Ein Bild vom Wesen der Softwareentwicklung: Erfahrungen aus zwei agilen Projekten

Leonore Dietrich[1], Andreas Gramm[2], Petra Kastl[3] und Ralf Romeike[3]

Abstract: Die beiden Projekte, die hier in getrennten Teilen beschrieben werden, fanden in sehr unterschiedlichen Kontexten statt und zeigen, wie Prozesse unter Verwendung agiler Methoden so an die Bedürfnisse der Beteiligten und die Besonderheiten des Projekts angepasst werden können, dass die intendierten Ziele erreicht werden. Zunächst wird ein „Forschungsprojekt" vorgestellt, das vier Oberstufenschüler mit geringen fachlichen Vorkenntnissen realisierten. Ihre Aufgabe war die Reflexion der erlebten Vorgehensweise. Im zweiten Projekt erlebten Programmieranfänger/innen einer 9. Klasse, wie sie kooperativ selbst Informatiksysteme erschaffen und nach ihren Wünschen gestalten können. Gemeinsam ist beiden Projekten also ihr Fokus auf dem Erlebbar machen eines zeitgemäßen Bilds vom Wesen der Softwareentwicklung.

Keywords: Agile Methoden, Softwareentwicklungsprojekt, Unterrichtsprojekt, Informatikprojekt

1 Einleitung

In diesem Bericht sind zwei Projekte zusammengefasst, die in vielerlei Hinsicht verschieden sind: ein kleines Forschungsprojekt, in dem vier Oberstufenschüler eines Hochbegabtenseminars agile Methoden reflektieren und ein Projekt in einer 9. Klasse, in dem Programmieranfänger/innen mit der visuellen Programmentwicklungsumgebung Scratch ein kleines Computerspiel entwickeln. Gemeinsam ist beiden Projekten, dass sie prozessbezogene Aspekte der Schaffung neuer Informatiksysteme in den Vordergrund stellen und so für Schüler/innen näherungsweise erfahrbar machen, wie Informatiker/innen heute ihren Beruf ausüben. Was aber sind wesentliche Aspekte eines solchen Prozesses, was sind Charakteristika moderner Softwareentwicklung? Die Sicht der Wissenschaft darauf hat sich über die Jahre stetig verändert. Zum Ingenieurswesen und seinem Fokus auf Planung und Problemlösungen ist ergänzend ein kreatives, gestalterisches Moment hinzugekommen und es werden Strategien entwickelt, um angemessen auf Anforderungsänderungen zu reagieren [Me14]. Die Idee von dem einen, idealen Entwicklungsprozess wird abgelöst von der Aufforderung: denke selbst und wähle die guten Ideen, die zu dir und deinem Projekt passen [Ru12, Me14]. Eine Aufforderung, die insbesondere an die Entwickler/innen geht, die den Prozess und ihre Rolle im Prozess mit ausgestalten, regelmäßig reflektieren und optimieren sollen. Hinzu kommt eine verstärk-

[1] Universität Heidelberg, Didaktik der Informatik, Im Neuenheimer Feld 326, 69120 Heidelberg, leonore.dietrich@uni-heidelberg.de
[2] Gymnasium Tiergarten, Altonaer Strasse 26, 10555 Berlin, gramm@gymnasium-tiergarten.de
[3] Friedrich-Alexander-Universität Erlangen-Nürnberg, Didaktik der Informatik, Martensstr. 3, 91058 Erlangen, petra.kastl@fau.de, ralf.romeike@fau.de

te Betonung von Eigenverantwortung und Teamorientierung in modernen Entwicklungsprozessen. Die hier beschriebenen Schulprojekte zeigen, dass dieses Wesen von Softwareentwicklung auch in Schulprojekten motivierend und realistisch erlebbar gemacht werden kann. Sie geben Antwort auf die Frage, welches Bild der Informatik, eines Softwareentwicklungsprozesses sowie der Anforderungen von Projektarbeit an Entwickler/innen wir Jugendlichen in einem „agilen Projekt“ vermitteln können und wie Schüler/innen ihr Erleben reflektieren.

2 Teil 1: Agile Methoden – nur cool oder mehr? Ein Projekt zur Reflexion des Prozesses durch die Schüler

Das Projekt „Agile Methoden in der Softwareentwicklung“ fand im Rahmen des Hector-Seminars statt, einem Baden-Württembergischen Programm zur langfristigen Hochbegabtenförderung, das in der Oberstufe mit einer einjährige Kooperationsphase abschließt, in der Teams, meist in Zusammenarbeit mit einer Hochschule, ein kleines Forschungsprojekt durchführen. Vier Schüler, aus Mannheim und Heidelberg, haben in dieser Phase ein Softwareprojekt mit einer agilen Vorgehensweise [RG12] realisiert. Ziel war, das Vorgehen und die agilen Methoden zu erleben, zu reflektieren und abschließend vorzustellen. Die Projektumsetzung begann im Januar und lief bis Ende des Schuljahres. Im Mittel fand in dieser Zeit ein begleitetes, vierstündiges Treffen pro Monat an der Universität statt, parallel dazu gab es selbständige Treffen. Die fachlichen Vorkenntnisse der Schüler waren überwiegend gering. Sie hatten teils in der Mittelstufe einzelne Kurse zur Robotik (Lego Mindstorms) und zum Einstieg in die Programmierung mit Java (Greenfoot Kara) belegt, aber kaum tiefere, objektorientierte Konzepte verinnerlicht. Ein Schüler hatte grundlegende Java-Kenntnisse, Prozessmodelle aus der Softwareentwicklungstechnik waren nicht bekannt. Allerdings sind die Schüler sehr reflektiert, verfügen über eine extrem schnelle Auffassungsgabe und haben eine außergewöhnlich ausgeprägte Fähigkeit zu analytischem und strukturiertem Denken sowie eine hohe Abstraktionsfähigkeit.

2.1 Der Projektverlauf

Das Spiel, welches mit Greenfoot im Projekt entwickelt wurde, wurde primär als Produkt der Anwendung der ausgewählten Methoden und Strategien thematisiert. Deshalb wurde im ersten Treffen zunächst erläutert, wie Softwareentwicklungsprojekte mit agilen Vorgehensweisen ablaufen. Ausgewählte Videos unterstützten die Ausführungen und vermittelten einen Einblick in das Vorgehen bei professionellen Projekten. Anschließend wurde besprochen, wie es im konkreten Projekt aussehen kann. Dabei sind wir auf folgende Praktiken und Artefakte eingegangen: Stand-Up-Meetings[4], User Stories und

[4] Treffen des Teams, um sich genseitig über die jeweiligen Aktivitäten in arbeitsteiligen Phasen zu informieren.

Tasks[5] sowie ihre Aufwandsabschätzung mit Hilfe des Planning Pokers[6], Sprints[7], Pair Programming mit den Rollen des Drivers und des Navigators, Refactoring[8], Prototypen, die Retrospektive und das Project Board als Planungs- und Informationsbereich.

Im konkreten Kontext hätte sich ein elektronisches Project Board angeboten, da die Treffen an verschiedenen Orten stattfanden und die Schüler von unterschiedlichen Schulen kamen. Allerdings wollte ich den Schülern die Möglichkeit bieten, nach getaner Arbeit für alle sichtbar und haptisch erlebbar einen Zettel umzuhängen. Deshalb bestand unser Project Board aus einem für den Transport gefalteten und zu den Treffen mitgebrachten Plakat (vgl. Abb. 1). Die vierstündigen betreuten Treffen umfassten zwei Sprints von je 45 Minuten. Das Stand-Up-Meeting zu Beginn diente dem Rekapitulieren des vorangegangenen Treffens und ging dann in die Planung des nächsten Sprints über. Während der Planungs- und Entwurfsphase wurde vor allem viel diskutiert. Dokumentiert wurden ein Klassendiagramm, das im Rahmen eines Refactorings erstellt wurde und einige Abläufe in Form von Struktogrammen (vgl. Abb. 1). Für das Projekt und seine Ziele war diese Form der Planung passend, denn die Schüler hatten wenig Vorkenntnisse im Bereich von Spezifikationssprachen und das Produkt war bezogen auf die Fähigkeiten der Schüler zu strukturierendem und abstrahierendem Denken wenig komplex. Außerdem wurde so die “besondere „Schülerklientel“ zum Kommunizieren gebracht und es entstand viel Gelegenheit zum Wissenstransfer. Die Reflexionen der Sprints am Ende der Treffen hingegen wurden gut dokumentiert und die Schüler haben Schlussfolgerungen für ihren Entwicklungsprozess schriftlich festgehalten.

Abb. 1: Project Board und Struktogramme

2.2 Beobachtung und Bewertung

Aus Lehrersicht[9]:. Problematisch war die Teamgröße von nur 4 Schülern. Wenn einer der vier Schüler kurzfristig ein Treffen absagte, waren sinnvolle Sprints kaum möglich. Eine Herausforderung stellte das agile Vorgehen bezüglich der Erwartungen an kommu-

[5] Funktionalitäten aus Kundensicht sowie Aufgaben, die zur Umsetzung aus Entwicklersicht notwendig sind.
[6] Praktik zum gemeinsamen und spielerischen Abschätzen des Arbeitsaufwands.
[7] Feste Zeitfenster, in denen der Prototyp inkrementell in Miniprojekten weiterentwickelt wird: auch Iteration.
[8] Das Umstrukturieren des Quelltextes ohne Änderung der Funktionalität.
[9] Hier werden nur die Schülersicht ergänzende Aspekte beschrieben, abweichende gab es nicht.

nikative und kooperative Fähigkeiten dar – zumindest für drei der vier Schüler. Beim ersten Stand-Up-Meeting beispielsweise standen sie da wie „bestellt und nicht abgeholt". Im Verlauf wurde die Kommunikation untereinander beobachtbar besser: nach drei Sprints war beispielsweise zu erkennen, dass die Diskussionen beim Planen und später beim Abschätzen des Aufwands für die Tasks wesentlich differenzierter wurden und die Beteiligung gleichmäßiger verteilt war. Unterstützt wurde die Entwicklung vermutlich auch durch eine wechselnde Zusammenstellung der Pairs beim Programmieren, die für den Wissenstransfer förderlich war. Insgesamt hatte ich den Eindruck, dass sie durch die agile Vorgehensweise und die sogenannten Zeremonien einen für sie praktikablen Fahrplan hatten und eine Struktur, in der sie Fehler erkennen und Probleme lösen konnten. Sie sind planvoll vorgegangen, ohne dass ich als Lehrerin immer den nächsten Schritt vorgeben und extra motivieren musste. Das Planning Poker z. B. traf spontan auf große Zustimmung, trotz anfänglicher Schwierigkeiten aufgrund geringer Programmiererfahrung. Die Schüler reflektierten die erlebten Abweichungen nach jedem Sprint und bezogen die Erfahrungen in die folgenden Abschätzungen ein: „Das denke ich ist etwas / viel mehr / weniger aufwändig als [...], weil [...]". Dabei fielen ihnen auch Aspekte auf, die sie beim Planen der Tasks zu ungenau besprochen hatten, weil dann die Schätzungen auseinander lagen, d. h. die Abschätzung hatte auch eine Kontrollfunktion. Erstaunt hat mich, dass die Schüler von sich aus Fehler, die sie im Test gefunden haben, strukturierten und klassifizierten, um diese dann bei den zukünftigen Prototypen für Tests wieder zu verwenden.

Insgesamt konnte ich viel zuhören und in den Diskussionen von der Planung bis zur Retrospektive verfolgen, wie die Schüler denken, wie sie an Problemstellungen herangehen, sich Unbekanntes erschließen und wie sich ihre Fähigkeiten und Fertigkeiten im Laufe des Projekts entwickeln. Der Aufbau agiler Entwicklung erwies sich für die Schüler als anregend, die Bezeichnungen der Zeremonien sind nah an der Schülersprache und Elemente wie Planning Poker integrieren eine Art Gamification-Ansatz in die Projektsteuerung. Über diese Elemente lassen sich die Schüler gut abholen. Für das nächste Projekt wird die Gruppe allerdings größer ausfallen, um auch bei Krankheit eines Teammitglieds wenigstens zwei Pairs für eine weitere Iteration zusammenstellen zu können. Eine Teamgröße von 6-7 Schülern halte ich auf der Basis der Projekterfahrung für sinnvoll. Interessant wäre auch die Erprobung einer grafischen Programmierumgebung, die den Fokus noch mehr weg von Implementierungsproblemen und hin zum agilen Prozess verschiebt.

Aus Schülersicht: Der erste Durchlauf eines Sprints war für uns eine neue Erfahrung. Die kurzen Phasen erforderten gezieltes Arbeiten am Projekt und damit auch klare Kommunikation und Rollenverteilung. Die Formulierung von User Stories schien uns zunächst schnell von der Hand zu gehen. Bei der Formulierung des Tasks wurde dann aber klar, dass die Stories zum Teil zu klein waren – nur eine Task enthielten – und zum Teil viel zu umfassend. Das anschließende Planning Poker brachte weitere Schwierigkeiten und einige Diskussionen um Zeit- und Implementierungsaufwand. Im ersten Sprint stellte sich dann auch heraus, dass wir uns in einigen Punkten verschätzt hatten, wobei insbesondere die zu großen User Stories dann im vorgegebenen Zeitfenster nicht um-

setzbar waren. Diese wurden in der Sprint Retrospektive angesprochen und für den nächsten Sprint wieder aufgenommen. Die zu Beginn des zweiten Sprints zunächst vorgenommene Neuformulierung brach nun die zu groß geratenen User Stories in kleinere auf. Bei dieser Aufgabe holten wir uns bisweilen auch Feedback bei der Lehrerin. Ein wirklicher Kunde oder eine entsprechende Rolle – diese wurde in unserem Fall dann von der betreuenden Lehrerin übernommen – sind für die Entwicklung sehr hilfreich.

Im Rahmen der weiteren Sprints konnten wir den Umgang mit den Zeremonien verinnerlichen und uns in die schnelle Entwicklung in kurzen Iterationen einarbeiten. Als motivierend wirkte das stets lauffähige Spiel, die Möglichkeit, jederzeit zu testen und, die Auswirkungen der eigenen Programmierung direkt zu sehen. Schwierigkeiten hatten wir beim unbetreuten Arbeiten insofern, als dass wir die engen Zeitvorgaben nicht konsequent einhielten – hier ist wohl ein Projektmanager hilfreich wie ihn die betreuende Lehrerin bei den Präsenzterminen darstellte. Gleiches gilt für den Rollenwechsel in den Pairs innerhalb der Implementierungsphasen. Bei weiteren Präsenzterminen galt deshalb dem Zeitplan und dem Pair-Wechsel besondere Aufmerksamkeit. Wir konnten durch den Pair-Wechsel auch von dem Teammitglied lernen, das die besten Programmierkenntnisse mitbrachte.

Nach drei Iterationen kamen wir an eine Stelle, an der wir einige Programmierprobleme nicht mehr selbständig lösen konnten und fachliche Hilfe benötigten. Der Seminarleiter zeigte uns verschiedene Möglichkeiten zur Lösung des Problems auf. Bei der Umsetzung stellte sich jedoch heraus, dass wir gleichartige Funktionalitäten an vielen verschiedenen Stellen brauchten und somit eine Überarbeitung unserer Gesamtstruktur nötig wurde. Hierfür unterbrachen wir das agile Szenario[10] und entwickelten mit Unterstützung durch die Lehrerin ein Klassendiagramm aus Metaplankarten und Post Its. Das Experimentieren mit den Karten, aus denen wir das Klassendiagramm zusammenbauten, war einfach und schnell und man konnte gut sehen, wie die einzelnen Programmteile zusammenhängen. Indem wir Klassen und Methoden verschoben und verschiedene Strukturen diskutierten, fanden wir eine sinnvolle Struktur. Die Umsetzung des Refactorings übernahm ein Pair, während das andere sich mit einem weiteren identifizierten Problem befasste: der einheitlichen und standardkonformen Benennung von Klassen und Methoden. Als relativ einfach nahmen wir das Zusammenführen des Codes wahr, wobei die zwei Pairs beim Implementieren versuchten, jeweils in verschiedenen Klassen zu arbeiten. So mussten nur die einzelnen Klassen zusammenkopiert werden. Insgesamt war es spannend, den agilen Ansatz kennenzulernen. Bessere Vorkenntnisse in der Programmiersprache und der -umgebung hätten uns allerdings den Einstieg sicher erleichtert.

[10] In einer Iteration wurde kein Produktinkrement erstellt sondern ein Refactoring geplant und umgesetzt.

3 Teil 2: Das Henne-Ei Problem – Wie Programmieranfänger/innen ein Projekt stemmen können

Eine zentrale Frage ist, was wir mit Projekten im Informatikunterricht erreichen wollen. Grundsätzlich finde ich es schwierig, im gegebenen zeitlichen Rahmen Vorgehensweisen bei der Entwicklung größerer Softwaresysteme für Lernende erlebbar zu machen und sie über die Phasen hinweg zu motivieren. Ein typisches Problem ist, dass sie gar keine Analyse machen können, bevor sie nicht ein Projekt durchlaufen haben und wissen, was die eine oder andere Modellierungsentscheidung später bewirkt, was es bedeutet, etwas als Klasse, Interface oder Entität zu modellieren. Ohne ein ordentliches Modell kann ich kein sinnvoll strukturiertes Produkt erstellen und ohne einmal ein Produkt gesehen zu haben, kann ich nicht sinnvoll modellieren – ein Henne-Ei-Problem. Unzufrieden bin ich auch damit, dass Tests aus Zeitgründen zurücktreten müssen und für eine Reflexion trotzdem viel zu wenig Zeit bleibt. Bis aber die Schüler/innen überhaupt über die Voraussetzungen verfügen, um linear verlaufende Projekte angehen zu können, ist oft schon bei vielen ihr Interesse für das Fach der Auffassung gewichen, dass Informatik viel zu schwer ist.

Wichtig ist mir deshalb, ihnen am Anfang zu zeigen, was man mit Informatik machen kann und wie große Informatiksysteme entstehen. Ich möchte ihnen zeigen, dass sie etwas schaffen können, sie für das Fach begeistern und ihnen ein inspirierendes Bild vom Beruf eines Informatikers / einer Informatikerin vermitteln. Gerade im Anfangsunterricht möchte ich erlebbar machen, dass es in diesem Beruf um Gestalten und um Teamwork geht und dass man sich in der Informatik gut organisieren und zusammen tolle Sachen machen kann. Deshalb finde ich ein iteratives Vorgehen sehr verlockend. Man hat kurze Zyklen, sodass die Schüler/innen in zwei Wochen in einem Miniprojekt alles kennenlernen und dabei ein interessantes Zahnrädchen einer größeren Lösung entwickeln. In der nächsten Iteration kommt ein weiteres Zahnrädchen dazu. Es ist mehr als die Aneinanderreihung von Umsetzungen kleiner, unabhängiger Probleme, denn es fügt sich zu etwas Größerem zusammen. Wenn daran sechs Leute in Kooperation arbeiten, entsteht schnell etwas Spannendes. Das Endprodukt muss nicht perfekt sein, es muss nicht jede Idee umgesetzt sein. Trotzdem haben die Schüler/innen einen lauffähigen Prototypen, den sie selbst gemeinsam entwickelt haben und sie wissen, dass und wie es weitergehen könnte. Auch der Test verändert sich dadurch, dass er an einem inkrementell wachsenden, realen Produkt stattfinden kann, nicht auf einer Kommandozeilenebene weit weg vom eigentlichen Produkt.

3.1 Rahmenbedingungen

Das Projekt fand in einer 9. Klasse in einem Wahlpflichtkurs mit einer Doppelstunde pro Woche statt. Die Gruppe bestand aus 20 Schüler/innen, die zum größten Teil nichtdeutscher Herkunftssprache waren, weshalb sich die Kommunikation in kooperativen Arbeitsformen auch immer zur Förderung der Sprachkompetenz im Sinne einer Sprachbil-

dung im Fachunterricht anbietet. Für die Schüler/innen war es ihr erstes Schuljahr mit Informatikunterricht. Das Projekt wurde in der frühen Phase des Kurses mit Scratch durchgeführt. Aufgabe war es, exemplarische Komponenten eines Jump-and-Run-Spiels zu entwickeln. Das Projekt musste umständehalber zwei Mal unterbrochen werden. Die Vorübungen fanden im Herbst statt, die Formulierung der User Stories im Januar und die Realisierung im Frühjahr.

3.2 Einbettung in den Unterricht und Anpassungen an das konkrete Projekt

Vor dem Projekt wurden Kollisionen und Variablen in Scratch behandelt. Die Schüler/innen konnten Zustände und Zustandsveränderungen speichern und wussten, wie eine Kollision als Ereignis erfasst werden kann. Auf dazugehörige Arbeitsblätter mit erklärenden Beispielen konnten sie im Projekt zurückgreifen. Außerdem wurde vorbereitend auf das Projekt geübt, wie einzelne Objekte in Scratch exportiert und importiert werden, wobei deutlich wurde, dass eine Codeintegration relativ einfach abläuft, solange man nicht in gleichen Sprites arbeitet. Diese Übung haben die Schüler/innen als Pair Programming durchgeführt und damit diese Praktik bereits vorbereitend geübt.

Die agile Vorgehensweise sollte eine effektive Produktentwicklung mit Scratch unterstützen und wurde entsprechend angepasst: Da die User Stories sehr klein waren und nur einige wenige Aufgaben umfassten, bedurfte es in Verbindung mit Scratch keiner zusätzlichen Formulierung von Teilaufgaben aus Entwicklersicht. Deshalb wurde nicht zwischen User Stories und Tasks unterschieden. Die User Stories wurden notiert und am Project Board befestigt, ihr Aufwand wurde mit ein bis drei Sternchen[11] und ohne Planning Poker abgeschätzt (vgl. Abb. 2). Eine logische Bearbeitungsreihenfolge ergab sich im Verlauf von selbst, d. h. auch auf eine Priorisierung konnte verzichtet werden, wodurch die Vorarbeiten weiter verkürzt wurden. Stand-Up Meetings fanden zu Beginn jeder Doppelstunde statt und gaben den Lernenden eine Struktur vor, in der sie selbstständig den Stand ihrer Arbeit rekapitulieren und anschließend die Doppelstunde planen konnten, wobei die wesentliche Aufgabe der Planung in der Auswahl geeigneter User Stories bestand. Während der Stunde konnten in der Gruppe auftretende Fragen und Probleme auch in einem Stand-Up-Meeting vor dem Board besprochen werden (vgl. Abb. 2). Da es keinen festgelegten Funktionsumfang gab, der im gegebenen Zeitrahmen umgesetzt werden musste, genügte das Board, um den Projektfortschritt zu verfolgen, auf ein Burn Down Chart zur graphischen Darstellung konnte verzichtet werden. Die konkrete Umsetzung mit Scratch erfolgte im Pair Programming, wobei die Rolle des Navigators aufgrund der wenigen Konzepte und Datenstrukturen, die bekannt waren, eingeschränkt war. Wichtig war, dass der Driver stets seine Ideen bei der Umsetzung ausdrückte und die Schüler/innen so lernten, ihre Programmierung zu beschreiben bzw. kritisch zu hinterfragen.

Es wurden zwei Prototypen entwickelt, von denen nur der zweite zusammen mit einer

[11] Wenig, mittel bzw. viel Aufwand.

Reflexion über das im Projekt Gelernte im Plenum vorgestellt wurde, um eine längere Phase der konzentrierten Projektarbeit zu haben, nachdem das Projekt im Vorfeld zwei Mal unterbrochen worden war. Ihre individuellen Leistungen haben die Schüler/innen abschließend in den Gruppen besprochen und bewertet.

3.3 Die Projektdurchführung

Als Einstieg wurde das Ball Point Game [Gl15] mit allen 20 Schülerinnen und Schülern gemeinsam gespielt (vgl. Abb. 2). Sie hatten einen Heidenspaß und eine deutlich sichtbare Optimierung. Es war eine gelungene Motivation, die zeigte, wie wichtig Absprachen für eine erfolgreiche Kooperation sind und wie man einen iterativen Prozess durch stringentes Planen, Handeln und Reflektieren optimieren kann. In der verbleibenden Zeit dieser Doppelstunde wurde der Kurs in drei Gruppen geteilt, welche erste Ideen für ihr Jump-and-Run-Spiel sammelten, diskutierten und als User Stories formulierten. In einer weiteren Doppelstunde wurden diese User Stories konkretisiert und ergänzt. Die Umsetzung erfolgte in zwei Iterationen von jeweils zwei Doppelstunden an deren Ende jeweils ein Prototyp vorliegen sollte. Teilweise wurden in den Planungen weitere User Stories ergänzt, soweit dies die Umsetzung der Spielidee erforderlich machte. Wenn sie wollten, konnten die Schüler/innen sich ihr Projekt auf einen Stick kopieren und daran zu Hause weiterarbeiten. Die dabei umgesetzten Funktionalitäten wurden in der folgenden Doppelstunde der Gruppe vorgestellt und in das Projekt integriert. Zwei oder drei Mal sind in dieser Phase Probleme aufgetreten, die entweder alle hatten oder auf die sie in Kürze stoßen würden. In diesen Situationen habe ich das Thema zu Beginn der folgenden Doppelstunde kurz im Plenum aufgegriffen und von einzelnen Gruppen erarbeitete Lösungen für das Problem dem gesamten Kurs vorstellen lassen.

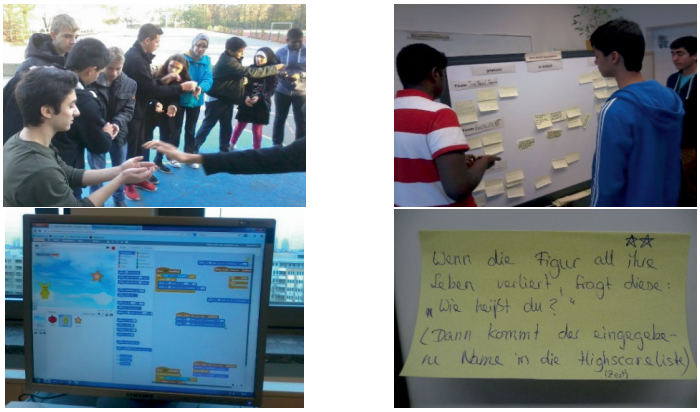

Abb. 2: Ball Point Game, am Project Board, Implementierung, User Story

In der abschließenden Präsentation sollten die Schüler/innen die Umsetzung einzelner Funktionalitäten erläutern können und jede Gruppe sollte eine Zusammenfassung ihrer

individuellen Reflexionen zur der Frage „Was habe ich im Projekt gelernt?“ vorstellen.

Eine Herausforderung stellt die Bewertung von Leistungen dar, die von der Gruppe bzw. individuell in einem Projekt erbracht werden, da zwei unterschiedliche Interessen kollidieren: Zum einen will ich sie befähigen, etwas zu tun und gleichzeitig messe ich, was sie tun. Ein weiterer Punkt ist die Frage, welchen Wert eine individuelle Idee hat und welchen Wert die Fähigkeit der Gruppe, sie erfolgreich umzusetzen. Nachdem die Schüler/innen ihre Projekte selbstorganisiert durchgeführt haben, war es mir wichtig, sie in den Prozess der Bewertung einzubeziehen. Gemäß dem Prinzip einer Poolnote bekam jede Gruppe von mir entsprechend ihrer Gesamtleistung eine Anzahl von Punkten. Die Aufgabe der Gruppe bestand nun darin, sich auf eine Verteilung der Punkte auf die Gruppenmitglieder zu einigen und sich so mit der Frage auseinandersetzen, welche Fähigkeiten und Fertigkeiten jedes einzelnen Mitgliedes die Gruppe im Projekt besonders gut vorangebracht haben und welche eher hemmend waren. Die so erzielte Note umfasste sowohl die Leistung bezüglich des Produkts als auch den Beitrag zur Zusammenarbeit.

3.4 Beobachtungen und Erfahrungen

Die Entscheidung, das Projekt früh im Schuljahr durchzuführen war gut, weil es bei den Lernenden eine positive Einstellung zum Fach erzeugte. Auch das Thema würde ich so wiederwählen, zum einen waren sie von Anfang an interessiert und konnten ihre Erfahrung mit Spielen einbringen, zum anderen entstand im Laufe der Zeit eine Art Wettbewerbssituation. Sie interessierten sich mehr und mehr für die Produkte der anderen. Diese Wettbewerbssituation wirkte zusätzlich anregend, ebenso wie das Peerfeedback. Motivation und Engagement führten dazu, dass schnell gute Prototypen da waren. Die positive Bestärkung der Selbstwirksamkeit durch die iterative Vorgehensweise und die Prototypen sehe ich sehr positiv. Und da sich der Ablauf wiederholt und eine klare Struktur hat, musste ich nicht wie früher moderieren und den nächsten Schritt vorgeben. Die Schüler/innen arbeiteten eigenständiger als in Projekten nach dem Wasserfallmodell und zunehmend selbstbewusster. Mir hat gefehlt, dass sie während des Projekts in fachlicher Hinsicht nicht nur bekannte Fähigkeiten ausbauen und vertiefen, sondern sich im Rahmen des Projekts auch neue Konzepte erarbeiten. Im nächsten Projekt werde ich daher Anregungen bereitstellen, die die Erarbeitung neuer Fachkonzepte initiieren.

Für mich sehr entlastend war, dass von Anfang an klar kommuniziert war: Wir werden nur ein Stück des Spiels implementieren, es wird funktionieren, aber nicht ausgereift sein. Es bestanden weder Anspruch noch Notwendigkeit, in der gegebenen Zeit alle Funktionalitäten umsetzen zu müssen, um ein sinnvolles Produkt zu haben. Darüber, wie sich das bei Oberstufenprojekten umsetzen lässt, muss ich noch nachdenken, hier jedenfalls war ich entspannt und die Schüler/innen waren mit ihren Ergebnissen zufrieden.

Verantwortlich für ihr Projekt haben sich fast alle Schüler/innen gefühlt. Einzelne, die sich herausnehmen, gibt es wahrscheinlich immer, aber hier ist ein solches Verhalten

deutlich aufgefallen und es war für diejenigen unangenehmer als gewöhnlich. In einer Gruppe hat es mit der Teamarbeit nicht gut funktioniert. Auch dort haben sich alle bemüht, etwas beizutragen. Allerdings war in diesem Team eine sehr leistungsstarke Schülerin mit einer schnellen Auffassungsgabe. Sie fand schnell Lösungen und setzte sie um. Leider hat es die Gruppe aber nicht geschafft, dieses Wissen zu transferieren und gemeinsam zu nutzen, obwohl ich dies wiederholt angeregt habe. Diese Schülerin hatte in der Gruppe keinen leichten Stand. Ich vermute, dass sich die anderen Gruppenmitglieder nicht von ihr unterstützt fühlten bzw. es als besondere Herausforderung empfanden, neben ihr auch gute Leistungen zeigen zu können.

Bei der Bewertung haben sich zwei Gruppen sehr schnell geeinigt, die Punkte gleich aufzuteilen. An dieser Stelle war es mir nicht wichtig, einzugreifen oder mir die Überlegungen darlegen zu lassen, da sich meiner Beobachtung nach alle für ein gutes Arbeitsergebnis engagiert hatten. Interessant war wiederum die Gruppe, in der es Spannungen gegeben hatte. In dieser Gruppe gab es entsprechend große Diskussionen: Die leistungsstärkere Schülerin meinte, dass ihr Beitrag mehr Anerkennung finden müsste, die anderen Gruppenmitglieder betonten, dass erst die Tatsache, dass sie der Gruppe eine Überarbeitung zugesagt, aber nicht eingehalten hat, ein besseres Gruppenergebnis verhindert hat. Nach langer Diskussion mit wenig Verständnis für die jeweils andere Position, haben sie sich als Kompromiss auf die gleiche Punktzahl für alle Gruppenmitglieder geeignet. Die Schüler/innen haben damit eine sehr ambivalente Situation bewältigt, die ihnen auch im späteren Leben begegnen wird. Am Ende formulierten sie dann auch in der Reflexion: Absprachen treffen ist wichtig, aber mühsam und teilweise schwierig. Aber ohne Absprachen kann eine fruchtbare Zusammenarbeit nicht funktionieren. Neben dieser Erfahrung war die Präsentation der Arbeitsergebnisse aber vor allem von Stolz auf das Erreichte geprägt. Auch wenn es sich dabei nur um kleine Funktionen handelte: Die Schüler/innen haben erfahren, dass sie gemeinsam selbst Informatiksysteme erschaffen und nach ihren eigenen Wünschen und Interessen gestalten können und waren nun hoch motiviert, weitere Themengebiete der Informatik zu erarbeiten.

Literaturverzeichnis

[Gl15] Glogler B.: Ball Point Game. http://borisgloger.com/scrum/materialien/tools/, 24.04.2015.

[Me14] Meyer, B.: Agile! The good, the hype and the ugly. Springer, 2014.

[RG12] Romeike, R.; Göttel, T.: Agile Projects in High School Computing Education: Emphasizing a Learners' Perspective. In (Knobelsdorf, M.; Romeike, R. Hrsg.): Proceedings of the 7th Workshop in Primary and Secondary Computing Education. ACM, New York, NY, USA, 2012; S. 48–57.

[Ru12] Rumpe, B.: Agile Modellierung mit UML. Codegenerierung, Testfälle, Refactoring. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.