Erfassung, Strukturierung und Überprüfung von Qualitätsanforderungen durch aktivitätenbasierte Qualitätsmodelle

Stefan Wagner, Florian Deissenboeck, Sebastian Winter

Institut für Informatik
Technische Universität München
wagnerst@in.tum.de
deissenb@in.tum.de
winterse@in.tum.de

Abstract: Die Behandlung von Qualitätsanforderungen ist ein zentrales Thema bei der Entwicklung von Software-Systemen. Aufgrund der Komplexität und Vielschichtigkeit der Thematik „Software-Qualität", stellt bereits die geeignete Formulierung dieser Anforderungen eine Herausforderung dar. Qualitätsmodelle haben sich hierbei als ein vielsprechender Ansatz zur Beherrschung der Komplexität erwiesen. Eine neue Art von Qualitätsmodellen verwendet die Aktivitäten, die auf und mit dem System ausgeführt werden, als eine explizite Dimension zur Strukturierung von Qualitätsmodellen. Dieses Papier beschreibt die Verwendung von Qualitätsmodellen dieser Art zur Erfassung, Verfeinerung und Überprüfung von Qualitätsanforderungen.

1 Qualitätsanforderungen

Qualitätsanforderungen werden üblicherweise als Teil der *nichtfunktionalen* Anforderungen an ein System gesehen. Diese nichtfunktionalen Anforderungen beschreiben Eigenschaften eines Systems, die nicht zur primären Funktionalität gehören. Obwohl die Begrifflichkeit „nichtfunktionale Anforderungen" teilweise umstritten ist, existieren immer Anforderungen, die zu spezifischen Qualitäten eines Systems Bezug nehmen [Gli05]. Diese Art von Anforderungen werden *Qualitätsanforderungen* genannt.

Qualitätsanforderungen werden in der Anforderungsanalyse von Softwaresystemen oft nur unbefriedigend behandelt. Ein Hauptgrund dafür ist, dass diese Anforderungen grundsätzlich schwieriger in einer messbaren Art und Weise ausdrückbar sind, was sie wiederum schwieriger analysierbar macht [NE00]. Zugrunde liegt das Problem, dass Qualität selbst ein komplexes und vielschichtiges Konzept ist [Gar84] und deren Ausmaß oder Existenz schwierig zu analysieren ist. Dadurch wird auch die Definition von Qualitätsanforderungen eine anspruchsvolle Aufgabe. Besonders die vollständige Integration der verschiedenen Aspekte aller Stakeholder ist aufwändig und fehleranfällig. Das Problem besteht also darin, wie Qualitätsanforderungen in einer strukturierten und vollständigen Art und Weise erfasst und überprüft werden können.

Es wird ein fünfstufiges Vorgehen zur Behandlung von Qualitätsanforderungen unter Nutzung eines aktivitätenbasierten Qualitätsmodells [DWP+07b] vorgeschlagen. Das Qualitätsmodell verwendet Aktivitäten als eine explizite Dimension und beschreibt die Einflüsse der Systementitäten und ihrer Attribute auf diese Aktivitäten. Diese beiden Dimensionen können zweckdienlich als Strukturierungsmittel für Qualitätsanforderungen verwandt werden, da sie eine abstrakte Sicht auf Qualitätsanforderungen bieten. Ihre Verfeinerung kann durch eine Analyse der Systementitäten und durch welche Aktivitäten diese beeinflusst sind erfolgen. Schließlich ist dadurch eine direkte Nachverfolgbarkeit von der Qualitätssicherung zurück zu den Qualitätsanforderungen durch das Qualitätsmodell gegeben, da das Modell wiederum als Basis für Qualitätssicherungsmaßnahmen, wie Reviews, verwendet werden kann.

2 Qualitätsmodellierung

Um Qualitätsanforderungen effizient handhaben zu können, braucht man eine Möglichkeit, sie in einer präzisen und konsistenten Art und Weise ausdrücken zu können. Seit den 1970ern wurden eine Vielzahl von Qualitätsmodellen entwickelt, die das Ziel hatten genau dies zu erreichen [BBK+78, ISO03]. Jedoch wird in [DWP+07b] argumentiert, dass diese Ansätze eine Reihe von Schwachpunkten besitzen. Am wichtigsten ist hierbei, dass sie es versäumen, die Zusammenhänge zwischen Systemeigenschaften und den auf und mit dem System durch die Stakeholder ausgeführten Aktivitäten mit in Betracht zu ziehen. Diese Auslassung der Aktivitäten ist ein ernstes Problem, da sie den größten Teil der Gesamtkosten von Software bestimmen. Zusätzlich erlauben gerade die Aktivitäten ein natürliches Kriterium zur Dekomposition des komplexen Konzepts *Qualität*, das anderen Ansätzen fehlt.

Um diesen Problemen zu begegnen, wurde eine konsequente Unterscheidung von Aktivitäten und Systementitäten vorgeschlagen [DWP+07b]. Diese Unterscheidung unterstützt die Identifizierung von fundierten Qualitätskriterien und erlaubt über deren Zusammenhänge Schlussfolgerungen zu treffen. Zur Illustration verwenden wir im Folgenden das Qualitätsattribut *Wartbarkeit*, das bekannterweise einen starken Einfluss auf die Gesamtkosten im Lebenszyklus eines Softwaresystems hat und darüber hinaus oft als eines der vielfältigsten, am schwierigsten zu beherrschenden, Qualitätsattribute wahrgenommen wird.

Die erste Dimension des Qualitätsmodells besteht aus den von den Stakeholdern auf und mit dem System ausgeführten Aktivitäten. Im Falle von Wartbarkeit hängt die Menge der relevanten Aktivitäten vom jeweiligen Entwicklungs- und Wartungsprozess der Organisation ab. Da Aktivitäten einfach in Teilaktivitäten strukturiert werden können, formt diese erste Dimension einen Baum: den *Aktivitätenbaum*.

Die zweite Dimension des Modells, der *Entitätenbaum*, beschreibt die Dekomposition der *Entitäten*, vor allem des Softwaresystems. Um Wartbarkeit in einem gegebenen Projekt messen zu können, müssen Verbindungen zwischen den Entitäten und Aktivitäten etabliert werden. Dieser Zusammenhang kann durch eine Matrix wie in Abb. 1 dargestellt werden.

Wie die Abbildung zeigt, muss man die Entitäten mit grundlegenden *Attributen* wie *Kon-*

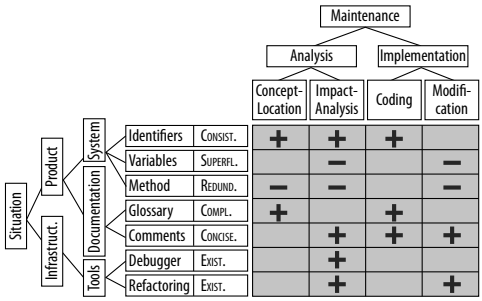

Abbildung 1: Wartbarkeitsmodell

sistenz, Vollständigkeit, Prägnanz oder *Redundanz* ausstatten, um die Zusammenhänge ausdrücken zu können. Diese Zusammenhänge werden nun durch die Identifizierung von *Einflüssen* dargestellt. Ein Einfluss ist definiert als die Relation zwischen einem Tupel bestehend aus Entität und Aktivität und einer Aktivität, wobei das Zeichen „+“ einen positiven und „−“ einen negativen Einfluss darstellt.

Ein Beispiel in obiger Abbildung ist die Prägnanz von Bezeichnern im Programmtext, die einen positiven Einfluss auf das Auffinden von Konzepten im Code hat oder der negative Einfluss von überflüssigen Variablen auf die Änderung existierenden Codes. Das Beispiel zeigt, dass sogar das offensichtlich einfache Attribute *Existenz* sehr mächtig sein kann, wenn gezeigt werden soll, dass eine passende Infrastruktur, wie zum Beispiel ein Debugger, einen Einfluss auf bestimmte Aktivitäten hat.

Solch ein Modell ist gut geeignet, um Qualitätsanforderungen handhabbar zu machen, da die Aktivitäten eine direkte Verbindung zwischen den Stakeholdern, die schließlich Qualitätsanforderungen definieren, und dem Software-System liefert. Im Beispiel der Wartbarkeit ist der betroffene Stakeholder ein Entwickler. Seine Hauptaktivität *Wartung* kann in handhabbarere Teilaktivitäten heruntergebrochen werden. Diese Teilaktivitäten können dann durch einfache Attribute den Systementitäten zugeordnet werden. In [WWD08] wurde gezeigt, wie solch ein Modell zur Beschreibung der *Gebrauchstauglichkeit* angewandt werden kann. Dort ist der zentrale Stakeholder der Nutzer der Software und seine Hauptaktivität *Nutzung* kann wiederum leicht in spezifischere Subaktivitäten aufgeteilt werden. Diese können dann zu konkreten Entitäten, wie den Schriftarten in der Nutzerschnittstelle, zugeordnet werden.

3 Erfassung und Verfeinerung

Die Hauptansätze zur Erfassung von Qualitätsanforderungen bestehen entweder aus dem Abarbeitung verschiedener Anforderungstypen und der abgeleiteten Erstellungen von Prototypen [RR99] oder der Verwendung von positiven und negativen Szenarios [Ale03]. Beide Ansätze haben jeweils ihre Vorteile und sollten am besten in Kombination eingesetzt werden. Jedoch kann die zusätzliche Einbindung von Qualitätsmodellen, wie sie im Abschnitt 2 dargestellt wurden, die Ergebnisse durch eine verfeinerte Struktur verbessern.

Ebert klassifiziert in [Ebe97] Anforderungen in *benutzerorientert* und *entwicklungsorientiert*, was bereits in Richtung einer Strukturierung nach Aktivitäten geht. Diese Unterscheidung ist aber noch zu grob. Doerr et al. verwenden in [DKK+05] bereits Qualitätsmodelle in für nichtfunktionale Anforderungen. Die verwendeten Modelle liefern aber keine direkte Verbindung zu Aktivitäten und damit auch keine direkte Verbindung zu Stakeholdern.

Die Struktur, die durch das aktivitätenbasierte Qualitätsmodell induziert wird, bietet die Grundlage, um Qualitätsanforderungen zu erfassen und zu verfeinern. Der Erfassungs- und Verfeinerungsprozess besteht aus fünf Hauptschritten, die von den etablierten Erfassungstechniken unterstützt werden sollten. Einen Überblick verschafft Abb. 2. Die Schritte sind stark an der Verwendung der beiden Bäume im Qualitätsmodell orientiert und haben das Ziel – soweit möglich – strukturiert bis zu quantitativen Werten für die Anforderungen zu gelangen. Natürlich hängt der Ansatz von der Verfügbarkeit eines geeigneten Qualitätsmodells ab. Idealerweise exisitiert bereits ein solches, aber es kann auch parallel zu den Anforderungen entwickelt werden. Die oberen Ebenen der Bäume lassen sich erfahrungsgemäß aus anderen Qualitätsmodellen der Literatur [DWP+07b] wiederverwenden.

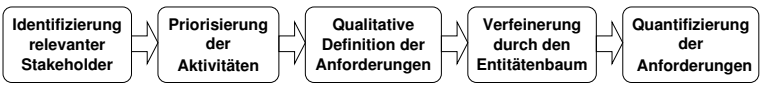

Abbildung 2: Die Prozessschritte zur Erfassung und Verfeinerung von Qualitätsanforderungen

Identifizierung relevanter Stakeholder Der erste Schritt ist, in Anlehnung an andere Ansätze, die Identifikation aller Stakeholder des Software-Systems. Für Qualitätsanforderungen sind dies üblicherweise die Nutzer, Entwickler und Warter, Operateure und Nutzertrainer. Weitere Stakeholder können natürlich, abhängig vom konkreten Projekt, zusätzlich relevant sein. Nach dieser Identifzierung kann das Qualitätsmodell zur Ableitung ihrer jeweiligen Aktivitäten verwendet werden. Beispielsweise gehören *Konzeptlokalisierung*, *Auswirkungsanalyse*, *Kodierung* oder *Änderung* zu den Aktivitäten des Wartungsingenieurs [DWP+07b]. Insbesondere die spezifischen Aktivitäten des Nutzers können durch zusätzliche Nutzungsszenarien detailiert werden.

Priorisierung der Aktivitäten Im nächsten Schritt werden die Aktivitäten der relevanten Stakeholder im Bezug auf ihre Wichtigkeit priorisiert. Falls nicht alle notwendigen Aktivitäten im Modell definiert sind, wird es entsprechend erweitert. Dies resultiert in einer Liste aller Aktivitäten der relevanten Stakeholder, sortiert in der Reihenfolge der Wichtigkeit. „Wichtigkeit“ meint hierbei die Häufigkeit der Ausführung und die Aufwändigkeit der Aktivitäten. Die Rechtfertigung der Wichtigkeit kann Expertenmeinung oder Erfahrungen aus ähnlichen Projekten sein. Diese Liste wird in den folgenden Schritten zur Fokusierung der Definition und Verfeinerung der Anforderungen verwendet.

Qualitative Definition der Anforderungen Im nächsten Schritt muss entschieden werden, wie stark die unterschiedlichen Aktivitäten unterstützt werden sollen. Die Antworten stellen schließlich qualitative Anforderungen and das Software-System dar. Zum Beispiel

kann es gewünscht sein, dass die Aktivität *Konzeptlokalisierung* einfach durchzuführen ist, da die Problemdomäne der Software viele komplexe Konzepte enthält. Abhängig von der Zahl der Aktivitäten, die berücksichtigt werden müssen, können die Aktivitäten mit der geringsten Priorität ignoriert werden.

Verfeinerung durch den Entitätenbaum Wie in Abschnitt 2 enthält der Entitätenbaum die Entitäten der Software, der Umgebung und des Entwicklungsprozesses, die in irgendeiner Form relevant für die Software-Qualität sind. Der Baum organisiert sie in einer definierten, hierarchischen Weise, die ideal zur Verfeinerung der auf Basis der Aktivitäten erfassten Anforderungen geeignet ist. Das Qualitätsmodell selbst hilft hierbei, da es in den Einflüssen den Zusammenhang zwischen Entitäten und Aktivitäten dokumentiert. Daher muss zur Identifzierung der Entitäten, die die identifizierten Aktivitäten beeinflussen nur den dokumentierten Einflüssen gefolgt werden. Falls die Einflüsse im aktuellen Modell noch unvollständig sind, kann dieser Schritt wiederum zur Erweiterung des Modells verwendet werden. Dadurch ist es auch bedeutend einfacher Konsistenz mit der späteren Qualitätssicherung herzustellen, da diese auch auf dem Modell basiert. Zur Definition der verfeinerten Qualitätsanforderungen können die definierten Attribute der Entitäten herangezogen werden. Beispielsweise kann eine detailierte Anforderung sein, dass jedes Objekt einen von aussen zugänglichen Zustand besitzen muss, damit dieser im Test überprüft werden kann, was direkt aus einem Modellelement abgeleitet werden kann. Eine Verfeinerung ist vor allem für die Anforderungen erforderlich, die in den vorigen Schritten als wichtig eingestuft wurden.

Quantifizierung der Anforderungen Das endgültige Ziel ist es, die Anforderungen in einer quantitativen und damit überprüfbaren Form vorliegen zu haben. Durch die direkte Zuordnung zu einer Aktivität ist es auch möglich Anforderungen, wie beispielsweise *geeignetet Modularisierung des Systems*, spezifisch zu quantifizieren. Die Quantifizierung kann sowohl auf der Aktivitätenebene als auch auf der Entitätenebene passieren. Auf der Aktivitätenebene wäre das zum Beispiel, dass eine durchschnittliche *Änderungsaktivität* nicht mehr als vier Personenstunden an Aufwand brauchen darf. Anforderungen auf der Entitätenebene können abhängig vom gewählten Attribut quantitativ überprüfbar sein (vgl. [DWP+07b]). Zum Beispiel können *überflüssige Code-Variablen* gezählt und damit ein oberes Limit geprüft werden. Möchte man eine Anforderung, wie beispielsweise die *Angemessenheit der Datenstrukturen*, bewusst nicht quantifizieren, besteht die Möglichkeit dies direkt im Qualitätsmodell zu vermerken. In jedem Fall bietet das Qualitätsmodell dann die Möglichkeit der Überprüfung der Qualitätsanforderungen, da es eine direkte Verbindung zu qualitätssichernden Massnahmen darstellt [DWP+07b].

4 Zusammenfassung

Wir haben bereits in [DWP07a] vorgeschlagen, Qualitätsanforderungen anhand der Kosten und damit der Stakeholder zu klassifizieren. In diesem Papier haben wir diesen Weg konse-

quent fortgesetzt und eine konkrete Methode zur Erhebung und Strukturierung solcher Anforderungen mit Hilfe von aktivitätenbasierte Qualitätsmodellen angegeben. Die verwendeten Qualitätsmodelle stellen einen wiederverwendbaren „Speicher“ an Qualitätswissen dar, der die Erhebung erleichtert und später auch zur Überprüfung verwendet werden kann, da Qualitätssicherungsmaßnahmen direkt auf dem Qualtätsmodell aufbauen können. Wir planen die einzelnen Schritte der Methode noch genauer durch konkrete Vorlagen auszuarbeiten. Diese werden in Fallstudien erprobt und auch die Verbindung zu funktionalen Anforderungen wird hergestellt werden.

Literatur

[Ale03] Ian Alexander. Misuse Cases: Use Cases with Hostile Intent. *IEEE Software*, Seiten 58–66, 2003.

[BBK+78] Barry W. Boehm, John R. Brown, Hans Kaspar, Myron Lipow, Gordon J. Macleod und Michael J. Merrit. *Characteristics of Software Quality*. North-Holland, 1978.

[DKK+05] Joerg Doerr, Daniel Kerkow, Tom Koenig, Thomas Olsson und Takeshi Suzuki. Non-Functional Requirements in Industry – Three Case Studies Adopting an Experience-based NFR Method. In *Proc. 13th International Conference on Requirements Engineering (RE'05)*, Seiten 373–382. IEEE Computer Society, 2005.

[DWP07a] Florian Deissenboeck, Stefan Wagner und Markus Pizka. Kosten-basierte Klassifikation von Qualitätsanforderungen. In *Erhebung, Spezifikation und Analyse nichtfunktionaler Anforderungen in der Systementwicklung. Halbtägiger Workshop in Zusammenhang mit der SE Konferenz 2007*, 2007.

[DWP+07b] Florian Deissenboeck, Stefan Wagner, Markus Pizka, Stefan Teuchert und Jean-François Girard. An Activity-Based Quality Model for Maintainability. In *Proc. 23rd International Conference on Software Maintenance (ICSM '07)*. IEEE Computer Society, 2007.

[Ebe97] Christof Ebert. Dealing with Nonfunctional Requirements in Large Software Systems. *Ann. Softw. Eng.*, 3:367–395, 1997.

[Gar84] David A. Garvin. What Does »Product Quality« Really Mean? *MIT Sloan Management Review*, 26(1):25–43, 1984.

[Gli05] Martin Glinz. Rethinking the Notion of Non-Functional Requirements. In *Proc. Third World Congress for Software Quality (3WCSQ)*, Seiten 55–64, 2005.

[ISO03] ISO 9126: Product Quality – Part 1: Quality Model, 2003.

[NE00] Bashar Nuseibeh und Steve Easterbrook. Requirements Engineering: A Roadmap. In *Proc. Conference on the Future of Software Engineering (ICSE '00)*, Seiten 35–46. ACM Press, 2000.

[RR99] Suzanne Robertson und James Robertson. *Mastering the Requirements Process*. ACM Press, Addison-Wesley, 1999.

[WWD08] Sebastian Winter, Stefan Wagner und Florian Deissenboeck. A Comprehensive Model of Usability. In *Proc. Engineering Interactive Systems 2007 (EIS '07)*. Springer, 2008. Zur Veröffentlichung angenommen.