Reflexionsförderung bei Lehramtsstudierenden durch den Einsatz von videobasierten Aufgaben

Johannes Fischer[1], Martin Weinert[2]

Abstract: Im Informatikunterricht ist es notwendig, dass Lehrpersonen die mentalen Modelle ihrer Schülerinnen und Schüler diagnostizieren und auf diese reagieren können. Damit sie dies tun können, ist die Reflexionsfähigkeit der Lehrperson von entscheidender Bedeutung. Um diese bereits im Lehramtsstudium zu fördern, entwickeln wir verschiedene videobasierte Aufgabenformate mit dem fachlichen Schwerpunkt *Programmierung*. Im vorliegenden Beitrag stellen wir diese Aufgabenformate zusammen mit ihrem Entwicklungsprozess vor. Wir präsentieren dabei auch Erkenntnisse aus dem Einsatz dieser Aufgaben und den daraus resultierenden Plänen für ihre Weiterentwicklung.

Keywords: Lehrerbildung; videobasierte Aufgaben; Reflexion; Programmierung

1 Einleitung

Ein Problem im Lehramtsstudium ist der Mangel an schulischer Praxis. Lehramtsstudiernde haben während ihres Studiums nur sehr begrenzte Möglichkeiten, realen Unterricht zu beobachten und eigene praktische Erfahrungen zu sammeln. Sie lernen beispielsweise nur aus abstrakter, forschungsorientierter Perspektive, wie Schülerinnen und Schüler lernen zu programmieren. Dadurch steigen sie mit wenig Erfahrung in den Vorbereitungsdienst ein und müssen die Denk- und Lernprozesse ihrer Schülerinnen und Schüler zunächst selbst kennenlernen.

Um diese Lücke zwischen universitärer Bildung und schulischer Praxis zu schließen, bieten sich videobasierte Aufgaben an. Hierzu gibt es bereits verschiedene Ansätze in unterschiedlichen Fächern [CT03; Ka06; SS14] und auch in der Informatik [BLR19]. Während diese Ansätze die Lehrtätigkeiten der Lehrerinnen und Lehrer fokussieren, richten wir den Blick auf die Lernprozesse der Schülerinnen und Schüler.

Dazu entwickeln wir videobasierte Aufgaben, die wir in Seminaren im Lehramtsstudium der Informatik einsetzen. Der fachliche Schwerpunkt liegt dabei auf dem Thema *Programmierung*. Durch die Aufgaben lernen die Studierenden, wie Schülerinnen und Schüler beim Programmieren vorgehen, wie man sie dabei unterstützen kann und welche Rolle die Prozessfokussierung beim Thema *Programmierung* spielt. Die Aufgabenformate sind dabei

[1] TU Dortmund, Fakultät für Informatik, Otto-Hahn-Straße 14, 44227 Dortmund, Deutschland johannes.fischer@cs.tu-dortmund.de
[2] TU Dortmund, Fakultät für Informatik, Otto-Hahn-Straße 14, 44227 Dortmund, Deutschland martin.weinert@cs.tu-dortmund.de

doi:10.18420/infos2021_p228

so gewählt, dass die Teilnehmenden zur Reflexion der Verhaltens der Schülerinnen und Schüler als auch des eigenen angeregt werden, um so im späteren Beruf auch auf neue Situationen angemessen reagieren zu können. In diesem Beitrag stellen wir die Aufgaben vor, die wir für das Seminar *Diagnose und individuelle Förderung* entwickelt haben und diskutieren Verbesserungsmöglichkeiten.

2 Projektstruktur und Begriffe

Das vorgestellte Projekt ist Teil einer fächerübergreifenden Kooperation, die vom Bundesministerium für Bildung und Forschung gefördert[3] wird. Das Ziel ist, eine videobasierte Lernplattform für Lehramtsstudierende verschiedener Fachrichtungen zu entwickeln. Die Lernplattform stellt dazu Funktionen zum Annotieren, Codieren und Schneiden von Unterrichtsvideos bereit. Ein zentraler Aspekt ist dabei die Entwicklung von Aufgabenformaten zur Förderung von Reflexionskompetenzen. Die Aufgaben werden im Teilprojekt Informatik mit einem *design-based research*-Ansatz [GC06] entwickelt und erforscht. Sie basieren auf Videos programmierender Schülerinnen und Schüler, in denen ihr Computerbildschirm zu sehen ist. In einigen Aufnahmen sind zusätzlich Nebenansichten von Eingabegeräten und Notizen zu sehen [FRW20].

Im Bereich der Lehrerbildung herrscht einerseits Einigkeit über die Wichtigkeit von *Reflexionskompetenz*, andererseits ist die Bedeutung des Begriffs *Reflexion* aber immer noch nicht genau geklärt [Cl15]. Basierend auf John Deweys Definition von Reflexion [De10, S.6] assoziieren wir den Begriff mit drei zentralen Fokusse: Diese sind (a) Vorstellungen und (vermeintliches) Wissen, (b) Gründe bzw. Ursachen und (c) abgeleitete Konsequenzen. Bei Fokus (a) wird zusätzlich noch zwischen Beschreibungen und Interpretationen unterschieden. Die Studierenden sollen also beispielsweise beschreiben (a), dass die Schülerinnen und Schüler das Schlüsselwort *String* nacheinander durch *init*, *Init* und *Int* ersetzen. Dies sollen sie als fehlschlagenden Versuch der Deklaration einer Integer-Variable interpretieren (a). Als Ursache (b) könnte fehlende Kenntnis der Schlüsselwörter für Datentypen oder sogar der grundlegenden Bedeutung von Datentypen genannt werden, was zur Folge (c) hätte, dass das Thema *Datentypen* mehr Beachtung im Unterricht finden muss.

3 Lerneinheiten

Im folgenden Abschnitt beschreiben wir die entwickelten Aufgaben und ihre Entstehungshintergründe. Neben den fachspezifischen Zielsetzungen, die zu den Einheiten jeweils beschrieben werden, legen wir bei der Planung der Aufgaben die folgenden allgemeinen Designprinzipien zugrunde:

[3] Förderkennzeichen *16DHB2130*

DP1: *Konstruktivistische Lernendenorientierung*: Basierend auf der Idee, dass Wissen immer auf bestehendem Wissen aufgebaut wird, lassen wir die Studierenden immer ihr bisheriges Wissen aktivieren und versuchen, dieses in die Arbeit zu integrieren.
DP2: *Tragen verschiedener Hüte* [HLR14, S.17]: Das *Tragen verschiedener Hüte* bedeutet, in der Lage zu sein, die Perspektive wechseln und Situationen aus verschiedenen Blickpunkten betrachten zu können. Dies gilt insbesondere für die Perspektive der Schülerinnen und Schüler und die der Lehrpersonen.
DP3: *Active-Learning-Based Teaching Model* [HLR14, S. 19]: Das Modell sieht die Stufen *Trigger, Lernaktivität, Diskussion* und *Zusammenfassung* vor. Die Studierenden werden also zunächst mit einer neuen Fragestellung konfrontiert (*Trigger*), bevor sie das zur Beantwortung notwendige Wissen erarbeiten (*Lernaktivität*), diskutieren und sichern.

Die entwickelten Aufgaben wurden in einem Seminar mit elf Lehramtsstudierenden der Informatik eingesetzt. Teil der Prüfungsleistung dieses Seminares war es, den eigenen Lernfortschritt in den einzelnen Einheiten zu beschreiben. Diese Beschreibungen wurden vom Seminarleiter hinsichtlich der Analyse-Fragestellungen *Wodurch wurden Lernprozesse ausgelöst?* (AQ1), *Wie haben die Studierenden die Aufgaben wahrgenommen?* (AQ2) und *Zu welchen Erkenntnissen sind die Studierenden gekommen?* (AQ3) untersucht. In den folgenden Unterabschnitten präsentieren wir die wichtigsten Antworten auf diese Fragen und diskutieren, welche Konsequenzen wir aus diesen Erkenntnissen ziehen.

3.1 Einheit 1: Produkte vs. Prozesse

In der Einheit *Produkte vs. Prozesse* sollen die Studierenden erkennen, dass die Produkte, die Schülerinnen und Schüler beim Bearbeiten von Aufgaben produzieren, alleine wenig Aussagekraft über ihre Programmierfähigkeiten besitzen. Sie sollen außerdem die Beobachtung der Schülerarbeit als nützliches Instrument zur Ermittlung individueller Vorstellungen und Wissensstände erkennen.

Bei der Tätigkeit des Programmierens handelt es sich um einen Prozess. Sowohl historische als auch neuere Definitionsversuche haben gemeinsam, dass sie von einem Vorgang sprechen [Bl02]. Die Programmierfähigkeiten einer Person dürfen dann aber nicht nur am Endprodukt gemessen werden, da Lernende manchmal auch trotz nicht tragbarer Vorstellungen die zugehörigen Programmieraufgaben lösen können [Lu18; Ma07; MG02]. Für Lehrkräfte ist es daher wichtig, das Verhalten ihrer Schülerinnen und Schüler zu verstehen, damit sie sie bei eventuellen Schwierigkeiten unterstützen können [Lu18].

Um dies zu vermitteln, nutzt die vorgestellte Einheit Videoaufnahmen von zwei Schülern, die gemeinsam eine Diagnoseaufgabe bearbeiten. Die beiden Schüler gehen dabei verschiedenen Ideen nach, scheitern aber zunächst immer wieder. Letztendlich gelingt es ihnen, die Aufgabe zu lösen, was jedoch nicht auf planvollem Vorgehen basiert. Ausschnitte aus diesem Arbeitsprozess bilden das Videomaterial für die vorgestellte Einheit.

Verlauf

Zu Beginn tragen die Studierenden ihr Vorwissen zu Diagnose und Diagnoseaufgaben zusammen und ergänzen dieses durch Literatur [HLP07]. Im nächsten Schritt beschäftigen sich die Studierenden mit einer konkreten Diagnoseaufgabe und bearbeiten diese zunächst selbst. Sie versetzen sich also in die Lernendenperspektive, bevor sie in die Lehrendenperspektive wechseln und überlegen, welches Wissen und Können notwendig ist, um die Aufgabe zu lösen. Zusätzlich betrachten sie hypothetische Abgaben einer Schülergruppe und bewerten diese. Bei der anschließenden Diskussion stellen sie sich die Frage, ob die betrachteten Abgaben ausreichen, um wirklich auf den Wissensstand der Lernenden zu schließen.

Der nun folgende Schritt stellt den Kern der beschriebenen Einheit dar. In dieser betrachten die Studierenden Ausschnitte aus dem Arbeitsprozess zweier Schüler, die versucht haben, die zuvor untersuchte Diagnoseaufgabe zu lösen. Dabei sind sie aufgefordert zu beschreiben, welches Verhalten die Schüler zeigen, welchen Problemen sie begegnen und wie sie mit diesen umgehen. Außerdem sollen sie mögliche Vorstellungen und Ursachen für das beobachtete Verhalten vorschlagen und überlegen, ob und wie sie ggf. in die Situation eingreifen würden. In einer Diskussion kontrastieren sie ihre Ergebnisse wieder mit denen ihrer Kommilitoninnen und Kommilitonen.

In einem letzten Schritt sollen die Studierenden ihr Verständnis der Situationen weiter vertiefen. Dazu kodieren sie einen Teil der Sequenzen unter Anwendung eines induktiven Kodierverfahrens [Ma14] zunächst alleine. Anschließend erarbeiten sie mit je einer weiteren Person eine Teamlösung. Schließlich erstellen sie in Gruppen von je fünf bis sechs Studierenden Gruppenlösungen, die sie im Plenum vorstellen.

Beobachtungen

Die Videos wurden als nützlich empfunden, da sie es ermöglichten, sich mit der Denkweise der Schülerinnen und Schüler zu beschäftigen, was in Praktika bis dahin nicht möglich gewesen ist. Sie konnten zeigen, wie (Fehl-)Vorstellungen die Arbeitsweise der Schülerinnen und Schüler beeinflussen und so die Probleme der Lernenden ersichtlich machen. Das Betrachten der Arbeitsprozesse hat gezeigt, wie abwegig erscheinende Lösungsversuche aus Lernendensicht durchaus Sinn ergeben können. Die Studierenden kritisierten ferner, dass die Videoanalysen erst durch das Hinzuziehen der Literatur aus der zweiten Einheit (3.2) nützlich wurden und dass das Analysieren mehrerer Szenen hintereinander als kognitiv stark belastend empfunden wurde.

Die Studierenden gaben auch an, dass die häufigen Diskussionsphasen den Lernfortschritt stark beförderten. Der Grund hierfür war, dass in diesen Phasen ersichtlich wurde, dass das Lernendenverhalten unterschiedlich interpretiert werden kann. Dieser Effekt wurde insbesondere bei der letzten Aufgabe beobachtet, in der die Studierenden in immer größeren

Gruppen gemeinsame Codierungen erstellen mussten, beobachtet. Die Aufgabe wurde daher als sehr nützlich empfunden, obwohl das Identifizieren von zu codierenden Stellen besonders herausfordernd war.

Diskussion

Die überwiegend positiven Rückmeldungen zu dieser Einheit sprechen dafür, dass ihre Grundstruktur nicht verändert werden sollte. Die Schwierigkeit, bei der Codieraufgabe Schlüsselmomente zu identifizieren, könnte durch das Ersetzen der induktiven Vorgehensweise durch eine deduktive [Ma14] reduziert werden. Als Kodierschema könnten beispielsweise bekannte Tracing-Strategien [FST05] herangezogen werden. Die Problematik der fehlenden Literaturbasis wird in Abschnitt 4 diskutiert.

3.2 Einheit 2: Vortragsblock

Ziel des Vortragsblockes ist es, den Studierenden eine theoretische Wissensbasis zu vermitteln, damit sie ihre naiven Analysen zu fundierten weiterentwickeln können. Diese Basis besteht aus Wissen über Fehlvorstellungen von Programmieranfängerinnen und -anfängern, über Faktoren, die deren Entstehung begünstigen, sowie über mögliche Maßnahmen, um ihnen entgegenzuwirken. Dadurch werden die mit Reflexion assoziierten Aspekte adressiert. Im ersten Durchgang des Seminares wurde festgestellt, dass die meisten Studierenden von den Vorträgen ihrer Kommilitoninnen und Kommilitonen wenig behalten haben.

Verlauf

Die Studierenden wählen einen Abschnitt aus *Students' Misconceptions and Other Difficulties in Introductory Programming: A Literature Review* [QL17] aus und bereiten einen Vortrag zu diesem vor. Damit die Kommilitoninnen und Kommilitonen mehr vom Vortrag behalten, stellen sie ihnen anschließend ein Handout und drei Multiple-Choice-Fragen als optionalen Selbsttest zur Verfügung.

Beobachtungen

Die Studierenden berichten, dass sie die Vortragsmethode grundsätzlich für geeignet halten, da sie durch das Vorbereiten eines Vortrags ihr eigenes Verständnis verbessert haben und die Inhalte insgesamt anschaulich dargestellt wurden. Ein Studierender bemerkte, dass er einige der im Text angesprochenen Fehlvorstellungen in den Videos wiedererkennen konnte. Gleichzeitig geben andere Studierende an, dass sie von den Vorträge nur wenig in Erinnerung behalten hätten, was sich auch in den Ausarbeitungen zeigt (s. u.).

Diskussion

Das Problem, dass die Studierenden aus den Vorträgen ihrer Kommilitoninnen und Kommilitonen wenig mitnehmen, wurde durch Handouts und Selbsttests nicht gelöst. Daher erscheint eine Neustrukturierung dieser Einheit als sinnvoll. Denkbar wären hierzu eine Postersession, Videovorträge oder ganz andere Methoden, wie z. B. ein Gruppenpuzzle.

3.3 Einheit 3: Erklärvideos

Die Einheit *Erklärvideos* soll den Studierenden die Komplexität des Programmierprozesses bewusst machen. Sie soll außerdem zeigen, dass Videos geeignet sind, um den Lösungsprozess von Programmieraufgaben zu vermitteln und sie somit eine nützliche Unterstützung für den Schulunterricht sein können.

Bennedsen und Caspersen [BC05] postulieren, dass es ein Lernziel von Programmierunterricht sei, eine systematische Vorgehensweise zu vermitteln. Hierzu sei es notwendig, den Lernenden die Vorgehensweise explizit zu zeigen. Die Autoren illustrieren die Notwendigkeit mit einem Zitat von David Gries [Gr74], der einen Programmierkurs mit einem Schrankbau-Kurs vergleicht. Dort wäre es offensichtlich absurd, den Lernenden die Werkzeuge und das finale Produkt zu zeigen und anschließend zu erwarten, dass sie in der Lage wären, einen Schrank zu bauen. Watkins und Hufnagel [WH07] setzten in einem Hochschulkurs zum Thema Programmierung Videoaufnahmen ein, die den Lösungsprozess der Übungsaufgaben zeigten. Sie stellten fest, dass anschließend mehr Studierende eine Verbesserung in ihrer Leistung zeigten und weniger Fragen zu den in den Videos behandelten Themen gestellt wurden.

So entstand die Idee, dass Videos, in denen der Lösungsprozess erklärt wird, auch im Schulunterricht ein nützliches Unterstützungswerkzeug sein könnten. Ein erster Versuch zeigte, dass die Produktion eines solchen Videos die Komplexität des eigenen Vorgehens und der eigenen Gedankengänge bewusst macht. Durch diese Selbstbeobachtung könnte den Studierenden verdeutlicht werden, wie ihnen selbst simpel erscheinende Programmieraufgaben aus Schülersicht eine große Herausforderung darstellen können.

Verlauf

Den Studierenden werden zwei Programmieraufgaben und die zugehörigen Musterlösungen in unterschiedlicher Form präsentiert: Die Lösung der ersten Aufgabe besteht lediglich aus kommentiertem Programmcode, während die Lösung der zweiten zusätzlich ein Video bereitstellte, in dem der Programmierer die Lösung Schritt für Schritt entwickelt und verbal kommentiert. Die Studierenden vergleichen die Lösungen miteinander und überlegen, wie nützlich sie jeweils für Lernende sind. Zusätzlich überlegen sie, welche Informationen das

Video enthält, die in der rein textbasierten Lösung fehlen. Anschließend lernen sie eine mögliche Vorgehensweise beim Programmieren kennen [CK06] und vergleichen diese mit dem zuvor gesehenen Video.

Daraufhin folgt die zentrale Aufgabe dieser Einheit, die daraus besteht, selbst ein Erklärvideo anzufertigen. Dazu wählen die Studierenden eine geeignete Programmieraufgabe aus und zeichnen ihren Lösungsprozess zusammen mit verbalen Kommentaren auf. Anschließend geben sich die Studierenden in einem Peer-Feedback-Verfahren gegenseitig Rückmeldungen zu den Videos. Schließlich diskutieren sie die Möglichkeiten und Grenzen solcher Videos.

Beobachtungen

Einige Studierende stellten fest, dass der Lösungsprozess in Erklärvideos besser nachvollziehbar ist als allein auf Textbasis. Sie halten Erklärvideos für gut geeignet, um strukturierte Programmierung und bestimmte Verhaltensweisen zu vermitteln. Dabei bezogen sie sich speziell die Möglichkeit, die Nicht-Linearität des Programmierprozesses verdeutlichen zu können. Ebenso stellten sie fest, dass durch die Videos deutlich wird, dass der Lösungsprozess nicht fehlerfrei verlaufen muss. Dies kann als Gelegenheit genutzt werden, um beispielsweise den Umgang mit Fehlermeldungen und die Fehlersuche zu demonstrieren. Die Studierenden wiesen außerdem auf die Gefahr hin, dass auch schlechtes Verhalten durch Videos vermittelt werden kann.

Die Meinung der Studierenden zur Nützlichkeit von Erklärvideos ist nicht einheitlich. So gab ein Studierender an, die Erstellung eigener Erklärvideos als nicht sinnvoll anzusehen, da der Nutzen den Aufwand nicht rechtfertigt. Ein anderer Studierender wiederum berichtete, dass sich seine Meinung durch die Einheit völlig ins Gegenteil verändert hat. Während er zuvor keinen Nutzen gesehen hatte, sah er Erklärvideos nach der Einheit als nützliches Werkzeug für den Unterricht an.

Diskussion

Die Einheit scheint nützlich gewesen zu sein, da mehrere Studierende davon berichten, ihre Meinung geändert und überdacht zu haben. Einige bewerten sie als nützlichste Einheit des Seminars. Im nächsten Durchgang könnten Videos analysiert werden, in denen Schülerinnen und Schüler ihren Programmcode erklären, um den Einsatz als Diagnoseinstrument vorzuführen. Diese werden momentan in Kooperation mit einer Realschule entwickelt.

3.4 Einheit 4: Interviewanalysen

In dieser Einheit sollen die Studierenden Einzelgespräche mit Schülerinnen und Schülern sowohl als Diagnose-, als auch als Unterstützungsinstrument kennenlernen. Sie sollen dabei

üben, die mentalen Modelle der Lernenden aus ihren Aussagen zu rekonstruieren. Dazu analysieren sie Gespräche sowohl in text- als auch videobasierter Form.

Verlauf

Zunächst wird die Aufgabe *Interview with David* [HLR14] im Plenum unter Einsatz der Think-Pair-Share-Methode durchgeführt. Dabei lösen die Studierenden zunächst selbst eine Programmieraufgabe, bevor sie den Lösungsvorschlag des Schülers *David* analysieren. Anschließen untersuchen sie zwei Gesprächspassagen und versuchen, sowohl Davids mentale Modelle zu rekonstruieren, als auch Fragen vorzuschlagen, die ihm helfen sollen, seine Vorstellungen weiterzuentwickeln.

Bei der nächsten Aufgabe untersuchen sie ein Coachingvideo, in dem ein Lernender Programmieraufgaben bearbeitet und dabei Hilfestellungen von einem Coach bekommt. Die Studierenden suchen nach Szenen, in denen sie dem Verhalten des Coaches lernförderliches oder lernhinderliches Verhalten zuschreiben. Anschließend diskutieren sie die gewählten Szenen im Plenum.

Beobachtungen

Es wurde berichtet, dass die Diskussion von Schüleraussagen wichtig ist, da sie oft das einzige sind, das Lehrpersonen für die Diagnose nutzen können. Diesen Diskussionen und den Analysen wird zugeschrieben, zu einem besseren Verständnis der Denkprozesse geführt zu haben. Die Studierenden berichten, dass das Interviewvideo gezeigt hat, dass ein Coach auf sehr viele Aspekte achten muss, die Verständnisprobleme hervorrufen können. Ein Studierender berichtet, dass es dadurch manchmal nur schwer oder gar nicht möglich ist, ad-hoc-Diagnosen durchzuführen. Ein weiterer Studierender kritisierte die Länge des verwendeten Videos (ca. 1h).

Diskussion

Insgesamt scheint die Einheit zielführend gewesen zu sein. Dem zuletzt genannten Problem könnte begegnet werden, indem das Video in kurze Sequenzen geteilt wird, um gezieltere Analysen zu ermöglichen.

4 Ausblick

Es wurde deutlich, dass die entwickelten Aufgaben Wirkung gezeigt haben. Die Studierenden gaben an, die Versuche, Lern- und Denkprozesse nachzuvollziehen, als sehr nützlich erlebt

zu haben und halten sie aufgrund fehlender Praxiserfahrungen für wichtig. Wegen der COVID-19-Pandemie wurde das Seminar auf ein Onlineformat umgestellt. Dadurch wurden Kollaborationswerkzeuge stärker eingesetzt, was die Studierenden sehr positiv bewerteten.

Insgesamt hat der Einsatz des Mediums *Video* gezeigt, dass es die gestellten Erwartungen durchaus erfüllen kann, wenn es richtig eingesetzt wird. Daher werden wir in Zukunft versuchen, weitere Designprinzipien zu finden, die einen effektiven Einsatz von Videos in der Lehrkräftebildung unterstützen. Außerdem werden wir die vorgestellten Aufgabenformate weiter- und zusätzliche Einheiten neuentwickeln. Dazu werden wir zusätzliche Videos produzieren, analysieren und so aufbereiten, dass sie zur Verbesserung der Lehrveranstaltungen im Lehramtsstudium und langfristig auch in der 2. Ausbildungsphase beitragen.

Literatur

[BC05] Bennedsen, J.; Caspersen, M. E.: Revealing the Programming Process. In: Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education. SIGCSE '05, S. 186–190, 2005.

[Bl02] Blackwell, A. F.: What is programming? In (Kuljis, J.; Baldwin, L.; Scoble, R., Hrsg.): Proc. PPIG 14. 2002.

[BLR19] Broneak, C.; Lucarelli, C.; Rosato, J.: Exploring the Use of Video Reflection as a Professional Development Tool. In: Proceedings of the 2019 ACM Conference on International Computing Education Research. ICER '19, Association for Computing Machinery, Toronto ON, Canada, S. 293, 2019.

[CK06] Caspersen, M. E.; Kölling, M.: A Novice's Process of Object-Oriented Programming. In: Companion to the 21st ACM SIGPLAN Symposium on Object-Oriented Programming Systems, Languages, and Applications. OOPSLA '06, S. 892–900, 2006.

[Cl15] Clarà, M.: What is Reflexion? Looking for Clarity in an Ambiguous Notion. Journal of Teacher Education 66/3, S. 261–271, 2015.

[CT03] Cannings, T.; Talley, S.: Bridging the Gap between Theory and Practice in Preservice Education: The Use of Video Case Studies. In: Proceedings of the 3.1 and 3.3 Working Groups Conference on International Federation for Information Processing: ICT and the Teacher of the Future. Bd. 23. CRPIT '03, Australian Computer Society, Inc., Melbourne, Australia, S. 17–20, 2003.

[De10] Dewey, J.: How we think. D. C. Heath & Co., Boston, 1910.

[FRW20] Fischer, J.; Romahn, N.; Weinert, M.: Fostering Reflection in CS Teacher Education. A Video-Based Approach to Unveiling, Analyzing and Teaching Novices' Programming Processes. In (Kori, K.; Laanpere, M., Hrsg.): Proceedings of the International Conference on Informatics in School: Situation, Evaluation and Perspectives. Bd. 2755. CEUR Workshop Proceedings, CEUR-WS.org, S. 128–139, 2020.

[FST05] Fitzgerald, S.; Simon, B.; Thomas, L.: Strategies That Students Use to Trace Code: An Analysis Based in Grounded Theory. In: Proceedings of the First International Workshop on Computing Education Research. ICER '05, Association for Computing Machinery, Seattle, WA, USA, S. 69–80, 2005, ISBN: 1595930434.

[GC06] Gravemeijer, K.; Cobb, P.: Design research from a learning design perspective. Educational design research/, Jan. 2006.

[Gr74] Gries, D.: What Should We Teach in an Introductory Programming Course? In: Proceedings of the Fourth SIGCSE Technical Symposium on Computer Science Education. SIGCSE '74, S. 81–89, 1974, ISBN: 9781450374835.

[HLP07] Hußmann, S.; Leuders, T.; Prediger, S.: Schülerleistungen verstehen - Diagnose im Alltag. PM : Praxis der Mathematik in der Schule 49/, S. 1–9, Jan. 2007.

[HLR14] Hazzan, O.; Lapidot, T.; Ragonis, N.: Guide to Teaching Computer Science: An Activity-Based Approach. Springer, London, 2014.

[Ka06] Kale, U.; Hur, J. W.; Yerasimou, T.; Brush, T.: A Model for Video-Based Virtual Field Experience. In: Proceedings of the 7th International Conference on Learning Sciences. ICLS '06, International Society of the Learning Sciences, Bloomington, Indiana, S. 944–945, 2006.

[Lu18] Luxton-Reilly, A.; Simon; Albluwi, I.; Becker, B. A.; Giannakos, M.; Kumar, A. N.; Ott, L.; Paterson, J.; Scott, M. J.; Sheard, J.; Szabo, C.: Introductory Programming: A Systematic Literature Review. In: Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education. ITiCSE 2018 Companion, ACM, Larnaca, Cyprus, S. 55–106, 2018, ISBN: 978-1-4503-6223-8.

[Ma07] Ma, L.; Ferguson, J.; Roper, M.; Wood, M.: Investigating the viiability of mental models held by novice programmers. In: Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education (SIGCSE'07). ACM, New York, NY, USA, S. 499–503, 2007.

[Ma14] Mayring, P.: Qualitative content analysis: theoretical foundation, basic procedures and software solutions. Klagenfurt, 2014.

[MG02] Madison, S.; Gifford, J.: Modular programming: novice misconceptions. Journal of Research on Technology in Education 34/3, S. 217–229, 2002.

[QL17] Qian, Y.; Lehman, J.: Students' Misconceptionsand Other Difficulties in Introductory Programming: A Literature Review, 2017.

[SS14] Seidel, T.; Stürmer, K.: Modeling and Measuring the Structure of Professional Vision in Preservice Teachers. American Educational Research Journal 51/4, S. 739–771, 2014.

[WH07] Watkins, A.; Hufnagel, E. M.: Video Vignettes: Teaching Computer Programming to the MTV Generation. Decision Sciences Journal of Innovative Education 5/2, S. 391–395, 2007.