Projekthafte Entwicklung eines regelbasierten Auswertungstools zur Bestimmung der Qualität von funktionalen Anforderungen

Alexander Bartel, Georg Hagel

Fakultät Informatik
Kempten University of Applied Sciences
Bahnhofstraße 61
87435 Kempten
alexander.bartel@hs-kempten.de
georg.hagel@hs-kempten.de

Abstract: Dieser Beitrag beschreibt die Erfahrungen von Lehrenden, welche bei einem Einsatz eines projektbasierten Lehr- Lernarrangements im Bereich des Requirements Engineering mit Studierenden gesammelt werden konnten. Thematisiert werden, neben den Schwerpunkten des Projekts, die Ziele und Rahmenbedingungen der Umsetzung, sowie die eigentliche Durchführung. Es folgt eine abschließende Zusammenfassung des Projekts, in der die gewonnenen Erkenntnisse aus Sicht der Studierenden und Lehrenden dargestellt werden.

1 Einleitung

Die Anforderungsanalyse ist ein grundlegender Bestandteil der Entwicklung eines Softwareprodukts. Hierfür ist es notwendig, dass textuell formulierte Anforderungen ein hohes Maß an Qualität aufweisen, da Fehler oder Inkonsistenzen bei deren Erstellung umso aufwändiger zu beheben sind, je weiter das Projekt fortschreitet. Nach Boehm ist der Aufwand, einen Anforderungsfehler während der Programmierung zu beheben, um 20 Mal höher als es während der Anforderungsanalyse der Fall ist [Boe81]. Dieser Umstand begründet die Notwendigkeit und Wichtigkeit einer qualitativ ausgereiften Anforderungsanalyse, nicht zuletzt aus ökonomischer Sicht.

2 Themenschwerpunkte des Projekts

Um die Qualität von Anforderungen zu sichern, sind in der Literatur zahlreiche Hilfestellungen zu finden, wobei der Qualitätsbegriff in seiner Definition sowie der Messung durch Metriken variiert [RS09, Coc00, Jac95]. Repräsentativ soll für diesen Beitrag und das im Folgenden beschriebene Projekt das SOPHIST REgelwerk herangezogen werden [RS09].

Es besteht aus insgesamt 18 Regeln, welche allesamt das Ziel haben, auf „definierte, systematische Art und Weise mehrdeutige, unvollständige und widersprüchliche Aussagen in Anforderungsdokumenten zu finden“ [RS09, 125]. Das SOPHIST REgelwerk retrospektiv auf bereits erstellte Anforderungen anzuwenden, gestaltet sich in der Regel als äußerst aufwändig [RS09, 160]. Deshalb schlägt Rupp weiterhin eine Qualitätssicherung durch sogenannte Anforderungsschablonen vor. Dabei handelt es sich um „einen Bauplan, der die Struktur eines einzelnen Anforderungssatzes festlegt“ [RS09, 161]. Eine derartige Schablone dient als Unterstützung für Analysten, verschiedene Arten von Systemaktivitäten strukturiert zu erfassen und damit die Qualität von Anforderungen zu steigern. Sowohl das SOPHIST REgelwerk als auch die Anforderungsschablonen sollen in einem Softwareprojekt umgesetzt werden.

3 Projekthafte Umsetzung

3.1 Organisatorischer Rahmen

Das Softwareprojekt ist im 6. Semester des Bachelorstudiengangs Informatik an der Hochschule Kempten angesiedelt. Studierende bearbeiten in Kleingruppen (in diesem Fall: 5 Personen) ein Semester lang selbst-koordiniert ein Thema, welches unter praxisnahen und möglichst realen Bedingungen (beispielsweise definierte Rollen mit einschlägigen Methoden des Projektmanagements) vollzogen wird. In diesem Fall sollten die Studierenden eine Software für Requirements Engineering und Management entwickeln. Sie soll in der Lage sein, die Regeln des SOPHIST REgelwerks auf bestehende Anforderungen automatisiert anzuwenden und sie anhand von diesen zu bewerten. Ebenso sollen Benutzer der Software bei der Eingabe von neuen Anforderungen mittels Anforderungsschablonen unterstützt werden.

3.2 Ziele

Seitens der Lehrenden werden bei diesem lernenden-zentrierten Ansatz die folgenden Ziele verfolgt:

Studierende sollen lernen...

- ... Probleme in Teilprobleme zu strukturieren und damit beherrschbar zu machen.
- ... Entscheidungen zu treffen, daraus Ziele abzuleiten und Verantwortung für die Ergebnisse zu übernehmen.
- ... mit der begrenzten Zeit umzugehen und das Spannungsfeld zwischen Zeit und Qualität zu bewerten.
- ... bekannte Werkzeuge und Techniken aus dem Software Engineering und Projektmanagement anzuwenden und zu kombinieren.

- ... kollaborativ, kooperativ zu arbeiten, miteinander zu kommunizieren und durch gegenseitige Anregungen voneinander zu profitieren.
- ... (Zwischen-) Ergebnisse vor Kunden zu präsentieren.

Ob und inwieweit die Studierenden diese Ziele erreichen, ist unter anderem abhängig von ihrer motivationalen Ausrichtung. Unsere bisherigen Erfahrungen haben gezeigt, dass sich die Motivation von Studierenden durch Maßnahmen steigern lässt, welche sich auf den projektbasierten Ansatz anwenden lassen [FHB13]. Abbildung 1 zeigt vier Felder der Motivation von Studierenden, welche im Lehrkontext positiv beeinflusst werden können [FHB13].

Abbildung 1: Vier Felder der Motivation (nach [FHB13])

Da sich die Studierenden auf eigenen Wunsch hin für ein Projekt anmelden, kann davon ausgegangen werden, dass ein gewisses Maß an Grundinteresse für das zu bearbeitende Thema vorhanden ist. Die Unabhängigkeit von Studierenden ist dadurch gegeben, dass diese frei entscheiden dürfen, wie das Projektziel erreicht werden soll. Beispielsweise werden Kommunikationsprozesse oder Entscheidungen für oder gegen eine Entwicklungssoftware oder ein Vorgehensmodell vollständig in die Hände der Studierenden gelegt. Als Bedingung wird jedoch formuliert, dass jede getroffene Entscheidung begründet werden muss. Lehrende haben lediglich eine beratende Funktion. Es wird nur bei fachlichen Fehlern durch Lehrende eingeschritten. Die Schwierigkeit des Projekts lässt sich anpassen und es wird sich in zwei Stufen dem Projektziel genähert: In der ersten Stufe soll die Eingabeassistenz in Form von Anforderungsschablonen im System umgesetzt werden. Anschließend soll eine Umsetzung des SOPHIST REgelwerks erfolgen, was den Schwierigkeitsgrad entsprechend steigert. Anreize werden auf verschiedene Art und Weise geschaffen. Da den betreuenden Lehrenden zu Beginn des Projekts kein derartiges Produkt bekannt ist, könnte das Projektergebnis auch außerhalb der Hochschule Kempten auf Interesse stoßen. Somit besitzt dieses Projekt eine gewisse Ernsthaftigkeit und Relevanz, was – sofern

die Studierenden sich mit dem Projekt identifizieren – einer der maßgebenden Faktoren für eine erfolgreiche Umsetzung darstellt. Zudem handelt es sich um eine Studienleistung, welche mit 15 ECTS bewertet wird und damit einer Einzelleistung von umgerechnet 450 Personenstunden entspricht.

3.3 Durchführung

Während der Durchführung des Projekts finden regelmäßige Lenkungsausschüsse mit den betreuenden Lehrpersonen statt, welche gleichzeitig als Kunden bzw. Auftraggeber auftreten. Ebenso gibt es projektinterne Treffen, an denen nur die Studierenden teilnehmen. Diese Art des Vorgehens ist orientiert an agilen Vorgehensmodellen wie Scrum. welche unterschiedliche Prozessaktivitäten (z.B. Daily-Scrums) mit unterschiedlichen Projektrollen vorsehen [SS11]. Dadurch können sowohl die fachliche Richtigkeit durch Lenkungsausschüsse als auch realitätsnahe Projektbedingungen geschaffen werden, indem beispielsweise gezielte Anforderungsänderungen in einer fortgeschrittenen Phase des Projekts eingestreut werden. Bisherige Erfahrungen haben gezeigt, dass seitens des Kunden geänderte Anforderungen einen großen Effekt auf das Verständnis der Studierenden in Bezug auf Vorgehensmodelle und erstellte Artefakte haben. In diesem Fall wurden konkret zwei Änderungen festgelegt:

1. Die Studierenden waren dazu angehalten kurz nach Abschluss der Anforderungsanalyse eine Schnittstelle zu schaffen, welche für spätere Weiterentwicklungen relevant ist. Hierbei war es unabdingbar bisher erstellte Artefakte entsprechend anzupassen, sodass die neu hinzugekommene Schnittstelle umgesetzt werden konnte. Neben den Abhängigkeiten der Artefakte untereinander, konnten zudem wichtige Designprinzipien von Software-Architekturen verdeutlicht und angewendet werden.
2. Ebenso wurde nach Abschluss der Implementierung der Wunsch seitens des Kunden geäußert, die schriftliche Dokumentation des Systems in einem anderen Format als ursprünglich vorgesehen auszuhändigen. Dabei ging es nicht darum die Studierenden zu beschäftigen, sondern vielmehr die Dynamik hin zum Kunden bzw. Auftraggeber aufzuzeigen und ein gewisses Maß an Flexibilität einzufordern.

Während der Durchführung entwickelte das Projekt immer mehr Dynamik, da die Studierenden Gefallen daran gefunden haben. So wurden freiwillig weitere Zwischenergebnisse erstellt, wie beispielsweise ein entsprechender Webauftritt über welchen die Software benutzt werden kann.

3.4 Ergebnis

Das Projekt konnte erfolgreich abgeschlossen werden. Das eingangs formulierte Thema wurde vollständig bearbeitet. Lediglich wenige Einzelheiten wurden bewusst ausgelassen,

da sich diese nicht programmiertechnisch abbilden ließen. Dafür wurde die Software um weitere Funktionalitäten ergänzt. Diese wenigen inhaltlichen Anpassungen sprechen für die eingesetzte Methode, da dies die Wichtigkeit von Flexibilität und Anpassungsfähigkeit betont, was auch in der Praxis relevant ist. Abbildung 2 zeigt beispielhaft einen Screenshot der Software. Dargestellt ist der Anforderungsschablonenmodus, in dem ein Benutzer Anforderungen mit Hilfe der genannten Schablonen erfassen kann (1), welche anschließend nach dem SOPHIST REgelwerk bewertet werden (2).

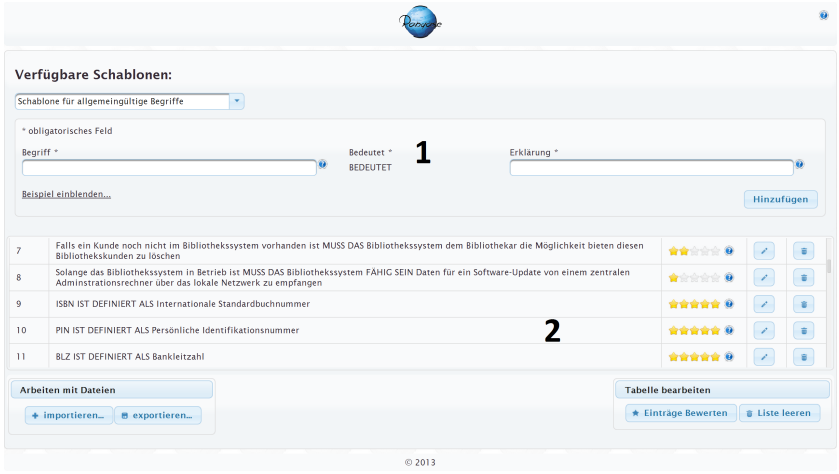

Abbildung 2: Screenshot - Schablonenmodus

Die Abbildung 3 zeigt beispielhaft den Freitextmodus der Software. Hier können beliebige Anforderungen eingegeben werden (3) und nach ausgewählten Regeln des SOPHIST REgelwerks bewertet werden (4). Das Ergebnis der Bewertung wird dem Benutzer angezeigt (5).

4 Lessons Learned für die Lehre von Requirements Engineering

Die Nachbesprechung dieses Lehr- Lernarrangements fand in Form eines Lessons Learned Meetings statt, an welchem die betreuenden Lehrenden und die Studierenden teilnahmen. Diese qualitative Herangehensweise schien aufgrund der geringen Grundgesamtheit (von N=5 Personen) angebracht. Hierbei konnten folgende subjektiven Einschätzungen seitens der Studierenden festgehalten werden:

- Das Verständnis des SOPHIST REgelwerks und damit für qualitätsgesicherte Anforderungen wurde erheblich gesteigert. Der praktische Einsatz wurde zunächst selbstständig explizit von den Studierenden erlernt und während des Projekts implizit angewendet.

Abbildung 3: Screenshot - Bewertung von Anforderungen

- Weiterhin konnte ein iterativ-inkrementeller Prozess der stetigen Verbesserung angestoßen werden. Es war zu beobachten, dass Anforderungen an die Software, die zu Beginn des Projekts formuliert wurden, zunehmend qualitativ hochwertiger wurden. Dies weist darauf hin, dass das erworbene Wissen auf die erstellten Artefakte angewendet wurde, was nach Baddeley eine nachhaltigere Verinnerlichung von Wissensinhalten begünstigt [Bad86, 183].

- Die realitätsnahen Bedingungen mit einer festlegten Rollenverteilung, eingebettet in ein Vorgehensmodell, umgesetzt in einer Kombination aus Werkzeugen, sowie der geringe Grad an Steuerung durch die Lehrenden, wurden zudem als besonders positiv hervorgehoben. Auch die Wichtigkeit einer intakten Kommunikation zwischen den Teammitgliedern und mit den Lehrenden (in diesem Fall: Kunden), wurde durch die Studierenden erkannt und positiv bewertet.

- Ebenso wurde von den Studierenden berichtet, dass als Nebeneffekt das Verständnis für die deutsche Grammatik verbessert wurde. Dies lag zum Großteil an der Entwicklung von Algorithmen für die Anwendung des SOPHIST REgelwerks auf einen deutschen Satz.

Abschließend kann festgehalten werden, dass die gesammelten Erfahrungen und die abschließende Evaluation des Projekts durchweg positiv ausfiel. Die Studierenden bestärken uns durch ihre Äußerungen dieses Lehr-Lernarrangement auch für zukünftige Gruppen einzusetzen. Dabei soll auch für Folgesemester der Fokus auf der thematischen Rückkopplung während eines möglichst realitätsnah durchgeführten Projekts liegen. Dies könnte zum Beispiel der Fall sein, wenn eine zukünftige Projektgruppe an das Projektergebnis anknüpft, in dem sie die vorhandene Syntax durch die SOPHIST Schablonen nutzt, um daraus automatisiert, von einer schriftlichen zu einer grafischen Dokumentation in

Form von Use Cases zu gelangen. Auch wäre es denkbar erweiterte und umfangreichere (Qualitäts-)Metriken zu entwickeln und diese zur automatisierten Bewertung von Anforderungen heran zu ziehen. Solche Metriken können beispielsweise logische bzw. kausale Abhängigkeiten der Regeln des SOPHIST REgelwerks aufzeigen und entsprechend gewichten.

Literatur

[Bad86] A.D. Baddeley. *So denkt der Mensch: unser Gedächtnis u. wie es funktioniert*. Droemer Knaur, 1986.

[Boe81] Barry W. Boehm. *Software Engineering Economics*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st. Auflage, 1981.

[Coc00] Alistair Cockburn. *Writing Effective Use Cases*. Addison-Wesley, Amsterdamm, 2000.

[FHB13] Paula Figas, Georg Hagel und Alexander Bartel. The Furtherance of Motivation in The Context of Teaching Software Engineering. In Institute of Electrical and Electronics Engineers, Hrsg., *Global Engineering Education Conference*, Seiten 1299–1304, Berlin, 2013.

[Jac95] Michael Jackson. *Software requirements and specifications - a lexicon of practice, principles and prejudices.* Addison-Wesley, University of Michigan, 1995.

[RS09] Chris Rupp und die SOPHISTen. *Requirements Engineering und Management*. Hanser, München, 2009.

[SS11] Ken Schwaber und Jeff Sutherland. *The Scrum Guide*. scrum.org, 2011. URL: https://www.scrum.org/Portals/0/Documents/Scrum/Guides/ScrumGuide.pdf, letzter Zugriff: 03.01.2014.

Dieses Vorhaben wird aus Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01PL12022C gefördert.