gleichzeitig bietet Isabelle eine sehr große Bibliothek an bereits vorhandenen Beweisformalisierungen und entsprechenden fortgeschrittenen Bedienungs- und Beweistechniken an, die sehr ambitionierte Formalisierungsprojekte erlauben, jedoch für die grundlegende Bedienung nicht notwendig sind. Hier können also besonders begabte Lernende schnell in komplexere Projekte einsteigen, ohne dass Lernende mit niedrigerem Begabungsniveau überfordert werden. Am Ende des Einsatzes eines solchen Beweisassistenzsystems in einer Lehrveranstaltung steht immer das Ziel, den Lernenden durch angemessene, vom Beweisassistenzsystem unterstützte Projekte interaktiv an den Lerninhalt heranzuführen und damit die Qualifikationsziele durch praktische Auseinandersetzung zu erreichen.

3.2 Automatische Verifikation als E-Assessment

Automatische Verifikation als Ergebnis des Einsatzes von automatischen und interaktiven Theorembeweisern kann auch losgelöst von dem zuvor beschriebenen Lernarrangement des Lerndialogs sinnvoll und gewinnbringend eingesetzt werden.

Hier reduzieren wir den Einsatz dieser Systeme auf ihre Funktion der Rückmeldung zum Zwecke des E-Assessment. So kann ein System wie Isabelle mittels einer Online-Abgabeplattform als ein Assessment-Werkzeug eingesetzt werden. In unserem Falle benutzen wir eine Kombination aus DOMjudge[2] und Isabelle, um ein entsprechendes Angebot zu erstellen.

DOMjudge ist eine Online-Plattform, ursprünglich entwickelt für Programmierwettbewerbe, die es den Benutzern (hier: den Studierenden) erlaubt, Lösungsversuche zu einer bestimmten Aufgabe hochzuladen. Diese Lösungsversuche sind in unserem Fall formalisierte Beweise in Isabelle-Dokumenten. Nach dem Hochladen durchlaufen diese Beweisabgaben dann automatisiert einen Isabelle-Verifizierungs- und Validierungssprozess (gegen eine vom Lehrenden festgesetzte Ziel-Schnittstelle) und melden alle Probleme zurück an DOMjudge. Sobald die Verifikation beendet ist – typischerweise in weniger als einer Minute – können die Studierenden dann auf der Online-Plattform das Ergebnis der Prüfung nachlesen. War der Verifizierungsschritt erfolgreich, so ist die Abgabe fertig gestellt. Andernfalls bekommen die Studierenden eine Rückmeldung über den Fehlschlag und können weiter ihre Beweisführung verbessern bzw. korrigieren.

Wir sind der Überzeugung, dass die Lernenden auch von dem Einsatz von Theorembeweisern zum Zwecke des E-Assessment stark profitieren können. Im gängigen Stift- und Papier-Abgabeschema von Mathematik- und Informatikübungen im universitären Studium bekommen die Studierenden eine einzige finale Rückmeldung (z. B. von der Tutorin bzw. dem Tutor). In unserem Fall wird in Minutenschnelle eine konstruktive Rückmeldung generiert, die die Lernenden für den weiteren Lern- und Bearbeitungsprozess – auch vor der eigentliche Abgabe – einbeziehen können. Insbesondere in großen Veranstaltungen fehlt oftmals die Zeit, allen Lernenden ein angemessenes und differenziertes Feedback für das jeweilige Lernergebnis

[2] DOMjudge ist eine freie open-source Software und kann unter https://www.domjudge.org/ heruntergeladen werden. Hier kann man ebenfalls eine ausführliche Dokumentation finden.

zu geben. Durch den Einsatz eines solchen E-Assessment-Systems können die Lehrenden auch hierbei stark unterstützt bzw. entlastet werden.

Einer Angliederung an evtl. vorhandene Blended-Learning-Prozesse steht in dieser Konfiguration ebenfalls nichts im Wege.

4 Fallbeispiel: Komputationale Metaphysik

Mit dem Ziel, die interdisziplinäre Logikausbildung voranzutreiben, haben wir ein Lehrvorhaben skizziert, dass die oben genannten Ideen aufgreift und exemplarisch im Themenbereich der computergestützen Analyse von Argumenten der Metaphysik einsetzt.[3] Mit dieser Lehrveranstaltung richten wir uns gleichzeitig an Studierende der Philosophie, Mathematik und Informatik. Ziel ist es, eine fachübergreifende Einführung in verschiedene Logikformalismen mit einer praktisch motivierten Einführung in moderne, computerbasierte Beweisassistenzsysteme zu kombinieren und diese auf anspruchsvolle Themen der Metaphysik anzuwenden.

In der Veranstaltung werden die Studierenden nach einer Einführung in die theoretischen Grundlagen dialogisch an die Beweissysteme heran geführt. In kleinen Gruppen sollen die Studierenden sich graduierlich komplexeren Problemen stellen. Dabei sollen sie gemeinsam ihre Ideen diskutieren und können Isabelle jeder Zeit als Prüfstein verwenden. Rückmeldung über den Erfolg und den Fortschritt in einem Problem kommen somit unmittelbar und nicht erst in der nächsten Übungssitzung. Als Kontrollmedium wollen wir den oben genannten DOMjudge als E-Assessment Werkzeug verwenden. In den Übungssitzungen diskutieren die Studierenden ihre Ideen und Lösungen aus den Kleingruppen und können anschließend Variationen aus den Diskussionen in ihrer Isabelle-Formalisierung ergänzen und analysieren.

Die Lehrveranstaltung soll darin münden – als Höhepunkt der Veranstaltung –, die Studierenden zu befähigen, aktuelle Forschungsresultate zur Verifikation des ontologischen Gottesbeweises (Metaphysik) mit einem Computersystem selbst nachzuvollziehen. Wir sind der Meinung, dass selbst bei solch komplexen Argumentationen Isabelle als Assistenzwerkzeug eine hilfreiche und produktive Ergänzung der Argumentationsführung ist. Die syntaktische Aufbereitung der Beweisargumentation innerhalb von Isabelle unterscheidet sich nur an sehr wenigen Stellen von der ursprünglichen Beweisführung aus der theoretischen Philosophie.

[3] Vgl. Veranstaltungsseite unter http://inf.fu-berlin.de/~lex/lehre/compmeta/.

In Projektarbeiten sollen die Studierenden versuchen, die erlernten Techniken auf weitere, ähnliche Argumente der Metaphysik zu übertragen und anzuwenden. Dabei sollen die vorher erbrachten formalisierten Lernresultate in die Auswahl der konkreten Themen einbezogen werden. Die angesprochenen Probleme und die Themen der studentischen Projekte adressieren hochaktuelle und interessante Forschungsinhalte.

Das Leitmotiv des ontologischen Gottesbeweis eignet sich besonders als roter Faden des Lehrvorhabens: Es ist einerseits ein philosophisch höchst anspruchsvolles und belebtes Thema und führte zu vielen kontroversen Forschungsarbeiten. Andererseits demonstriert es verständlich wichtige Konzepte der Modallogik (einer Erweiterung der klassischen Aussagen- bzw. Prädikatenlogik mit besonderer Bedeutung in der Philosophie), motiviert den Einsatz von Beweisassistenzsystemen aus der Informatik und zeigt gleichermaßen die Relevanz des formalen Ansatzes der maschinengestützten Argumentation.

5 Fazit und Ausblick

Wir haben skizziert, wie interaktive und automatische Theorembeweiser sinnvoll in die universitäre Lehre integriert werden können und dabei den Studierenden den Zugang zu herausfordernden Bereichen der Logik (und angrenzender Gebiete) erleichtern. Dabei können die Beweiser in ein Konzept dialogischen Lernens integriert werden und unterstützen dessen erfolgreiche Implementierung. Die Studierenden werden dabei von den Systemen durch Hinweise und sofortige Rückmeldungen unterstützt. Die automatische Überprüfung des Beweisziels erlaubt dabei, alle denkbaren Lösungswege zu unterstützen, da nur die logische Konsequenz als Zielsetzung, nicht aber der Beweisweg geprüft wird. Gleichzeitig haben wir illustriert, wie man diese Konzepte in einem Lehrprojekt im Bereich der theoretischen Philosophie anwenden kann. Dieses Lehrprojekt greift Themen der Metaphysik auf, um Studierende aus der Informatik, Mathematik und Philosophie gleichermaßen in den Bereich der formalen Logik einzuführen. Gleichzeitig profitiert das Projekt aus den Synergien der unterschiedlichen Fachausbildungen der Studierenden im interdisziplinären Dialog.

Nach Beendigung des skizzierten Lehrprojekts wird evaluiert werden, in wie weit sich der Einsatz von Theorembeweisern auf das Verständnis und den Umgang mit formalen Beweisen ausgewirkt hat. Insbesondere sind wir daran interessiert, ob die Studierenden nach Absolvieren des Kurses in der

Lage sind, an Forschungsthemen im Bereich der Komputationalen Metaphysik zu arbeiten und somit also Anschluss an die aktuelle Forschung erlangen konnten. Neben diesem forschungsorientierten Konzept ist es ebenso sinnvoll zu untersuchen, wie sich der Einsatz von Theorembeweisern in Grundlagenmodulen zum Thema Logik und theoretischer Informatik auf den Lernerfolg der Studierenden auswirkt.

Literatur

[AH89] K. Appel, W. Haken: Every Planar Map Is Four Colorable. Amer. Mathematical Society, 1989.

[Ben11] C. Benzmüller: Combining and Automating Classical and Non-Classical Logics in Classical Higher-Order Logic. Annals of Mathematics and Artificial Intelligence, 62(1-2):103–128, 2011.

[B+15] C. Benzmüller et al.: The Higher-Order Prover LEO-II. Journal of Automated Reasoning, 55(4):389–404, 2015.

[Chu40] A. Church: A formulation of the simple theory of types. Journal of Symbolic Logic, 5(2), 1940.

[H+15] T. C. Hales et al.: A formal proof of the Kepler conjecture. CoRR, abs/1501.02155, 2015.

[Hen50] L. Henkin: Completeness in the Theory of Types. Journal of Symbolic Logic, 15(2):81–91, 1950.

[Kru10] O. Kruse: Kritisches Denken im Zeichen Bolognas: Rhetorik und Realität. Neue Impulse in der Hochschuldidaktik: Sprach- und Literaturwissenschaften, 1991.

[NPW02] T. Nipkow, L. C. Paulson, M. Wenzel: Isabelle/HOL – A Proof Assistant for Higher-Order Logic. LNCS 2283, Springer, 2002.

[RG99] U. Ruf, P. Gallin: Dialogisches Lernen in Sprache und Mathematik ½. Kallmeyer, 1999.

[Rob65] J. A. Robinson: A Machine-Oriented Logic Based on the Resolution Principle. Journal of the ACM, 12(1):23–41, 1965.

[Ruf08] U. Ruf: Das Dialogische Lernmodell auf dem Hintergrund wissenschaftlicher Theorien und Befunde – Besser lernen im Dialog. Dialogisches Lernen in der Unterrichtspraxis, 2008.

[WSB14] M. Wisniewski, A. Steen, C. Benzmüller: The Leo-III Project. Joint Automated Reasoning Workshop and Deduktionstreffen, 2014.

Short Papers

Kooperative und kompetenzorientierte Übungen in der Softwaretechnik

Kai Gebhardt

Lehrstuhl für Softwaretechnik
Friedrich-Schiller-Universität Jena
07745 Jena
Email: kai.gebhardt@uni-jena.de

Zusammenfassung: Die Unterrichtsmethode Stationsarbeit kann verwendet werden, um Individualisierung und Differenzierung im Lernprozess zu ermöglichen. Dieser Beitrag schlägt Aufgabenformate vor, die in einer Stationsarbeit über das Klassendiagramm aus der Unified Modeling Language verwendet werden können. Die Aufgabenformate wurden bereits mit Studierenden erprobt.

1 Motivation

Moderne Informationssysteme werden in der Regel durch einen objektorientierten Softwareentwicklungsprozess erstellt. In der Praxis hat sich die Unified Modeling Language (UML) durchgesetzt, um ein Softwaresystem zunächst konzeptionell zu entwerfen. Um die statische Struktur des Systems zu beschreiben, wird das Klassendiagramm verwendet. Es beschreibt welche Daten verarbeitet werden sollen, wie diese miteinander in Beziehung stehen und welche Operationen mit den Daten zulässig sind. Dobing und Parson haben in einer Studie ermittelt, dass ein Großteil der befragten Analysten die Notation nur unzureichend beherrscht [DP06]. In Kapitel 2 werden Kompetenzen definiert, die im Umgang mit Klassendiagrammen erforderlich sind, und passende Aufgaben vorgestellt, die beim Erwerb der Kompetenzen unterstützen könnten. In Kapitel 3 folgt eine Evaluation und ein Ausblick auf weitere Forschungsarbeiten.

2 Aufgabenkonstruktion

Bei der Konstruktion von Aufgaben für Analyse-Klassendiagramme ist zu berücksichtigen, welche Kompetenzen die Studierenden erwerben sollen. Daher ist eine Orientierung an einem Kompetenzmodell für informatisches Modellieren sinnvoll. Aus dem vom DFG-geförderten Projekt MoKoM ist ein solches Modell mit zugehörigen Messinstrumenten entstanden und wurde in der Dissertation von Rhode vorerprobt [R13]. Die Aufgabenvorschläge werden sich auf die folgenden Kompetenzkomponenten des Modells beziehen:

> *„K1.3.2.3.2 Die Lernenden sind in der Lage […] von gleichartigen Objekten Klassen (im Hinblick auf das Klassendiagramm) abzuleiten.[…]*
> *K1.3.2.3.4 Die Lernenden sind in der Lage Analyse-Klassendiagramme zu entwickeln […], sie können Klassen inklusive Attribute und Methoden definieren, Assoziationen festlegen und sinnvolle Vererbungsstrukturen entwickeln.“ [R13, S. 139].*

Um jeden einzelnen Lerner stärker einzubeziehen und individuell zu fördern, bieten sich Unterrichtsmethoden aus der Freiarbeit an. Eine Möglichkeit besteht im Stationenlernen. Nachfolgend werden Aufgabenformate vorgestellt, welche an einzelnen Stationen angeboten werden könnten.

2.1 UML-Memory

Bei UML-Memory handelt es sich um ein Teamspiel. Die Aufgabe der Spieler besteht darin, Kartenpaare zu finden, die inhaltlich zueinander passen. Anders als beim traditionellen Memory sind alle Karten zu Beginn bereits aufgedeckt. Für UML-Memory sind vier Spielkartentypen entwickelt worden. Auf der Klassenkarte ist ein Klassendiagramm abgebildet. Eine Objektkarte zeigt ein Objektdiagramm, während eine Textkarte eine Aussage enthält. Pro Spiel gibt es eine Papierkorbkarte. Nur Karten verschiedenen Typs können ein Paar bilden. Ein Klassendiagramm passt zu einem Objektdiagramm, falls das Objektdiagramm eine gültige Instanziierung zum Klassendiagramm zeigt. Analog passt eine Textkarte zu einer Klassen- oder Objektdiagrammkarte, wenn die Aussage zu dem dargestellten Domänenmodell passt. Falls für eine Karte keine Partnerkarte mehr auf dem Spielfeld existiert, ist der Papierkorb eine gültige Wahl.

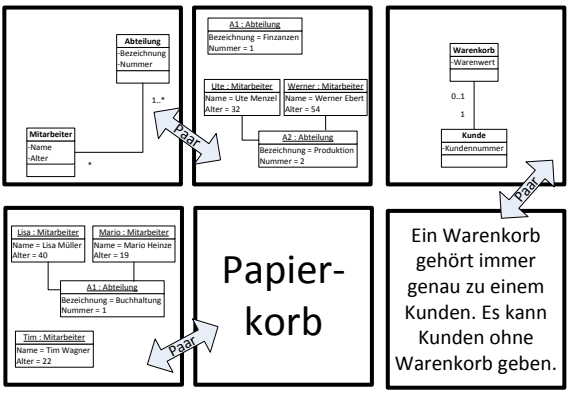


Abb. 1: Spielkarten aus dem UML-Memory

Der Papierkorb kann Partner mehrerer Karten sein und bleibt daher bei erfolgreicher Zuordnung auf dem Spielfeld erhalten. Abb. 1 zeigt exemplarisch sechs der derzeit 36 entwickelten Spielkarten und deren korrekte Zuordnung. Die ersten zwei Karten in Reihe eins und die erste Karte in der zweiten Reihe könnten zueinander passen. Um zu erkennen, dass die untere Karte nicht passt, ist es erforderlich, die richtige Positionierung der Multiplizitäten zu erkennen. Dieses Aufgabenformat adressiert vorwiegend die Kompetenzkomponente K1.3.2.3.2, jedoch sind auch Teilaspekte aus K1.3.2.3.4 ansprechbar, wie das Beispiel zeigt, da Wissen über Assoziationen abgerufen wird. Ähnliche Aufgaben lassen sich für Vererbungsstrukturen konstruieren.

2.2 UML-Krimi

Hierbei handelt es sich ebenfalls um ein Teamspiel. Jeder der vier Spieler pro Gruppe bekommt einen Text, der eine Anwendungsdomäne für ein Softwaresystem beschreibt. Drei Spieler haben den selben Text, während in dem vierten Text leicht abweichende Anforderungen an das Softwaresystem auftauchen. Beispielsweise soll für eine Schule ein AG-Verwaltungssystem entwickelt werden, wobei ein Schüler mehrere AGs belegen kann. In dem veränderten Text darf ein Schüler nur genau eine AG belegen. Ziel des Spieles ist es, den nicht passenden Text herauszufinden, ohne dass sich die Spieler ihre Texte zeigen oder vorlesen dürfen. Hierfür sollen sie enthaltene Anforderungen in einem UML-Klassendiagramm abbilden und dann ihre

Diagramme vergleichen. Damit wird hier eine kooperative Lernform umgesetzt. Nur wenn jeder Teilnehmer ein Klassendiagramm zeichnet, kann eine Gesamtlösung erarbeitet werden. Unterschiede in den Diagrammen können entweder Zeichen einer Lösung sein oder dass ein Teilnehmer noch nicht sicher im Umgang mit der UML ist. Durch die Diskussionen über die Unterschiede haben die anderen Teilnehmer die Gelegenheit ihrem Gruppenmitglied den Sachverhalt zu erklären. Damit wird das Wissen innerhalb der Teilnehmer transferiert. Die adressierte Kompetenzkomponente dieses Aufgabenformates ist K1.3.2.3.4.

4 Evaluation und Ausblick

Bei der Testgruppe (n = 10) handelt es sich um Bachelorstudierende aus der Informatik, Angewandten Informatik und den Wirtschaftswissenschaften im vierten bis siebten Fachsemester. Die Teilnehmer sind zwischen 20 und 30 Jahre alt. Für das Memory wurden im Mittel 20 Minuten und für einen Krimi-Fall 10 Minuten benötigt. Die anschließende Befragung soll die intrinsische Motivation der Teilnehmer in beiden Aufgabenformaten ermitteln, da diese direkten Einfluss auf den Lernerfolg hat. Um den Befragungsaufwand hierfür in einem akzeptablen Rahmen zu halten, wurde eine Kurzskala zur Erfassung genutzt [W09]. Zur Erfassung der Antworten wurde eine Likert-Skala mit vier Antwortmodi verwendet. Diese wurden von 1 (stimme nicht zu) bis 4 (stimme voll zu) kodiert. Die mittlere intrinsische Motivation besitzt bei UML-Memory einen Score von 3 und bei UML-Krimi einen Score von 3,2. Die Ergebnisse sprechen dafür die Aufgabenformate in weiteren Übungen zu verwenden und in einer größeren Studie zu evaluieren.

Literatur

[DP06] Dobing, B.; Parsons J.: How UML is used. Communications of the ACM – Two decades of the language action perspective, Volume 49 Issue 5, 109–113, 2006.

[R13] Rhode, T.: Entwicklung und Erprobung eines Instrumentes zur Messung informatischer Modellierungskompetenz im fachdidaktischen Kontext, Universität Paderborn, Diss., 2013.

[W09] Wilde, B. et. al.: Überprüfung einer Kurzskala intrinsischer Motivation (KIM). Zeitschrift für Didaktik der Naturwissenschaften, Jg. 15, S. 31–45, 2009.

Synergieeffekte zwischen Fach- und Lehramtsstudierenden in Softwarepraktika

Matthias Ehlenz, Nadine Bergner, Ulrik Schroeder

Lehr- und Forschungsgebiet Informatik 9, RWTH Aachen
52074 Aachen
Web: learntech.rwth-aachen.de
Email: <Nachname>@cs.rwth-aachen.de

Zusammenfassung: Dieser Beitrag diskutiert die Konzeption eines Software-Projektpraktikums im Bereich E-Learning, welches Lehramts- und Fachstudierenden der Informatik ermöglicht, voneinander zu profitieren und praxisrelevante Ergebnisse generiert. Vorbereitungen, Organisation und Durchführung werden vorgestellt und diskutiert. Den Abschluss bildet ein Ausblick auf die Fortführung des Konzepts und den Ausbau des Forschungsgebietes.

1 Motivation

Das Konzept des Software-Projektpraktikums (SPP), welches im Wintersemester 2015/16 erstmalig durchgeführt wurde und hier diskutiert wird, versucht die Synergieeffekte zwischen Studierenden des Faches und des Lehramts Informatik zu identifizieren und nach Möglichkeit systematisch zu nutzen.

2 Zielsetzung und Rahmenvorgaben

Ziel des SPP war die Entwicklung von Lernspielen für Multitouchgeräte, mit denen informatische Inhalte an Grundschulkinder vermittelt werden, wobei die unterschiedlichen Vorkenntnisse beider Zielgruppen (breiteres Fachwissen bei den Fachstudierenden und didaktische Vorkenntnisse bei den Lehramtsstudierenden) verknüpft werden. Am Pilotdurchgang nahmen fünf Lehramts- und sieben Fachstudierende teil. Unter den vier Projektgruppen mit

jeweils drei Studierenden waren schließlich zwei gemischte sowie je eine reine Lehramts- und Bachelor-Gruppe.

Das übergeordnete **Ziel** des SPP war die Zusammenführung von E-Learning, Web-Technologien und Fachdidaktik. Praktisch betrachtet bestand das Ziel in der digitalen Aufbereitung einzelner Stationen aus dem Modul „Zauberschule Informatik“ des Schülerlabors Informatik InfoSphere[1], welches sich an SchülerInnen aus der Grundschule richtet und in einem Stationenlernen spielerisch grundlegende Konzepte der Informatik vermittelt.

Technische Rahmenvorgaben ergaben sich durch die 27 Zoll großen Multitouch-Displays, die bereits im Schülerlabor im Einsatz waren. Insbesondere die gleichberechtigte Eingabemöglichkeit von bis zu vier wie auch die intuitive Bedienbarkeit begründeten die Wahl der Technik. Mit Hinblick auf die Programmiersprache fiel die Entscheidung auf HTML5 und JavaScript. Diese ermöglichen jedem, den Quellcode einzusehen und zu modifizieren. Für die Grafikdarstellung im HTML5-Canvas-Element wurde die Verwendung der freien JavaScript-Suite CreateJS verbindlich festgelegt.

3 Vorgehen

Basierend auf den Herausforderungen durch eine neue Programmiersprache, neue Inhalte und didaktische Aspekte fand die Veranstaltung mit einem hohen Präsenzanteil betreut durch zwei wissenschaftliche MitarbeiterInnen des Lehrstuhls statt. Neben einem wöchentlichen Termin über 90 Minuten wurden jeder Gruppe Einzeltermine ermöglicht.

3.1 Struktur der Präsenztermine und Dokumentation

Die Gesamtgruppensitzungen dienten dem zentralen Input durch die BetreuerInnen sowie der wechselseitigen Reflexion des Fortschrittes der Einzelgruppen. Die Struktur einzelner Schlüsselsitzungen wird im Folgenden kurz erläutert: In der **ersten Sitzung** wurden die Art der technischen Zusammenarbeit vorgestellt und anhand von Minimalbeispielen ein Einblick in die Verwendung der eingesetzten Technologien gegeben. Als Aufgabe wurde allen Gruppen die Konzeption eines „Prototypen“ aufgetragen, mit dem sie in der **zweiten Sitzung** jeweils ihre Ideen zur Gestaltung der Benutzerführung vorstellten. Zum Zeitpunkt der **dritten Sitzung** hatten sich

[1] http://schuelerlabor.informatik.rwth-aachen.de/.

alle Gruppen bereits intensiv mit den Minimalbeispielen auseinandergesetzt. Es wurden JavaScript-spezifische Eigenheiten veranschaulicht, um resultierende Probleme zu vermeiden sowie ein Überblick über Best-Practices in der Modularisierung größerer JavaScript-Projekte gegeben. Die **vierte Sitzung** beschäftigte sich mit der inhaltlichen Aufbereitung und didaktischen Konzeption der Spiele. In den **Folgesitzungen** wurde zunächst der aktuelle Entwicklungsstand durch jede Gruppe selbst präsentiert. Dabei wurden aktuelle Herausforderungen betont und Feedback durch die Mitglieder der anderen Gruppen formuliert.

Im Verlauf des SPP wurde der Dokumentation ein hoher Stellenwert zugeschrieben. Bei der begleitenden Dokumentation standen zwei Aspekte im Fokus. Zum einen sollte gleichmäßig rotierend jeweils ein Gruppenmitglied die **Wochenverantwortung** über den Entwicklungsprozess übernehmen. In der Verantwortlichkeit lagen die Festlegung von Wochenzielen und die Gliederung in Teilaufgaben. Zum anderen sollte bei der Bearbeitung dieser Teilaufgaben konsequent ein **Vier-Augen-Prinzip** umgesetzt werden. Ein Gruppenmitglied sollte die Erfüllung der Teilaufgabe übernehmen, ein weiteres die Ergebnisse kontrollieren und testen.

3.2 Testläufe

Neben den kontinuierlichen Rückmeldungen in der Entwicklungsphase wurden die Lernspiele in **drei Iterationen** getestet. Der erste Schritt war ein *Peer-Review*-Verfahren zur Semestermitte, anschließend testeten *Schülerpraktikanten und Auszubildende* die Spiele. Nach der Überarbeitung endete das SPP mit einem finalen Testlauf mit SchülerInnen der intendierten Zielgruppe. Die Kinder testeten jedes Spiel unter Beobachtung der Studierenden.

4 Arbeitsergebnisse

In der finalen Abgabe der Studierenden wurde neben dem kommentierten Code der Lernspiele eine fünfteilige Dokumentation erwartet. Bei der **Arbeitsprozessdokumentation** handelt es sich um jenes Dokument, welches die Studierenden im Verlauf des Entwicklungsprozesses kontinuierlich gepflegt haben. Sie stellt die Grundlage für die Individualbewertung der Studienleistungen dar. Die **Entwicklerdokumentation** dient der langfristigen Verwendbarkeit der Praktikumsergebnisse. Hierin wird der Kontrollfluss grafisch dargestellt, Schlüsselmethoden in ihrer Funktionalität erläutert und

Möglichkeiten zur Anpassung des Spielverhaltens und der Darstellung dokumentiert. Die **Endkundendokumentation** richtet sich an Lehrkräfte, die die Lernspiele einsetzen möchten, und zeigt Besonderheiten auf, die im Einsatz zu bedenken sind. In der **didaktischen Erläuterung** stellen die Studierenden die informatischen Lernziele der Spiele dar. Zudem erfolgt hier eine Begründung didaktischer Entscheidungen. Die **Testlaufdokumentation** dient dazu, die Testlauf-Beobachtungen zu reflektieren. Hier sollten Schlussfolgerungen für die weitere Entwicklung gezogen werden.

Im Endergebnis erfüllen die entstandenen Lernspiele[2] die Erwartungen, sie sind (von der intendierten Zielgruppe) gut bedienbar, laufen fehlerfrei und sind im Code hinreichend gut dokumentiert. Eine inhaltliche Diskussion wurde bereits in [B16] veröffentlicht.

5 Reflexion und Ausblick

Das SPP „E-Learning in der Schule“ war der Versuch, eine praxisrelevante Lehrveranstaltung zu schaffen, von der angehende Lehrkräfte sowie InformatikerInnen im Studium profitieren. Die hier nicht diskutierte Evaluation zeigt: Das Konzept ist ein voller Erfolg und wird künftig fortgeführt. Insbesondere die Weiterverwendung der Ergebnisse in Schüler-Workshops macht, so die Rückmeldung der Studierenden, den Reiz dieses Praktikums aus und hilft auch langfristig die Motivation zu erhalten. Es wird angestrebt, das Konzept künftig mit anderen Fachinhalten und Zielgruppen zu verstetigen.

Darüber hinaus soll der Bereich „Multitouch-Lernspiele“ deutlich ausgebaut werden: In einigen anlaufenden Bachelorarbeiten wird ein Framework für diese Spiele konzipiert und implementiert, so dass künftig eine Fragmentierung verhindert und klare Leitlinien vorgegeben werden. Drei Studierende aus dem SPP führen dort ihre Arbeit fort.

Literatur

[B16] N. Bergner; M. Ehlenz; U. Schroeder: Kooperatives E-Learning am Multitouch-Display für Grundschulkinder. In (Thomas, M.; Weigend, M. Hrsg.): 7. Münsteraner Workshop zur Schulinformatik. Informatik für Kinder, Münster 2016 (in press).

[2] http://schuelerlabor.informatik.rwth-aachen.de/spp/.

Bisher sind in dieser Reihe erschienen:

1 Schwill, A. (Hrsg.): Hochschuldidaktik der Informatik. HDI2008 – 3. Workshop des GI-Fachbereichs Ausbildung und Beruf / Didaktik der Informatik 2008
2009 | ISBN 978-3-940793-75-1

2 Stechert, P.: Fachdidaktische Diskussion von Informatiksystemen und der Kompetenzentwicklung im Informatikunterricht
2009 | ISBN 978-3-86956-024-3

3 Freischlad, S.: Entwicklung und Erprobung des Didaktischen Systems Internetworking im Informatikunterricht
2010 | ISBN 978-3-86956-058-8

4 Engbring, D., Keil, R., Magenheim, J., Selke, H. (Hrsg.): HDI2010 – Tagungsband der 4. Fachtagung zur „Hochschuldidaktik Informatik“
2010 | ISBN 978-3-86956-100-4

5 Forbrig, P., Rick, D., Schmolitzky, A. (Hrsg.): HDI 2012 – Informatik für eine nachhaltige Zukunft
2013 | ISBN 978-3-86956-220-9

6 Diethelm, I., Arndt, J., Dünnebier, M., Syrbe, J. (Eds.): Informatics in Schools – Local Proceedings of the 6th International Conference ISSEP 2013 – Selected Papers
2013 | ISBN 978-3-86956-222-3

7 Brinda, T., Reynolds, N., Romeike, R., Schwill, A. (Eds.): KEYCIT 2014. Key Competencies in Informatics and ICT
2015 | ISBN 978-3-86956-292-6

8 Dörge, C.: Informatische Schlüsselkompetenzen. Konzepte der Informationstechnologie im Sinne einer informatischen Allgemeinbildung
2015 | ISBN 978-3-86956-262-9

9 Forbrig, P., Magenheim, J. (Hrsg.): HDI 2014 – Gestalten von Übergängen. 6. Fachtagung Hochschuldidaktik der Informatik. 15.–16. September 2014, Universität Freiburg
2015 | ISBN 978-3-86956-313-8

10 Schwill, A., Lucke, U. (Hrsg.): Hochschuldidaktik der Informatik. HDI2016 – 7. Fachtagung des GI-Fachbereichs Informatik und Ausbildung/Didaktik der Informatik. 13.–14. September 2016 an der Universität Potsdam
2016 | ISBN 978-3-86956-376-3

In dieser Reihe erscheinen Tagungsbände und ausgewählte Forschungsberichte zu Themen aus der Didaktik der Informatik in Schule und Hochschule.

Die 7. Fachtagung für Hochschuldidaktik, die 2016 erneut mit der DeLFI E-Learning Fachtagung Informatik stattfand, setzte das erfolgreiche Modell einer Tagung fort, die sich mit hochschuldidaktischen Fragen und der Gestaltung von Studiengängen der Informatik beschäftigt.

Thema der Tagung waren alle Fragen, die sich der Vermittlung von Informatikgegenständen im Hochschulbereich widmen.
Dazu gehörten u. a.:

- fachdidaktische Konzepte der Vermittlung einzelner Informatikgegenstände
- methodische Lösungen, wie spezielle Lehr- und Lernformen, Durchführungskonzepte
- empirische Ergebnisse und Vergleichsstudien
- E-Learning-Ansätze, wenn sie ein erkennbares didaktisches Konzept verfolgen
- Studienkonzepte und Curricula, organisatorische Fragen, wie Gewinnung von Studierenden, Studieneingangsphase, Abbrecher.

Die Fachtagung widmete sich ausgewählten Fragestellungen dieses Themenkomplexes, die durch Vorträge ausgewiesener Experten und durch eingereichte Beiträge intensiv behandelt wurden.

ISSN 1868-0844
ISBN 978-3-86956-376-3