Entwicklung eines agilen Frameworks für Projektunterricht mit Design-Based Research

Petra Kastl und Ralf Romeike[1]

Abstract: Fachdidaktische Innovationen stellen Forscher und Praktiker grundsätzlich vor die Herausforderung, Theorie mit Praxis in Einklang zu bringen. Besonders in der Informatik bergen kontinuierliche fachliche Weiterentwicklungen bedeutendes Potential für didaktische und methodische Neuerungen. Der Beitrag skizziert am Beispiel eines agilen Modells für Projekte einen Forschungsprozess, der die Implementierung und Weiterentwicklung des Modells unter Einbeziehung unterrichtspraktischer Expertise begleitet. Zusätzlich werden Erfahrungen aus der ersten Iteration des dem Prozess zugrunde liegenden Design-Based Research-Ansatzes beschrieben.

Keywords: agile Methoden, agile Praktiken, Design-Based Research, Projektunterricht

1 Motivation

In der professionellen Softwareentwicklung hat sich in den letzten Jahren herausgestellt, dass sequentielle Vorgehensmodelle wie das Wasserfallmodell oft zu mangelhafter Qualität, größeren Terminverschiebungen und wenig Kundenzufriedenheit führen – für Entwickler eine frustrierende Situation. Ähnliche Probleme treten auch in Schulprojekten auf, die sich am Wasserfallmodell orientieren: Projekte sind schwer planbar, werden aus Zeitmangel oft nicht mit einem nutzbaren Produkt beendet und lange Modellierungs- und Testphasen sind kaum zu motivieren [HNR07, We05]. In der IT Branche setzt man zur Lösung mehr und mehr auf agile Methoden, also verstärkt auf Kommunikation und Kooperation, Mitarbeitermotivation sowie eine kontinuierliche Produkterstellung in kurzen, iterativen Entwicklungsphasen, die durch eine langfristige Produktvision gelenkt werden. Agile Methoden beruhen entsprechend des agilen Manifests [Be01] auf Werten und Praktiken, die auch für den Schuleinsatz angebracht erscheinen. Auf der WiPSCE 2012 stellten Romeike und Göttel ein auf agilen Methoden basierendes Modell für Projekte im Informatikunterricht (AMoPCE) vor [RG12]. Das Modell stellt für die Praxis einen leicht einsetzbaren Satz agiler Praktiken bereit, der es erlaubt im Unterricht agile, kommunikationsfördernde Prozesse einzusetzen, die schnell nutzbare Teilergebnisse liefern und die das Entwicklerteam ins Zentrum stellen.

Ein Problem bei der theoretischen Entwicklung eines so umfangreichen Modells ist, dass praxisrelevante Aspekte und auftretende Schwierigkeiten nur eingeschränkt vorhergesehen werden können. Um Schwierigkeiten zu vermeiden, wie sie bei der Umsetzung von

[1] Friedrich-Alexander-Universität Erlangen-Nürnberg, Didakitk der Informatik, Martensstr. 3, 91058 Erlangen
petra.kastl@fau.de, ralf.romeike@fau.de

am Wasserfallmodell orientierten, linearen Modellen im Schulunterricht auftreten, soll das theoretisch entwickelte agile Modell in verschiedenen schulpraktischen Kontexten erprobt und mit Hilfe eines Design-Based Research-Ansatzes in einem zyklischen Prozess verfeinert und weiterentwickelt werden. Ziel ist es, die berufsspezifischen Kompetenzen erfahrener Lehrkräfte in den Prozess einzubinden und ihre Beobachtungen bei der Umsetzung, ihre Ideen, Interpretationen und individuellen Anpassungen in jedem Zyklus zu bündeln und zu reflektieren, um die Weiterentwicklung mittels formativer Evaluation empirisch abzusichern. Schrittweise und forschungsgeleitet soll AMoPCE so zu einem praxiserprobten, offenen und flexiblen „Methodenkoffer" weiterentwickelt werden, mit dessen Hilfe Projekte mit einer individuell angepassten, sinnhaften agilen Vorgehensweise erfolgreich durchgeführt werden können. Das vorgestellte Vorgehen und die zugrundeliegenden Überlegungen können als Basis dienen, auch andere Innovationen forschungsgeleitet in der Praxis des Informatikunterrichts zu verankern.

2 Projekte im Informatikunterricht

Projektunterricht beschreibt unabhängig vom Schulfach eine ganzheitliche, offene Lernform, die mit hohem Anteil an Mitbestimmung der Teilnehmer in Phasen abläuft [FS82]. Beschrieben wird er meist über eine Liste von Arbeitsschritten und/oder Merkmalen, wie beispielsweise bei Gudjons oder Frey. Ausgehend von komplexen Aufgaben oder offenen Fragestellungen entwickeln und konkretisieren die Teilnehmer entsprechend ihren Neigungen und Interessen Ideen, bei deren Umsetzung sie selbstorganisiert und zielgerichtet arbeiten und die Verantwortung für das Projekt gemeinsam übernehmen [Gu14]. Sie benötigen dazu vielfältige fachliche, methodische und soziale Fähigkeiten und Fertigkeiten, deren Erwerb, Anwendung und Weiterentwicklung Ziele von Projektunterricht sind [Fr83]. Eine bekannte Schwierigkeit ist, dass Schülerinnen und Schüler (SuS) mit der Komplexität und den Anforderungen eines Projekts oft überfordert sind. Aufgabe der Lehrkraft ist es dann, das Projekt in kleinere Teilprojekte zu gliedern [SS05].

Bei Softwareentwicklungsprojekten im Informatikunterricht liegt eine besondere Situation vor: Da professionelle Softwareentwicklung in Projekten stattfindet, gibt es hierfür seit den 1970iger Jahren wissenschaftlich verankerte Vorgehensmodelle. Will man den Lernenden einen Einblick in die „echte Welt" vermitteln, muss man diese Modelle berücksichtigen. Gleichzeitig stehen sie zum Teil im Widerspruch zu wesentlichen Kriterien und Zielen von Projektunterricht. Daher findet man – historisch bedingt – in der Informatik überwiegend adaptierte Modelle, die versuchen, die Kernideen und Ziele von Projektunterricht bestmöglich mit dem Wasserfallmodell zu verweben [Fr83, SS11]. Unterhält man sich mit Lehrkräften über Projekte und studiert man didaktische Literatur, legt das die Vermutung nahe, dass solche linearen Modelle aufgrund schulischer Anforderungen und Organisationsstrukturen in der Umsetzung häufig zu vielerlei Problemen führen [HNR07, Hu05, MH10, SS11, We05]. SuS sind regelmäßig mit der Komplexität überfordert und können Inhalte und Ressourcen kaum selbständig organisieren bzw. planen. Eine Folge ist, dass sie die Verantwortung für das Projekt nicht gemeinsam

übernehmen können, sodass oft geringe Eigeninitiative und Motivation bei Teilen des Teams beobachtet wird[2]. Gleichzeitig muss die Lehrkraft in hohem Maße motivieren, unterstützen, planen und leiten[3,4]. Aber auch für Lehrkräfte ist die Planung innerhalb der Organisationsstrukturen von Schule schwierig. Unfertige Produkte und schlechte Produktqualität sind die Folge[5,6]. Diese können dann auch nur sehr eingeschränkt von SuS reflektiert und bewertet werden. Eine weitere Schwierigkeit für Lehrer besteht darin, dass das Wasserfallmodell keine konkreten Techniken zur Verfügung stellt, die Kommunikation und Interaktion unterstützen, um soziales Lernen zu ermöglichen.

Theoretische Überlegungen zu agilen Methoden lassen vermuten, dass sie den Spagat zwischen den Zielen professioneller Softwareentwicklung (effiziente Erstellung nützlicher Software von hoher Qualität zur Zufriedenheit des Kunden) und den oben dargestellten Zielen von Schulprojekten müheloser schaffen als lineare Modelle, weist man ihnen doch Eigenschaften wie verstärkte Betonung von Kommunikation und Kooperation sowie individueller Fähigkeiten und Bedürfnisse zu, ebenso die Stärkung der Eigenverantwortung und das Setzen auf motivierte Projektbeteiligte, die von sich aus verantwortungsvoll und couragiert handeln [Be01, Ru04]. Von einem agilen Modell für Schulprojekte könnten demnach Lernende und Lehrende gleichermaßen profitieren zumal eine an das individuelle Projekt angepasste „schlanke“ Vorgehensweise sinnhaft, zeitgemäß und realitätsnah ist. Vereinzelte Untersuchungen in der Vergangenheit bestärken diese Annahmen: Weigend [We05] beschreibt positive erste Erfahrungen bei der Anwendung einzelner agiler Elemente. Meerbaum-Salant und Hazzan [MH10] haben eine Studie durchgeführt, in der sie agile Werte nutzen, um Lehrer bei der Betreuung von Softwareprojekten zu unterstützen. Göttel [Gö12] verwendet einige agile Praktiken in verschiedenen Projekten mit der Intention, soziale Interaktionen in der modernen Softwareentwicklung hervorzuheben und so Lernende ein attraktives Bild der Informatik zu vermitteln.

3 Das agile Framework AMoPCE

Agile Methoden stellen konkrete Vorgehensweisen für die Projektorganisation und Projektarbeit zur Verfügung, die flexibel und nach den individuellen Bedürfnissen eines Projekts ausgewählt werden. Wesentliche innovative Neuerungen gegenüber dem linearen Modell werden im Folgenden kurz dargestellt[7].

[2] „Einer hat dann zu Hause den größten Teil programmiert.“ (Interview mit einem Lehrer)
[3] „In jedem Kurs das Gleiche: Die Leute wollen einfach nicht zuhören, wenn es darum geht, zuerst die Struktur einer Website auf Papier zu überlegen.“ [HNR07]
[4] „Obwohl ich nach den Doppelstunden regelmäßig nass geschwitzt war, haben die Schülerinnen und Schüler vor allem die langen Wartezeiten bemängelt, bis Unterstützung kam.“ (Interview mit einem Lehrer)
[5] „Zum Testen sind wir nicht mehr gekommen.“ (Interview mit einem Lehrer)
[6] „Ein paar Fehler sind noch drin, so dass es noch nicht läuft. Aber zum Fertigstellen hatten wir keine Zeit mehr.“ (Interview mit einem Lehrer)
[7] Eine detaillierte Beschreibung insbesondere auch der Anpassung professioneller Elemente an den Schulkontext durch didaktischen Transposition findet man bei Romeike und Göttel [RG12].

Iteratives Vorgehen und Prototypen: Ausgehend von einer Produktvision werden die gewünschten Funktionalitäten in User Stories aus Nutzersicht beschrieben und priorisiert. In den anschließenden Iterationen werden jeweils einige User Stories mit hoher Priorität ausgewählt und die zugehörigen Tasks formuliert, die aus Entwicklersicht bei der Umsetzung der jeweiligen Story zu erledigen sind. In einem (linearen) Mini-Projekt mit Planungs-, Entwurfs- Implementierungs- und Testphase werden dann die Tasks realisiert. Am Ende eines jeden Mini-Projekts steht ein lauffähiger und getesteter Prototyp, der ausprobiert, vorgestellt und bewertet werden kann (vgl. Abb. 1).

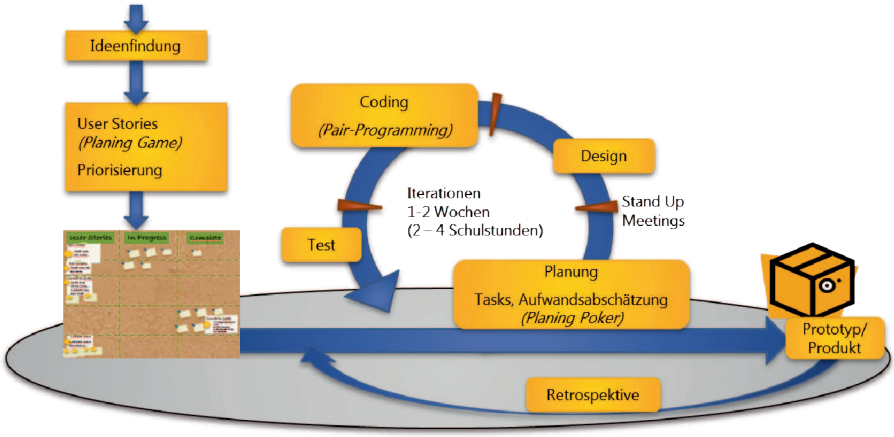

Abb. 1: Ein agiles Framework für Schulprojekte

Solche Mini-Projekte sind wesentlich leichter zu überblicken und sollen den Schülerinnen und Schülern das autonome Planen und Arbeiten erleichtern. Durch die iterative Vorgehensweise können die Projektteams auch mit geringer fachlicher und methodischer Kompetenz beginnen, und Erkenntnisse, die sie durch Reflexion und Peerfeedback gewinnen, in der nächsten Iteration anwenden. Der Abschluss jeder Phase mit einem lauffähigen Prototyp führt zu sichtbaren Ergebnissen, die die Motivation erhalten und erleichtert gleichzeitig die Gesamtplanung durch die Lehrkraft.

Anforderungen als variable Projektgröße: Während in professionellen agilen Projekten in der Regel Zeit, Personal und Produktqualität fixe Größen darstellen, sind Anforderungen flexibel, folgen einer Produktvision und werden durch intensive Zusammenarbeit mit dem Kunden nach und nach konkretisiert. Der Vorgang der Softwareentwicklung wandelt sich: Aus einem stark prozessfokussierten Vorgang wird ein dynamischer und kreativer [FST13], der die Projektbeteiligten zu gestalterischem und verantwortungsvollem Handeln ermutigt. Reichen die Ressourcen nicht aus, werden keine Kompromisse hinsichtlich Qualität (Testen), Zeit (Endtermin) oder mehr Personal eingegangen, sondern niedrig-priorisierte Anforderungen nicht umgesetzt. Diese innovative Herangehensweise erleichtert die Planungsarbeit in Schulprojekten enorm, weil sie Planungsfehler unerfahrener Schüler verzeiht. Darüber hinaus kann die Lehrkraft die Rolle des Kun-

den übernehmen und damit steuernd eingreifen.

Kommunikationsfördernde Praktiken: Einige Praktiken wie Pair-Programming und Stand-Up Meetings, in denen sich die Teammitglieder gegenseitig kurz über Erledigtes, Probleme und Geplantes informieren, fördern die Kommunikation und geben der Lehrkraft einen Einblick in den individuellen Lernfortschritt.

Praktiken, die den Planungs- und Organisationsprozess unterstützen: Durch Praktiken wie das Planning Poker erhalten SuS Handlungsanweisungen, die sie in der schwierigen Phase der Aufwandsabschätzung unterstützen sollen. Regelmäßige Stand-Up Meetings, feste Iterationslängen und Retrospektiven am Ende jeder Iteration strukturieren den Prozess. Das Project Board ist zentraler Informations- und Planungsbereich, an dem der Projektfortschritt visualisiert und begreifbar gemacht wird und auftretende Probleme und Fragen notiert werden können. Es soll Lernenden und Lehrenden einen Überblick über den Stand des Projekts bieten, die Planung der nächsten Schritte erleichtern und helfen, Entscheidungen zu treffen. Im Zusammenspiel sollen diese Praktiken Schülerinnen und Schülern ein selbständiges Arbeiten und Planen ermöglichen und in Konsequenz eine starke Änderung der Lehreraufgaben bewirken.

Die theoretischen Überlegungen zeigen, dass ein agiles Framework Projektarbeit für Lernende und Lehrende im positiven Sinne verändern kann. Neben der begründeten Relevanz und der Konsistenz gilt es in den nächsten Schritten die Praxistauglichkeit sicher zu stellen, um dann – möglichst umfassend und gestützt durch theoretische und empirische Argumente – die Bedingungen zu beschreiben, die zum Gelingen führen. Ein dazu geeignetes Forschungsformat erlaubt es, Lern- und Arbeitsprozesse während des Projektverlaufs bezüglich hinderlicher und fördernder Faktoren zu untersuchen und liefert idealerweise neben dem Erkenntnisgewinn für die Theorie auch ein für die Praxis nützliches Artefakt, das zukünftig eine Verbesserung der Unterrichtspraxis erleichtert.

4 Forschungsgeleitete Adaption und Weiterentwicklung

Als Problem der Unterrichtsentwicklung hat sich gezeigt, dass allein durch theoretische Überlegungen erlangte Erkenntnisse und Modelle oft nur unzureichend zu nachhaltiger Innovation in der Unterrichtspraxis führen [Re05]. Die Gründe hierfür sind vielfältig. So können beispielsweise verschiedenste Probleme bei der Implementierung eines theoretisch erarbeiteten Modells in realen Kontexten auftreten, wie beim Verwenden des adaptierten Wasserfallmodells. Diese können inhärent dem Entwurf innewohnen oder abhängig vom Kontext auftreten. Manchmal fehlt rein theoretisch begründeten Ansätzen auch einfach die Praxisrelevanz. Gleichzeitig ist zeitgemäßer Informatikunterricht auf kontinuierliche didaktische Aufarbeitung der Themen und auf nachhaltige Innovation in der Unterrichtspraxis angewiesen, da Informatik eine dynamische Wissenschaft ist. Beispiele hierfür finden sich in der fortwährenden Entwicklung neuer Software- und Hardwarewerkzeuge zur Verbesserung der Unterrichtspraxis oder der Weiterentwicklung zentraler Unterrichtsgegenstände, wie sie z. B. derzeit beim Thema Datenmanagement, -schutz

und -sicherheit zu beobachten ist. Die agilen Methoden schließlich haben den berufstypischen Prozess der Softwareentwicklung ebenso wie die wissenschaftliche Sicht auf ihn enorm verändert und vermutlich haben sie auch das Potential, Unterrichtsprojekte gewinnbringender zu gestalten. Einer engen Theorie-Praxis-Verschränkung scheint deshalb in der Informatik eine substantielle Bedeutung inne zu wohnen, damit in der Theorie begründete und erarbeitete Entwürfe forschungsgeleitet so weiterentwickelt werden können, dass sie zu nachhaltiger Innovation und konkreter Unterrichtsweiterentwicklung führen.

Ein vielversprechendes Forschungsformat, das hilft generalisiertes Wissen über das Entwerfen und das nachhaltige Implementieren innovativer Unterrichtsmodelle zu entwickeln, ist Design-Based Research (DBR) [De03]. Reed und Guzdial [RG12] halten DBR für die Informatikdidaktik für besonders geeignet, da hiermit Polarisierungen und Diskussionen des „einzigen richtigen Vorgehens“ vermieden werden können und stattdessen Interventionen hinsichtlich der Ursachen erfolgreicher Lernprozesse beschrieben werden. Charakteristische Merkmale des DBR sind die Motivation, Unterrichtspraxis verbessern zu wollen, das Ziel, reale Probleme zu lösen und dabei Erkenntnisse für Theorie und Praxis zu gewinnen, sowie der Stellenwert des Designs als zentraler Teil des Forschungsprozesses. DBR ermöglicht es, verschiedenste Aspekte einer Intervention in unterschiedlichen Kontexten zu untersuchen, um Designprinzipien zu generieren, die durch formative Evaluation der Beobachtungen und Erfahrungen empirisch gestützt sind [CJB04]. Die Arbeit verläuft typischerweise zyklisch und zusammen mit Praktikern, wobei der Entwurf der Intervention flexibel ist – sowohl innerhalb eines Zyklus als auch über den Forschungsprozess hinweg. Deshalb ist insbesondere bei der Analyse und der Reflexion der Beobachtungen und beim Re-Design ein intensiver Austausch zwischen Forschern und Lehrerinnen und Lehrern wichtig.

Aus den oben beschriebenen Gründen sind wir der Meinung, dass die Theorie-Praxis-Verschränkung und eine Zusammenarbeit von Forschern und Lehrkräften auf Augenhöhe für unsere Arbeit besonders wichtig sind. Das Forschungsvorhaben fußt deshalb auf der berufsspezifischen Kompetenz von Lehrkräften, fachliches Wissen, allgemeindidaktisches, fachdidaktisches sowie pädagogisches Wissen in einem Wissensbereich zu integrieren [St10]. Erfahrene Lehrkräfte passen das agile Modell individuell an ihren Kontext an, setzten ergänzend eigene Ideen um, beobachten und sammeln Erfahrungen.

Der DBR-Prozess zu den agilen Methoden gliedert sich in Phasen des Inputs und der Vernetzung (Workshops) sowie Phasen der individuellen Anpassung, Weiterentwicklung und Erprobung durch die Lehrkräfte (vgl. Abb. 2). Wichtigste Funktion der Workshops ist die Bündelung der Beobachtungen und Erfahrungen aus den Implementierungen, um in gemeinsamer Analyse und Reflexion die Erkenntnisse herauszuarbeiten, auf deren Basis die Theorie angepasst und Lösungsansätze und Anregungen für die weitere Verbesserung der Praxis entwickelt werden. Eine Datenerhebung findet in jedem Zyklus statt und umfasst Leitfadeninterviews, exemplarische Projektdokumentationen und -ergebnisse sowie die Ergebnisse des Workshops. Ziel ist es, die Interventionen als Fallstudien bezüglich möglichst vieler beeinflussender Faktoren zu rekonstruieren und die

theoretischen Überlegungen, die Ergebnisse aus den Workshops und die Beobachtungen in den Interventionen auf Konsistenz zu prüfen.

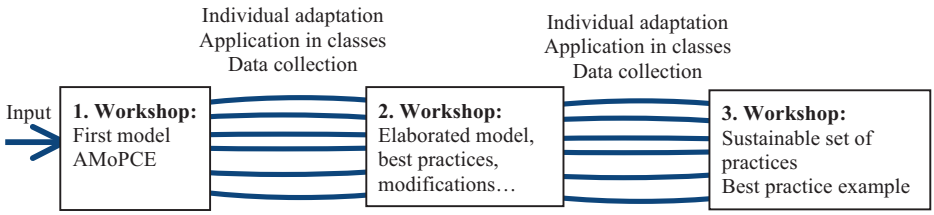

Abb. 2: Forschungsprozess zur Weiterentwicklung von AMoPCE

5 Durchführung des ersten Zyklus und erste Ergebnisse

Im Folgenden beschreiben wir wesentliche Teile der konkreten Umsetzung des ersten Zyklus, insbesondere mit Blick auf die Ausgestaltung der Zusammenarbeit mit den Lehrkräften, eine Einordnung in den Gesamtkontext sowie erste Erkenntnisse.

Die Idee einer Zusammenarbeit von Theoretikern und Praktikern auf Augenhöhe und die aktive Einbeziehung des berufsspezifischen Wissens der Lehrkräfte von Beginn an waren entscheidend für die Gestaltung des ersten Workshops, der zugleich die Funktion einer Fortbildung haben musste, damit die Lehrkräfte nach der Teilnahme in der Lage waren, das agile Modell zu implementieren. Interessierte Lehrkräfte sollten dazu ermutigt werden, das agile Modell in einer an ihren Kontext angepassten Form in einem Unterrichtsprojekt anzuwenden und dabei auch eigene Ideen und Interpretationen einzubauen. Darüber hinaus sollten sie motiviert werden, ihr berufsspezifisches Wissen in einen theorie-praxis-verschränkten Prozess der Weiterentwicklung einzubringen. Am ersten, zweitägigen, Workshop, nahmen elf Lehrerinnen und Lehrer sowie drei Forscher teil. Zur Motivierung, Verdeutlichung der Praxisrelevanz und thematischen Einstimmung beschrieb ein Softwareentwickler zunächst an einem konkreten Projekt, wie in seinem Team professionelle agile Softwareentwicklung umgesetzt wird und begründete deren Sinnhaftigkeit an konkreten, eingängigen Beispielen. Im Anschluss wurden typische Probleme in Schulsoftwareprojekten besprochen, AMoPCE vorgestellt und Möglichkeiten praktischer Umsetzungen der einzelnen Vorgehensweisen des Modells ausführlich und offen diskutiert. In dieser Phase haben die Lehrkräfte aktiv mögliche Schwierigkeiten und Probleme herausgearbeitet, die bei der Implementierung von AMoPCE im Klassenraum auftreten können und erste Ideen zu deren Lösung entwickelt. Nach einer anschließenden praktischen Erprobung wurden zu allen Vorgehensweisen des agilen Modells die diskutierten Punkte fixiert sowie weitere Ideen und Lösungsansätze dazu erarbeitet. Im weiteren Verlauf des ersten Zyklus haben sechs Lehrerinnen und Lehrer aus drei Bundesländern die agilen Methoden in sechs Mittelstufenklassen und drei Oberstufenkursen mit insgesamt ca. 170 SuS umgesetzt. Als Werkzeuge wurden Scratch, Greenfoot, BlueJ und eine Processing-IDE eingesetzt, die Projektlänge variierte von wenigen

Wochen bis hin zu acht Monaten und ebenso unterschiedlich waren die Vorkenntnisse der SuS, die vom Programmieranfänger bis zum fortgeschrittenen Programmierer reichten. Vorrangiges Ziel der ersten Implementierungsphase war es, die Stellen im Prozess zu identifizieren, die den SuS bzw. den Lehrkräften noch generell oder kontextbedingt Probleme bereiten. Ein weiteres Ziel war es herauszufinden, inwiefern sich die theoretischen Überlegungen in der Praxis an Stellen zeigen, an denen die Umsetzung bereits ohne Schwierigkeiten verlief und welche Kontextfaktoren dabei eine Rolle spielen. Auch hier galt es, die Perspektiven der Lernenden und die der Lehrenden zu berücksichtigen. Die Interpretationen, Ideen und Anpassungen der Lehrer stellen einen wesentlichen Schritt hin zu einem an Best Practices orientierten Leitfaden dar, der Lehrer bei Kontextanpassungen unterstützen soll. Diese Aspekte wurden in Leitfadeninterviews erfasst.

Generell zeigte sich in den Interviews, dass die Erfahrungen und Beobachtungen unabhängig vom Kontext motivierend gut mit den theoretischen Überlegungen übereinstimmten. Klare Schwierigkeiten zeigten sich lediglich bei SuS mit wenig Implementierungserfahrung die Java als Programmiersprache verwendeten. Diesen SuS fiel das Formulieren von Tasks schwer, also der Perspektivwechsel vom Nutzer zum Entwickler. Abgesehen von diesem Hemmnis berichten die Lehrer übereinstimmend, dass die Teams mit Hilfe der agilen Praktiken ihre Projektarbeit von Beginn an selbst organisiert haben. So berichtet ein Lehrer zwar von anfänglichen Schwierigkeiten einer Gruppe, die „innerhalb von den ersten 45 Minuten des Unterrichts fünf Stand-Up Meetings brauchte“ aber auch, dass sie rasch lernten „zielgerichtet [zu] planen“. Ein anderer formuliert den Vorteil für SuS so: „Sie haben halt eine Struktur, in der sie ihre Fehler, ihre Probleme lösen können. Ich fand […], dass sie auch mehr Selbständigkeit erlebt haben.“ Weil sie ihnen einen regelmäßigen, guten Einblick in die fachlichen und sozialen Fähigkeiten der SuS boten, schätzten die Lehrer die Stand-Up Meetings auch für sich als sehr wertvoll ein. So berichtet ein Lehrer „ich lerne auch was über die, ja, wie denken die, wie ticken die“ und weiter führt er aus „du siehst Entwicklungen, wo ich der Meinung bin, die hätte ich sonst eben nicht gesehen“ und fügt hinzufügt „ich hab viel mehr Zeit auch, die Schüler zu beobachten“, womit er die geänderte Lehrerrolle anspricht. „Ich fand es total interessant,“ führt ein anderer Lehrer aus, „…also du hängst nicht mitten drin, du musst nicht moderieren, sondern du kannst einfach zuhören. Klar, man kriegt sehr viel mehr mit, darüber wie sie arbeiten, wie sie denken, wie dieser Prozess abläuft.“ Im Zusammenspiel mit dem iterativen Vorgehen konnten sie die Lernfortschritte der SuS erkennen: „Also die Kommunikation und die Kooperation ist von Mal zu Mal intensiver geworden, also von Iteration zu Iteration.“ Sagt ein Lehrer und fügt erklärend an, dass sich immer mehr SuS an fachlich immer tiefergehenden Gesprächen beteiligten: „Wenn du so zuhörst, was da an Problemerörterungen nach dem gegenseitigen Testen lief, ist [das] beeindruckend.“ Auch der organisatorische Aspekt wurde durchweg als hilfreich empfunden. Dazu führt ein Lehrer aus: „[Alle] zusammen gekommen sind wir immer am Ende der Stunde […], da gab es immer diese Vorstellung der Prototypen“. Testen wurde nach Aussagen der Lehrerinnen und Lehrer zu einem normalen Bestandteil der Projektarbeit, der uneingeschränkt positiv von den SuS aufgenommen wurde und keiner weiteren Motivierung bedurfte: „Dann waren die gegenseitigen Tests immer erst – rein aus Motivation: Ich will mal sehen, was die gemacht haben“ berichtet ein Lehrer und kontrastiert die

aktuelle Situation etwas später zu Erfahrungen mit dem Wasserfallmodell; „weil wenn sie überhaupt am Testen sind, ist ja das schon mal was." Ein Lehrer berichtet, dass eine Gruppe von sich aus typische, wiederkehrende Fehler identifizierte „und diese Fälle haben sie sich […] auch aufgeschrieben, auch dokumentiert, welche Fehler es sind. Und die bei jeder Iteration wieder getestet. Also dann schon so ein bisschen Regressionstest". Die Prototypen gaben Gelegenheit zu (Peer-)Feedback, so schwärmt ein Lehrer, dass „dieser Kurs von mir permanent gelobt wurde, was für Schüler ich da habe […] weil die haben sich wirklich auch Mühe gegeben, […] tolle Sachen gebracht".

6 Diskussion

Das Gesamtbild aller Interviews, von dem wir oben einen Ausschnitt skizziert haben, zeigt, dass sich die theoretischen Überlegungen in der Unterrichtspraxis unabhängig vom Kontext widerspiegelten. Die Anpassungen an das spezifische Projekt waren durchweg gelungen und werden in Best Practices festgehalten. Interessant war auch, dass die SuS selbst erkannten, welche Praktiken für sie sinnvoll und hilfreich waren. So hat ein Lehrer beispielsweise das Planning Poker erst während des laufenden Projekts eingeführt, als die SuS schon ein Gefühl für sinnvolle Größen von User Stories und Tasks hatten. Diese SuS haben in der Aufwandsabschätzung keinen planerischen Mehrwert gesehen und die Praktik nach zwei Iterationen aufgegeben – gleiches wurde uns auch von professionellen Softwareentwicklern berichtet. In einem anderen Projekt wurde das Abschätzen am Anfang eingeführt und von den SuS als hilfreich empfunden, wobei sie die Erfahrung aus der ersten Iteration nutzten, um im Folgenden vergleichend abzuschätzen.

Interessant ist auch der Aspekt der Nachhaltigkeit. Ein Lehrer wurde von einer Kollegin gefragt, was er mit seinen SuS im Vorjahr gemacht habe. „Die fangen direkt an zu arbeiten, wenn ich eine Aufgabe stelle. Meine heben erst mal alle den Finger." sagte sie. Möglicherweise hat sich die Selbstwirksamkeit und/ oder die Herangehensweise der SuS an komplexere Aufgaben durch das Projekt verändert.

Wir waren überrascht, mit welcher Leidenschaft und mit welchem Engagement die Lehrerinnen und Lehrer sich beteiligen. Dies manifestierte sich beispielsweise in der mutigen und kreativen Umsetzung, die viel Vorbereitungszeit beanspruchte und unter anderem die Idee der „Student Story" hervorbrachte, ein vom Lehrer zu den User Stories der Schüler hinzugefügter Lernauftrag. Auch die inspirierende Atmosphäre und die engagierte Zusammenarbeit in den Workshops zeigt, wie sehr sich die Lehrerinnen und Lehrer mit dem Thema identifizieren. Möglicherweise ist dieses Format auch geeignet, um Lehrerfortbildungen über Zusammenarbeit auf Augenhöhe nachhaltiger zu gestalten, als durch rein inputorientierte Formate. Im Weiteren ist vorgesehen, diesen Aspekt mit zu betrachten, um festzustellen, ob aus der Anlage und Durchführung des Projekts auch Aussagen getroffen werden können, wie theorie-praxis-verschränkte Lehrerfortbildungen effektiv und nachhaltig durchgeführt werden können. Durch den zyklischen Verlauf und das flexible Design ermöglicht es DBR, diese interessanten, unerwartet aufgetreten Aspekte in einem nächsten Zyklus mit zu untersuchen.

Literaturverzeichnis

[Be01] Beck, Kent; Beedle, Mike; Bennekum, Arie van; et al.: Manifesto for Agile Software Development. http://www.agilealliance.org/the-alliance/, abgerufen am 10.04.2015.

[CJB04] Collins, Allan; Joseph, Diana; Bielaczyc, Katerine: Design Research: Theoretical and Methodological Issues. Journal of the Learning Sciences 13 (2004), Nr. 1, S. 15-42.

[De03] Design-based Research Collective: Design-Based Research: An Emerging Paradigm for Educational Inquiry. In: Educational Researcher vol. 32 (2003), Nr. 1, S. 5–8.

[Fr83] Frey, Karl: Die sieben Komponenten der Projektmethode - mit Beispielen aus dem Schulfach Informatik. In: Log in vol. 3 (1983), Nr. 2, S. 16 – 20.

[FS82] Frey, Karl; Schäfer, Ulrich: Die Projektmethode. Beltz, Basel, 1982.

[FST13] Fuch, Alexander; Stolze, Carl; Thomas, Oliver: Von der klassischen zur agilen Softwareentwicklung. In: Praxis der Wirtschaftsinformatik vol. 290 (2013), S. 17–26.

[Gö12] Göttel, Timo: Agiler Informatikunterricht : Soziale Aspekte der professionellen Softwareentwicklung im Schulunterricht erfolgreich erfahrbar machen. University of Hamburg, 2012.

[Gu14] Gudjons, Herbert: Handlungsorientiert lehren und lernen. Verlag Julius Klinkhardt, Bad Heilbrunn, 2014.

[HNR07] Hartmann, Werner; Näf, Michael; Reichert, Raimond: Informatikunterricht planen und durchführen. Springer, 2007.

[Hu05] Humbert, Ludger: Didaktik der Informatik. Teubner, 2005.

[MH10] Meerbaum-Salant, Orni; Hazzan, Orit: An Agile Constructionist Mentoring Methodology for Software Projects in the High School. In: ACM Transactions on Computing Education vol. 9 (2010), Nr. 4, S. 1–29.

[Re05] Reinmann, Gabi: Innovation ohne Forschung? Ein Prädoyer für den Design-Based Research-Ansatz in der Lehr-Lernforschung. In: Unterrichtswissenschaft vol. 33 (2005), Nr. 1, S. 52–69.

[RG12] Romeike, Ralf; Göttel, Timo: Agile Projects in High School Computing Education – Emphasizing a Learners' Perspective. In: Proceedings of the 7th Workshop in Primary and Secondary Computing Education (WiPSCE'12). ACM, Hamburg, 2012.

[Ru04] Rumpe, Bernhard: Agile Modellierung mit UML. Codegenerierung, Testfälle, Refactoring. Springer, 2004.

[SS11] Schubert, Sigrid; Schwill, Andreas: Didaktik der Informatik. Springer, 2011.

[SS05] Schneider, Daniel K; Synteta, Paraskevi: Conception and implementation of rich pedagogical scenarios through collaborative portal sites, 2005. http://tecfa.unige.ch/proj/seed/catalog/docs/schneider-icool-final.pdf, abgerufen am 10.04.2015.

[St10] Strunz-Maireder, Edith: Pedagogical Content Knowledge. In: wissenplus vol. 28 (2010), Nr. 5, S. 41–44.

[We05] Weigend, Michael: Extreme Programming im Klassenraum. In: S. Friedrich (ed.): INFOS 2005. Dresden: GI- Edition - Lecture Notes in Informatikcs [LNI], 2005.