Software-Engineering Projekte in der Ausbildung an Hochschulen – Konzept, Erfahrungen und Ideen

Dr. Jens Liebehenschel, metio GmbH, Liederbach

Jens.Liebehenschel@metio.de

Überblick

In vielen Studiengängen wurden gegen Ende des Studiums Entwicklungs-Projekte in den Studienplan aufgenommen.

Der Autor beobachtete vor mehreren Jahren zwei Dinge: Für viele Absolvierende von Ingenieur- und Informatik-Studiengängen sind wesentliche Schritte im Entwicklungsprozess und deren Zusammenhänge nicht klar. Trotz oft detaillierter Vorgaben in den Firmen müssen noch immer viele Aspekte der Software-Entwicklung ausgestaltet werden.

Basierend darauf wurde ein Projekt mit starkem Praxisbezug konzipiert und seit 2007 sieben Mal an der FH Frankfurt und der TH Mittelhessen durchgeführt und kontinuierlich verbessert. Die Studierenden sollen die Möglichkeit haben, in einem „geschützten Raum“ eigene Erfahrungen zu machen, da Erfahrung im Rahmen einer Vorlesung schwer weitergegeben werden kann.

Das Ziel des Papers ist eine ausführliche Beschreibung der einzelnen Aspekte des Konzepts und eine Diskussion der Erfahrungen. Des Weiteren wird ein kurzer Ausblick zu angedachten Erweiterungsmöglichkeiten gegeben und am Ende werden die wesentlichen Punkte zusammengefasst.

Beschreibung des Projekts

Das Projekt *Objektorientierte Softwareentwicklung für Steuergeräte* wurde für Studierende in Informatik-Studiengängen am Ende ihrer Hochschulausbildung konzipiert. Eines der wichtigsten Kriterien bei der Konzeption war die Praxisnähe.

Drei Entwicklungsteams mit jeweils vier Personen arbeiten im Projekt an der gleichen Aufgabe, die Benotung erfolgt individuell. In den Teams ist eine Steuerung von Ampeln an einer Kreuzung oder eine Aufzugsteuerung zu entwickeln.

Für das Projekt existieren zwei entsprechende Modelle, außerdem drei unterschiedliche Sätze von Evaluationsboards mit automotive Controllern unterschiedlicher Leistungsklassen. Bilder der Modelle sind unter (metio Webseite, 2012) zu finden.

Das Projekt deckt die typischen Schritte in der Embedded-Software-Entwicklung von der Anforderungsanalyse mit dem Kunden bis hin zur Implementierung in C++ und zum Test ab (Hruschka, 2002; Automotive SPICE, 2007).

Durch Abgaben und Präsentationen sowie die Arbeit in Entwicklungsteams samt Planung der Arbeitspakete wird die Realität von Projekten bereits in der Hochschulausbildung vermittelt. Insbesondere die Praxisnähe macht das Projekt für Studierende interessant.

Alle Teilnehmer müssen mindestens zwei Mal die selbst erarbeiteten Abgaben vor der Gruppe präsentieren.

Jedes Team ist für die rechtzeitige Abgabe mindestens der folgenden Arbeitsprodukte verantwortlich:

- Planung des Projekts
- Anforderungsanalyse
- Architektur und Design
- Dokumentierte Implementierung
- Testkonzept
- Testfälle
- Lessons Learned

Darüber hinaus gibt es optionale Arbeitspakete mit entsprechenden Abgaben, zum Beispiel

- die erneute Anforderungsanalyse und die Überarbeitung von Architektur und Design, wenn eine Anforderungsänderung in das Projekt kam,
- die Erstellung einer Testsuite, mit deren Hilfe der Compiler der zur Verfügung stehenden Entwicklungsumgebung auf die Güte der C++-Unterstützung untersucht wird oder

- die Erstellung einer Simulation der Hardware wegen der Verzögerung ihrer Verfügbarkeit.

Diese zusätzlichen Abgaben erlauben die Variation des Projekts und die Anpassung an unterschiedliche Studiengänge mit unterschiedlichem zeitlichem Umfang für die Studierenden. Neben unterschiedlichen Arbeitspaketen wird die Lehrveranstaltung abgewandelt durch die Verwendung von verschiedenen Controllern oder durch abgewandelte Anforderungen an das System.

Innerhalb der Teams erfolgen zu zwei Zeitpunkten gegenseitige Bewertungen. Diese sind nur dem Dozenten bekannt. Außerdem finden in dem Rahmen kurze Einzelgespräche mit allen Teilnehmern statt. Natürlich ist dies bei Bedarf auch zu anderen Zeitpunkten möglich.

Ziele des Projekts

Das Projekt hat mehrere Ziele, die im Anschluss detaillierter beleuchtet werden:

- Verständnis der Besonderheiten eingebetteter Systeme bei den Studierenden
- Begeisterung für die Welt eingebetteter Systeme
- Fachliche Weiterentwicklung der Studierenden durch einen Schritt von der Hochschulausbildung hin zur Praxis
- Persönliche Weiterentwicklung der Studierenden durch Übernahme von Eigenverantwortung
- Spaß am Lernen und an der Wissensvermittlung

Eingebettete Systeme und deren Entwicklung haben viele Gemeinsamkeiten mit nicht eingebetteten Systemen, aber auch einige Unterschiede. In erster Linie sind es oft harte Echtzeit-Anforderungen, begrenzte Ressourcen und eine starke Nähe zur Hardware (Barr, 1999; Kienzle, 2009). Die Entwicklung orientiert sich meist stark an den Qualitätsmerkmalen Verfügbarkeit, Sicherheit (Betriebssicherheit und Angriffssicherheit), Kosten und Management der Varianten- und Versionenvielfalt. In der Realisierung kommen häufig zeitgesteuerte Systeme zur Anwendung und die erstellte Software läuft nicht auf dem Computer, auf dem sie entwickelt wurde. Diese Unterschiede und deren Auswirkungen für die Entwicklung sind insbesondere Studierenden der Informatik häufig nicht bewusst. Das Projekt schafft dieses Bewusstsein.

In der Praxis werden im Zeitalter einer immer höheren Durchdringung der Welt mit Computern (Mattern, 2012) noch mehr Personen für die Entwicklung eingebetteter Systeme benötigt. Das Projekt zielt darauf ab, bei den Studierenden Begeisterung für dieses interessante Themengebiet zu wecken. Die Begeisterung resultiert auch aus dem zu entwickelnden Endprodukt: Die Ampeln leuchten und lassen sich durch Knöpfe beeinflussen und der Aufzug bewegt sich.

Am Ende der Hochschulausbildung ist es notwendig, den Einstieg der Studierenden in die Praxis zu erleichtern. Dazu gibt es in vielen Informatik-Studiengängen Projekte oder Praktika. Die im Laufe des Studiums erlernten Bausteine werden individuell bei Bedarf vertieft. Insbesondere müssen sie aber kombiniert werden, um die Abhängigkeiten und den Nutzen besser zu verstehen. Dies unterstützt die Einordnung der späteren Tätigkeit in den gesamten Entwicklungsprozess und damit auch das Verständnis für das Zusammenwirken aller notwendigen Schritte in der Produktentwicklung.

Neben der fachlichen Weiterentwicklung wird auch im Rahmen der Möglichkeiten an geeigneten Stellen an der persönlichen Weiterentwicklung der Studierenden gearbeitet (Vigenschow, 2007). Im Projekt treten immer wieder schwierige Situationen wie beispielsweise Konflikte auf. Es werden bei Bedarf erste Lösungsansätze vermittelt und die Umsetzung in der Praxis diskutiert. Außerdem wird den Studierenden das Verständnis Ihrer Eigenverantwortung näher gebracht.

Bei vielen Studierenden konnte eine große Weiterentwicklung der praxisrelevanten Fähigkeiten beobachtet werden – sicherlich unterstützt durch die große Freude an der Projektarbeit. Nicht zuletzt wird das zeitaufwändige Projekt immer wieder durchgeführt, weil die Wissensvermittlung sehr viel Spaß macht.

Konzept und Erfahrungen

In diesem Kapitel werden die im Projekt verwendeten didaktischen Konzepte vorgestellt. Mehrere Punkte entsprechen den Rahmenbedingungen aus (Kleuker, 2011).

Praxisnähe

Ein wesentlicher Aspekt des Konzeptes ist die Fokussierung auf die Praxisnähe. In der Umsetzung wird die zur Verfügung stehende Zeit weniger für ein schwierig zu erstellendes System verwendet. Vielmehr soll ein Verständnis für die Phasen im Projekt, die notwendigen Arbeitsprodukte und Ihre

Abhängigkeiten geschaffen werden. Am Anfang des Studiums liegt der Fokus eher auf einzelnen Bausteinen, am Ende des Studiums sollten sie geeignet zusammengesetzt werden. Das zu lösende Problem und die Implementierung sind wenig komplex, aber doch nicht trivial. Dies zeigt sich sowohl bei der Anforderungsanalyse als auch im weiteren Projektverlauf, wenn im Team festgestellt wird, dass die Anforderungen doch nicht eindeutig oder komplett waren.

Wie schon eingangs beschrieben werden die typischen Schritte in der Embedded-Software-Entwicklung von der Anforderungsanalyse mit dem Kunden bis hin zur Implementierung und zum Test abgedeckt. Realistische Projektsituationen werden partiell durchlebt. Dazu müssen entsprechende Arbeitspakete bearbeitet und vor der Abgabe durch die Studierenden aus dem Team einem Review unterzogen werden. Dies dient dem Verständnis des Inhalts von Arbeitsprodukten und zeigt den Studierenden den Zusammenhang der in Entwicklungsprojekten benötigten Dokumentation.

Die Ergebnisse müssen durch die Ersteller vor allen Gruppen präsentiert werden. Dabei ist nicht unbedingt das Ziel, aus den Abgaben Folien zu erstellen, sondern die Präsentation basierend auf den abgegebenen Arbeitsprodukten zu gestalten. So werden die Anforderungen anhand eines Textdokuments oder die Architektur anhand von Architektursichten in Architektur-Werkzeugen oder in anderen Dokumenten vorgestellt. In der Praxis werden Themen oft anhand der Arbeitsprodukte besprochen, ohne aufwändige Präsentationen zu erstellen. An diesen Stellen werden die Studierenden immer wieder angeleitet, Dokumentation im sinnvollen Umfang anzufertigen. Sie soll der Kommunikation dienen und verständlich sein.

Schließlich ist für die Projekte in der Praxis gegenüber den kleinen Projekten im Studium an den Hochschulen die Teamarbeit sehr viel wichtiger. Dies wird den Studierenden nahe gebracht, in dem einerseits die Aufteilung der Arbeit in parallel zu erstellende Arbeitspakete notwendig ist, andererseits aber die Arbeitspakete auch voneinander abhängen, so dass Abhängigkeiten der verschiedenen Mitglieder in den Teams bestehen und eine gute Zusammenarbeit unerlässlich ist.

Überraschendes

Ein weiterer wichtiger Baustein ist das Erzeugen von Aha-Effekten. Dabei wird das Ziel verfolgt, bei den Studierenden wichtige Aspekte im Gedächtnis zu verankern, weil sie diese nicht nur erzählt bekommen, sondern selbst erleben. Dies geschieht zum Beispiel durch starke Übertreibung, bewusste Fehlinterpretationen oder das „Vorhalten eines Spiegels“.

Bei der Anforderungsanalyse gibt sich der Kunde (der Autor) zunächst bewusst unkooperativ und dreht den Studierenden das Wort im Mund herum oder ignoriert Ihre Fragen. So merken die Studierenden, dass die Ermittlung von Anforderungen schwierig ist. Nachdem dieses Verhalten des Kunden beleuchtet wurde, gibt sich der Kunde kooperativer, damit die Studierenden ihrem Ziel nach eindeutigen und vollständigen Anforderungen näher kommen können.

Wie zuvor angesprochen ist ein weiterer wichtiger Aspekt die Vermittlung von Wissen über den Zusammenhang der Arbeitsprodukte im Software Engineering. Beispielsweise basiert das Design auf Entscheidungen, diese wiederum auf Anforderungen. Diese Zusammenhänge verstehen die Studierenden durch ihre Abgaben. Meistens wird in das Projekt mindestens eine Anforderungsänderung eingebaut. Das Ziel ist es, den Studierenden Konsequenzen dieser in der Praxis üblichen Situation zu zeigen. Es soll das Bewusstsein geschärft werden, dass in solchen Situationen verschiedenste Arbeitsprodukte zu überarbeiten sind.

Oft ist bei Studierenden der genaue Anteil der Implementierung am Gesamtumfang des Projektes nicht bekannt. Die Studierenden werden der Reihe nach gefragt, wie hoch sie den Anteil schätzen. Am Ende nennt der Autor einen Anteil von 10-15%. Natürlich hängt dies stark vom Umfang des Projekts, der Domäne und der Entwicklungsansätze ab. Da meist die Schätzungen deutlich darüber liegen, wird der Aufwand für andere Aufgaben wie beispielsweise Anforderungsanalyse oder Änderungsmanagement erläutert.

Insbesondere für Studierende ohne vorherigen Kontakt mit der Entwicklung eingebetteter Systeme ist die Erfahrung interessant, dass die Arbeit mit Evaluationsboards sehr zeitraubend sein kann. Die Ausführungsumgebung für die Programme ist auf einem vom Entwicklungs-PC getrennten Computer, der in Betrieb genommen werden muss. Außerdem muss die Software geladen werden, und darüber hinaus ist die Dokumentation oft unvollständig oder fehlerhaft.

Rollenspiele

Das Projekt lebt von Rollenspielen. Der Dozent schlüpft wie im vorherigen Abschnitt beschrieben beispielsweise in die Rolle des Kunden, erläutert immer zu geeigneten Zeitpunkten was gerade passiert ist und gibt Lösungsmöglichkeiten vor, die dann im weiteren Verlauf erprobt werden können. Eine wichtige Erfahrung bei den Rollenwechseln ist, den Studierenden jederzeit klar zu machen, in welcher Rolle man sich gerade befindet, da es sonst verwirrend ist.

Gerade in der Rolle eines Kunden kann bewusst eine ganz andere „Sprache“ verwendet werden. Damit kann den Studierenden aufgezeigt werden, dass es oft nicht möglich ist, sich mit Kunden über das System mit den technischen Begriffen der Entwickler zu unterhalten, sondern die Unterhaltung in der Sprache des Kunden geführt werden muss.

Ein Erlebnis des Autors war die Bemerkung einiger Studierender, dass das Verhalten in der Rolle des Kunden während der Anforderungsanalyse absurd sei. Dies wurde vom Autor abgeschwächt – natürlich hatte er aus didaktischen Gründen übertrieben. Erstaunlicherweise berichteten andere Studierende mit bereits vorhandener Praxiserfahrung, dass genau solche Situationen so oder so ähnlich wirklich in der Praxis vorkommen.

In einem Projekt gab es in einem Team sehr große Probleme, so dass diese trotz mehrerer Gespräche mit einzelnen Personen und im Team nicht gelöst werden konnten. In diesem Fall musste der Autor eingreifen, um den Projekterfolg nicht zu gefährden und allen Beteiligten realistische Noten geben zu können. Der Dozent schlüpfte dazu in die Rolle des Managers und unterteilte das Team. Zwei neue Teams mussten nur noch einen Teil der Aufgaben bearbeiten. Die Kommunikation-Schnittstelle wurde auf das absolut notwendige Maß reduziert.

Eigenverantwortung

In vielen Firmen existieren für System- und Software-Entwicklungs-Projekte sehr detaillierte Regelungen und andere Vorgaben, welche Arbeitsprodukte wie aussehen, welche Inhalte vorhanden sein müssen, wie sie ineinandergreifen, usw. Dennoch ist es oft nicht klar, wie genau ein zu erstellendes Arbeitsprodukt aussieht oder wie an die Erstellung eines Arbeitsprodukts herangegangen werden kann. Es ist in der Praxis nicht sinnvoll – wenn nicht sogar unmöglich – das Vorgehen in jedem einzelnen Spezialfall ganz feingranular zu beschreiben. Daher sind die an der Entwicklung beteiligten Personen immer wieder in der Situation, eigenverantwortlich die richtigen Entscheidungen zu treffen. Das Vorgehen muss pragmatisch sein. Einerseits muss im Sinne der Vorgaben gearbeitet werden, andererseits sollte der Aufwand in einem vertretbaren Maß bleiben. Diese Abwägung ist in der Praxis sehr wichtig.

Die Studierenden machen eine Projektplanung, ohne dass zunächst detaillierte Vorgaben zu den erforderlichen Schritten – wie die zu erstellenden und abzugebenden Arbeitsprodukte – vorhanden sind. Auch zu zeitlichen Zusammenhängen im Projekt ist ihnen außer den Präsenzterminen und dem nächsten Abgabetermin nichts bekannt. Nach der ersten Abgabe erhalten sie alle weiteren Abgabetermine mit den Arbeitsprodukten. Die manchmal dadurch notwendige Überarbeitung der Planung wird zusammen mit den anderen Arbeitsprodukten nach der Anforderungsänderung abgegeben.

Durch die geforderte Planung der Arbeitspakete ist eine intensive Beschäftigung mit Inhalt und Dauer der erforderlichen Schritte und deren Abhängigkeiten unerlässlich. Diese Herangehensweise erscheint vielen Studierenden zunächst sehr herausfordernd, weil die Freiheitsgrade in vorherigen Studienabschnitten eher gering sind. Durch diese Herausforderungen haben die Studierenden die Möglichkeit der Weiterentwicklung ihres Verständnisses der Software-Entwicklung, sie werden spezifisch gefördert.

Nicht nur das Management des Projektes mit Aspekten wie Zeitplanung und Verteilung von Arbeitspaketen liegt komplett in der Hand der Studierenden. Auch viele technische Aspekte wie geeignete Ablage der entstehenden Arbeitsprodukte in einer Versionsverwaltung und verwendete Werkzeuge und Inhalte der einzelnen Arbeitsprodukte werden durch die Studierenden eigenverantwortlich gestaltet. Zum Beispiel werden für die Dokumentation der Architektur keine Sprachen wie UML (UML, 2012) oder SysML (SysML, 2012) vorgegeben. Und wenn ein Team sich für eine dieser Modellierungssprachen entscheidet, dann muss es außerdem ein geeignetes Werkzeug auswählen und entscheiden, welche Sichten auf das System vorhanden sind und wie sie ausgestaltet werden (Zörner, 2012).

Erfahrungsgemäß fällt es den Studierenden schwer, ein Testkonzept zu erstellen. Es scheint zunächst schwierig zu sein, die Inhalte eines Testkonzeptes zu erfassen. Nach Diskussion über das Thema verstehen die Studierenden, dass es nichts anderes ist, als das Vorgehen in den Testphasen zu beschreiben und die Art der Tests samt Hintergründen zu erläutern.

Der Weg ist das Ziel

Gerade die genaue Ausgestaltung von Arbeitsprodukten ist auch in der Praxis oft ein Problem, wie am Anfang des letzten Abschnitts angesprochen.

Daher wird im Projekt als Aufgabenstellung ein Ziel vorgegeben, jedoch nicht der Weg dorthin. Dies wird den Studierenden am Anfang der Veranstaltung auch mitgeteilt, damit sie von Anfang an Klarheit über den Ablauf des Projektes haben.

Den Studierenden werden immer wieder – auch angeregt durch Abgaben oder Fragen – Denkanstöße zum Weg der Problemlösung gegeben. Einige Beispiele dazu sind im nächsten Abschnitt zu finden. Durch Hinterfragen der gegangenen Schritte werden die Studierenden an strukturierte Herangehensweisen an Probleme herangeführt. Sie erhalten Anregungen für mögliche Lösungswege. Diese können im Projekt zu einem späteren Zeitpunkt noch genutzt werden, zum Beispiel bei der Überarbeitung der Arbeitsprodukte nach einer Anforderungsänderung. Den Studierenden werden auf diese Weise immer wieder Anregungen zur Selbstreflektion gegeben, um Vorgehen und Verhalten zu analysieren und zu verbessern.

In der Praxis werden Arbeitsergebnisse immer wieder Reviews unterzogen. Auch dies wird im Projekt geübt. Zu jeder Abgabe ist vom gesamten Team ein Review durchzuführen. Neben den Abgaben sind zusätzlich die im Review gefundenen Aspekte mit abzugeben. Damit kann frühzeitig geübt werden, von anderen Personen erstellte Arbeitsprodukte zu verstehen und Fehler aufzudecken. Auch wird klar, dass ein Review auf das Aufdecken von Fehlern zielt und nicht etwa den Ablauf oder Inhalt der Arbeitsprodukte beschreibt.

Bei der Kurzpräsentation der einzelnen abzugebenden Arbeitsprodukte durch den Dozenten ergeben sich bei den Studierenden Fragen. Es werden nicht direkt Antworten gegeben, welche Schritte in welcher Reihenfolge durchgeführt werden müssen. Jedoch merken sie schnell, dass die Antworten sie in die richtige Richtung leiten. Wenn die Studierenden gute Fragen zu stellen, erhalten sie vom Dozenten immer mehr Information. Dies ist auch ein Lernziel, Fragen zu stellen, wenn Dinge nicht komplett verstanden sind.

Vertiefung spezieller Themen bei Bedarf

Bei Bedarf werden im Projekt spezielle Themen vertieft, zum Beispiel bei Fragen der Studierenden oder durch den Dozenten erkannten Unklarheiten. Die Erfahrung zeigt, dass manche Themen in jedem Projekt diskutiert werden müssen. Dies liegt zum Teil an speziellen Eigenschaften von eingebetteten Systemen, aber auch zu allgemeinen Begriffen der Informatik besteht oft Klärungsbedarf.

Beispielsweise werden meistens die Begriffe Architektur und Design (Goll, 2011; Starke, 2008), Safety und Security (Löw, 2010), synchrone und asynchrone Kommunikation (Bengel, 2008), Nebenläufigkeit und Synchronisation (Vogt, 2012) und kritischer Pfad (Wikipedia: Kritischer Pfad, 2012) anhand von Beispielen diskutiert, um ein besseres Verständnis bei allen Studierenden zu erreichen.

Oft werden verschiede Entwicklungsansätze, vom Wasserfallmodell bis hin zu agilen Ansätzen (Summerville, 2007; Vliet, 2008) mit ihren Eigenschaften erläutert und Einsatzbereiche diskutiert.

Meistens besteht bei den Studierenden kein Wissen über unterschiedliche Klassen von Anforderungen (Pohl, 2007). Funktionale Anforderungen sind diejenigen, die vom System komplett erfüllt werden müssen. Qualitätsanforderungen (oder oft auch nicht-funktionale Anforderungen genannt) werden immer zu einem gewissen Grad erfüllt. Wichtig für (Architektur-)Entscheidungen sind die Qualitätsanforderungen (Bass, 2003). Dies kann anhand des folgenden Beispiels veranschaulicht werden.
Es ist ein System zum Sortieren von Daten zu entwickeln (funktionale Anforderung). Dabei ist die mittlere Laufzeit wichtiger als der Speicherplatzbedarf (Qualitätsanforderung). Die Bewertung sieht wie folgt aus:

	Mittlere Laufzeit	Speicher (Stack)
Bubblesort	–	+
Quicksort	+	–

Tabelle 1: Alternativen und Bewertung

Nur mit der Qualitätsanforderung kann eine Entscheidung für eine Alternative getroffen werden:

Quicksort, weil die Laufzeit wichtiger als der Speicherplatzbedarf ist.

Die bei den meisten eingebetteten Systemen vorhandene zeitgesteuerte Ausführung (Kienzle, 2009) ist vielen Studierenden nicht bekannt. Es werden in der Praxis übliche Lösungen diskutiert, die Hintergründe der Ansätze beleuchtet und mit anderen Ausführungsparadigmen verglichen.

Alle diese technischen Diskussionen werden generell mit allen Studierenden gemeinsam geführt. Wenn möglich erklären die Studierenden die Sachverhalte, oder sie werden zumindest angehalten, zur Diskussion beizutragen. Am Ende der Diskussion wird durch Rückfragen geklärt, ob die Inhalte verstanden worden sind.

Die Dauer der Diskussionen richtet sich nach dem Inhalt – von wenigen Minuten bis zu einer Stunde. Natürlich können manche Themen nicht erschöpfend erörtert werden, aber das Verständnis oder Problembewusstsein kann deutlich erhöht werden.

Auch nicht-technische Aspekte werden bei Bedarf aufgegriffen. Wenn beispielsweise bei der Präsentation von Ergebnissen Aspekte verbessert werden können, werden nach Rücksprache mit den Vortragenden diese Themen in den Veranstaltungen angesprochen. Dies können zum Beispiel Hinweise zur Gestaltung der präsentierten Unterlagen oder verwendeten Tools sein. Aber auch Tipps zur Verbesserung der Präsentationsfähigkeiten, zum Beispiel der Rhetorik, werden gegeben.

Oft gehen mehrere Personen eines Teams gemeinsam vor die Gruppe, um Ihre Ergebnisse vorzustellen. Jede Person weiß genau, welchen Teil sie vorstellen wird, jedoch ist nicht klar, wer eine kurze Einführung gibt oder wer beginnt. Das wird vor der Gruppe besprochen. Dieses häufiger beobachtete Phänomen wird abgestellt, nachdem die Studierenden darauf hingewiesen werden. Dadurch werden zukünftige Vorträge – sei es an der Hochschule oder in Unternehmen – professioneller.

Organisatorisches

Am Anfang der Lehrveranstaltung ist es notwendig, den Studierenden das Handwerkszeug aus verschiedenen Themengebieten mitzugeben, um die Aufgaben im Projekt meistern zu können. Dazu findet ein ganztägiger Präsenztermin vor dem Beginn der Vorlesungszeit statt. Jedoch werden schon ab dem zweiten Präsenztermin die Studierenden immer mehr der zur Verfügung stehenden Zeit mit Präsentationen ausfüllen, und spätestens ab dem vierten Präsenztermin reagiert der Dozent nur noch, bringt aber von sich aus keine weiteren Themen ein.

Wie eingangs beschrieben, müssen die Teams regelmäßig Arbeitspakete abgeben. Diese werden nach der Abgabe bewertet und müssen zu einem Präsenztermin, der wenige Tage nach der Abgabe stattfindet, vorgestellt werden. Abgaben und Präsenztermine finden in Abhängigkeit des Aufgabenumfangs alle zwei bis vier Wochen im Umfang von vier Vorlesungsstunden statt.

Als Teamgröße wird vier festgesetzt. Drei Personen pro Team wäre auch möglich. Bei weniger Personen ist der Arbeitsaufwand der Einzelnen zu hoch, bei mehr Personen wird die Aufteilung der Arbeitspakete schwierig. Eine durchgängige Mitarbeit jedes Teammitglieds könnte nicht sichergestellt werden.

Ein weiterer wichtiger Aspekt ist die individuelle Benotung jedes Teilnehmers. Natürlich geht in die Benotung zum kleinen Teil das Teamergebnis mit ein, aber wesentlich sind die Beteiligung und die erstellten Arbeitsergebnisse. Validiert wird die Fairness der individuellen Benotung durch eine gegenseitige Bewertung der Teammitglieder, die zwei Mal durchgeführt wird, in der Mitte und am Ende der Lehrveranstaltung. Diese liefert dem Autor fast immer eine Bestätigung, dass die Sicht im Team auf die Leistungen der einzelnen Mitglieder sich mit seiner deckt.

Der Lernerfolg kann nur dann sichergestellt werden, wenn die Studierenden die Aufgaben selbst bearbeiten und dadurch ihre eigenen Erfahrungen machen. Da oft Arbeitsergebnisse in der Studierendenschaft weitergegeben werden, ist es erforderlich, jedes Semester Änderungen am Projekt zu machen. Diese Änderungen sind zum einen unterschiedliche Themen, aber auch für jedes Thema unterschiedliche Anforderungen und immer wieder andere abzugebende Arbeitsergebnisse. Die Variationsmöglichkeiten wurden eingangs bereits beschrieben.

Das komplette Material für die Veranstaltung erhalten die Studierenden nach dem ersten und zweiten Präsenztermin, das Evaluationsboard gegebenenfalls später – je nach Ausgestaltung des Projektes. Literaturverweise gibt der Autor ausschließlich nach Bedarf aus.

Schließlich soll ein Punkt nicht unerwähnt bleiben. Der Dozent führt die Lehrveranstaltung mit viel Freude durch. Auch bei den Studierenden soll der Spaß an der Arbeit nicht zu kurz kommen. Aber trotz eines guten Arbeitsklimas und einem freundschaftlichem Umgang ist die Wahrung der notwendigen Distanz wichtig – dies ist immer wieder eine Gradwanderung.

Feedback der Studierenden

Letztendlich ist es nicht nur notwendig, dass die richtigen Dinge vermittelt werden, sondern auch eine möglichst große Nachhaltigkeit erreicht wird. Das Projekt soll den Studierenden den Einstieg ins Berufsleben vereinfachen. Dies ist nur möglich, wenn die Studierenden meinen, etwas aus der Veranstaltung „mitzunehmen". Daher wird regelmäßig im Projekt um Feedback gebeten – auch nach der Bekanntgabe der Endnote, um ein möglichst ehrliches Feedback zu erhalten.

Die folgenden Äußerungen von Studierenden werden sinngemäß und verkürzt wiedergegeben. Zunächst eher allgemeine Punkte zum Projekt und im Anschluss ein paar Punkte zu inhaltlichen Aspekten.

- Das Projekt hatte einen hohen Praxisbezug – das gab es sonst im Studium nicht.
- Es war gut, ein komplettes Projekt von Anfang bis Ende zu machen.
- Es war gut, dass keine Prozessanforderungen vorgegeben waren.
- Die Arbeit im Team mit der eigenen Teamorganisation war gut.
- Das Ziel war klar, aber der Weg nicht. Dennoch gab es viel Unterstützung beim Finden des richtigen Wegs.
- Oft wurden nicht alle Informationen gegeben, aber am Ende war doch alles rechtzeitig da.
- Die Praxisnähe war größer als bei allen Veranstaltungen vor dem Projekt.
- Es ist schwierig, vom Kunden gute Anforderungen zu erhalten.
- Die Rollenspiele waren sehr interessant, zum Beispiel die andere Sprache des Kunden.
- Die Einstiegsschwelle in das Thema war niedrig, weil das System einfach ist und keine komplexen Anforderungen besitzt.
- Inhaltlich war das Thema gut: Zunächst einfach und später kamen dann doch noch viele Details und Unklarheiten zu Tage.
- Die Anforderungsänderung war gut, die Auswirkungen auf das Design wurden klar.
- Die Arbeit an der Architektur war gut; ein Mittelweg zwischen zu viel und zu wenig Architektur musste gefunden werden.

Anhand des Feedbacks und vor allem der Diskussionen mit den Studierenden erhält der Dozent den Eindruck, dass der Ansatz zielführend ist. Die Studierenden können die im Studium gelernten Inhalte kombinieren, lernen etwas über deren Abhängigkeiten und erhalten so einen Einblick in die Praxis. Außerdem kann aus dem großen Engagement vieler Studierender geschlossen werden, dass auch die Begeisterung für die Welt eingebetteter Systeme gelungen ist.

Ein letzter Punkt soll nicht unerwähnt bleiben. Das Projekt ist erfahrungsgemäß sehr schnell ausgebucht. Dies gibt dem Autor das gute Gefühl, mit dem Projekt auf dem richtigen Weg zu sein. Die Teilnehmenden entscheiden sich aktiv für das Projekt, um mehr über eingebettete Systeme zu lernen, und bringen daher auch eine hohe Motivation für das Themengebiet mit. Auch diese Tatsache kann ein Grund für das positive Feedback sein.

Einige der ehemaligen Teilnehmer durfte und darf der Autor weiter begleiten. Ein paar Details werden dazu im nächsten Abschnitt vorgestellt.

Ehemalige Projektteilnehmer

Immer wieder haben Studierende aus den Projekten die Möglichkeit, in der Firma des Autors oder bei ihren Kunden Praktika abzuleisten, ihre Abschlussarbeiten anzufertigen, oder als Angestellte in herausfordernden Kundenprojekten zu arbeiten.

In der Zusammenarbeit zeigt sich das erlernte Verständnis für die in der Entwicklung zu erstellenden Arbeitsprodukte. Ein Beispiel ist die Entwicklung von kleinen Werkzeugen. Zunächst wird am Verständnis der Anforderungen gearbeitet, danach werden grundsätzliche Designüberlegungen gemacht. Erst dann wird mit der Umsetzung begonnen.

Auch an einer anderen Stelle zeigt sich eine strukturierte Vorgehensweise. An vielen Stellen müssen Entscheidungen getroffen werden. Dies kann die Auswahl eines Werkzeuges oder eines Frameworks sein, oder auch Designentscheidungen bei der Erstellung von Software. Alternativen werden bewertet und die Entscheidung wird dokumentiert – sie entsteht nicht einfach so und ist nachvollziehbar.

Eine weitere Beobachtung hat der Autor gemacht. Auch wenn die Mitarbeiter sehr selbstständig arbeiten, bitten sie an wichtigen Stellen in den Projekten von sich aus um Reviews, damit sie eine höhere Sicherheit erhalten, sich auf dem richtigen Weg zu befinden.

Ideen für die Zukunft

Das Projekt hat sich über die Jahre immer weiter entwickelt. Drei kombinierbare Ideen für die zukünftige Entwicklung werden kurz aufgelistet:

- Ein Roboter ist anzusteuern, dies erfordert in der Software gegebenenfalls ein Umweltmodell.
- Die Steuerung erfolgt über eine ebenfalls zu erstellende App. Dazu würden neue Controller mit entsprechenden Schnittstellen benötigt.
- Der Teambuilding-Aspekt wird in das Projekt eingebracht, angelehnt an (Schmedding, 2011).

Checkliste als Zusammenfassung

Das Ziel dieser Arbeit ist es, die Konzepte hinter dem mehrfach durchgeführten und verbesserten Projekt offen zu legen und von den Erfahrungen zu berichten. Daher werden als Zusammenfassung abschließend die aus Sicht des Autors wesentlichen Erfolgsfaktoren aufgelistet, um Projekte mit gutem Lernerfolg für die Studierenden durchzuführen.

- ☐ Ist das zu lösende Problem nicht zu komplex?
- ☐ Werden essentielle Schritte der Software-Entwicklung „im Kleinen“ erlebt?
- ☐ Sind die Ziele und nicht die Wege vorgegeben?
- ☐ Werden die Studierenden beim Finden guter Wege hinreichend unterstützt?
- ☐ Präsentieren die Studierenden ihre Ergebnisse?
- ☐ Gibt es eingeplante „Fallstricke“?
- ☐ Werden elementare Aspekte durch Konzepte wie Rollenspiele nachhaltig vermittelt?
- ☐ Werden die Studierenden angeregt, über ihr Vorgehen nachzudenken?
- ☐ Sind die Dozierenden für notwendige Unterstützung gut erreichbar?
- ☐ Besteht ein Team aus drei oder vier Personen?
- ☐ Müssen die Teams das Projekt im Wesentlichen eigenverantwortlich durchführen?
- ☐ Erfolgt die Benotung individuell?
- ☐ Macht die Veranstaltung allen Spaß?

Literatur

Automotive SPICE Prozessassessment (2007), http://www.automotivespice.com/AutomotiveSPICE_PAM_v23_DE.pdf

Barr, M. (1999): Programming Embedded Systems in C and C++

Bass, L. et al. (2003): Software Architecture in Practice, 2nd edition

Bengel, G. et al. (2008): Masterkurs Parallele und Verteilte Systeme

Goll, J. (2011): Methoden und Architekturen der Softwaretechnik

Hruschka, P. et al. (2002): Agile Softwareentwicklung für Embedded Real-Time Systems mit der UML

Kienzle, E. et al. (2009): Programmierung von Echtzeitsystemen

Kleuker, S. et al. (2011): Vier Jahre Software-Engineering-Projekte im Bachelor – ein Statusbericht, SEUH 2011

Löw, P. et al. (2010): Funktionale Sicherheit in der Praxis

Mattern, F., GI Webseite (2012) http://www.gi.de/service/informatiklexikon/detailansicht/article/pervasiveubiquitous-computing.html

metio Webseite mit Bildern der Modelle (2012): Ampel: http://www.metio.de/ger/ampel.html, Aufzug: http://www.metio.de/ger/aufzug.html

Pohl, K. (2007): Requirements Engineering

Schmedding, D. (2011): Teamentwicklung in studentischen Projekten, SEUH 2011

Sommerville, I. (2007): Software Engineering

Starke, G. (2008): Effektive Software-Architekturen

SysML (2012), http://www.omg.org/spec/SysML/

UML (2012), http://www.omg.org/spec/UML/

Vigenschow U. et al. (2007): Soft Skills für Softwareentwickler

Vliet, H. van (2008): Software Engineering

Vogt, C. (2012): Nebenläufige Programmierung

Wikipedia: Kritischer Pfad (2012), http://de.wikipedia.org/wiki/Methode_des_kritischen_Pfades

Zörner, S. (2012): Software-Architekturen Dokumentieren und Kommunizieren