

# Software-Qualitätsmodelle in der Praxis: Erfahrungen mit aktivitätenbasierten Modellen

Stefan Wagner, Florian Deißeböck, Martin Feilkas, Elmar Jürgens

Institut für Informatik  
Technische Universität München  
{wagnerst,deissenb,feilkas,juergens}@in.tum.de

**Zusammenfassung** Software-Qualität ist ein komplexes und vielschichtiges Gebiet, das eine hohe und sich ständig verstärkende Praxis-Relevanz aufweist. Dieser Komplexität wird üblicherweise mit Qualitätsmodellen begegnet, die Qualität in Attribute und Subattribute aufspalten. Die Modelle haben aber in der Anwendung verschiedene Probleme, wodurch sie sich nicht allgemein durchsetzen lassen. Eine neuartige Strukturierung von Qualitätsmodellen basiert auf der Trennung von technischen Eigenschaften und Aktivitäten. Das Papier faßt die Vorteile dieser Art von Qualitätsmodellen zusammen und stellt die umfangreichen industriellen Erfahrungen damit kurz dar.

## 1 Qualitätsmodelle für Software

Qualität ist grundsätzlich ein komplexes und vielschichtiges Konzept [1]. Aufgrund ihrer großen praktischen Relevanz müssen aber trotzdem Mittel und Wege gefunden werden, mit dieser Komplexität umzugehen.

### 1.1 Motivation und Probleme

Qualitätsmodelle beschreiben abstrakt, was Qualität bei Software-Systemen bedeutet. Dazu wird Qualität typischerweise in verschiedene Faktoren zerlegt, wie beispielsweise in der ISO-Norm 9126 [2]. Solche hierarchischen Strukturen als Basis für Qualitätsmodelle sind bereits seit langem im Einsatz. Ein Modell, das zuerst von Boehm et al. [3] und McCall et al. [4] benutzt wurde, besteht aus drei Schichten: Faktoren, Kriterien und Metriken. Die Faktoren auf der obersten Hierarchieebene modellieren die Hauptqualitätsziele. Diese Faktoren werden in Kriterien und Subkriterien weiter zergliedert. Typischerweise werden dabei drei Ebenen definiert, wobei für die unterste Ebene eine Metrik zur Messung des Kriteriums angegeben wird. Jedoch ist die genutzte Dekomposition oft zu abstrakt und unpräzise, um in der Analyse und der Messung direkt einsetzbar zu sein.

Darüber hinaus fehlt in solchen Modellen typischerweise der Bezug zu den Aktivitäten im Entwicklungsprozess und somit zu Kosten und Nutzen. Auch neuere Arbeiten zur Messung und Modellierung von Software-Qualität, wie beispielsweise Seffah et al. [5] ändern an den genannten Kritikpunkten nichts.

Zusammenfassend weisen Qualitätsmodelle heute typischerweise vier Probleme auf:

1. *Bewertbarkeit*. Die meisten Qualitätsmodelle besitzen viele Kriterien, die zu grob-granular sind, als dass sie direkt bewertet werden könnten.
2. *Begründung*. Oft fehlt auch für viele Kriterien die Erklärung des Einflusses, den sie auf die Entwicklung, den Betrieb oder die Nutzung des Systems haben.
3. *Homogenität*. Da eine eindeutige Aufteilung in verschiedene Qualitätsaspekte fehlt, besitzen die meisten Qualitätsmodelle heute Qualitätskriterien, die teilweise nicht disjunkt sind und auf sehr unterschiedlichen Abstraktionsebenen liegen.
4. *Operationalisierung*. Typische Qualitätsmodelle existieren nur in Form von Text und Graphiken und sind kein lebendiger Teil des Entwicklungsprozesses.

## 1.2 Aktivitätenbasierte Qualitätsmodelle

Aus den genannten Problemen ist ersichtlich, dass es notwendig ist die Strukturierung von Qualitätsmodellen zu verbessern. Zwei der Probleme, ungenügende Begründung und Homogenität, stammen aus der nicht ausreichenden Struktur gängiger Modelle. Aus diesem Grund wird in [6] und [7] vorgeschlagen, ein explizites Metamodell für Qualitätsmodelle zu verwenden. Dieses Metamodell trennt technische Eigenschaften von den auf und mit dem System auszuführenden Aktivitäten. Weiterhin werden die Einflüsse der Systemeigenschaften auf diese Aktivitäten explizit modelliert. Dies resultiert in einer „Qualitätsmatrix“, wie sie in Abb. 1 dargestellt ist.

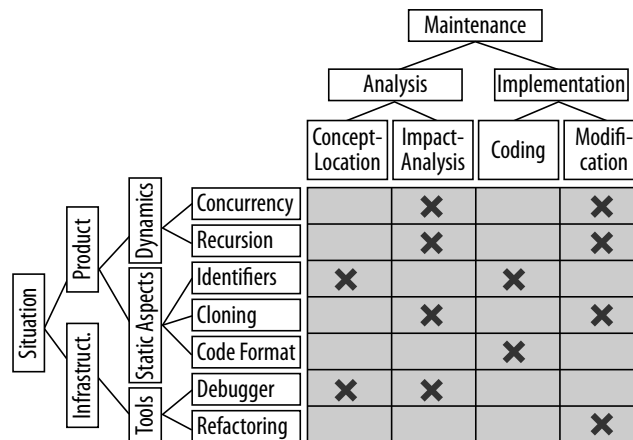


Abbildung 1. Eine beispielhafte Qualitätsmatrix für Wartungsaktivitäten

Ein Kreuz zeigt hier den Einfluss von bspw. „Cloning“ auf die Aktivität „Modification“ an. Typischerweise werden solche Einflüsse noch weiter spezifiziert,

z.B. in positive und negative Einflüsse. Dies (zusammen mit textuellen Beschreibungen) reicht bereits aus, um Review-Richtlinien zu generieren. Für weitere Analysen können Qualitäts-Cockpits, wie z.B. ConQAT<sup>1</sup>, angebunden werden. Das Modell selbst kann in einem eigens dafür entwickelten Editor bearbeitet werden und liegt als XML-Datei vor. Dadurch wird das Modell zu einem lebendigen Artefakt in der Entwicklung.

### 1.3 Modellbasiertes Qualitätscontrolling

Um den oben genannten Problemen gebräuchlicher Qualitätsmodelle zu begegnen, ist die stärkere Operationalisierung von Qualitätsmodellen als lebendiges Artefakt im Software-Entwicklungsprozess wichtig [6]. Dies wird auch „modellbasiertes Qualitätscontrolling“ genannt, wobei hier nicht klassische Software-Modelle im Sinne der UML gemeint sind, sondern ein Qualitätsmodell die Basis von Qualitätsanalysen darstellt (Abb. 2).

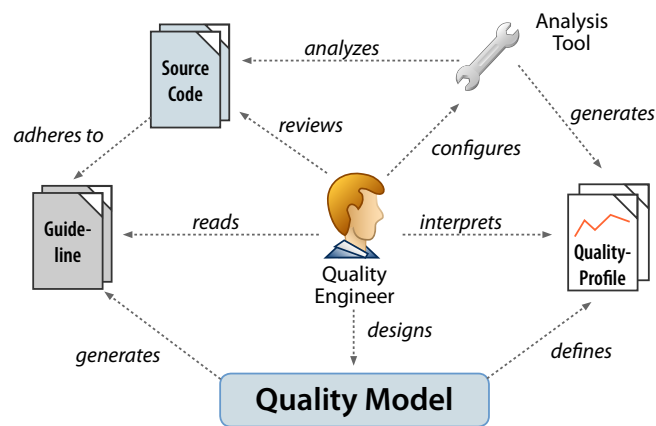


Abbildung 2. Modellbasiertes Qualitätscontrolling

Im Zentrum steht ein „Qualitäts-Ingenieur“, der ein Qualitätsmodell entwirft und auf dieser Basis Richtlinien generiert und Qualitätsprofile definiert. Weiterhin setzt er Analysewerkzeuge ein, um Quellcode und andere Artefakte zu analysieren und deren Konformität mit dem Qualitätsmodell zu prüfen.

## 2 Praxiserfahrungen

Der aktivitätenbasierte Ansatz zur Qualitätsmodellierung wurde inzwischen in einer Reihe von Industriekooperationen eingesetzt. Die wichtigsten Beispiele sind die folgenden drei.

<sup>1</sup> <http://conqat.in.tum.de/>

*Siemens SBS* Die Siemens Business Services arbeiten an betrieblichen Informationssystemen, die teilweise sehr lange Einsatzzeiten vorweisen können. Gerade bei diesen langlebigen Systemen ist eine strukturierte Bewertung der Wartbarkeit von großer Bedeutung. In Zusammenarbeit mit SBS sind die Grundstrukturen unserer Qualitätsmodellierung entstanden [8]. Es wurden Richtlinien gesammelt, vervollständigt und Inkonsistenzen und Widersprüche aufgedeckt.

*MAN Nutzfahrzeuge* Die MAN Nutzfahrzeuge ist ein internationaler Hersteller von Lastkraftwagen und Bussen mit eigener Elektronik- und Software-Entwicklung. Dort wird ein Qualitätsmodell eingesetzt, dass vor allem die Qualität von Matlab/Simulink und Stateflow abdeckt [6]. Aus diesem Qualitätsmodell, werden Review-Richtlinien generiert, die direkt im Entwicklungsprozess eingesetzt werden. Bei der Erstellung des Modells wurden wiederum in bestehenden Richtlinien Inkonsistenzen und Widersprüche entdeckt.

*Münchener Rück* Die Münchener-Rück-Gruppe gehört weltweit zu den größten Rückversicherern und entwickelt und betreibt eine Vielzahl betrieblicher Informationssysteme zur Unterstützung ihrer Geschäftsprozesse. In einer Kooperation mit der Münchener Rück ist ein Qualitätsmodell entstanden, das Programmierrichtlinien und Qualitätskriterien für diese betrieblichen Informationssysteme beschreibt. Das Qualitätsmodell bildet die Grundlage von statischen Code-Analysen, die automatisiert zentrale Qualitätskriterien überprüfen. Die Qualitätsanalysen werden mittels ConQAT in den Entwicklungsprozess integriert und unterstützen dadurch Reviews und ermöglichen ein kontinuierliches Qualitäts-Controlling.

In allen praktischen Anwendungen hat es sich aber gezeigt, dass das Erstellen und Pflegen solcher Modelle aufwändig ist. Ein operationalisiertes Modell, das zur Ableitung von Analysen verwendet wird, muss entsprechend umfangreich sein, um die verschiedenen Qualitätsfacetten abbilden zu können. Es ist also ein Kosten/Nutzen-Verhältnis abzuwägen. Dies ist nach unseren Erfahrungen positiv, hängt aber stark von (1) der Werkzeugunterstützung und (2) der Einbindung in den Entwicklungsprozess ab. Speziell bei MAN hat sich die Verwendung des Modells als vorteilhaft gezeigt, da sowohl automatische Analysen als auch Review-Richtlinien abgeleitet werden und auch Messungen auf dieser Basis geplant sind.

### 3 Zusammenfassung

Die strukturierte Definition von Software-Qualität in Qualitätsmodellen ist essentiell für ihre Handhabung in der Praxis. Nur ein Modell, das als operationalisiertes Artefakt während der Entwicklung auch weiterentwickelt wird, kann langfristig Qualität sicherstellen. Aus den gemachten Erfahrungen in den industriellen Anwendungen kann geschlossen werden, dass dies möglich ist. Wichtig ist aber, dass sowohl eine sinnvolle Struktur, wie auch eine umfangreiche Werkzeugunterstützung vorhanden ist. Die Strukturierung auf Basis von Aktivitäten

hat sich in der Praxis bewährt, weil die Aktivitäten als Hauptkostentreiber eine direkte Verbindung zu ökonomischen Bewertungen ermöglichen. Weiterhin hat sich herausgestellt, dass ein Qualitätsmodell nur erfolgreich sein kann, wenn es sich stark auf die jeweiligen Bedürfnisse der Domäne und Technologien zuschneiden lässt.

## Literatur

1. Garvin, D.A.: What does product quality really mean? MIT Sloan Manage. Rev. **26**(1) (1984) 25–43
2. ISO: ISO 9126: Product Quality – Part 1: Quality Model (2003)
3. Boehm, B.W., Brown, J.R., Kaspar, H., Lipow, M., Macleod, G.J., Merrit, M.J.: Characteristics of Software Quality. North-Holland (1978)
4. McCall, J.A., Richards, P.K., Walters, G.F.: Factors in software quality. Reports NTIS AD/A-049 014, 015, 055, US Rome Air Development Center (1977)
5. Seffah, A., Donyaee, M., Kline, R.B., Padda, H.K.: Usability measurement and metrics: A consolidated model. Software Quality Control **14**(2) (2006) 159–178
6. Deissenboeck, F., Wagner, S., Pizka, M., Teuchert, S., Girard, J.F.: An activity-based quality model for maintainability. In: Proc. 23rd International Conference on Software Maintenance (ICSM '07), IEEE Computer Society Press (2007)
7. Winter, S., Wagner, S., Deissenboeck, F.: A Comprehensive Model of Usability. In: Proc. Engineering Interactive Systems 2007 (EIS '07), Springer (2008) Zur Veröffentlichung angenommen.
8. Broy, M., Deissenboeck, F., Pizka, M.: Demystifying maintainability. In: Proc. 4th Workshop on Software Quality (4-WoSQ), ACM Press (2006) 21–26