

Kontinuierliches Qualitäts-Controlling langlebiger Softwaresysteme

Florian Deissenböck

Institut für Informatik, TU München
Boltzmannstr. 3, 85748 Garching bei München
deissenb@in.tum.de

Abstract: Um wirtschaftlich erfolgreich zu sein, müssen Softwaresysteme an sich stetig ändernde Anforderungen angepasst werden. Daher ist die Wartbarkeit von Softwaresystemen von zentraler Bedeutung. Dennoch werden dedizierte Prozesse, Methoden und Werkzeuge zur Sicherung der Wartbarkeit nur von wenigen Unternehmen eingesetzt. Dies ist teilweise dadurch bedingt, dass derzeit keine präzise Definition des Begriffs Wartbarkeit existiert. Diese Dissertation schlägt einen neuen Ansatz zur Modellierung von Wartbarkeit vor, der Systemeigenschaften explizit mit Wartungsaktivitäten in Verbindung setzt und damit eine strukturierte Dekomposition des Begriffs Wartbarkeit ermöglicht. Mit Hilfe von Werkzeugen können Wartbarkeitsmodelle, die mit diesem Ansatz definiert sind, für das kontinuierliche Qualitäts-Controlling eingesetzt werden um einen schleichenden Verfall der Qualität zu vermeiden. Die Eignung des Ansatzes wird im Rahmen von Fallstudien im industriellen Umfeld demonstriert.

1 Einführung

Zwischen 60% und 80% der Gesamtlebenszykluskosten langlebiger Softwaresystem treten während der Wartungsphase und nicht bei deren initialen Entwicklung auf [BBEB80, NP90]. Entscheidend hierbei ist, dass etwa die Hälfte der Aufwände nicht für die Fehlerbehebung und die Anpassung an die sich ändernde technische Umgebung anfallen. Vielmehr werden sie aufgewandt um existierende Funktionalität zu ändern und neue Funktionen zu implementieren, die es Unternehmen ermöglichen, ihre Geschäftsprozesse an die sich ändernde Marktsituation anzupassen oder neue, innovative Produkte und Dienstleistungen anzubieten. Aufgrund der Dynamik der Anforderungen in den meisten Wirtschaftszweigen ist die Fähigkeit, neue Funktionalität kosteneffizient und zeitnah zu entwickeln einer der Schlüsselfaktoren für den wirtschaftlichen Erfolg heutiger Softwaresysteme.

Die Eigenschaft eines Softwaresystems, effizient geändert und erweitert werden zu können, wird üblicherweise mit dem Begriff *Wartbarkeit* beschrieben. Trotz der anerkanntermaßen zentralen Rolle der Wartbarkeit, mangelt es noch immer vielen Unternehmen an dedizierten Prozesse, Methoden und Werkzeugen zur Sicherstellung der Wartbarkeit. Dies ist insbesondere kritisch, da sich die Wartbarkeit im Rahmen der Softwareevolution erfahrungsgemäß verschlechtert und daher kontinuierlich gesteuert¹ werden muss. Dieses

¹Der deutsche Begriff „Steuerung“ wird im Rahmen dieser Zusammenfassung äquivalent zum englischen Be-

Versäumnis ist teilweise dadurch bedingt, dass derzeit keine präzise Definition des Begriffs Wartbarkeit existiert. Obwohl in den letzten vier Jahrzehnten unterschiedliche Methoden vorgeschlagen wurden, existiert bisher kein umfassender Ansatz zur Definition, Bewertung und Steuerung der Wartbarkeit großer, langlebiger Softwaresysteme. Infolgedessen kommt es bei langlebigen Softwaresystemen oft zu einem schleichenden Qualitätsverfall, der die Reaktionsfähigkeit von Unternehmen hinsichtlich sich ändernder Anforderungen einschränkt und damit deren wirtschaftlichen Erfolg gefährdet.

Diese Dissertation [Dei09] schlägt einen neuen Ansatz zur Modellierung von *Wartbarkeit* vor, der Systemeigenschaften explizit mit Wartungsaktivitäten in Verbindung setzt und damit eine strukturierte Dekomposition des Begriffs Wartbarkeit ermöglicht. Die klare Trennung von Systemeigenschaften und Wartungsaktivitäten ermöglicht die Identifikation schlüssiger Qualitätskriterien und erlaubt es, ihre Abhängigkeiten zu erörtern. Der vorgestellte Ansatz basiert auf einem Qualitätsmetamodell, das den systematischen Entwurf von Wartbarkeitsmodellen unterstützt und zu deren Vollständigkeit beiträgt. Darüber hinaus beschreibt diese Arbeit wie Wartbarkeitsmodelle, die auf dem vorgestellten Metamodell basieren, für das *kontinuierliche Qualitäts-Controlling* operationalisiert werden können. Dies umfasst die Definition eines Qualitätssicherungsprozesses sowie die für dessen Instantiierung benötigten Werkzeuge. Diese Werkzeuge unterstützen den Metamodellkonformen Entwurf von Qualitätsmodellen und bieten die automatische Generierung von Qualitätsrichtlinien sowie *Review-Checklisten*. Zur Durchführung einer zeitnahen Qualitätsüberprüfung dienen *Qualitätsleiststände*, die die Ergebnisse automatischer Analysewerkzeuge zusammenfassen und mit dem Qualitätsmodell in Bezug setzen. Das modellgestützte Qualitäts-Controlling erlaubt es, die Wartbarkeit langlebiger Softwaresysteme langfristig sicherzustellen und somit ihren Wert zu erhalten.

Die Eignung des Modellierungsansatzes sowie der unterstützenden Werkzeuge wird im Rahmen mehrerer Fallstudien, die vorwiegend im industriellen Kontext durchgeführt wurden, demonstriert.

2 Stand der Forschung

Derzeit existiert keine integrierte Methode zur umfassenden Definition und Sicherstellung der Wartbarkeit langlebiger Softwaresysteme. In unterschiedlichen Teildisziplinen des Software-Engineerings wurden jedoch unabhängig voneinander zahlreiche Ansätze entwickelt, deren Ziel es ist, Wartbarkeit zu definieren, zu analysieren und zu erreichen. Beispiele für analytische Ansätze sind Inspektionen [GD93], der Einsatz von Softwaremetriken zur Überprüfung von Wartbarkeitskriterien [CLO95] sowie eine Vielzahl von Qualitätsanalysewerkzeugen [DJH⁺08]. Zur Erreichung von Wartbarkeit werden konstruktive Maßnahmen wie Qualitätsrichtlinien und Entwicklerschulungen eingesetzt. Aus Platzgründen konzentriert sich die folgende Diskussion jedoch auf Ansätze, die zur *Definition* von Wartbarkeit verwendet werden.

Den wichtigsten Ansatz hierfür stellen sog. *Qualitätsmodelle* dar, die in den 70er Jahren zur Definition von Softwarequalität im Allgemeinen eingeführt wurden. Die prominenteste Begriff „control“ verwendet, der sowohl die Überprüfung als auch die Ergreifung geeigneter Maßnahmen umfasst.

ten Vertreter hierfür sind die Qualitätsmodelle, die von McCall&Walter [MW77] sowie Boehm et al. [BBK⁺78] vorgeschlagen wurden und einen starken Einfluss auf die heute gebräuchliche ISO Norm 9126 hatten. Daneben existieren eine Reihe von Ansätzen, die keine konkreten Qualitätsmodelle, sondern Methoden zur Definition spezifischer Modelle darstellen. Bekannte Vertreter sind die Arbeiten von Kitchenham et al. [KLPN97] und Dromey [Dro95]. Die meisten Ansätze versuchen, den komplexen Begriff „Wartbarkeit“ (bzw. „Qualität“) zu definieren, indem sie ihn in hierarchischer Art und Weise in vermeintlich konkretere Begriffe wie „Änderbarkeit“ bzw. „Erweiterbarkeit“ zerlegen. Auf unterster Ebene stehen dabei idealerweise Begriffe, die so konkret sind, dass sie mit Metriken quantifiziert werden können. Obwohl diese Ansätze einen wichtigen Beitrag zur Strukturierung des komplexen Begriffs der Wartbarkeit geleistet haben, offenbaren Sie eine Reihe von Schwächen, die den erfolgreichen Einsatz in der Praxis verhindern:

In vielen Fällen definieren existierenden Qualitätsmodelle Kriterien, die zu unspezifisch sind um tatsächlich überprüft zu werden. Ein Beispiel ist die „Änderbarkeit“, die u. a. von der ISO 9126 gefordert wird. Während dieses Kriterium zweifellos wichtig ist, ist es zu unpräzise, als dass für ein gegebenes System überprüft werden könnte, ob es wirklich änderbar ist. Da es nur bedingt hilfreich ist, Kriterien zu definieren, deren Einhaltung nicht überprüft werden kann, schränkt dies den Nutzen aktueller Qualitätsmodelle stark ein.

Viele Modelle definieren Qualitätskriterien ohne zu erläutern welche Konsequenzen deren (Nicht-)Einhaltung für die Softwarewartung hat. Dies behindert nicht nur die strukturierte Auseinandersetzung mit Softwarequalität, sondern erschwert es, Softwareentwickler von der Wichtigkeit der Kriterien zu überzeugen, was wiederum deren Einhaltung gefährdet und somit zu einem Qualitätsverfall beiträgt.

Bereits Kitchenham et al. merkten an [KLPN97], dass hierarchische Modelle an einer „teilweise willkürlichen Auswahl von Qualitäts-Charakteristiken und Sub-Charakteristiken“ leiden. Das zugrunde liegende Problem ist, dass diese Modelle versuchen, unscharfe Begriffe wie „Wartbarkeit“ durch ähnlich schwammige Begriffe wie „Erweiterbarkeit“ oder „Änderbarkeit“ zu definieren. Aufgrund der Unschärfe der Begriffe ist es nicht möglich, ein präzise definiertes *Dekompositionskriterium* anzugeben, das eine eindeutige hierarchische Zerlegung vorgeben würde. Darüber hinaus weist nahezu keiner der existierenden Ansätze ein explizites *Metamodell* auf, das die Beschaffenheit der Modellelemente und deren Beziehung klar definieren würde. Durch die mangelhafte Strukturierung wirken die Modelle oft willkürlich und sind außerdem schwer bzgl. Vollständigkeit zu überprüfen.

In den meisten Fällen sind die Modelle ausschließlich in Form von Prosa und Abbildungen beschrieben. Sie begleiten den Wartungsprozess in Form von Dokumenten, werden jedoch nicht in konkretem Zusammenhang zu den Qualitätssicherungsaktivitäten gestellt. Insbesondere bleibt meist unklar, welche Rolle die Qualitätsmodelle im Rahmen von analytischen Maßnahmen wie Reviews und Inspektionen bzw. beim Einsatz automatischer Qualitätsanalysewerkzeuge spielen. Die fehlende *Operationalisierung* der Modelle führt dazu, dass diese ihrer Aufgabe als zentrale Instanz zur Definition von Qualitätskriterien nicht gerecht werden.

Die existierenden Arbeiten zeigen auf, dass die Menge der Einflussfaktoren auf die Softwarewartung sehr groß und mannigfaltig ist. Insgesamt muss jedoch festgestellt werden,

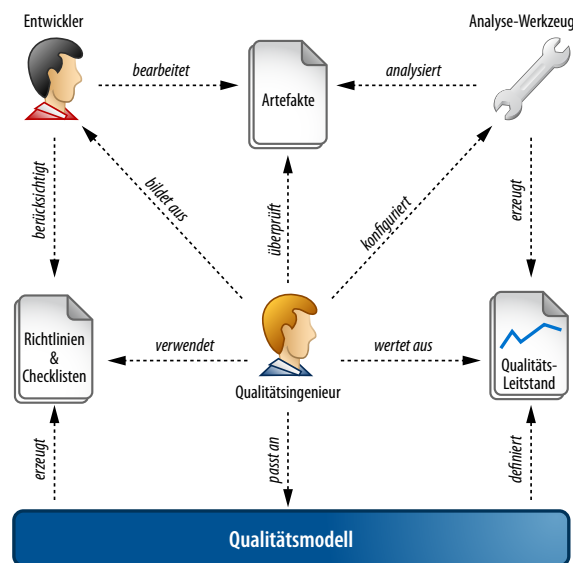


Abbildung 1: Operationalisierung des Qualitätsmodells

dass bisher kein Ansatz existiert, der es erlaubt, überprüfbare und begründete Qualitätskriterien in einer strukturierten Form zu definieren und diese Definition so mit den Qualitätssicherungsaktivitäten in Bezug zu stellen, dass eine langfristige Einhaltung der Kriterien sichergestellt werden kann.

3 Ein operationalisierbares Qualitätsmodell für Wartbarkeit

Dieser Abschnitt beschreibt den Hauptbeitrag der Dissertation: Ein neuartiger Ansatz zur Definition, Bewertung und Verbesserung der Wartbarkeit von Softwaresystemen. Der zentrale Bestandteil des vorgestellten Ansatzes ist ein Qualitätsmetamodell mit dessen Hilfe Qualitätsmodelle entwickelt werden können, die im Rahmen des Wartungsprozesses operationalisiert werden. Abbildung 1 zeigt, dass dabei eine Person (*Qualitätsingenieur*) die zentrale Rolle für Sicherstellung der Wartbarkeit einnimmt. Dieser definiert das *Qualitätsmodell* auf dem schließlich alle Qualitätssicherungsaktivitäten basieren. So werden aus dem Qualitätsmodell z. B. *Richtlinien* generiert, die dazu dienen, die Qualitätsanforderungen an die Entwickler zu kommunizieren. Ebenso werden *Checklisten* generiert, die im Rahmen von manuellen Qualitätssicherungsaktivitäten wie Reviews und Inspektionen verwendet werden. Darüber hinaus wird das Qualitätsmodell zur Konfiguration von Qualitätsanalysewerkzeugen verwendet, die ihre Ergebnisse in Form eines integrierten *Qualitätsleitstands* präsentieren, der den Qualitätsingenieur in seiner Arbeit unterstützt.

Aus der Diskussion der existierenden Qualitätsmodellierungsansätze in Abschnitt 2 ergibt sich, dass ein Qualitätsmodell für seinen praktischen Einsatz bewertbar, begründet und wohlstrukturiert sein muss. Die zentrale Herausforderung liegt daher in der Definition eines *Qualitätsmetamodells* mit dessen Hilfe Qualitätsmodelle entworfen werden

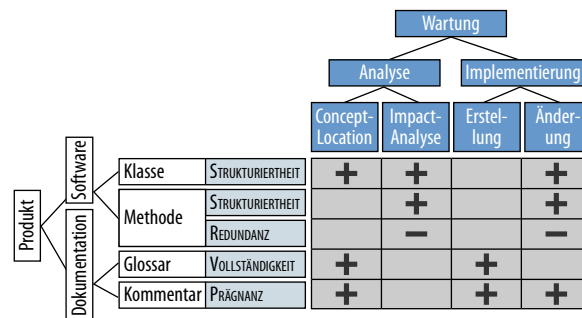


Abbildung 2: Aktivitäten-basiertes Qualitätsmodell

können, die diese Anforderungen erfüllen und auf den jeweiligen Projektkontext zugeschnitten sind. Um eine Operationalisierung, wie oben beschrieben, zu ermöglichen, muss das Modell zudem in einer maschinenlesbaren Form vorliegen. Die folgenden Absätze beschreiben das in dieser Dissertation vorgeschlagene Qualitätsmetamodell für Wartbarkeit.

Eine weit verbreitete Definition gibt an, dass die Wartbarkeit eines Systems definiert ist durch den Aufwand, der anfällt um spezifizierte Änderungen durchzuführen². Daraus ergibt sich, dass der oft formulierte Wunsch nach *hoher Wartbarkeit* bei genauer Betrachtung der Wunsch nach *niedrigen Wartungskosten* ist. Ausgehend von dieser Beobachtung fokussiert das vorgeschlagene Modell nicht auf den abstrakten Begriff der Wartbarkeit, sondern auf den greifbareren Begriff des *Wartungsaufwands*. Um einen Zusammenhang zwischen der großen und mannigfaltigen Menge an Systemeigenschaften, die einen Einfluss auf die Wartungskosten haben und den Wartungskosten selbst herzustellen, greift der Ansatz auf ein Konzept zurück, das im Bereich des Rechnungswesens unter dem Begriff *Activity-Based-Costing (ABC)* bekannt ist [CK88]. Idee des ABC ist es, die komplexen Kostenstrukturen einzelner Produkte zu analysieren indem die für Entwicklung, Fertigung, Distribution und Kundenbetreuung durchzuführenden *Aktivitäten* und deren Kostentreiber identifiziert werden.

Entsprechend sieht das vorgeschlagene Metamodell eine explizite Modellierung der Wartungsaktivitäten und deren Beeinflussung durch Produkteigenschaften vor. Wie in dem beispielhaften Modell in Abbildung 2 dargestellt, werden die Wartungsaktivitäten dabei hierarchisch verfeinert. Um Systemeigenschaften zu beschreiben wird eine weitere hierarchische Dekomposition verwendet, die unterschiedliche Artefakttypen eines Systems beschreibt, die jeweils mit relevanten Eigenschaften in Bezug gesetzt werden. Der Zusammenhang zwischen solchen Tupeln aus Systemartefakt und Eigenschaft auf der einen, und Wartungsaktivitäten auf der anderen Seite, wird über die in der Abbildung als Matrix dargestellte Relation hergestellt. Zum Beispiel wird aufgezeigt, dass die Redundanz einer Methode sich negativ auf die Aktivitäten „Impact-Analyse“ und „Änderung“ auswirkt. Im Beispiel wird hierfür eine dreiwertige Skala verwendet, die einen positiven, einen negativen oder das Fehlen eines Zusammenhangs ausdrückt. Das zugrunde liegende Metamodell erlaubt jedoch auch die Verwendung komplexerer Skalen, die z. B. die Stärke des Einflusses

²SEI Open Systems Glossary (<http://www.sei.cmu.edu/opensystems/glossary.html>)

ses ausdrücken können. Darüber hinaus erfordert das Metamodell detaillierte Beschreibungen der einzelnen Modellelemente, die über das in der Abbildung gezeigte hinausgehen und z. B. angeben, warum die Redundanz einer Methode die Änderungsaufwände erhöht.

Auf dem Metamodell basierende Qualitätsmodelle adressieren die zentralen Kritikpunkte bisheriger Modelle: Durch den Zusammenhang zu den Aktivitäten wird für jedes Qualitätskriterium erläutert, welche Auswirkung es auf die Wartung hat und damit inhärent eine Begründung angegeben. Das Metamodell trennt Aspekte, die in bisherigen Modellen vermengt sind (Aktivitäten, Artefakte und Eigenschaften) und schafft dadurch Hierarchien mit klar definierten Dekompositionskriterien. Dies erlaubt eine saubere Strukturierung der Modelle und bietet damit erstmals auch die Möglichkeit, die Modelle tatsächlich soweit zu verfeinern, dass überprüfbare Eigenschaften beschrieben werden können.

Darüber hinaus ermöglicht das explizit definierte Metamodell eine umfassende Werkzeugunterstützung, die für die Operationalisierung des Modells, wie in Abbildung 1 dargestellt, benötigt wird. So wurde ein graphischer Editor entwickelt, mit dem Metamodell-konforme Qualitätsmodelle entworfen und gewartet werden können. Ein Werkzeug ist hierfür essentiell, da realistische Qualitätsmodelle oft mehrere hundert Modellelemente umfassen und daher nicht mehr manuell gepflegt werden können. Dieser Editor bietet zudem die Möglichkeit, Qualitätsrichtlinien und Review-Checklisten direkt aus den Modellen zu generieren. Um die Ergebnisse automatischer Qualitätsanalysewerkzeuge in direkten Zusammenhang zu einem definierten Qualitätsmodell zu stellen, werden Qualitätsleitstände verwendet, die mit dem im Folgenden beschriebenen Werkzeug erstellt werden können.

4 Qualitätsleitstände

In realistischen Qualitätsmodellen für Wartbarkeit sind oft mehrere hundert Qualitätskriterien definiert, die, um einem Qualitätsverfall entgegenzuwirken, kontinuierlich überprüft werden müssen. Ein signifikanter, wenn auch bei weitem nicht vollständiger, Teil dieser Kriterien kann mit Hilfe von automatisierten Qualitätsanalysewerkzeugen überprüft werden. Beispiele hierfür sind die Erhebung von Softwaremetriken, das Auffinden von Architekturverletzungen [FRJ09] oder die Identifikation von redundantem Quelltext [JD-HW09]. Entsprechend existiert eine große Zahl von kommerziellen wie akademischen Werkzeugen.

Aufgrund der Vielfalt der Qualitätskriterien fällt bei den Analysen eine enorme Datenmenge an, die durch geeignete Aggregation und Visualisierung so aufbereitet werden muss, dass sie effektiv und effizient für das kontinuierliche Qualitäts-Controlling eingesetzt werden kann. Darüber hinaus müssen die Analysedaten verschiedener Werkzeuge zueinander in Kontext gesetzt und der direkte Bezug zu einem Qualitätsmodell, das letztendlich die Soll-Qualität definiert, hergestellt werden. Aus Gründen der Übersichtlichkeit erfolgt dies idealerweise in einem *Qualitätsleitstand*, der den Qualitätsingenieur zeitnah mit den für ihn relevanten Daten versorgt. Zu diesem Zweck wurde das Qualitätsbewertungs-Framework ConQAT (Continuous Quality Assessment Toolkit) entworfen. ConQAT unterstützt den effizienten Aufbau von Qualitätsleitständen und deren Anpassung auf Projektspezifika. Es kann autonom eingesetzt werden, aggregiert Analyseergebnisse, bezieht

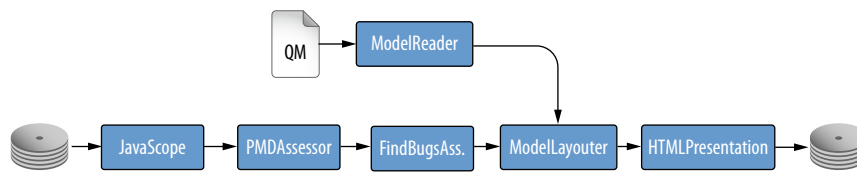


Abbildung 3: ConQAT-Konfiguration für die Anbindung an ein Qualitätsmodell

vielfältige Datenquellen ein und ist flexibel erweiterbar. Die Flexibilität beruht auf dem Ansatz, spezialisierte Analysen (sog. Prozessoren) zu entwickeln und mit Hilfe einer eigens entwickelten graphischen Datenflussprache zu kombinieren. Die in Abbildung 3 dargestellte Analysekonfiguration analysiert beispielsweise ein Java-System mit Hilfe der bekannten Analysewerkzeuge PMD³ sowie FindBugs⁴ und setzt die Ergebnisse in Kontext mit dem ebenfalls automatisch eingelesenen Qualitätsmodell. Die Ergebnisse werden in Form von HTML-Seite ausgegeben, die dem Qualitätsingenieur dabei unterstützen, den Überblick über den aktuellen Qualitätsstand des Systems zu behalten.

ConQAT bietet mehr als 300 Prozessoren mit unterschiedlichster Funktionalität für Analyse, Aggregation, Historisierung und Visualisierung. Es wird in vielfältigen industriellen wie akademischen Kontexten eingesetzt und ist als Open-Source-Werkzeug verfügbar⁵.

5 Fallstudien

Die Eignung des in der Arbeit präsentierten Ansatzes für das kontinuierliche Qualitäts-Controlling wurde in vier industriellen Fallstudien mit der BMW AG [DP08], der MAN Nutzfahrzeuge AG [DWP⁺07], der Interasco GmbH [Gar07], der Münchener Rückversicherungs-Gesellschaft (Munich RE) und der ABB Gruppe [DJH⁺08] sowie in einer Fallstudie im akademischen Kontext evaluiert. Aus Platzgründen beschränkt sich diese Zusammenfassung auf zwei der durchgeführten Fallstudien.

Modellbasierte Entwicklung eingebetteter Systeme MAN entwickelt einen signifikanten Anteil der eingebetteten Software für Nutzfahrzeuge mit Hilfe eines modellbasierten Entwicklungsansatzes, der auf den Werkzeugen Simulink bzw. TargetLink beruht. Ziel dieser Fallstudie war es, explizite Kriterien für die Wartbarkeit von Simulink-Modellen zu definieren und diese in Form von Qualitätsrichtlinien an die Entwickler zu kommunizieren.

Zu diesem Zweck wurde ein Qualitätsmodell auf Basis des in dieser Dissertation vorgeschlagenen Metamodells erstellt. Hierzu wurde ein bereits existierendes Modell, das im Bereich der Telekommunikation entworfen wurde, um Kriterien bezüglich der modellbasierten Entwicklung erweitert. Diese Erweiterungen basieren auf (1) existierenden Richtlinien für Simulink, (2) allgemeinen wissenschaftlichen Erkenntnisse im Bereich der modellbasierten Entwicklung und (3) Expertenwissen der MAN-Ingenieure. Die Aufnahme

³<http://pmd.sourceforge.net>

⁴<http://findbugs.sourceforge.net>

⁵<http://www.conqat.org>

der Kriterien führte zu einer Erweiterung des Qualitätsmodell um ca. 240 Modellelemente. Um die definierten Qualitätskriterien an die Entwickler zu kommunizieren wurden aus dem Qualitätsmodell Richtlinien in dem MAN-üblichen Dokumentformat generiert. Um verschiedene Lesertypen optimal zu unterstützen, wurde das Richtliniendokument in zwei Bereiche unterteilt. Der erste enthält die wichtigsten Informationen in einer sehr kompakten, listenartigen Darstellung und richtet sich an erfahrene Entwickler, die die Richtlinie als Referenz benutzen. Der zweite Teil des Dokuments richtet sich an jüngere Entwickler und enthält weitergehende Information, z. B. über die Auswirkungen eines Qualitätskriteriums auf die Wartungsaktivitäten.

Die Formalisierung von Qualitätskriterien mit Hilfe eines expliziten Qualitätsmodells erwies sich aus mehreren Gründen als vorteilhaft. Durch die Modellierung wurde die *inkonsistente Terminologie* der existierenden Richtlinien vereinheitlicht und *inhaltliche Widersprüche* zwischen Richtlinien aufgedeckt. Darüber hinaus traten im Rahmen der Modellierung offensichtliche Lücken in den bisher verwendeten Richtlinien zu Tage. Durch die automatische Generierung von Richtlinien wurde die Konsistenz zum Qualitätsmodell sichergestellt und außerdem ermöglicht, verschiedene Richtlinienformate für Entwickler mit unterschiedlicher Erfahrung zu erstellen. Aufgrund der inhärenten Redundanz zwischen diesen Richtlinien wäre die Sicherstellung der Konsistenz ohne automatische Generierung sehr aufwändig und fehleranfällig.

Aufgrund der erreichten Resultate hat sich MAN, entgegen der ursprünglichen Planung, entschieden, nicht nur die generierten Richtlinien, sondern den vollständigen Qualitätsmodellierungsansatz schrittweise in seine Prozesse zu integrieren.

Qualitätsleitstände Der Einsatz von ConQAT-basierenden Qualitätsleitständen zur Unterstützung des Qualitäts-Controllings wurde in Zusammenarbeit mit der Rückversicherung Munich RE und einer finnischen Entwicklungsabteilung des Technologiekonzerns ABB evaluiert. In beiden Unternehmen war es das Ziel, ausgewählte Qualitätskriterien mit Hilfe eines Leitstands über einen längeren Zeitraum zu überwachen und, bei Bedarf, entsprechende Maßnahmen einzuleiten. Für beide Unternehmen war wichtig, dass der Einsatz des Leitstands möglichst zeitnah zu erkennbar positiven Ergebnissen führt und dass dies mit möglichst wenig Veränderungen an existierenden Entwicklungs- und Qualitätssicherungsprozessen erreicht wird. Daher wurden nur wenige, aktuell relevante Qualitätskriterien in den Leitstand aufgenommen. Auf Basis von im Vorfeld durchgeführten Qualitätsuntersuchungen der Systeme fiel bei beiden Organisationen die Wahl auf eine Analyse der Quelltextredundanz (*Cloning*), d. h. das Auffinden von Code, der durch Copy&Paste-Programmierung entstanden ist. Bei der Munich RE wurde außerdem die Untersuchung von Architekturverletzungen, d. h. der Abgleich zwischen der Soll-Architektur des Systems und der tatsächlich implementierten Ist-Architektur in den Leitstand aufgenommen. Beide Analysen werden von ConQAT direkt unterstützt. Im Falle von ABB wurde zusätzlich eine Untersuchung von ungenutztem Quelltext sowie eine Analyse von bestimmten Kriterien hinsichtlich der Internationalisierbarkeit des Quelltextes in den Leitstand integriert. Die zugrunde liegenden Analysen werden hierbei vom Analysewerkzeug FxCop⁶ durchgeführt und dessen Ergebnisse in den ConQAT-Leitstand integriert.

⁶[http://msdn.microsoft.com/en-us/library/bb429476\(VS.80\).aspx](http://msdn.microsoft.com/en-us/library/bb429476(VS.80).aspx)

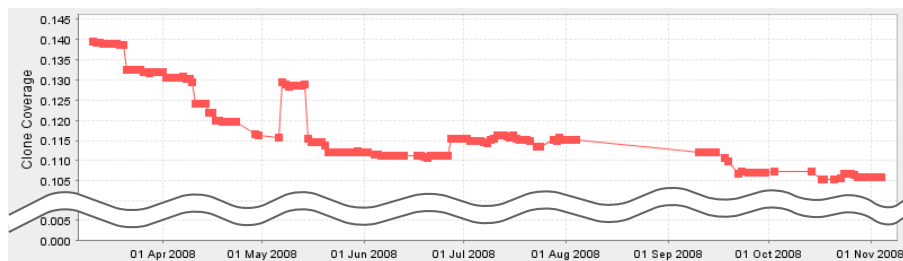


Abbildung 4: Trendkurve für Quelltext-Cloning in einem Projekt der Munich RE

Da die Prozessintegration des Leitstands vorwiegend den Firmen überlassen wurde, entwickelten sich hierfür unterschiedliche Vorgehensweisen. In manchen Projekten werden die Leitstände ausschließlich von den Entwickler selbst konsultiert, in anderen existiert ein dedizierten Qualitätsingenieur. In wieder anderen, werden die Leitstände gemeinsam im Rahmen von Teambesprechungen ausgewertet. Auf Basis eines beobachteten Zeitraums von jeweils ca. 6 Monaten kann geschlossen werden, dass der Einsatz der Leitstände bei beiden Unternehmen nicht nur half, den vorgefunden Qualitätsstand beizubehalten, sondern auch zu einer schrittweisen Verbesserung beigetragen hat. Dies zeigt sich z. B. an der Trendkurve in Abbildung 4. Diese zeigt, dass die *Clone Coverage*, die ein Maß für den Umfang der Quelltext-Redundanz darstellt, im beobachteten Zeitraum um ca. 25% fiel. Dies ist, insbesondere weil bekannt ist, dass der gleiche Wert in den meisten Softwaresystemen kontinuierlich zunimmt, als positiv zu betrachten. Beide Unternehmen waren von den Ergebnissen überzeugt und haben sich daraufhin entschlossen, den Einsatz ConQAT-basierte Leitstände auf mehrere Projekte auszudehnen.

6 Zusammenfassung & Ausblick

Da der weitaus größte Teil der Gesamtlebenszykluskosten langlebiger Softwaresysteme während der Pflege und Weiterentwicklung anfällt, ist die Wartbarkeit dieser Systeme heutzutage entscheidend für deren kommerziellen Erfolg. Trotzdem setzten bisher nur wenige Unternehmen dedizierte Prozesse, Methoden und Werkzeuge zur Sicherstellung der Wartbarkeit ein. Dies ist zum Teil darauf zurückzuführen, dass keine Methoden zur präzisen Definition von Wartbarkeit existieren. Diese Dissertation schlägt einen auf einem expliziten Qualitätsmetamodell basierenden Ansatz vor, der eine eindeutige Definition dieser zentralen Systemeigenschaft und deren Überprüfung ermöglicht. Konstruktive Methoden der Qualitätssicherung werden von dem Ansatz durch die automatische Generierung von Qualitätsrichtlinien unterstützt. Die Integration von Qualitätsanalysewerkzeugen in einen Qualitätsleitstand, der den aktuellen Qualitätsstand im Bezug auf das Qualitätsmodell aufzeigt, ermöglicht ein *kontinuierliches Qualitäts-Controlling* in einer effizienten und effektiven Art und Weise. Nachdem die Eignung des Ansatzes für die Wartbarkeit in mehreren Fallstudien nachgewiesen wurde, stellt die Ausdehnung der entwickelten Methoden auf andere Qualitätsattribute wie z. B. *Sicherheit* den wichtigsten nächsten Schritt dar. Als Teilnehmer des BMBF-geförderten Projektes „Quamoco“ bietet sich dem Autor dieser Dissertation die Möglichkeit, diese und andere Herausforderungen der Qualitätsmodellierung in Zusammenarbeit mit hochrangigen Forschungsinstituten sowie wichtigen Vertretern der Softwareindustrie im Rahmen des dreijährigen Projektes adressieren zu können.

Literatur

- [BBEB80] B.Lientz, P. Bennet, E.Swanson und E. Burton. *Software Maintenance Management: A Study of the Maintenance of Computer Application Software*. Addison Wesley, 1980.
- [BBK⁺78] B. Boehm, J. Brown, H. Kaspar, M. Lipow, G. Macleod und M. Merrit. *Characteristics of Software Quality*. North-Holland, 1978.
- [CK88] R. Cooper und R.. Kaplan. Measure Costs Right: Make the Right Decisions. *Harvard Bus. Rev.*, 67(Sept.-Oct.):96–103, 1988.
- [CLO95] D. Coleman, B. Lowther und P. Oman. The application of software maintainability models in industrial software systems. *J. Syst. Softw.*, 29(1):3–16, 1995.
- [Dei09] F. Deissenboeck. *Continuous Quality Control of Long-Lived Software Systems*. Dissertation, Technische Universität München, 2009.
- [DJH⁺08] F. Deissenboeck, E. Juergens, B. Hummel, S. Wagner, B. Mas y Parareda und M. Pizka. Tool Support for Continuous Quality Control. *IEEE Softw.*, 25(5):60–67, 2008.
- [DP08] F. Deissenboeck und M. Pizka. Probabilistic Analysis of Process Economics. *Softw. Process Improve. Pract.*, 13(1):5–17, 2008.
- [Dro95] R. Dromey. A Model for Software Product Quality. *IEEE Trans. Software Eng.*, 21(2):146–162, 1995.
- [DWP⁺07] F. Deissenboeck, S. Wagner, M. Pizka, S. Teuchert und J.-F. Girard. An Activity-Based Quality Model for Maintainability. In *Proc. of the ICSM*. IEEE CS Press, 2007.
- [FRJ09] M. Feilkas, D. Ratiu und E. Juergens. The Loss of Architectural Knowledge during System Evolution: An Industrial Case Study. In *Proc. of the ICPC*. IEEE CS, 2009.
- [Gar07] A. Gardener. Effiziente Wartung und Weiterentwicklung der Benutzeroberflächen betrieblicher Informationssysteme. Diplomarbeit, Technische Universität München, 2007.
- [GD93] T. Gilb und D.Graham. *Software Inspection*. Addison-Wesley, 1993.
- [JDHW09] E. Juergens, F. Deissenboeck, B. Hummel und S. Wagner. Do Code Clones Matter? In *Proc. of the ICSE*. IEEE CS, 2009.
- [KLPN97] B. Kitchenham, S. Linkman, A. Pasquini und V. Nanni. The SQUID approach to defining a quality model. *Software Qual. J.*, 6(3):211–233, September 1997.
- [MW77] J. McCall und G. Walters. *Factors in Software Quality*. The National Technical Information Service (NTIS), Springfield, VA, USA, 1977.
- [NP90] J. Nosek und P. Palvia. Software maintenance management: changes in the last decade. *Journal of Software Maintenance*, 2(3):157–174, 1990.



Florian Deibenböck, geboren am 1. Oktober 1977 in München, studierte Informatik an der TU München und am Asian Institute of Technology in Thailand. Seit 2004 arbeitet er am Lehrstuhl von Prof. Dr. Dr. h.c. M. Broy an der TU München und schloss dort 2009 seine Promotion ab. Seine Forschungsinteressen liegen im Bereich der Softwarewartung und der Wirtschaftlichkeitsbetrachtung von Software. Als geschäftsführender Gesellschafter der CQSE GmbH widmet er sich dem Transfer von Forschungsergebnisse in die industrielle Praxis. Er ist verheiratet und Vater von zwei Töchtern.