Quantitativer Vergleich von Anforderungsspezifikationen

Eric Knauss (eric.knauss@inf.uni-hannover.de)
Leibniz Universität Hannover
FG Software Engineering

1 Motivation

In der Informatikausbildung werden an der Leibniz Universität Hannover im 5. Semester Software Projekte durchgeführt. In diesen Projekten haben wir die *Anzahl der Anforderungen* untersucht, um den Effekt eines Use Case Editors [Kna07] zu messen. Die Ausgangshypothese war, dass durch Verwendung des Editors ein größerer Anteil der funktionalen Anforderungen einer Spezifikation in Use Cases spezifiziert wird. In [Hor07] sind wir daher der Frage nachgegangen, wieviele Anforderungen diese Softwareprojekte in der Regel haben, mit dem Ziel, unterschiedliche Projekte miteinander zu vergleichen. Die hier vorgestellten Messergebnisse erlauben nicht, die ursprüngliche Hypothese zu stützen oder zu widerlegen: Die Art des Projektes (Eingebettetes System oder Informationssystem) scheint mehr Einfluss zu haben, als die Werkzeugunterstützung.

Wir halten die Ergebnisse dennoch für interessant:

1. Die Zahl der funktionalen Anforderungen ist unabhängig von der technischen Komplexität ihrer Umsetzung. Da die Bearbeitungszeit in den untersuchten Projekten konstant ist, sind Anforderungsspezifikationen trotz unterschiedlicher Projekte vergleichbar.
2. Die Art der Erhebung scheint verhältnismäßig robust zu sein. Es ist möglich, die Anforderungen von Projektunkundigen zählen zu lassen.
3. Auch wenn wir keine direkten Ergebnisse aus der Zählung ziehen können, so eignet sich die Datenbasis dennoch als Grundlage für weitere Hypothesen und somit als Vorstudie im Sinne der *Goal Question Metric* Methode (GQM, [vSB99]).

Unsere neue Ausgangshypothese ist dementsprechend: Die Art eines Projekts beeinflusst die Anzahl der funktionalen Anforderungen und ihre Verteilung auf Use Cases und restlicher Spezifikation.

Dieser Beitrag enthält **a)** die Definition unserer Metrik (Anzahl der funktionalen Anforderungen) sowie Hinweise zur Erfassung, **b)** die Ergebnisse dieser Untersuchung und **c)** eine Diskussion der Robustheit der Erfassung.

2 Evaluation der Softwareprojekte

Ziel unserer Erhebung war es, eine Aussage über die *Anzahl* der Anforderungen in den Spezifikationen zu machen. Für die Zählung wurde definiert:

Eine funkt. Anf. ist jede Anforderung nach der Anforderungsschablone (Abb. 1) mit *einem* Prozesswort, die sich sinnerhaltend aus einem Satz einer Anforderungsspezifikation ableiten läßt.

In Tabelle 1 wurden auf diese Weise drei Anforderungen aus einer Projektzielsetzung gewonnen:

Ziel dieses Softwareprojektes ist es, Fragebögen für Umfragen komplett elektronisch per Internetbrowser erstellen, ausfüllen und auswerten zu können.					
1.	muss	System	die Mögl. bieten	Fragebögen	erst.
2.	muss	System	die Mögl. bieten	Fragebögen	ausf.
3.	muss	System	die Mögl. bieten	Fragebögen	ausw.

Tabelle 1: Ein Satz der Spezifikation enthält 3 Anf.

An dieser Stelle ist anzumerken, dass die Abstraktionsebene der Anforderungen (wie zum Beispiel nach [GW05]) nicht betrachtet wurde. Uns ist klar, dass diese Metrik daher im Vergleich zu gängigen Aufwandschätzverfahren (siehe [BHM+00]) naiv ist. Die Gründe für diese Vereinfachungen sind, dass wir auf diese Weise

1. ... die Interpretationsleistung beim Zählenden auf ein Minimum reduzieren können und
2. ... den Erfassungsaufwand mit etwa 2 h für eine 25-seitige Spezifikation auf ein erträgliches Maß reduzieren konnten.

Um weitere Projekte in die Zählung mit aufnehmen zu können, wollten wir wissen, ob das Ergebnis der Messung von der Person des Zählenden abhängt. Daher haben wir ein Referenzprojekt von drei verschiedenen Personen analysieren lassen. Das Ergebnis:

Das Zählen der funktionalen Anforderungen mit Hilfe der Anforderungsschablone hat sich als erstaunlich robust herausgestellt. Es ist unerheblich, ob der Zählende im Umgang mit der Anforderungsschablone vertraut ist. Kenntnis vom Inhalt des Projekts scheint eher hinderlich zu sein. Insgesamt wurden jedoch stets die gleichen Anforderungen gefunden.

Tabelle 2 zeigt die Ergebnisse unserer ersten Messung [Hor07]. Abbildung 2 veranschaulicht drei Werte aus Tabelle 2: Die Anzahl der Anforderungen, die Anzahl der ausformulierten Use Cases und die Anzahl der im UML-Diagramm spezifizierten Use Cases.

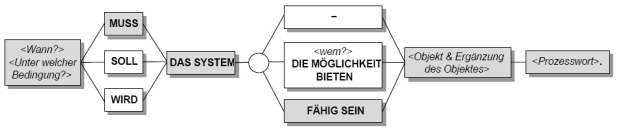

Abbildung 1: Anforderungsschablone nach [Rup04].

Projekt	Anforderungen			red.	Anzahl
	ges.	UC	Dok.	Anf.	UC
NEH-A	38	14	24	11 / 4	4 / 6
NEH-B	42	12	30	15 / 5	2 / 2
NEH-C	32	11	21	11 / 3	3 / 9
PPM	42	16	26	15 / 4	8 / 8
SOA-Me Client	55	26	29	17 / 4	6 / 6
SOA-Me Editor	61	26	35	20 / 4	6 / 6
SOA-Me Server	42	32	10	12 / 2	8 / 8
WüSin	93	54	39	48 /25	10 / 17

Projekt: Kurzname des Projekts.

ges.: Zahl der funktionale Anforderungen in der gesamten Spezifikation.

UC: Zahl der funktionalen Anforderungen in den Use Cases der Spezifikation.

Dok.: Zahl der funktionalen Anforderungen außerhalb der Use Cases.

red. Anf.: Zahl der redundanten Anforderungen. Die erste Zahl gibt die Anzahl der Anf. an, für die es bereits eine wortgleiche Entsprechung lt. Anforderungsschablone gibt, die zweite Zahl gibt an, wieviele Anforderungen sowohl in den Use Cases als auch im restlichen Text definiert wurden.

Anzahl UC: Die erste Zahl gibt die Anzahl von Use Cases an, die in einem Template (nach [Coc00]) ausformuliert wurden. Die zweite Zahl gibt an, wieviele Use Cases in einem Use Case Diagramm modelliert wurden.

Tabelle 2: Anzahl von Anforderungen und Use Cases.

Berücksichtigt man nur die tatsächlich ausformulierten Use Cases, so lassen sich mehrer Cluster bilden:

1. Eher wenige funktionale Anforderungen und sehr wenige Use Cases. In diesen Bereich fallen die Projekte NEH-A, -B und -C, bei denen eingebettete Systeme (Legoroboter) programmiert wurden.
2. Wenige funktionale Anforderungen und überdürchschnittlich viele Use Cases. In diesen Bereich fallen zwei Projekte: Ein algorithmenlastiges Projekt zur Mustersuche in Graphen (PPM) sowie ein Server, um Geschäftsprozesse als Webservice-Orchestrierung auszuführen (SOA-ME Server).
3. In diesen Bereich fallen ein EPK[1]-Editor für Geschäftsprozesse (SOA-Me Editor) und ein Framework zur Generierung von Benutzerschnittstellen für die so modellierten Geschäftsprozesse (SOA-Me Client). Beide Projekte arbeiten mit dem SOA-Me Server zusammen.
4. In den letzten Bereich fällt ein Informationssystem zur Verwaltung von Lehrveranstaltungen (WüSin). Als Besonderheit war in diesem Projekt der Kunde nicht vor Ort, sondern nur per E-Mail und (seltener) per Telefon erreichbar war.

Auch wenn diese Einteilung in die genannten Bereiche plausibel erscheint, kann sie keinesfalls als allgemeingültig angenommen werden. Dazu ist die Menge der untersuchten Projekte zu klein. Es ist auch nicht

[1]Ereignisgesteuerte Prozesskette

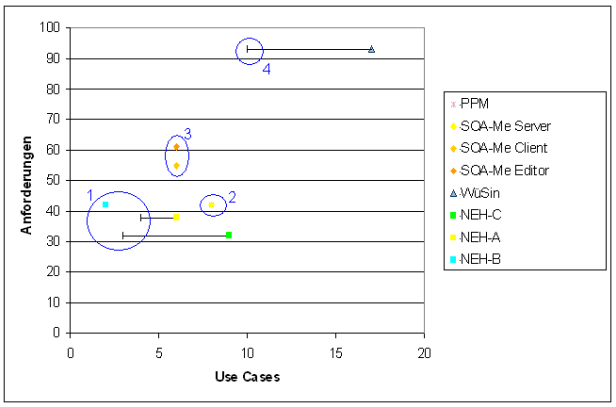

Abbildung 2: Klassifikation von Softwareprojekten.

klar, ob beispielsweise in Bereich 4 das entscheidende Merkmal *entfernter Kunde* oder *Informationssystem* war. Die hier gezeigten Daten können allenfalls als Grundlage für weiterführende Hypothesen dienen.

3 Fazit

Die Art des Projekts (Embedded Software, Informationssystem, SOA, usw.) hat einen entscheidenden Einfluss auf die Struktur der Anforderungen. Es scheint möglich zu sein, diesen Einfluss mit relativ einfachen Mitteln wiederholbar zu messen. Auf der Basis von 25 weiteren Projekten haben wir so Richtwerte für unsere Projekte ermittelt. Demnach sind durchschnittlich 69 % der funktionalen Anforderungen in den Use-Cases spezifiziert. Weiterhin sind durchschnittlich 33 % der funktionalen Anforderungen redundant.

Diese Ergebnisse erlauben es, Anforderungsspezifikationen quantitativ miteinander zu vergleichen. Die Bedeutung einer Abweichung von diesen Richtwerten sowie die Ursachen für die Redundanz müssen aber noch weiter untersucht werden.

Literatur

[BHM+00] Barry W. Boehm, Ellis Horowitz, Ray Madachy, Donald Reifer, Bradford K. Clark, Bert Steece, Winsor A. Brown, Sunita Chulani, and Chris Abts. *Software Cost Estimation with Cocomo II*. Prentice Hall PTR, January 2000.

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

[GW05] Tony Gorschek and Claes Wohlin. Requirements abstraction model. *Requir. Eng.*, 11(1):79–101, 2005.

[Hor07] Nicko Horst. Quantitative und qualitative analyse von anforderungen in software-projekten. Studienarbeit, Leibniz Universität Hannover, 2007.

[Kna07] Eric Knauss. Einsatz computergestützter Kritiken für Anforderungen. *GI Softwaretechnik-Trends*, 27(1), Februar 2007.

[Rup04] Chris Rupp. *Requirements-Engineering und -Management : professionelle, iterative Anforderungsanalyse für die Praxis*. Hanser, 2004.

[vSB99] Rini van Solingen and Egon Berghout. *The Goal/Question/Metric Method: a practical guide for quality improvement of software development*. McGraw-Hill Publishing Company, 1999.