Erfahrungen mit agilem Software-Engineering im Projekt "Smart Grid @ Home"

Barbara Dörsam
Hochschule der Medien
Stuttgart, Deutschland
doersam@hdm-stuttgart.de

***Zusammenfassung*—Dieses Paper beschreibt das Vorhaben, das im Sommersemester 2019 an der Hochschule der Medien in Stuttgart durchgeführt wurde, um Studierenden eines Ingenieur-Studienganges Softwareengineering-Themen nahe zu bringen. Da es sich dabei um keine reine Informatik-Ausbildung handelt, wurde vor allem Wert auf Praxiserfahrungen und nicht nur auf Vermittlung der Theorie gesetzt. Im Folgenden werden die wichtigsten Aspekte eines solchen Projektes (Themenwahl, Vorbereitung, Begleitung des Projektes) aus der Betreuersicht geschildert und die Rückmeldungen der beteiligten Studierenden reflektiert.**

***Abstract*—This paper describes a project, which was carried out in year 2019 at the Media University in Stuttgart. The goal of the project was to familiarize students of an engineering degree with software engineering topics. Since the students involved were mostly beginners in the topic of computer sciences, the emphasis was placed on practical experience and not just on teaching theory of software engineering. In the following, the most important aspects of such a project (the choice of the subject, the preparation and the supervision of the project) are described from the supervisor's point of view and the feedback from the students is reflected on.**

Keywords — Software-Engineering, agile Praktiken, Entwicklung verteilter Softwaresysteme, Smart Grid

I. Zielsetzung der Projektarbeit und die Projektidee

Im Folgenden wird eine Projektarbeit beschrieben, die zum Ziel hatte, Studierende des Studiengangs „Wirtschaftsingenieurwesen Medien" an der Hochschule der Medien an das Thema „Software Engineering" heranzuführen. Die Lernziele, die mit dem Modul (8 SWS / 12 ECTS) erreicht werden sollten, entsprechen den typischen Teilgebieten des Software-Engineerings [1]:

- Analyse und Entwurf verteilter Informationssysteme,
- Softwareentwicklung im Team mit aktuellen Technologien, z.B. Implementierung asynchroner Abläufe mit node.js,
- Projektmanagement in einem agilen Umfeld sowie
- Softwarequalitätsthemen.

Ziel der hier beschriebenen Veranstaltung war es, diese Themen an Studierende zu vermitteln, die bis dahin lediglich das reine Programmieren und Entwickeln von Webanwendungen in zwei bis drei Semestern gelernt haben. Anders als in vorhergehenden Semestern wurde diesmal Wert darauf gelegt, die Kenntnisse nicht im Rahmen einer Vorlesung theoretisch zu vermitteln, sondern sie von Anfang an im Rahmen eines größeren Projektes zusammen mit den Studierenden zu erarbeiten.

Für dieses Vorhaben war die Wahl des Themas für die Projektarbeit besonders wichtig: frühere Projektarbeiten haben gezeigt, dass heutige Studierende sich wenig für Themen begeistern können, welche sie nicht als relevant für ihren Alltag empfinden und welche ihnen nicht komplex genug erscheinen. Diverse kleinere Projekte aus der Vergangenheit haben bereits gezeigt, dass die Motivation der Studierenden sinkt, wenn sie auf technische oder organisatorische Probleme bei solchen „Spielprojekten" stoßen: da sie wissen, dass ihre Projektergebnisse nach Projektabschluss nicht weiter verwendet werden, werden Probleme oft durch eine Reduktion der Projektkomplexität gelöst und der geplante Projektumfang sinkt damit im Laufe eines Semesters.

Deswegen war die Wahl eines realitätsnahen Themas, bei dem auch Aussicht auf eine Weiterentwicklung in späteren Projekten besteht, von besonderer Bedeutung für dieses Vorhaben. Aber auch weitere Aspekte spielten bei der Wahl des Projektthemas eine wichtige Rolle:

Ein besonders wichtiger Aspekt der heutigen Lehre ist die Wahrnehmung der „Third Mission" der Hochschulen. Dahinter steckt der Anspruch an die Hochschullehre, einen direkten oder indirekten Beitrag zu gesamtgesellschaftlichen Problemstellungen und Entwicklungen zu leisten. Dieser spiegelt sich auch in aktuellen Projektausschreibungen wider, z.B. beim Projekt HUMUS plus der GHD (Geschäftsstelle der Studienkommission für Hochschuldidaktik der Hochschule Karlsruhe – Technik und Wirtschaft) [2].

Einen weiteren Aspekt liefert eine aktuelle Studie des VDI vom April 2019 [3]: Dort ergab eine Umfrage, dass 56% der befragten Studierenden als Hemmnis beim digitalen Wandel vor allem „konservativ eingestellte oder neuen Inhalten ablehnend gegenüberstehende Professorinnen und Professoren wahrnehmen".

Im Rahmen des Projektes, das im Folgenden vorgestellt wird, standen beide Aspekte im Vordergrund: es wurde versucht, sowohl Themen des Software-Engineerings anhand aktueller gesamtgesellschaftlichen Problemstellungen und Entwicklungen zu vermitteln als auch den technologischen Wandel in Richtung digitaler Zukunft vorzuleben und voranzutreiben. Deswegen fiel die Entscheidung auf das Projektthema „Smart Grid @ Home". Es bietet Studierenden die Möglichkeit, an einem gesellschaftlich verantwortlichen Thema (erneuerbare Energien / Reduzierung des CO_2-Ausstoßes) mit zu arbeiten und dabei ihre Fachkenntnisse zu vertiefen und in die Praxis umzusetzen. Dadurch sollte die intrinsische Motivation der Studierenden geweckt werden.

II. Hintergrund: „Smart Grid"-Anwendungen

In den USA wurde der Begriff Smart Grid im Rahmen eines Gesetzes bereits 2007 eingeführt, um mit Hilfe digitaler Techniken die Verfügbarkeit, Sicherheit und Effizienz des elektrischen Netzes zu erhöhen [4]. Die dort getroffene Charakterisierung eines Smart Grids enthält auch ein aktives

Energiemanagement - sowohl auf Seiten der Netzbetreiber als auch auf Seiten der Haushalte und der Industrie.

Heute ist der Aspekt der Vernetzung von Stromverbrauchern und –erzeugern im Haushalt ein hoch aktuelles Thema, in das nicht nur große Unternehmen investieren, sondern das auch in den aktuellen Medien immer wieder aufgegriffen wird. Bei der Vernetzung von Haushalts- und Entertainmentgeräten stand bisher größtenteils der Komfort von Vernetzungslösungen und digitaler Steuerung im Vordergrund („Smart Home“). Inzwischen wandelt sich hier jedoch die Sichtweise und der Energiebedarf für den Betrieb der Haushaltsgeräte rückt in den Fokus der Betrachtungen. Vor allem private Haushalte erzeugen immer häufiger eigenen Strom und speisen immer mehr überschüssige Strommengen ins Netz, die von diesem verkraftet werden müssen. Hier stellt sich die Frage, ob durch eine intelligente Steuerung der Haushaltsgeräte die selbst erzeugte Energie so verteilt werden kann, dass sie größtenteils vom Erzeuger selbst verbraucht und die Einspeisung sowie der Energiebezug vom Netzbetreiber reduziert werden kann.

Ansätze, bei denen versucht wird, den aus erneuerbaren Energien erzeugten Strom (z.B. aus einer privaten Solaranlage) so weit wie möglich selbst zu verbrauchen und dadurch bezüglich der Stromversorgung autark und CO_2-neutral zu werden, bilden einen Schwerpunkt der „Smart Grid“-Forschung. Ziel von „Smart Grid“-Haushaltslösungen ist es daher, eine Vernetzung von Energieverbrauchern und –erzeugern zu schaffen und mit einem Energiemanagementsystem (EMS) eine intelligente Stromzuteilung zu erreichen. Die Zuteilung der solar erzeugten Energie wäre dann optimal, wenn sie direkt beim Erzeuger verbraucht oder gespeichert und der Energiebezug damit so weit wie möglich reduziert werden könnte (Fig. 1).

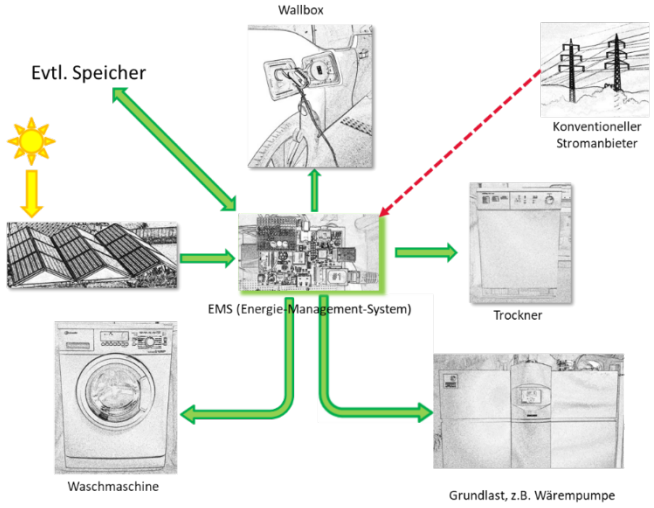

Fig. 1. Beispiel einer vernetzten Smart Grid Lösung.

Das hier vorgestellte Projekt „Smart Grid @ Home“ hatte zum Ziel, Software-Engineering-Kenntnisse am Beispiel eines „Smart Grid“-Systems zu vertiefen und auszubauen. Dadurch konnten sich Studierende im Rahmen ihrer eigenen Möglichkeiten beim Thema Klimaschutz (z.B. Reduzierung des CO_2-Ausstoßes) positiv einbringen und gleichzeitig ihre Kenntnisse in curriculum-relevanten Themen ausbauen.

Nach der erstmaligen Durchführung des Projektes in diesem Jahr sollten Randbedingungen geschaffen sein, um das Projekt in nachfolgenden Semestern mit ähnlichen oder erweiterten Fragestellungen durch weitere Studierende erneut bearbeiten zu lassen. Dadurch stand in dem Projekt von Anfang an auch das Thema Nachhaltigkeit im Vordergrund, so dass allen Beteiligten bewusst war, dass ihre Ergebnisse in Zukunft weiter verwendet werden.

Das gesamte Projekt besteht aus vier Phasen, die im Folgenden beschrieben sind. Davon sollten Phase 1 und 2 in einem ersten Semester und Phasen 3 sowie 4 im darauf folgenden Semestern umgesetzt werden.

A. *Phase 1*

Hier steht zunächst das Erlernen und Erkennen der Zusammenhänge zwischen der klassischen Erzeugung elektrischer Energie z.B. aus fossilen Energiequellen (CO_2-Ausstoß) und der volatilen Erzeugung elektrischer Energie aus alternativen Energiequellen im Vordergrund. Im nächsten Schritt soll Wissen über die zeitliche Abhängigkeit der elektrischen Energieflüsse im Haushalt bei Betrieb von Großverbrauchern wie Wärmepumpen, Elektrofahrzeugen, Waschmaschinen und Wäschetrocknern sowie über die zur Steuerung der Verbraucher benötigten Datenflüsse zur Koordination der alternativen Energieerzeuger und der Verbraucher erlangt werden.

B. *Phase 2*

Nach der Vermittlung der physikalischen und digitalen Zusammenhänge soll in einer Simulationsumgebung eines Haushaltes mittels Algorithmen die solare Energie zwischen den Großverbrauchern (Waschmaschine, Wäschetrockner, Elektrofahrzeug, Wärmepumpe) so intelligent verteilt werden, dass der Bezug vom Energieversorger minimiert wird. Dazu muss allerdings zuerst eine Simulationsumgebung aufgebaut werden, die Kern des Projektes ist.

C. *Phase 3*

In der dritten Phase sollen alle gesammelten Erkenntnisse in die Entwicklung eines Energiemanagement-Systems fliessen, das über eine zu entwickelnde Web-Oberfläche gesteuert werden kann. Zudem sollen hier bereits intelligente Algorithmen für das Energiemanagement-System entwickelt werden.

D. *Phase 4*

In der letzten Phase soll zudem ein Demonstrationslabor aufgebaut werden, in dem die physikalischen Zusammenhänge und die Steuerungsmöglichkeiten des Smart Grids anhand von Modellen der Haushaltsgeräte und Solaranlagen gezeigt werden sollen.

III. Die Projektumsetzung

Das Projekt ist in den typischen Phasen eines Lernprojekts abgelaufen [5]:

A. *Vorbereitung*

Der Rahmen für die Projektarbeit wurde von der betreuenden Dozentin festgelegt. Sie hat das Thema „Simulationsumgebung für Smart-Grid-Anwendungen“ definiert und ebenfalls festgelegt, welche Kompetenzen gefördert werden sollen: Analyse und Entwurf verteilter Informationssysteme, Softwareentwicklung mit aktuellen Technologien, Teamarbeit, Projektmanagement in einem agilen Umfeld sowie Softwarequalitätsthemen. Der zeitliche und organisatorische Rahmen waren durch die Struktur des Lernmoduls und vom Semesterplan vorgegeben.

Als ein wichtiger Vorbereitungsschritt für dieses Projekt wurde von der Betreuerin bereits ein Prototyp der Simulationsumgebung entwickelt. Dadurch konnte sie im Detail Vor- und Nachteile bestimmter Architekturen benennen,

kannte ebenfalls bereits viele mögliche Stolpersteine, über welche die Studierenden im Laufe des Projektes stolpern könnten, und wusste auch im Detail, welche Vorkenntnisse aus dem Studium die Studierenden bei dem Projekt nutzen können. Dadurch war der Lösungsweg für die Studierenden zwar nicht direkt vorgegeben, allerdings war die Betreuerin stets in der Lage, sie bei potentiellen Fehlentscheidungen durch Diskussionen und Hinweise wieder in die richtige Richtung zu lenken.

B. Einstieg

Das Thema hat bei den Studierenden einen sehr großen Anklang gefunden. Es war ursprünglich für eine Gruppe von fünf bis maximal sieben Studierenden geplant. Von 24 Studierenden wollten 22 das Thema bearbeiten. Schließlich wurden daraus zehn besonders motivierte Studierende für das Projekt ausgewählt, welche auch genügend Vorkenntnisse hatten, um sich im Rahmen des Projektes auf neue Themen konzentrieren zu können.

Die Studierenden wurden zu Semesterbeginn in die Problemstellung sowie in Methoden eingeführt, mit denen sie ein eigenes Projekt entwickeln können. Sie erhielten ebenfalls eine fachliche Einführung in das Thema der Projektarbeit. Eine thematische Gruppenteilung hat sich in dem Projekt relativ früh entwickelt.

C. Planung

Durch die zeitlichen Randbedingungen eines Semesters und die inhaltlichen Zielvorstellungen für das Projekt war der Releaseplan von der Betreuerin des Projektes vorgegeben – das ungefähre Releasedatum entsprach dem Ende der Vorlesungszeit mit einem potentiellen Puffer nach der Prüfungsphase. Auch die groben Features waren vorgegeben – die Betreuerin übernahm dabei die Rolle eines Kunden.

Den ersten groben Planungsschritt und alle weiteren Detailpläne für die einzelnen Iterationen mussten die Studierenden selbst vornehmen. Sie hatten die Aufgabe, das endgültige Projektziel selbst zu formulieren und grobe Meilensteine zu definieren, um es zu erreichen.

Die Planung wurde bei den wöchentlichen Statusmeetings immer wieder überprüft und die Meilensteine verfeinert: Für jeden Meilenstein wurden gemeinsam Lösungsschritte festgelegt und daraus Arbeitsaufgaben für einzelne Teammitglieder abgeleitet. Diese Aufgaben wurden vor allem von zwei Studierenden mit besonderen Rollen im Projekt getrieben:

Der Product Owner hat das Projektziel aus der fachlichen Sicht definiert und immer wieder kritisch hinterfragt, ob alle getroffenen Entscheidungen zum Ziel führen und neue Detailfragen gestellt, die im Team beantwortet wurden.

Der Projektleiter hat die Termin- und Ressourcenplanung für jede Iteration übernommen und wöchentlich den groben Projektplan angepasst und die anstehenden und zurückgestellten Features mit den Teammitgliedern abgestimmt und daraus resultierende Aufgaben in Form von Issues dokumentiert.

Die agile Vorgehensweise stand im Mittelpunkt des Projektes. Daher gab es keine explizite und abgeschlossene Planungsphase zum Projektanfang, sondern die Planung wurde jede Woche kritisch hinterfragt und angepasst.

D. Umsetzung

Die geplanten Arbeitsschritte wurden im wöchentlichen Rhythmus durchgeführt und sowohl der Prozess als auch die Ergebnisse dokumentiert. Die technischen Herausforderungen bei der Umsetzung waren vor allem:

- Konzeption einer verteilten Architektur mit asynchroner Kommunikation zwischen den Komponenten. Alle simulierten Komponenten (Waschmaschine, Trockner, Wallbox und die Solaranlage) wurden als voneinander unabhängige 3-Tier-Komponenten mit einem eigenen Web-Server und einer web-basierten Bedienoberfläche konzipiert und entwickelt. Das zentrale Energie-Management-System (EMS) wurde zudem als eine zusätzliche Komponente entwickelt, die mit allen anderen Komponenten kommuniziert.
- Konzeption einer losen Kopplung zwischen den Komponenten: die simulierten Komponenten melden sich eigenständig beim EMS an und ab, schicken ihm die notwendigen Informationen und bekommen vom EMS Nachrichten, wenn sie bestimmte Aktionen durchführen sollen. Gleichzeitig sind die Komponenten vom Benutzer bedienbar.
- Entwicklung von Standards für die Kommunikation zwischen den Komponenten: dadurch soll ein möglichst hoher Grad an Wiederverwendung erlangt werden.
- Entwicklung von Web-Oberflächen: Für die Bedienung der Komponenten mussten Web-Oberflächen entwickelt werden. Zudem mussten die Schnittstellen so beschaffen sein, dass eine Weiterentwicklung zu echten Hardwarekomponenten mit einem bitmapbasierten Display künftig möglich sein sollte.
- Nutzung aktueller Web-Technologien: HTML, CSS, JavaScript und node.js.
- Qualitätssicherung: Ein Build-Server sollte für die Continuous Integration (CI) aufgesetzt werden, um die Qualität der Projektergebnisse sowohl mit statischen als auch mit dynamischen Tests abzusichern.

Durch diese Rahmenbedingungen sollten die Studierenden vor allem die Aspekte der Analyse und des Entwurfs komplexer Informationssysteme, die Entwicklung verteilter Softwaresysteme sowie Themen der Informationsaufbereitung für unterschiedliche Medien unter Berücksichtigung von qualitätssichernden Maßnahmen kennenlernen.

Dabei wurde darauf geachtet, dass typische Tools aus der Praxis eingesetzt wurden:

- Als Versionsmanagement-Tool wurde GitLab eingesetzt, so dass die Studierenden von Anfang an gelernt haben, ihren Code mit anderen zu teilen und in verteilten Teams zu arbeiten. Zudem wurde von Anfang an der „collective code ownership“-Ansatz eingehalten, so dass alle sich schnell daran gewöhnt haben, dass der Sourcecode nicht einem Entwickler selbst, sondern dem gesamten Team gehört und damit auch von jedem verstanden werden muss.
- Der Entwicklungsprozess wurde mit Hilfe von Issues in GitLab geplant und dokumentiert. Auch die geschätzten und die tatsächlichen Aufwände für die Umsetzungsaufgaben wurden in den Issues hinterlegt, so dass die Arbeitsbelastung der einzelnen Teammitglieder stets transparent war.
- Die Studierenden wurden ebenfalls angeleitet, die entwickelte Software mit gängigen Standards zu doku-

mentieren (JSDoc für die Dokumentation des Sourcecodes sowie UML für die Erstellung von Architekturabbildern).

Dadurch mussten sich Studierende mit typischen Softwaretechnologie-Ansätzen auseinandersetzen und diese auch im Projektalltag nutzen.

E. Auswertung

Da es sich bei dem Projekt um ein Studentenprojekt handelte, das während eines Semesters stattfand, in dem die beteiligten Studierenden noch andere Lehrveranstaltungen zu besuchen hatten, konnten keine täglichen Abstimmungstreffen stattfinden. Nichtsdestotrotz wurden die Projekt-Zwischenergebnisse wöchentlich vorgestellt, neue Lösungsansätze sowie Erfolge und neue Problemstellungen mit dem gesamten Team diskutiert. Hierbei wurden oft auch gemeinsam Verbesserungsvorschläge erarbeitet und auch Hinweise zur weiteren Betreuung an die Dozentin weitergegeben.

F. Präsentation

Besonders wichtig war in dem Projekt die Vorgabe, alle Ergebnisse an der „MediaNight" der Hochschule vorzustellen. Die MediaNight ist eine hochschulweite Veranstaltung, an der alle Studierenden ihre Projektergebnisse zum Semesterende vorstellen. Da die Veranstaltung von vielen Studieninteressierten, Alumni und Industrievertretern besucht wird, finden die Präsentationen in einem öffentlichen Rahmen statt. Dadurch wird die Bedeutung der Qualität der Projektergebnisse noch mal besonders betont.

G. Abschluss

Mit einer gemeinsamen Feedbackrunde wurde die Projektarbeit beendet. Im Folgenden sind die Ergebnisse dieser Feedbackrunde zusammengefasst.

IV. Kritischer Rückblick

A. Positive Erfahrungen

Das Projektteam war vom Beginn des Projektes bis zum Schluss sehr motiviert. Der hohe Motivationsgrad hat sich schon bei der Anzahl der Teilnehmer bei der Teamzusammenstellung gezeigt und ist im Laufe des Projektes auch nicht geringer geworden. Dies lag vor allem an dem Thema des Projektes, wie typische Antworten einer anonymen Befragung unter den Teilnehmern ergeben haben:

- *Ich habe mich für das Projekt entschieden, weil es ein aktuelles und zukunftsorientiertes Thema ist und ich das Gefühl habe, wir machen nicht „irgendein Projekt" an der HdM, sondern etwas Besonderes.*
- *Ich habe mich für das Projekt Smart Grid entschieden, da die Nutzung von bzw. der Umstieg auf erneuerbare Energien ein immer wichtigeres Thema in der Zukunft im Rahmen des Klimawandels wird und unsere Simulation des Smart Grids dazu beitragen kann, Menschen auch mehr mit dieser zukunftsorientierten Technologie vertraut zu machen.*
- *Ein nachhaltiges Projekt, welches in den kommenden Semestern fortgeführt wird und damit nicht nur ein "Zeitvertreib" ist, bei dem man etwas lernt.*
- *Das Thema beschäftigt sich mit der Zukunft, die uns alle etwas angeht, und zeigt, was nachhaltiges Haushalten mit der Energie für Folgen haben könnte und welche Abhängigkeit doch von Energieversorgern ausgeht.*

B. Erzielte Ergebnisse

Der ursprüngliche Plan für das Projekt sah vor, in einem Semester die Simulationsumgebung aufzubauen (Phase 1 und 2) und die Entwicklung intelligenter Algorithmen bzw. den Aufbau eines Demonstrationslabors (Phasen 3 und 4) in darauf folgenden Semestern fortzuführen.

Dieser Plan war – auch aufgrund der hohen Motivation des Teams – sogar zu konservativ. Das Team hat es während des einen Semesters geschafft, die komplette Simulationsumgebung mit allen Komponenten aufzubauen sowie erste Ansätze für ein intelligentes EMS zu implementieren. Insbesondere wurden dafür generische Schnittstellen geschaffen, die es ermöglichen, in Zukunft beliebige intelligente Algorithmen einzubinden und auszutauschen. Aufgrund der verteilten Software-Architektur, die vom Beginn des Projektes als Paradigma beachtet wurde, konnte die Implementierung ohne weitere Hindernisse auf „Raspberry PI"-Computer übertragen werden. Damit wurde im Rahmen dieses Projektes bereits ein erster Schritt für die Umsetzung eines Demonstrationslabors gemacht.

C. Rückblick auf den Entwicklungsprozess

Am Anfang des Projektes haben die Beteiligten nicht gleich verstanden, warum sie einen geregelten und gut organisierten Entwicklungsprozess mit einer klaren Rollenverteilung benötigen, da sie sich alle aus dem Studium bereits kennen und in bisherigen Veranstaltungen auch ohne spezielle Vorgaben und Tools miteinander gearbeitet hatten. Dennoch haben sie im Laufe des Projektes die eingesetzten Methoden und Werkzeuge schätzen gelernt. Folgende Rückmeldungen gaben die Studierenden im Rahmen der anonymen Umfrage:

- *Ich habe gelernt, wie agiles Projektmanagement funktioniert und warum es in Softwareentwicklungsprojekten notwendig ist.*
- *Der Umgang mit Issues wird immer besser. Dadurch sieht man relativ gut, was noch direkt zu tun ist und wer dafür verantwortlich ist. Generell ist die Arbeit verhältnismäßig gut strukturiert, da das Ziel des Projekts ziemlich klar ist und sich nicht geändert hat. Fachlich hat sich die Gruppe gut in kleinere Kompetenzteams aufgeteilt, die ihre Arbeit jeweils meist gut erledigen.*
- *Die Mischung aus eigenem Aneignen von Fachwissen und fachlicher Unterstützung bei Fragen ist bei dem Projekt sehr ausgewogen. Die Aufgabenverteilung über Issues bzw. konkrete Verteilung von Aufgaben über Projektleitung funktioniert sehr gut.*
- *Zusätzlich finde ich es gut, dass es einmal in der Woche ein großes Meeting gibt, bei dem alles Wichtige angesprochen und gemeinsam überlegt wird, wie es weitergehen soll. Das wiederholte Durchsprechen des Anwendungsfalls in den Statusmeetings kam mir anfangs überflüssig und langweilig vor, inzwischen schätze ich es, um den Anwendungsfall mit all seinen Details jederzeit nachvollziehen zu können und auch mitzubekommen, an was andere im Team gerade arbeiten.*

D. Rückblick auf die Teamarbeit

Durch das für Studierende interessante Thema waren alle Beteiligten am Projekt interessiert und hatten so eine hohe Bereitschaft, sich dafür zu engagieren und einzubringen. Jeder einzelne war meistens motiviert und wollte ein möglichst gutes Ergebnis des Projektes erzielen.

Zudem haben die Studierenden gelernt, dass sie im Rahmen eines Projektes nie allein gelassen wurden. Sie bekamen Unterstützung von ihrer Betreuerin, wenn sie diese benötigt haben. Auch dieser Punkt war sehr wichtig und wurde im Rahmen der Projektumfrage von einigen Studierenden betont: „*Ich habe gelernt, mutig zu sein und mir Hilfe zu holen, wenn ich allein nicht weiter kam*“ – diese Aussage eines Beteiligten zeigt die ursprüngliche Einstellung von manchen Studierenden: Fragen zu stellen, wenn man allein nicht weiter kommt, wird oft noch als negativ empfunden. In dem Projekt haben die Studierenden jedoch am praktischen Beispiel erfahren können, dass der offene Umgang mit Schwierigkeiten das gesamte Vorhaben voranbringt.

Zudem haben die Projektmitglieder gelernt, dass die Kommunikation und die Verständigung mit anderen Teammitgliedern sehr wichtig sind. Zum Projektbeginn war die prinzipielle Einstellung im Team, dass alle Entscheidungen demokratisch im Team zu treffen seien und daher ein dedizierter Projektleiter nicht notwendig sei. Im Laufe des Projektes reifte jedoch die Erkenntnis, dass *„… man gerade bei einem sehr großen Team eine Person benötigt, die in verfahrenen Situationen klare Anweisungen gibt und die Gruppe voranbringt.*“.

E. Erreichung der Lernziele

Die Lernziele der Veranstaltungen waren, typische Vorgehensweisen, Werkzeuge und Technologien des Software-Engineerings kennenzulernen bzw. vorhandene Kenntnisse aus den theoretischen Vorlesungen auszubauen. Auch diese Ziele wurden im Projekt erreicht, auch wenn sie nicht explizit betont wurden. Folgende Rückmeldungen gaben die Studierenden auf die Frage, was sie in dem Projekt gelernt haben:

- *node.js, Ajax, Git,*
- *den Weg kennengelernt von: „Es gibt eine Idee, wie man so etwas umsetzen könnte“ bis zur tatsächlichen Umsetzung,*
- *komplett eigenständig arbeiten (in der Vorlesung wurde man durch die Übungen und klaren Aufgabenstellungen ein bisschen an der Hand gehalten),*
- *asynchrone Aufrufe von Funktionen,*
- *zu verstehen, wie Server miteinander kommunizieren,*
- *mit komplexen Softwareaufgabenstellungen umzugehen, diese anzugehen und möglichst gut umzusetzen,*
- *wie Software funktioniert, bei der mehrere Server zusammenarbeiten müssen,*
- *wie Software Design Patterns zum Einsatz kommen,*
- *sinnvolle Planung einer Webanwendung (z.B. wie man Frontend und Backend miteinander vereint).*

V. Herausforderungen

Trotz aller positiver Erkenntnisse aus dem Projekt gab es auch einige Herausforderungen, die bei künftigen Aufsetzen ähnlicher Projekte zu berücksichtigen sind.

Der wichtigste Aspekt betrifft die Vorbereitung des Projektes: Es erfordert eine sehr gute und detaillierte Vorbereitung: der Betreuer muss eine detaillierte Vorstellung vom Ziel und von der Vorgehensweise haben, um den Studierenden den Einstieg in Softwareengineering-Themen zu ermöglichen. Im vorliegenden Fall wurde die geplante Simulationsumgebung von der Betreuerin prototypisch umgesetzt, bevor sie die Aufgabe an die Studierenden weitergab. Zudem mussten alle notwendigen Daten, z.B. Ertragsprofile einer Solaranlage oder die Verbrauchsprofile typischer Haushaltsgeräte aufbereitet werden, damit die Studierenden sich eine bessere Vorstellung von den Daten machen können, mit denen sie im Projekt umgehen müssen.

Aber auch während der Projektlaufzeit muss der Betreuer Zeit investieren, um den Studierenden bei Problemen nicht allein zu lassen. Wie man bei den Rückmeldungen der Studierenden gesehen hat, ist dieser Punkt nicht zu vernachlässigen: In jedem Umsetzungsprojekt gibt es inhaltliche und organisatorische Schwierigkeiten. Es ist unablässig, diese als Betreuer rechtzeitig zu erkennen und den Studierenden unter die Arme zu greifen, um sie nicht zu frustrieren und zu demotivieren.

Bezogen auf den Entwicklungsprozess sollte auch nicht zu viel Entscheidungskompetenz auf die Studierenden übertragen werden. Die Organisation der Aufgabenverteilung gestaltet sich bei zehn Studierenden, welche unterschiedliche Kenntnisse und Arbeitsweisen haben, als eine große Herausforderung. Daher wäre es im Nachhinein für die Studierenden wahrscheinlich besser gewesen, sie hätten von Anfang an einen genaueren Leitfaden für die Vorgehensweise im Projekt und für das Lösen typischer Probleme gehabt. Insbesondere am Anfang des Projektes sollten Rollen und deren Aufgaben vor allem in Bezug auf die Rolle des Product Owners bzw. die Projektleitung im allgemeinen deutlich klarer definiert werden.

Im Nachhinein hätte die inhaltliche Planungsphase länger sein dürfen. Es wurde im Laufe des Projektes sehr viel Code geschrieben, der zum Schluss gar nicht mehr oder nur teilweise benutzt wurde. Das gehört zwar einerseits zum Lernprozess in der Softwareentwicklung und wurde bei den regelmäßigen Retrospektiven auch betont, auf der anderen Seite empfinden es gerade Anfänger als sehr frustrierend, wenn sie etwas implementieren, das später doch nicht mehr benötigt wird. Das hätte mit mehr inhaltlichen Diskussionen am Anfang des Projektes vermutlich verhindert werden können.

Bei den gesetzten Lernzielen mussten im Laufe des Projektes ebenfalls Abstriche gemacht werden: ursprünglich war es geplant, gerade aufgrund der großen Teamgröße von Anfang an qualitätssichernde Maßnahmen für den Sourcecode einzusetzen. Die Idee bestand darin, einen CI-Server einzusetzen, in dem automatische statische Codeanalysen und dynamische Tests durchgeführt werden sollten. Für solche Maßnahmen war das Verständnis der Studierenden zum Projektanfang gar nicht gegeben, deswegen wollte sich auch niemand um die entsprechenden Aufgaben kümmern. Auch wenn die Studierenden selbst im Laufe des Projektes erkannt haben, dass der Sourcecode von zehn Beteiligten unlesbar wird, wenn man keine gemeinsamen Code Conventions benutzt, war die Bereitschaft, sich Regeln zu setzen und diese zu befolgen, sehr gering. Das lag vor allem an der Tatsache, dass die Studierenden aufgrund ihrer geringen praktischen Erfahrung in der Softwareentwicklung sich damit zufrieden gaben, wenn ihr selbst geschriebener Code irgendwie funktionierte – schön musste er dabei nicht sein. Ähnlich verhielt es sich mit den dynamischen Tests: da alle Beteiligten erst im Laufe des Projektes die Erfahrung gemacht haben, dass Änderungen im Sourcecode auch Seiteneffekte haben können, konnten sie zum Projektbeginn nicht verstehen, warum es wichtig ist, automatische Tests auch für scheinbar trivialen Sourcecode zu hinterlegen. Das zweite Problem lag darin, dass unerfahrene Softwareentwickler oft nicht wissen, was sie testen müssen, um Fehler zu finden. So wurde das Projekt ohne automatische Tests und ohne regelmäßige automatische Integrationen durchgeführt. Hierbei war es jedoch wichtig, dass die

Betreuerin regelmäßig den aktuellen Stand des Sourcecodes überprüft und manuelle Tests des aktuellen Standes durchgeführt hatte: so war es in den wöchentlichen Statusmeetings immer möglich, größere Probleme im Source Code gemeinsam zu diskutieren und Auswirkungen typischer Implementierungsfehler zu zeigen. Dadurch haben die Studierenden zwar nicht selbst die Testerfahrung sammeln können, jedoch haben sie ein Gefühl dafür entwickelt, wie labil eine ungetestete Software ist und wie einfach es ist, Fehler in einem scheinbar funktionierenden Sourcecode zu finden, wenn man genau genug hinschaut. Ohne solche Maßnahmen wäre dieser Lernerfolg bei den Studierenden ausgeblieben und auch die entwickelte Software hätte nicht die Qualität, um sie in späteren Semestern weiter entwickelt zu können. Im Laufe des Projektes konnten dadurch auch Reviews zum Ende einer Iteration eingesetzt und sinnvolle Refactoring-Maßnahmen durchgeführt werden. Es ist daher davon auszugehen, dass die Teilnehmer aufgrund ihrer positiven Erfahrung mit regelmäßigen Tests bei einem weiteren Projekt eher bereit wären, die typischen agilen Engineering-Praktiken wie Code Conventions und automatische Tests mit einem CI/CD-Server einzusetzen als es im Rahmen dieses Projektes der Fall war.

Das von der Betreuerin gesetzte Ziel, komplett auf ein agiles Software-Engineering zu setzen, konnte daher nur zum Teil erreicht werden. Oberflächlich betrachtet wurden einige Projektmanagement-Praktiken umgesetzt (Releaseplanung, Iterationsplanung, Arbeiten mit User Stories, bei deren Erstellung die Betreuerin die Kundenrolle übernahm). Auf der anderen Seite mussten teilweise auch hier Abstriche gemacht werden: So haben sich die meisten Studierenden bis zum Schluss nicht getraut, selbst zu entscheiden, welche Aufgaben sie als Nächstes bearbeiten sollten, obwohl ihnen nie der Eindruck vermittelt wurde, sie könnten eine falsche Entscheidung treffen. Die Aufgabenverteilung wurde von den Teammitgliedern zwar immer mit dem Projektleiter oder Product Owner diskutiert, aber am Ende haben sie dann doch erwartet, dass der Projektleiter entscheidet, wann was gemacht wird. Dieses Problem kann nur mit der mangelnden Erfahrung der Studierenden und dem dadurch noch fehlenden Selbstbewusstsein bei Themen der Softwareentwicklung erklärt werden.

Ähnliches galt für das Thema Aufwandschätzung für einzelne Aufgaben: ursprünglich haben sich die Studierenden nicht getraut, eine Schätzung abzugeben. Deswegen wurden in den regelmäßigen Retrospektiven nicht nur die entwickelten Features diskutiert, sondern auch die geschätzten und die tatsächlichen Aufwände miteinander verglichen: dadurch lernten die Studierenden eher, die potentielle Komplexität einer Aufgabe zu erkennen und den möglichen Aufwand für ihre Lösung zu schätzen.

Zudem lag es in der Verantwortung der Betreuerin darauf zu achten, dass die Studierenden aufgrund großer Begeisterung für das Projekt ihre sonstigen Module in dem Semester nicht vernachlässigen. Deswegen war hier eine regelmäßige Kontrolle der Aufwände notwendig, um die übereifrigen Studierenden auch mal zu bremsen.

VI. Fazit

Das hier beschriebene Vorhaben wurde im Sommersemester 2019 erstmalig durchgeführt. Ziel war es, den beteiligten Studierenden auf der Basis eines Softwareentwicklungsprojektes Themen des agilen Software-Engineerings nahe zu bringen. Dazu gehörten neben der reinen Softwareentwicklung auch das Softwaredesign, Projektmanagement und Qualitätssicherungsthemen.

Das Vorhaben hat vor allem gezeigt, dass Studierende auch ein fachlich komplexeres Projekt hoch motiviert angehen, wenn sie sich mit dem behandelten Thema identifizieren können. Daher ist die Themenwahl für das Gelingen eines solchen Vorhabens besonders wichtig.

Zudem konnte gezeigt werden, dass Studierende mit Grundlagenkenntnissen sich selbst im Rahmen eines Entwicklungsprojektes neue Kenntnisse und Fähigkeiten erarbeiten können. Die Voraussetzung dafür ist allerdings, dass das Projekt dafür gut genug vorbereitet wird und die Studierenden bei Problemen rechtzeitig Unterstützung bekommen, wenn sie allein nicht weiterkommen. Sowohl fachlich als auch im Bereich der Softskills haben Studierende in dem Projekt sehr viel gelernt.

Als eine besondere Herausforderung für den Betreuer eines solchen Projektes wurden vor allem Planungsaspekte identifiziert: nicht nur das gesamte Vorhaben muss sehr detailliert vorbereitet werden, auch die Einarbeitungsphase während des Projektes darf nicht zu kurz ausfallen, um den Studierenden eine Chance zu geben, sich mit den Zielen und möglichen Lösungsalternativen auseinanderzusetzen. Zudem gibt es in Softwareentwicklungsprojekten Aspekte, deren Sinn von nicht erfahrenden Studierenden nicht immer erkannt wird. Im vorliegenden Fall waren es vertiefte Qualitätssicherungsmaßnahmen, welche von den Studierenden nicht im geplanten Umfang umgesetzt worden sind. Erst am Ende des Projektes wurde der Zweck von solchen Maßnahmen von den Studierenden langsam gesehen. Auch dieses Beispiel zeigt, dass Studierende agile Software-Engineering-Techniken in der Praxis sehr gut lernen können, allerdings muss die Aufgabenstellung zu den vorhandenen praktischen Erfahrungen der Studierenden passen.

References

[1] H. Balzert, Lehrbuch der Sofwtaretechnik, Heidelberg: Springer Spektrum, 1997.

[2] GHD, „HUMUS plus," 2019. [Online]. Available: http://www.hochschuldidaktik.net/index.php?lg=de&main=Foerderung_inno&site=08:01:00. [Zugriff am 15 April 2019].

[3] A. Gottburgsen, K. Wannemacher, J. Wernz und J. Willige, „INGENIEURAUSBILDUNG FÜR DIE DIGITALE TRANSFORMATION: Zukunft durch Veränderung, VDI-Studie," VDI, 2019.

[4] „Smart grid," Wikipedia, 7 Mai 2019. [Online]. Available: https://en.wikipedia.org/wiki/Smart_grid. [Zugriff am 12 Mai 2019].

[5] K. Reich, „Projektarbeit," [Online]. Available: http://methodenpool.uni-koeln.de/download/projektmethode.pdf [Zugriff am 30. Oktober 2019].