

# Softwarequalitätsmodelle

Clara Lange

Technische Universität München  
Fakultät für Informatik  
Boltzmannstraße 3  
85748 Garching-Forschungszentrum  
langecl@in.tum.de

**Abstract:** Softwarequalität ist ein komplexes und vielschichtiges Konzept, das aus verschiedenen Perspektiven betrachtet und durch Qualitätsmodelle beschrieben werden kann. Qualität in Softwareprojekten kann durch Qualitätsmanagement erreicht werden. Die Arbeit befasst sich mit Qualitätsmodellen und ihren Vor- und Nachteilen. Auch auf mögliche Lösungsansätze und auf verbesserte Qualitätsmodelle aufgrund der Nachteile von älteren Qualitätsmodellen sollen eingegangen werden.

## 1 Einleitung

Die vorliegende Arbeit umfasst die Themenbereiche Softwarequalität und Qualitätsmodelle. Im zweiten Kapitel soll der Begriff Softwarequalität definiert und auch in verschiedene Begriffe unterteilt werden. Im darauffolgenden dritten Kapitel werden die fünf Sichten beziehungsweise Perspektiven auf Softwarequalität behandelt. Das vierte Kapitel umfasst das Qualitätsmodell nach McCall, das Qualitätsmodell nach der ISO Norm 9126 und ein neueres Softwarequalitätsmodell. Und abschließend beinhaltet das fünfte Kapitel das Fazit.

## 2 Der Begriff Softwarequalität

Um den Begriff Softwarequalität definieren zu können, bedarf es zuerst einer Definition von Qualität allgemein. Eine mögliche Begriffserklärung ist folgende: „Qualität ist die Gesamtheit der Eigenschaften von Erzeugnissen, die den Grad ihrer Eignung für einen bestimmten Verwendungszweck bestimmt“ [WB84]. Alternativ ist eine mögliche Definition, dass Qualität die „Bezeichnung einer potentiell wahrnehmbaren Zustandsform von Systemen und ihrer Merkmale, welche in einem bestimmten Zeitintervall anhand bestimmter Eigenschaften des Systems in diesem Zustand definiert wird“ [L\_12], ist. Kurz: Qualität entspricht allen charakteristischen Eigenschaften eines Gegenstands.

Die Definition von Qualität im vorhergehenden Absatz, dass Qualität die charakteristischen Merkmale eines Erzeugnisses darstellt, lässt sich auf Softwarequalität übertragen. Hier besteht allerdings das Problem, welche Qualitätsmerkmale zu einer guten Definition von Softwarequalität gehören. Diese Problematik wird anhand der Qualitätsmodelle gezeigt. Eine weitere Dimension von Softwarequalität ist die Einhaltung der festgelegten Anforderungen. Zusammenfassend gibt es demnach keine einheitliche beziehungsweise universelle Definition von Softwarequalität. Man kann sich dem Begriff durch Strukturierung in Prozess- und Produktqualität oder in innere und äußere Qualität annähern.

## **2.1 Produktqualität gegenüber Prozessqualität**

Um Softwarequalität besser verstehen zu können, ist eine Unterscheidung zwischen Produktqualität und Prozessqualität sinnvoll. Produktqualität ist der Grad der Qualität eines Produkts, also die Qualität des entwickelten Produkts. Das eigentliche Ziel eines Softwareprojektes ist ein hochqualitatives Produkt, welches durch Prozessqualität beeinflusst wird. Weiterhin entspricht Produktqualität dem Erfüllungsgrad der Kundenwünsche.

Prozessqualität hingegen beschreibt die Qualität der Herstellungsprozesse für ein Produkt, demzufolge wie der Herstellungsprozess ablaufen soll, kurz die Qualitätsattribute des Softwareentwicklungsprozesses. In einem Softwareprojekt sollte man Prozesse systematisch definieren und verbessern. Wichtig hierbei ist, dass der Prozess ohne Störung und im Idealfall fehlerfrei ablaufen soll, um Nacharbeit und Fehlerkosten zu verhindern. Somit ist auch später die Kundenzufriedenheit gesichert.

Es wird angenommen, dass ein hochqualitativer Prozess zu einem hochqualitativen Produkt führt.

## **2.2 Innere Qualität gegenüber Äußerer Qualität**

Innere Qualität entspricht der Perspektive der Entwickler beziehungsweise des Entwicklungsteams. Die Betrachtung der inneren Qualität spielt eine große Rolle, wenn Software analysiert wird, die in einem Team bearbeitet wird, um sie zum Beispiel an neue Anforderungen anzupassen. Diese Anpassung einer Software wird erleichtert, wenn sie bereits gute Qualität aufweist. Nachfolgende Qualitätseigenschaften sind von Bedeutung: Wartbarkeit, Erweiterbarkeit, Wiederverwendbarkeit, Verständlichkeit und Lesbarkeit des Quellcodes.

Äußere Qualität bezeichnet die Kundenperspektive, das heißt aus der Sicht von Personen, die mit dem Softwareprogramm konfrontiert werden, also mit diesem arbeiten. Hierbei sind folgende Kriterien besonders wichtig: einfache Verwendbarkeit, Bedien- und Benutzbarkeit, Zuverlässigkeit, Robustheit und Fehlertoleranz.

### **3 Sichten auf Softwarequalität**

Um Softwarequalität besser verstehen und definieren zu können, muss sie aus verschiedenen Sichten betrachtet werden. Dabei ist das Messen von Qualität abhängig von den gewählten Perspektiven. Durch das Messen und die dadurch ermittelten Kennzahlen kann ein Vergleich zwischen verschiedenen Softwareprodukten ermöglicht werden.

#### **3.1 Transzendente Sicht**

Die Hauptidee bei der transzendenten Sicht ist, dass sich Qualität nicht präzise definieren und somit auch nicht messen lässt. Nur aufgrund von Erfahrung kann Qualität beurteilt werden. Des Weiteren stellt Qualität ein Ideal dar, welches zwar unmöglich zu erreichen ist, jedoch sollte man versuchen, sich diesem Ideal anzunähern.

#### **3.2 Benutzerorientierte Sicht**

Bei der benutzerorientierten Sicht handelt es sich um eine externe Sicht, das heißt aus der Sicht von Benutzern, Käufern, Konsumenten, aber auch Marketinggruppen. Der Gegensatz zur externen Sicht ist die interne Sicht, also die Entwickler- bzw. Herstellerperspektive. Bezüglich der benutzerorientierten Sicht ist Qualität ausgerichtet auf die Bedürfnisse des Benutzers, welche gegeneinander abgewogen werden müssen. Das dadurch entstehende Problem ist, dass es viele verschiedene Benutzergruppen gibt und somit viele verschiedene Bedürfnisse. Des Weiteren entscheidet der Benutzer subjektiv über die Qualität eines Produkts, somit liegt die Definition von Qualität im Auge des Betrachters.

Für das Messen von Softwarequalität sind vor allem Zuverlässigkeit und Benutzbarkeit von großer Bedeutung für den Benutzer. Zuverlässigkeit ist zum Beispiel messbar durch die Anzahl der Ausfälle über eine gewisse Zeitspanne oder durch die Zeit zwischen den Ausfällen eines Systems. Zur Benutzbarkeit gehört die leichte Installation und das schnelle Erlernen des Umgangs mit dem System. Die Benutzbarkeit ist beispielsweise messbar durch die verstrichene Zeit, bis der Benutzer sich eine gewisse Kompetenz angeeignet hat, um mit dem Softwareprodukt gut umgehen zu können, sprich die Dauer der Einarbeitung ist messbar. Benutzbarkeit ist auch messbar, durch die Anzahl der Klicks, um eine gewünschte Aktion durchzuführen.

### **3.3 Herstellerorientierte Sicht**

Die herstellerorientierte Sicht ist im Gegensatz zur benutzerorientierten Sicht eine interne Sicht, also aus der Entwicklungsabteilung. Qualität entspricht hierbei der Einhaltung der vorher festgelegten Anforderungen. Es ist von höchster Priorität von Beginn des Projektes an ein hochqualitatives Produkt herzustellen, um Nacharbeit und damit verbundene Kosten zu verhindern. Somit spielt die Prozessqualität eine große Rolle bei der Entwicklung von Software. Ein weiterer wichtiger Aspekt sind die Haltbarkeitsüberlegungen, was bedeutet, dass sich das Produkt möglichst lang auf dem Markt halten und somit viel Profit bringen soll.

Bei der Entwicklung spielen vor allem die Anzahl von Defekten und die damit verbundene Nacharbeitungszeit eine große Rolle. Messbar ist die Defektrate, also die Anzahl der Defekte hinsichtlich der Größe des Softwareprodukts und zwar während der Entwicklungszeit. Nach dieser, also nach der Auslieferung, sind noch die Anzahl der Produktverbraucher, die Anzahl der Installationen und der Umfang des Gebrauchs zu beachten, weil bei einem wenig genutzten oder ungenutzten Produkt keine Fehler von Seiten des Kunden bemängelt werden können oder weil bei hohem Gebrauch die Zahl der gefundenen Fehler und somit die Fehlerbehebung ansteigt. Daneben kann die Nacharbeitungszeit durch den Anteil dieser Zeit an der gesamten Entwicklungszeit gemessen werden. Bei der Nacharbeitungszeit entstehen zusätzliche Kosten, diese sollen minimal gehalten werden.

### **3.4 Produktorientierte Sicht**

Der Ansatz bei der produktorientierten Sicht ist, dass Qualität eine exakt definierbare und messbare Größe darstellt. Folglich ist eine Ordnung der Produkte hinsichtlich ihrer Qualität möglich. Die Sicht beurteilt Qualität im Gegensatz zur benutzerorientierten Sicht objektiv und kontextfrei.

### **3.5 Wertorientierte Sicht**

Qualität wird über die Herstellungskosten und einen angemessenen Einkaufspreis definiert, wobei beide Grenzen schwer zu beziffern sind. Resultierend entspricht Qualität einem guten Preis-Leistungs-Verhältnis.

## 4 Qualitätsmodelle

Ein Softwarequalitätsmodell ist eine strukturierte Sammlung von Merkmalen, Kriterien und Eigenschaften, um Softwarequalität systematisch definieren, beurteilen und messen zu können. Außerdem zeigen Qualitätsmodelle, wie die Qualitätsmerkmale miteinander verbunden sind. Somit sind Softwarequalitätsmodelle hilfreich, da man mit ihnen Softwarequalität definieren kann, damit Messungen ermöglicht und einen Vergleich zwischen verschiedenen Softwareprodukten erlaubt. Im Anschluss werden zwei bekannte Qualitätsmodelle erläutert, das Modell nach McCall und nach der ISO Norm 9126.

### 4.1 Qualitätsmodell nach McCall

Das Softwarequalitätsmodell nach McCall stammt aus dem Jahre 1977 und ist ein dreistufiges Modell, bestehend aus elf Qualitätshauptzielen, den Qualitätsfaktoren ("factors"), aus Qualitätskriterien ("criteria") zu jedem Faktor und aus Kenngrößen ("metrics") beziehungsweise Metriken.

Die Metriken kann man unterteilen in binäre und relative Messungen. Relative Metriken sind meist ein Quotient, wohingegen binäre Metriken entweder eine Ja-Nein-Frage darstellen oder hinterfragen, ob etwas existiert oder abwesend ist. Ein Beispiel für eine binäre Messung zur Benutzbarkeit ist, ob eine nützliche Hilfe vorhanden ist. Eine mögliche relative Messung zur Flexibilität ist die Anzahl der allgemein anwendbaren Module geteilt durch die Gesamtanzahl aller Module.

Im Folgenden wird aufgezeigt, welche Faktoren zum Qualitätsmodell nach McCall gehören und welche Kriterien mit welchen Faktoren verbunden sind, da durch diese Verknüpfungen mit den Kriterien die Qualitätsfaktoren nachvollziehbar definiert und gemessen werden können:

- **Korrektheit** bedeutet, inwieweit die Spezifikation und die Anforderungen erfüllt werden, demnach ob das System tut, was es tun soll. Zur Korrektheit gehören Verfolgbarkeit, Vollständigkeit und Konsistenz.
- **Zuverlässigkeit** ist definiert als Leistungsfähigkeit eines Softwaresystems, die angeforderte Funktionalität unter bestimmten Bedingungen zu erbringen und hinterfragt, ob das System dauerhaft stabil beziehungsweise beständig ist. Qualitätskriterien sind Konsistenz, Fehlertoleranz und Genauigkeit.
- **Effizienz** stellt den Umfang dar, in der ein System seine Leistung mit minimalen Ressourcenverbrauch erbringt. Ausführungs- und Speichereffizienz sind die zugehörigen Kriterien zur Effizienz.

- **Integrität** hinterfragt, ob ein System nicht berechtigte Zugriffe auf die Software, auf Daten und deren unberechtigte Veränderung verhindert, sprich ob das Softwaresystem sicher ist. Zu diesem Qualitätsfaktor gehören die Zugriffskontrolle und der Zugriffsnachweis.
- **Benutzbarkeit** bedeutet, wie leicht der Benutzer die Bedienung des Systems erlernen kann. Hierzu gehören die Qualitätskriterien Bedienbarkeit, Trainierbarkeit und Kommunikativität.
- **Wartbarkeit** ist definiert als Leichtigkeit, mit der ein Softwaresystem verändert werden kann, um Fehler zu beheben. Damit verbundene Qualitätskriterien sind Einfachheit, Kürze, Selbsterklärung, und Modularität.
- **Testbarkeit** zeigt, wie groß der Aufwand ist, um Tests durchzuführen und Testfälle zu erstellen. Zugehörige Qualitätskriterien sind Einfachheit, Instrumentierbarkeit, Selbsterklärung und Modularität.
- **Flexibilität** ist definiert als Leichtigkeit, mit der ein Softwaresystem verändert werden kann, um es an neue Anforderungen anzupassen. Zur Flexibilität gehören Einfachheit, Erweiterbarkeit, Allgemeinheit und Modularität.
- **Portabilität** meint, wie leicht man ein System in eine andere Hard- oder Softwareumgebung überführen kann. Zu diesem Qualitätsfaktor zählen Einfachheit, System- und Geräteunabhängigkeit.
- **Wiederverwendbarkeit** zeigt, in welcher Größenordnung ein System oder ein Teil davon erneut verwendet werden kann. Mit dem Qualitätsfaktor Wiederverwendbarkeit stehen Einfachheit, Allgemeinheit, Modularität, System- und Geräteunabhängigkeit in Zusammenhang.
- **Verknüpfbarkeit** ist darüber definiert, wie leicht ein System mit einem anderen gekoppelt werden kann, Informationen austauschen und mit diesen arbeiten kann. Mit der Verknüpfbarkeit sind die Qualitätskriterien Modularität, Kommunikationskompatibilität und Datenkompatibilität verbunden.

#### 4.2 Qualitätsmodell nach Norm ISO 9126

Auch das Qualitätsmodell nach der Norm ISO ("International Standardization Organization") 9126 ist ein dreistufiges Modell, das aus sechs produktbezogenen Qualitätsmerkmalen, den dazugehörigen Qualitätsteilmerkmalen und Qualitätsindikatoren, sprich den Metriken, aufgebaut ist. Im Gegensatz zum Qualitätsmodell nach McCall ist ISO 9126 ein streng hierarchisch angeordnetes Modell, d.h. jedes Teilmerkmal ist nur mit einem Hauptmerkmal verbunden.

Folgende Qualitätmerkmale sind Bestandteil des Qualitätsmodells:

- **Funktionalität:** Wurden die geforderten und benötigten Funktionen implementiert und funktionieren diese auch, sprich sind sie ausführbar?
- **Zuverlässigkeit:** Kann die Software ihr Leistungsniveau halten und ist eine angemessene Freiheit von Fehlern und Defekten garantiert?
- **Benutzbarkeit:** Wie hoch ist der Aufwand, bis ein Benutzer das System bedienen und den Umgang erlernen kann?
- **Effizienz:** Wie ist das Verhältnis zwischen der Systemleistung, sprich das zeitliche Verhalten, und den eingesetzten Betriebsmitteln?
- **Änderbarkeit:** Wie leicht lässt sich das System verändern, um es an neue Anforderungen oder neue Umgebungen anzupassen?
- **Übertragbarkeit:** Ist es realisierbar, das System in eine andere Hard- oder Softwareumgebung zu transferieren?

Folgende Grafik zeigt den Zusammenhang zwischen den sechs Qualitätsmerkmalen und den dazugehörigen Teilmerkmalen des ISO Modelles 9126:

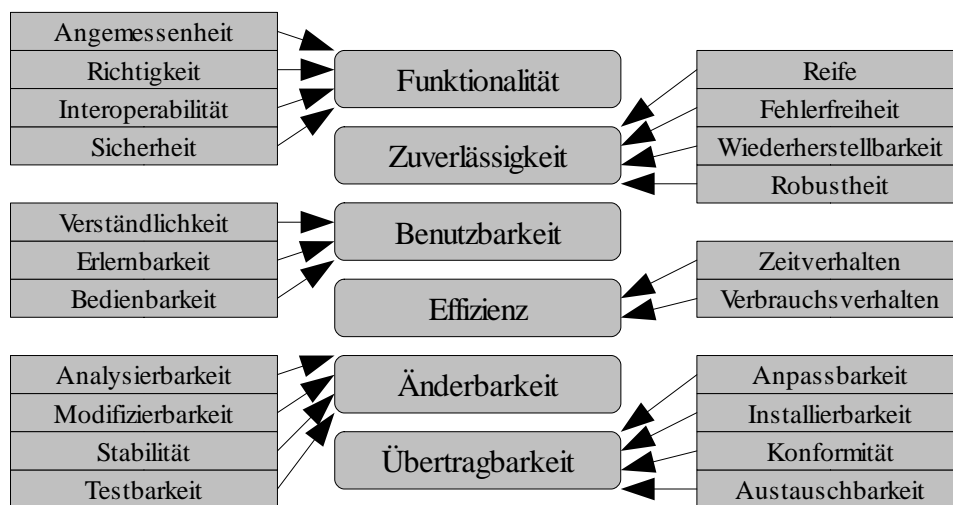


Abbildung 1: Merkmale (mittig) und Teilmerkmale (links & rechts) von ISO 9126

### **4.3 Kritik an Qualitätsmodellen**

Trotz des Vorteils der Qualitätsmodelle, dass sie es überhaupt ermöglichen, Softwarequalität definieren und messen zu können, weisen sie einige Probleme auf. Vor allem sind Qualitätsmodelle sehr standardisiert und nicht an den individuellen Entwicklungsprozess angepasst, somit besteht kein Bezug zu Kosten und Nutzen der jeweiligen Software. Auch auf die individuellen Anforderungen, die von Projekt zu Projekt stark variieren, wird keine Rücksicht genommen. Ein weiterer Kritikpunkt ist die Zuordnung der Qualitätsmerkmale zu einer Definition von Softwarequalität und der Zusammenhang zwischen Merkmalen und Teilmerkmalen. Beide Zuordnungen sind von Modell zu Modell unterschiedlich, somit gibt es keine eindeutige Definition von Softwarequalität. Der letzte große Kritikpunkt ist das Messen von Qualitätsmerkmalen, da dieses meist nicht eindeutig ist. Die Messungen sind sehr unpräzise und auch die Metriken sind nicht aussagekräftig genug.

### **4.4 Mögliche Lösungsansätze und verbesserte Qualitätsmodelle**

Mögliche Lösungsansätze sind, dass man Qualitätsmodelle hinsichtlich der individuellen Anforderungen eines Projekts anpassen und erweitern kann. Es wird empfohlen, Qualitätsmodelle zum Beispiel nach McCall oder der ISO 9126 Norm nicht Eins-zu-Eins zu übernehmen, sondern diese auf das jeweiligen Projekt zuzuschneiden. Des weiteren ist Prozesskontrolle ein wichtiger Aspekt, da man nur durch einen hochqualitativen Prozess ein hochqualitatives Produkt herstellen kann

Ein weiterer wichtiger Ansatz sind aktivitätenbasierte Qualitätsmodelle. Diese versuchen, die Problematik älterer Modelle, nämlich dass der Bezug zum individuellen Projekt und somit Projektverlauf fehlen, zu lösen. Außerdem wird jedes Qualitätsmerkmal einzeln betrachtet. Hinzu kommt eine explizite Trennung von technischen Eigenschaften und Aktivitäten im Entwicklungsprozess. Durch diese Spaltung resultiert eine Qualitätsmatrix mit den zwei Dimensionen Aktivitäten und technischen Eigenschaften zu jeweils einem Qualitätsmerkmal.

Ein Beispiel wäre die "Wartung". Der Prozess der Wartung gehört allgemein zum Entwicklungsprozess und dazu gehören die Subaktivitäten "Modifizieren", "Testen" und "Verstehen bzw. Analysieren" und die Eigenschaften "Strukturiertheit", "Kommunikativität", "Selbsterklärbarkeit" und "Kürze bzw. Prägnanz". Ein Kreuz in der Matrix kennzeichnet den Einfluss einer technischen Eigenschaft auf eine Aktivität. Das aktivitätenbasierte Qualitätsmodell besagt beispielsweise, wenn man ein System wartet, muss man es modifizieren und die Strukturiertheit beeinflusst das Modifizieren, also je strukturierter das Programm, desto besser lässt es sich modifizieren.



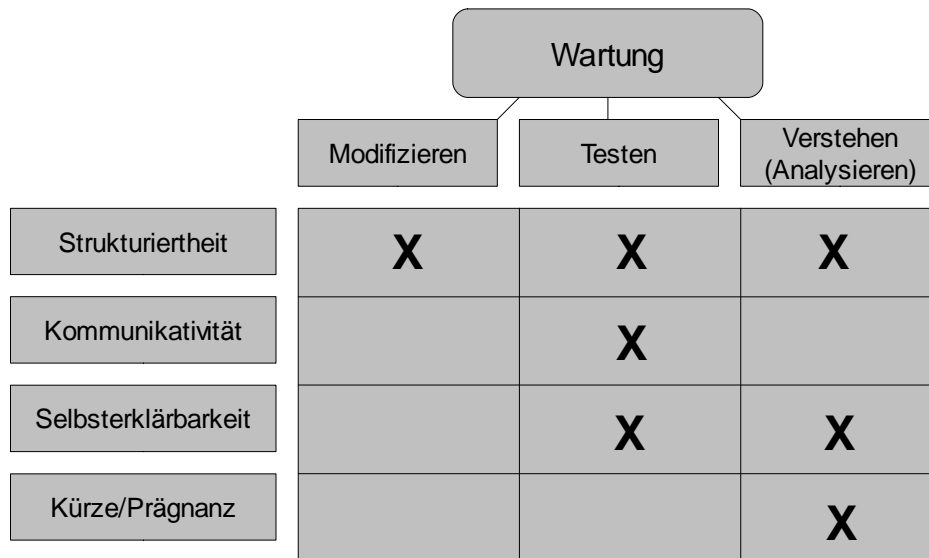


Abbildung 2: Ausschnitt eines aktivitätenbasierten Qualitätsmodells zur "Wartung" (Quelle: <http://mediatum2.ub.tum.de/doc/737380/737380.pdf>)

## 5 Fazit

Im Entwicklungsprozess einer Software spielt Qualität eine große Rolle - eine angemessene Qualität ist eines der Hauptziele bei Softwareprojekten.

Daher stellt Qualitätsmanagement einen großen Bestandteil von Softwareprojekten dar. Qualitätsmanagement ist definiert als alle organisatorischen und technischen Maßnahmen, die der Erschaffung und der Erhaltung von Softwarequalität dienen. Dazu gehören Qualitätsplanung, Qualitätssteuerung und Qualitätskontrolle beziehungsweise Qualitätsprüfung.

Im Bereich des Qualitätsmanagements kann man auf Qualitätsmodelle zurückgreifen. Häufig finden die Softwarequalitätsmodelle nach McCall und nach der ISO Norm 9126 in der Praxis Verwendung, zum Teil auch in abgewandelter Form. Diese Modelle haben es ermöglicht, Qualität zu definieren und zu messen. Die Ermittlung von Kennzahlen ist ein wichtiger Bestandteil.

In den letzten Jahren haben sich Qualitätsmodelle verändert und weiterentwickelt. Große Bedeutung gewannen dabei aktivitätenbasierte Modelle, diese unterstützen den Entwicklungsprozess besser.

Zusammenfassend sind Softwarequalitätsmodelle ein wichtiges Hilfsmittel im Qualitätsmanagement, welches alle Projektphasen betrifft, und den Entwicklungsprozess einer Software unterstützt.

## Literaturverzeichnis

- [KP96] Kitchenham, Barbara, Pfleeger, Shari L., Software Quality: The Elusive Target, IEEE Software, v.13 n.1, p.12-21, January 1996
- [M77] McCall, Jim A., Richards, Paul K., Walters, Gene F., Factors in Software Quality – Concept and Definition of Software Quality, November 1977
- [G84] Garvin, David A., What Does „Product Quality“ Really Mean?, Sloan Management Review, 1984
- [CM] Cavano, Joseph P., McCall, James A., A Framework for Measurement Of Software Quality
- [WB84] Großes Fremdwörterbuch, Bibliographisches Institut Leipzig, 1984
- [I\_01] [http://user.cs.tu-berlin.de/~sepradm/ws9900/projekt/referate/SQS\\_folien.pdf](http://user.cs.tu-berlin.de/~sepradm/ws9900/projekt/referate/SQS_folien.pdf), zuletzt aufgerufen am 10.07.2010
- [I\_02] <http://de.wikipedia.org/wiki/Softwarequalit%C3%A4t>, zuletzt aufgerufen am 10.07.2010
- [I\_03] [http://de.wikipedia.org/wiki/ISO/IEC\\_9126](http://de.wikipedia.org/wiki/ISO/IEC_9126), zuletzt aufgerufen am 10.07.2010
- [I\_04] [http://wwwmatthes.in.tum.de/file/Teaching/EIST/public/Teil2\\_1.pdf](http://wwwmatthes.in.tum.de/file/Teaching/EIST/public/Teil2_1.pdf), zuletzt aufgerufen am 10.07.2010
- [I\_05] [http://pi.informatik.uni-siegen.de/lehre/2007s/LM/lm\\_sqmo\\_20070505\\_a5.pdf](http://pi.informatik.uni-siegen.de/lehre/2007s/LM/lm_sqmo_20070505_a5.pdf), zuletzt aufgerufen am 10.07.2010
- [I\_06] [http://www.ifi.uzh.ch/terg/fileadmin/downloads/teaching/courses/swq\\_ss07/kapitel\\_06\\_Q\\_d\\_e.pdf](http://www.ifi.uzh.ch/terg/fileadmin/downloads/teaching/courses/swq_ss07/kapitel_06_Q_d_e.pdf), zuletzt aufgerufen am 10.07.2010
- [I\_07] [http://books.google.de/books?id=7zbSZ54JG1wC&pg=PA10&lpg=PA10&dq=kitchenham+pfleeger+mccall&source=bl&ots=fifrv61nf&sig=YphXrUXqaFONCuMdZEht0zg3D\\_8&hl=de&ei=CpfRS9zpFd-JOOzO4asO&sa=X&oi=book\\_result&ct=result&resnum=3&ved=0CBEQ6AEwAg#v=onepage&q=kitchenham%20pfleeger%20mccall&f=false](http://books.google.de/books?id=7zbSZ54JG1wC&pg=PA10&lpg=PA10&dq=kitchenham+pfleeger+mccall&source=bl&ots=fifrv61nf&sig=YphXrUXqaFONCuMdZEht0zg3D_8&hl=de&ei=CpfRS9zpFd-JOOzO4asO&sa=X&oi=book_result&ct=result&resnum=3&ved=0CBEQ6AEwAg#v=onepage&q=kitchenham%20pfleeger%20mccall&f=false), zuletzt aufgerufen am 10.07.2010
- [I\_08] <http://wirtschaftslexikon.gabler.de/Definition/qualitaetssicherung.html>, zuletzt aufgerufen am 10.07.2010
- [I\_09] [http://www.qm-infocenter.de/qm/o\\_bs.asp?task=4&basic\\_id=2335175111-79&bt=01000.00010](http://www.qm-infocenter.de/qm/o_bs.asp?task=4&basic_id=2335175111-79&bt=01000.00010), zuletzt aufgerufen am 10.07.2010
- [I\_10] <http://www4.informatik.tu-muenchen.de/publ/papers/TUMI0811.pdf>, zuletzt aufgerufen am 10.07.2010
- [I\_11] <http://mediatum2.ub.tum.de/doc/737380/737380.pdf>, zuletzt aufgerufen am 10.07.2010
- [I\_12] <http://de.wikipedia.org/wiki/Qualit%C3%A4t>, zuletzt aufgerufen am 10.07.2010