

Diplomarbeit

Wirtschaftliche Betrachtung von Qualitätsmaßnahmen in der Softwareentwicklung

ausgeführt im
Fachhochschul-Diplomstudiengang Internationale Unternehmensführung

von

Markus Rüt

Maria Theresienstraße 19

110116037

geboren am 27.02.1974 in Lauterach

Betreuer: Dipl.-Kfm. Dipl.-Ing. (BA) Christoph Benz

Eingereicht an der Fachhochschule Vorarlberg GmbH.



Sperrvermerk

Auf Wunsch der Firma ThyssenKrupp-Presta ist die vorliegende Diplomarbeit für die Dauer von zwei Jahren für die öffentliche Nutzung zu sperren.

Einsichtnahme, Veröffentlichung und Vervielfältigung sind ohne ausdrückliche Genehmigung der Firma ThyssenKrupp-Presta und des Verfassers bis zum 30. April 2007 nicht gestattet.

Eschen, 28. April 2005

Hinweis: Diplomarbeitsthema sowie die Kurzfassung/Abstract dürfen veröffentlicht werden.

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Diplomarbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Stellen sind als solche kenntlich gemacht.

Diese Arbeit wurde bislang weder in gleicher noch in ähnlicher Form einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Lustenau, 28. April 2005

Markus Rüb

Vorwort

Seit geraumer Zeit beschftige ich mich nun eingehend mit der Informatik. Ich entwickle und schreibe Software seit Beginn der 90er Jahre des vorigen Jahrhunderts. Stndig bin ich mit der Problematik „nicht korrekt funktionierender, den Anforderungen nicht entsprechender Software“ konfrontiert. Einerseits als Lieferant, andererseits als Projektleiter und schlielich auch als Kunde und Anwender. Aus diesen Grnden ist es mir ein besonderes Anliegen die Qualitt von Software zu frdern, aber auch einmal zu analysieren welche Manahmen zur Steigerung/Sicherung der Qualitt dienen und ob sie neben ihrer Effizienz auch wirtschaftliche Vorteile bringen. Als Software-Projektleiter beim Automobilzulieferer ThyssenKrupp-Presta bin ich tagtglich mit mangelnder Qualitt und Fehlern in Softwareprodukten konfrontiert und den Auswirkungen der daraus entstehenden Fehlerkosten auf den Geschftserfolg.

Mein besonderer Dank gilt fr die Betreuung der vorliegenden Diplomarbeit an der Fachhochschule Vorarlberg Herrn Dipl.-Kfm. Dipl.-Ing. (BA) Christoph Benz.

Bedanken mchte ich mich auch bei all jenen, die mich in irgendeiner Art und Weise untersttzt haben.

Die Arbeit widme ich meiner Freundin Sonja.

Anmerkungen

Aus Grnden der besseren Lesbarkeit kommen in dieser Arbeit fr beide Geschlechter die grammatikalisch maskulinen Bezeichnungen zur Anwendung. Sind bei den Abbildungen oder Tabellen keine Quellen angegeben, wurden diese vom Verfasser erstellt.

Inhaltsangabe

Sperrvermerk

Eidesstattliche Erklärung

Vorwort

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Kurzfassung

Abstract

1 Einleitung	14
1.1 Beispiele für die Verbreitung von Softwaresystemen.....	14
1.2 Problemstellung - Von der Qualität, den Kosten und der Wartung von Software.....	15
1.3 Bedeutung von Software.....	16
1.5 Fazit	17
2 Zielsetzung der Arbeit	18
3 Qualität als wichtiger unternehmerischer Erfolgsfaktor	19
3.1 Gründe für eine Wirtschaftlichkeits-betrachtung	20
3.2 „Null-Fehler“ Forderung (vgl. Crosby 1980).....	23
3.3 Steigende Kundenerwartungen.....	23
4 Vorstellung des Unternehmens ThyssenKrupp-Presta	25
5 Software	27
5.1 Definition Software.....	27
5.2 Softwarequalität	29
5.2.1 Versuch einer Definition von Qualität	29
5.2.3 Bedeutung des Qualitätsbegriffs	33
5.4 Operationalisierung der Qualität	36
5.5 Software Fehler	37
6 Auswirkungen der Softwarequalität	42
6.1 Gründe für Mängel an Softwarequalität?	42
6.2 Beispiele für mangelnde SW-Qualität	45
7 Anforderungsmerkmale von Softwarequalität (Ausprägung)	48

9 Maßnahmen um „gute“ Softwarequalität zu generieren.....	53
10 Wirtschaftlichkeitsanalyse	60
11 Qualitätskosten	65
11.1 Entwicklung der Qualitätskosten.....	67
11.2 Tätigkeitsorientierte Analyse qualitäts-bezogener Kosten.....	67
11.3 Wirkungsorientierte Analyse qualitäts-bezogener Kosten	70
11.4 Fehlerkostenrechnung.....	73
11.5 Zusammenfassung.....	74
12 Nutzen von Qualitätsmaßnahmen.....	75
12.1 Nutzenarten des Qualitätsmanagements.....	76
12.2 Nutzenkategorien des Qualitätsmanagements	77
13 Wirtschaftlichkeitsmodell des Qualitätsmanagements	80
14 Wirtschaftlichkeitsanalyse des QDA-Softwareprojekts.....	84
14.1 Vorstellung des Projektes.....	84
14.2 Erfassung der Fehlerkosten des QDA-Projekts	86
14.3 Untersuchung der Fehler	86
14.4 Aufstellung der Fehlerkosten.....	91
14.5 Notwendigkeit der Durchführung von Qualitätsmaßnahmen	96
14.6 Vorschläge und Kostenabschätzung	
für Qualitätsmaßnahmen im QDA-Projekt	97
14.6.1 Allgemeine Forderungen.....	97
14.6.2 Forderung von Maßnahmen aufgrund der	
Fehlerkosten-Untersuchung des QDA-Projekts	102
14.6.3 Wirtschaftlichkeit der Maßnahmen	107
14.6.4 Fazit.....	109
15 Zusammenfassung.....	110
Literaturverzeichnis	112
Anhang	118

Abbildungsverzeichnis

Abb. 1: Wartungskosten eines Softwareprojekts	S. 16
Abb. 2: Western European ICT market annual growth	S. 17
Abb. 3: Die Spannungsdreiecke	S. 19
Abb. 4: Qualität als Wettbewerbsfaktor: die PIMS Studie	S. 21
Abb. 5: Arten von Software (nach Standardisierungsgrad)	S. 28
Abb. 6: Faktoren des Qualitätsbegriffs	S. 29
Abb. 7: Einordnung der Softwarequalität in die Qualitätswelt	S. 32
Abb. 8: Spannungsdreieck der Software	S. 32
Abb. 9: FCM-Ansatz zur Operationalisierung der Qualität	S. 34
Abb. 10: Qualitätsmodell für Software	S. 35
Abb. 11: Fehlerentstehungskosten	S. 40
Abb. 12: Fehlerbeseitigungskosten	S. 40
Abb. 13: Kosten einer verzögerten Fehlerentdeckung	S. 41
Abb. 14: Fehlersummutationsauswirkung	S. 44
Abb. 15: Absturz der Ariane 5 am 4. Juni 1996	S. 46
Abb. 16: Erfolgsstatistik von IT-Projekten	S. 47
Abb. 17: Anforderungsmerkmale von Softwarequalität	S. 49
Abb. 18: V-Modell	S. 51
Abb. 19: Konstruktives QM	S. 59
Abb. 20: Analytisches QM	S. 59
Abb. 21: Arten der qualitätsbezogenen Wirtschaftlichkeitsanalyse	S. 62
Abb. 22: Phasenkonzept der Wirtschaftlichkeitsanalyse	S. 64
Abb. 23: Relative Anteile am Entwicklungsaufwand	S. 66
Abb. 24: Tätigkeitsorientiertes Modell qualitätsbezogener Kosten	S. 69
Abb. 25: Tätigkeits- vs. wirkungsorientierte Kostengliederung	S. 70
Abb. 26: Wirkungsorientiertes Modell qualitätsbezogener Kosten	S. 72
Abb. 27: Erweiterte Gliederung der Fehlerkosten nach der Fehlerkostenrechnung	S. 73
Abb. 28: Nutzenarten des Qualitätsmanagements	S. 76
Abb. 29: Nutzenkategorien des Qualitätsmanagements	S. 77

Abb. 30: Kostenkurve des Qualitätsmanagements	S. 80
Abb. 31: Zusammenhang zwischen Aktivitätsniveau und Nutzen	S. 81
Abb. 32: Optimales Aktivitätsniveau des Qualitätsmanagements	S. 82
Abb. 33: Funktionsprinzip und Datenfluss der QDA-Software	S. 85
Abb. 34: QDA Fehler beim Lieferanten	S. 88
Abb. 35: Paretodiagramm der QDA-Fehleranzahl	S. 90
Abb. 36: Neue V-Modell XT von der deutschen Bundesregierung	S. 98
Abb. 37: Beispielvorlage für Testprotokoll	S.100
Abb. 38: Beispielvorlage für Testspezifikation	S.100
Abb. 39: Folgen von Missverständnissen	S.118

Tabellenverzeichnis

Tab. 1: QDA Fehler bei TKA-Presta	S. 87
Tab. 2: QDA Fehlerklassifizierung bei TKA-Presta	S. 89
Tab. 3: QDA Fehlerkosten bei TKA-Presta	S. 93
Tab. 4: QDA Rollout Aufwände bei TKA-Presta	S. 93
Tab. 5: interne Kosten des Lieferanten bei externen Fehlern	S. 94
Tab. 6: Anteil der Fehlerkosten an Softwaregesamtkosten	S. 95
Tab. 7: Fehlerkosten in MM	S. 96
Tab. 8: Kosten der Qualitätsmaßnahmen	S.107
Tab. 9: Erlöspotenzial aus den Qualitätsmaßnahmen im QDA-Projekt	S.108
Tab.10: opportunitäts Gewinn durch Qualitätsmaßnahmen im QDA-Projekt	S.109

Abkürzungsverzeichnis

Abb.	Abbildung
aktual.	aktualisiert(e)
Aufl.	Auflage
Bd.	Band
bzw.	beziehungsweise
d.h.	das heißt
DV	Datenverarbeitung
DV-System	Datenverarbeitungs-System
erw.	erweitert(e)
etc.	et cetera
EUR	Euro
ff	folgend(e)
FM	Folgemonat
GJ	Geschäftsjahr
Hrsg.	Herausgeber
IO-Teile	In Ordnung Teile
IT	Informationstechnologie
LOC	Line of Code
MA	Mitarbeiter
MM	Mannmonat
Mio	Million(en)
NIO-Teile	Nicht in Ordnung Teile
PStk	Produzierte Stück
QM	Qualitätsmanagement
SW-PL	Software-Projektleiter
SW	Software
Tab.	Tabelle
TEUR	Tausend Euro
TKA	ThyssenKrupp Automotive
TKA-Presta	ThyssenKrupp-Presta AG

TQM	Total Quality Management
u.a.m.	und andere(s) mehr
usw.	und so weiter
überarb.	überarbeitet(e)
veränd.	verändert(e)
verb.	verbessert(e) ³
vgl.	vergleiche
Vgl.	Vergleich
vs.	versus
wesentl.	wesentlich
z. B.	zum Beispiel

Kurzfassung

Die Informatik zählt derzeit zu den dynamischsten Gebieten unserer Zeit. Unglaublich schnell hat die Software alle Lebensbereiche erobert. Durch diese „Revolution“ kommen manche Aspekte, die in anderen Bereichen als grundlegend angesehen werden, zu kurz.

Immer wichtiger wird deshalb die Softwarequalität. Wie kann es gelingen die Forderungen des Kunden nach geringen Kosten, rascher Verfügbarkeit und hohem Nutzen zu vereinbaren?

Diese Arbeit zeigt den aktuellen Stand des Softwarequalitätsbegriffs in der Literatur, und wie Softwarequalität fassbar gemacht werden kann. Denn eines ist zu beachten: die Softwareentwicklung unterscheidet sich sehr wesentlich von der industriellen Produktion, in der die Qualität schon lange als ständige Herausforderung in den Arbeitsprozess integriert ist. Qualität soll nicht nur gemessen werden können, sondern diese Arbeit definiert auch Maßnahmen wie Softwarequalität umgesetzt wird.

Es ist sehr wichtig entsprechende Softwarequalität zu generieren, doch zum unternehmerischen Handeln gehört auch - ökonomisch zu Denken. Deshalb bildet ein weiterer Schwerpunkt der Arbeit die wirtschaftliche Betrachtung von Qualität. Eine Theorie ist nur so gut, wie sie sich in der Praxis bewährt. Darum wird sie anhand eines praktischen Softwareprojekts in der ThyssenKrupp-Presta auf ihre Tauglichkeit überprüft.

Abstract

Informatics is at present one of the most rapidly changing areas of modern life. Unbelievably fast has software conquered all areas of life. In this "revolution" some aspects, which are fundamentally regarded within other ranges, fall short. Therefore the software quality becomes more and more important. How can the combined demands of the customer for less costs, less time until availability and added valued be met?

This thesis discusses the current definition of the software quality term in the literature and how to measure software quality.

One thing has to be considered: Software development differs substantially from the traditionally industry production. Where as in the industry, quality is already totally integrated as a continuous challenge of the working process. Quality has not to be only measurably, this thesis also describes methods how to get higher software quality.

It is very important to generate correct software quality, but economic thinking is another important requirement of business as well. Therefore a further part of this thesis will focus on the economic point of view of quality. A theory is only as good as the practice proofs. So the theory has to be evaluated on a real software project in the ThyssenKrupp-Presta.

1 Einleitung

Die Softwareentwicklung begann mit der Einführung von Informationsverarbeitenden DV-Systemen, z.B. Computer, in den 40er Jahren des vorigen Jahrhunderts (vgl. Liggesmeyer 2002: V). Die Informatik hat nicht zuletzt durch den gesellschaftlichen Wandel von der Produktions- zur Wissens-/ Dienstleistungsgesellschaft beträchtlich an Umfang gewonnen. In vielen Lebensbereichen haben DV-Systeme Einzug gehalten, diese werden oft maßgeblich durch Software beeinflusst oder sind sogar von Software abhängig.

1.1 Beispiele für die Verbreitung von Softwaresystemen

1. Haushaltselektronik

„einfachste“ Geräte wie Kaffeemaschinen, Waschmaschinen und Kühlschränke beinhalten DV-Systeme (Microcontroller).

2. Konsumelektronik wie Mobiltelefone, HiFi-Geräte und Digitalkameras beinhalten zum größten Teil Softwaresysteme

3. Automobilindustrie

Betriebliche Abläufe, Verwaltung und Produktion wären ohne Softwaresysteme nicht mehr möglich. Z.B. sind heute in einem Fahrzeug nahezu 200 Mikrocontroller integriert. Mehr als die Hälfte aller Fahrzeugpannen lassen sich auf Softwareprobleme zurückführen (vgl. Halfmann 2002: 12)

4. Informationssysteme

Medizinwesen, Finanzwesen, Verwaltung, etc.

Informationssysteme haben in der Unterstützung von Geschäftsprozessen einen Durchdringungsgrad von 60 % bis 90 % erreicht. An der Abwicklung eines Geschäftsprozesses sind unter Umständen mehr als 15 Großanwendungen beteiligt (vgl.

Hammerschall 2002: 5ff). Bei der ThyssenKrupp-Presta werden z.B. alle Qualitätsmaßnahmen für die Produktion durch ein elektronisches Qualitätsmanagement bzw. einen Workflow gelenkt. Qualitätsprüfungen in der Produktion werden zu 100 % mit Software analysiert.

1.2 Problemstellung - Von der Qualität, den Kosten und der Wartung von Software

Dadurch, dass Software in so vielen Bereichen eingesetzt wird, gewinnt deren Qualität im Bezug auf ihre korrekte und zuverlässige Funktion immer mehr an Bedeutung. Hielt schon vor längerer Zeit in industriellen Produktionsbetrieben, vor dem Hintergrund der Austauschbarkeit ihrer Leistungen (ihrer Produkte), Qualitätsmanagement in verschiedensten Ausprägungen Einzug in den gesamten Unternehmensprozess, so wird Qualitätsmanagement in der Softwarebranche erst seit kurzem gelebt. Softwarekosten waren früher im Verhältnis zu Hardwarekosten relativ gering, Software wurde früher z.B. beim Kauf eines Computers als „kostenlose“ Zugabe mitgegeben. Heutzutage sind die Softwarekosten sehr stark gestiegen, bzw. die Hardwarekosten sinken immer mehr. (vgl. Liggesmeyer 2002: 2). Nicht zuletzt auch deshalb, da bei der Hardware Produktion immer effizientere Produktionstechniken zum Einsatz kommen, bei der Softwareentwicklung die Kosten pro Line of Code (LOC) aber in etwa gleich geblieben sind. Dem zufolge sind Maßnahmen zur Kostenreduktion bei der Softwareentwicklung ein gewichtiger Hebel. Studien zeigen weiters, dass 50 % bis 80 % der Gesamtkosten in der Wartungsphase eines ausgelieferten, auf den Markt gebrachten Software Produkts entstehen (vgl. Liggesmeyer 2002: 4). Dies sind Kosten, die aufgrund mangelnder Qualität (zur Definition dazu später mehr) anfallen. Ursachen sind oft Fehler die bei der Entwicklung entstanden sind, die aber erst nach Auslieferung entdeckt werden. Weiters werden in der Nutzungsphase des Software-Lebenszyklus Mängel aufgedeckt, die auf

Fehler in der Entwurfs- oder Designphase zurückgeführt werden können. Beim Automobilzulieferer ThyssenKrupp-Presta werden zur Überwachung und Erstellung der Qualität in der Produktion Softwaresysteme eingesetzt, die einen sehr hohen Individualisierungsgrad aufweisen. Ist z.B. die mechanische Schaffung von Produktionsmaschinen stark standardisiert und kontrollierbar, so nimmt die Programmierung der Maschinen-Steuerungen inzwischen mehr als 60 % der gesamten Projektdurchführungszeit in Anspruch.

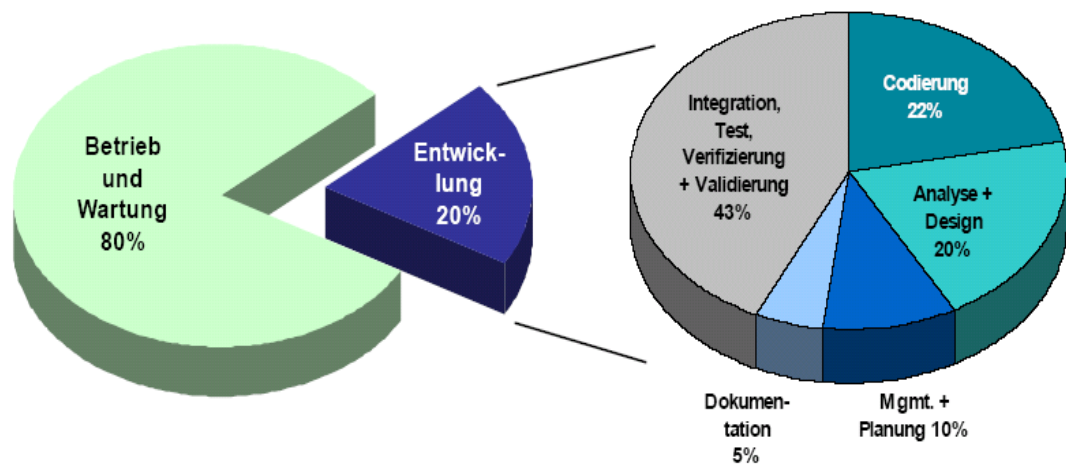


Abb. 1: Wartungskosten eines Softwareprojekts

Quelle: Robert N. Charlette, Software Engineering Risk Analysis and Management

1.3 Bedeutung von Software

Wie schon erwähnt, durchdringt Software zunehmend alle Lebensbereiche, trotzdem wird sie oft nicht in diesem Umfang wahrgenommen. Das Bewusstsein tritt erst dann zu Tage, wenn Systeme nicht so funktionieren wie sie sollten. Z.B. bei Verspätungen im Flugverkehr, - wenn Ampelanlagen ausfallen, - wenn das eigene Computersystem von Viren befallen wird und streikt, - wenn das neue Auto sich nicht mehr mit dem elektronischen Key starten lässt, etc.

Immer noch bin ich überrascht, dass in einem Produktionsbetrieb wie in jenem der TKA-Presta, so viele reine Software-Arbeitsplätze bestehen.

Z.B. in meinem Bereich der Fertigungsplanung, der sich mit der Schaffung von Produktionsmitteln (Neuanlagenbeschaffung, Typenintegrationen, Werkzeugbau) und deren Wartung beschäftigt, verbringen 100 % der MA durchschnittlich 60 % ihrer Arbeitszeit vor dem Computer.

1.5 Fazit

Immer mehr entscheidet die Software über den globalen Wettbewerb und wird zum bedeutenden Faktor. Zudem sichert Software Arbeitsplätze in der traditionellen Industrie (z.B. Kfz, Handel) und schafft neue Arbeitsplätze (z.B. Medien, Dienstleistungen) (Rombach and Martin 1995: 1ff).

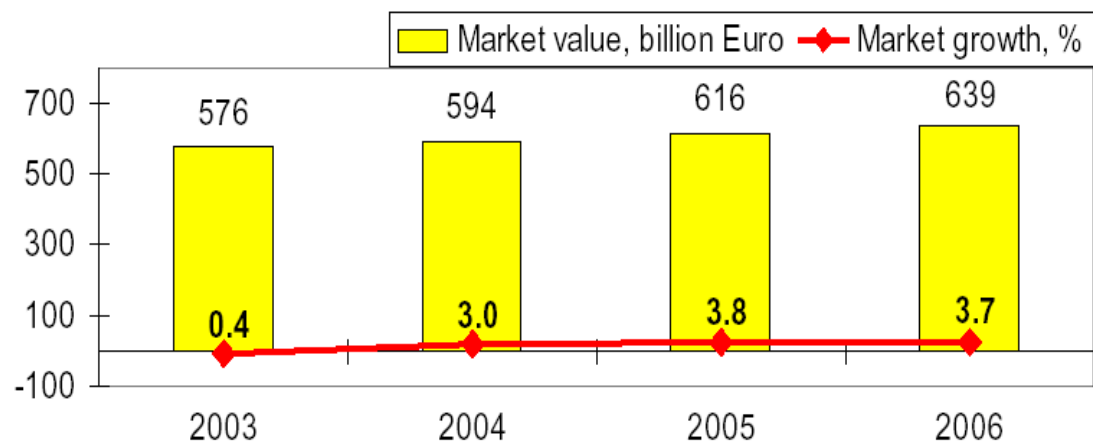


Abb. 2: Western European ICT market annual growth, 2003-2006

Quelle: <http://www.eito.com/tables.HTML>, (Zugriff am: 2005.04.28).

2 Zielsetzung der Arbeit

„Was man nicht messen kann... kann man nicht kontrollieren“
(DeMarco 1982: 1).

Zielsetzung der vorliegenden Diplomarbeit:

1. Einerseits soll der Begriff der Softwarequalität fassbar gemacht werden, denn, trotz inzwischen zahlreicher Normen, wird dieser noch immer in der Praxis sehr unterschiedlich interpretiert (und aufgrund der Kundensichtweise auch immer subjektiv wahrgenommen).
2. Andererseits soll aufgezeigt werden, welche Rahmenbedingungen/ Maßnahmen zur Steigerung der Qualität in der Softwareentwicklung in der wissenschaftlichen Literatur existieren.
3. Die Wirtschaftlichkeitsanalyse besteht aus der Gegenüberstellung von Kosten und Nutzen. Daher werden Erfassungsmodelle von Qualitätskosten und –nutzen beschrieben.
4. Danach soll für ein konkretes Software Projekt (QDA: Quality Data Analysis, eine Software-Anwendung zur rationellen Suche von Qualitätsdaten von Produkten (z.B. Lenksäulen) und deren Aufbereitung für eine statistische Auswertung) bei der TKA-Presta die Kosten der Fehler/ bzw. die Kosten für Nachbesserungen und Nichterfüllen von Kundenanforderungen aufgezeigt werden.
5. Aufbauend auf diesen eruierten Kosten sollen Qualitätsmaßnahmen definiert werden und mit Kosten bewertet werden. Die Gegenüberstellung der bisher angefallenen Fehlerkosten im QDA-Projekt zum Nutzen (Opportunitätserlöse) von neuen Qualitätsmaßnahmen soll Argumente für die Verpflichtung des Lieferanten zur Umsetzung dieser Qualitätsmaßnahmen liefern.

3 Qualität als wichtiger unternehmerischer Erfolgsfaktor

Qualität ist inzwischen nicht nur „ein“ Wettbewerbsfaktor, nein Qualität entscheidet darüber, ob man überhaupt am Wettbewerb teilnimmt. Qualität wird seit Mitte der neunziger Jahre des vorigen Jahrhunderts zunehmend von den verschiedenen Anspruchsgruppen der jeweiligen Organisationen eingefordert. Im Zuge dieser Entwicklung haben Aktivitäten des Qualitätsmanagements einen bedeutenden Stellenwert bei der Befriedigung der Kundenbedürfnisse eingenommen. Zentrale Kundenforderungen an ein Produkt sind:

1. **Nutzen,**
2. **Preis** und
3. **Verfügbarkeit.**

Unternehmerische Tätigkeiten die auf die Kundenbedürfnisse ausgerichtet sind, stehen wiederum im unternehmerischen Spannungsdreieck von

1. **Quantität / Zeit,**
2. **Geld** und
3. **Qualität** (vgl. Seghezzi 1993: 10ff).

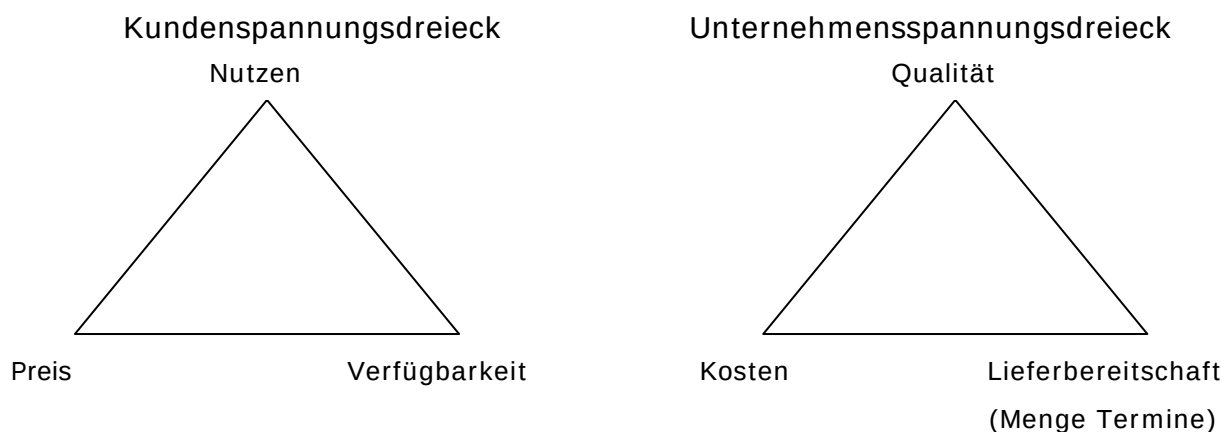


Abb. 3: Die Spannungsdreiecke

Quelle: Seghezzi 1993: 11 [eigene Bearbeitung]

Vor allem in der Automobilindustrie gilt das Qualitätsmanagement als wesentlichster Erfolgsfaktor. Es besitzt höchste Priorität noch vor Entwicklung, Produktentstehung und Innovation. Hohe Qualität ist die mit Abstand wichtigste Kompetenz, die ein Zulieferer als Kooperationspartner aufweisen sollte.

3.1 Gründe für eine Wirtschaftlichkeitsbetrachtung

Wurde früher in Europa, beginnend im industriellen Bereich (vor allem in der Automobilbranche) hohe Qualität teilweise als kostengünstige Maßnahme zur Konkurrenzabgrenzung und Wettbewerbsdifferenzierung angesehen, zeigen Erfahrungen aus der Praxis, dass eine umfassende Qualitätsausrichtung sehr wohl kostenintensive Auswirkungen nach sich zieht (vgl. Bartsch-Beuerlein 2000: 18).

Folgerichtig ist, dass auch Wirtschaftlichkeitskriterien bei der Gestaltung und Umsetzung einer Qualitätsorientierung im Unternehmen beachtet werden müssen. Deshalb ist es erforderlich, Qualitätsmaßnahmen – wie andere Unternehmensaktivitäten auch – nach Kosten-Nutzen-Gesichtspunkten zu beurteilen, um eine Ressourcenverschwendung zu vermeiden und Investitionschancen zu erkennen.

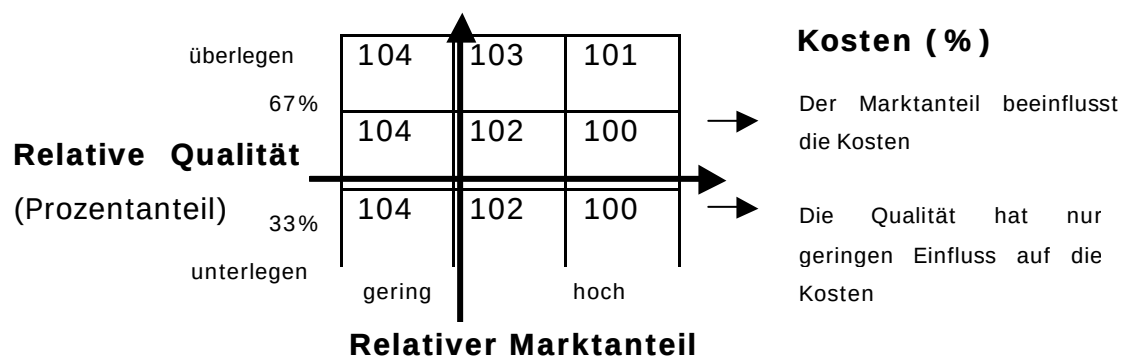
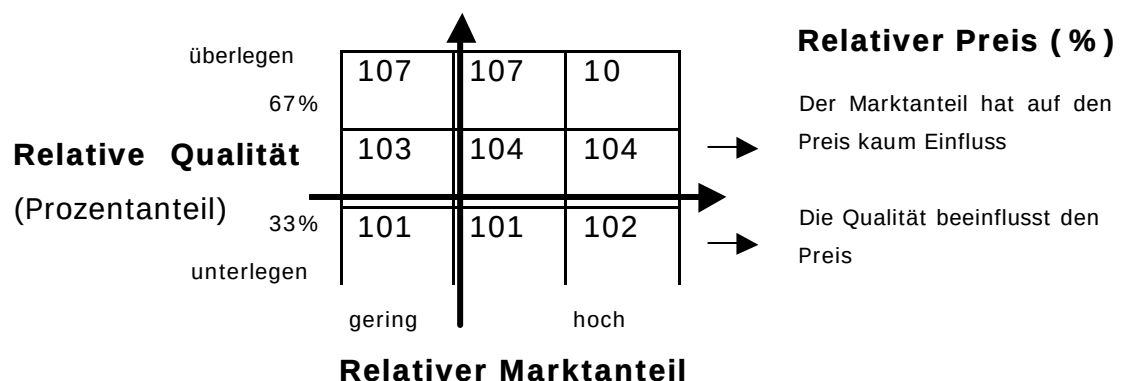
Die Erfolgskette des Qualitätsmanagements zeigt, dass Qualitätsmaßnahmen erlössteigernde und kostensenkende Auswirkungen auf den Unternehmenserfolg haben. Wenn durch das Qualitätsmanagement Kundenerwartungen besser erfüllt werden, und dadurch die Kundenzufriedenheit gesteigert wird, die sich wiederum in höherer Kundenbindung und positiver Mund zu Mund Kommunikation bemerkbar macht, können Erlössteigerungen erzielt werden. Weiters kann auch eine Effizienzsteigerung innerbetrieblicher Prozesse, sowie eine verstärkte Fehlervermeidung, was zu Kostensenkungen führen kann, realisiert werden. Die Erlössteigerungen und Kostensenkungen durch die

Qualitätsmaßnahmen sollten sich positiv auf den Unternehmenserfolg auswirken. (vgl. Bruhn 1999: 3ff).

Die PIMS-Studie analysierte verschiedene marketingbezogene Erfolgsfaktoren auf deren Auswirkung auf den Unternehmenserfolg:

- Unternehmen mit einer überdurchschnittlichen Leistungsqualität kann ein überdurchschnittliches Marktanteilwachstum nachgewiesen werden.
- Die Top-20 % Unternehmen hinsichtlich der Qualität erreichen um 8 % höhere Marktpreise als die Konkurrenz.
- Hohe Unternehmens-Qualität bewirkt wachsende Marktanteile und höhere Preisbereitschaft (vgl. Gale 1992: 34ff).

Qualität als Wettbewerbsfaktor PIMS-Studie



Qualität als Wettbewerbsfaktor:
PIMS-Studie III

ROS ROI

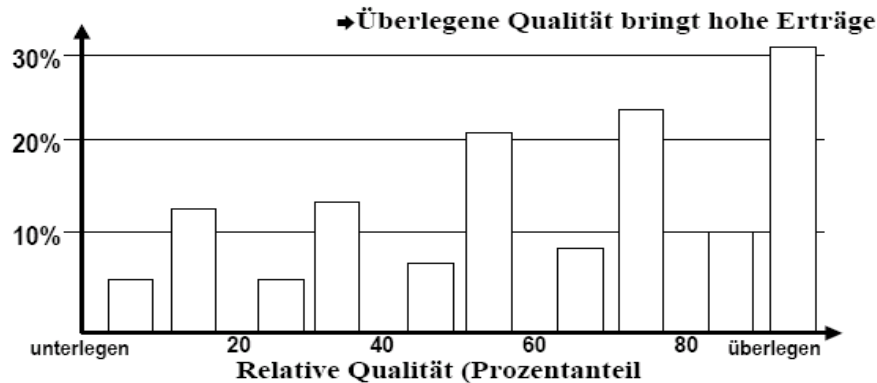


Abb. 4: Qualität als Wettbewerbsfaktor die PIMS Studie

Quelle: <http://www.qm-trends.de/fb1501.htm>, (Zugriff am 2005.02.26)

[eigene Bearbeitung]

Aber nicht immer müssen intensive Qualitätsbemühungen auch zu unternehmerischem Erfolg führen, weil nicht alle Faktoren der Erfolgskette linearer Natur sind. Nicht jede Qualitätsmaßnahme muss zu einer effizienteren Leistungserstellung oder Fehlervermeidung führen. Bruhn vermutet, „dass Qualitätsmaßnahmen - wie ein Grossteil unternehmerischer Aktivitäten – ab einem bestimmten Punkt (z.B. einem bestimmten Aktivitätsniveau) nur noch geringe Erlössteigerungs- bzw. Kostensenkungseffekte generieren, sodass die Maßnahmen aufgrund der durch sie hervorgerufenen Ausgaben ab diesem Punkt unprofitabel werden“ (vgl. Bruhn 1999: 10).

Qualitätsmaßnahmen sind deshalb als Investitionen zu sehen und betriebswirtschaftlich zu beurteilen um ihre Vorteilhaftigkeit richtig ein- und abschätzen zu können. Solche Investitionen, später als Qualitätsmanagementmaßnahmen zusammengefasst, haben einen entscheidenden Einfluss auf Technik und Wirtschaftlichkeit des gesamten Leistungserstellungsprozess, denn die Qualität steht im Spannungsfeld von Zeit, Kosten und Produktivität.

3.2 „Null-Fehler“ Forderung (vgl. Crosby 1980)

„Irren ist menschlich“ sagt ein altes Sprichwort – und lässt die Forderung nach einer „Null-Fehler“ Software illusorisch erscheinen. Dieses Sprichwort gibt oft Anlass zu Diskussionen, gerade unter Qualitätsfachleuten (und Programmieren). Die meisten Menschen meinen, Fehler seien unvermeidlich. Unter Software Programmierern gilt es als „bewiesen“, dass es „Bug-Freie“ Software nicht gibt. Die Akzeptanz der „Null-Fehler“-Forderung bezogen auf die eigene Arbeit ist zumeist nicht gegeben. Wir akzeptieren deshalb eigene Fehler nicht nur, sondern wir sehen sie sogar voraus. Gerade bei der Arbeit lassen wir uns gelegentlich einen Fehler „durchgehen“, einerseits im Vertrauen, ein anderer wird den Fehler entdecken und beseitigen, andererseits im Vertrauen darauf, dass die jeweiligen internen Auftraggeber und das Management Fehler sowieso einkalkuliert hat. **ABER** im privaten Leben akzeptieren wir Fehler anderer viel weniger. Wir setzen voraus und erwarten, dass:

- unser Konto auf der Bank richtig geführt wird.
- das Flugzeug auch bei seinem zehntausendsten Flug uns sicher zum Bestimmungsort transferiert.
- das Bauunternehmen die Arbeiten korrekt ausführt.

Aus diesen Gründen müssen wir aber auch die Forderung nach Qualität an unsere Arbeit(-sweise) akzeptieren.

3.3 Steigende Kundenerwartungen

Die Erwartungen von unseren Kunden steigen bezüglich der Qualität unserer Produkte. Der Kunde ist schon lange nicht mehr nur damit zufrieden, dass das Produkt funktionstüchtig ist, er stellt gerade an Softwareprodukte weitere Anforderungen, wie der Haltbarkeit, Zuverlässigkeit, Verfügbarkeit, Sicherheit (vgl. Balzert 1998: 56ff). Durch Qualität kann die Kundenzufriedenheit, das Kundenvertrauen gesteigert

werden. Z.B. ist es grundlegend, dass die eingesetzte Software zur Analyse von Qualitätsproblemen hundertprozentig richtige Ergebnisse liefert, auf deren Grundlage dann weitreichende Entscheidungen getroffen werden. Frühere Untersuchungen aus den USA haben ergeben, dass der Kunde einen deutlich höheren Preis zu zahlen bereit ist, wenn nur die Qualität stimmt (vgl. Haist 1989: 2). Dies wird durch neuere Untersuchungen inzwischen widerlegt. Der Kunde von heute erwartet sich Produkte mit „seinen“ Qualitätsanforderungen. Er ist zu meist nicht mehr bereit für Qualität mehr zu bezahlen, er wechselt einfach zu dem Lieferant, der ihm seine Qualität bietet. Dies ist auch eines der Ziele der Arbeit, ich möchte erreichen, dass sich meine Software Lieferanten wieder verstärkt der Fehlerprävention zuwenden und meine im Pflichtenheft spezifizierten Anforderungen auch umsetzen.

Dass die Null-Fehler Forderung gar nicht so illusorisch ist und erreicht werden kann zeigen genügend Beispiele auf. Das Null-Fehler Prinzip soll die Einstellung der Ausführenden verändern, sodass ein Fehler nicht mehr als notwendig hingenommen wird, sondern als ein Versagen im Arbeitsprozess aufgefasst und als Chance erkannt wird, die Ursache dieses Versagens zu beseitigen (vgl. Heist 1989: 46).

Typisch für diese Einstellung zum Fehler ist das fortwährende Fragen nach dem Warum, eine Vorgehensweise, um ganz konsequent nach den grundlegendsten Ursachen eines Problems zu suchen.

4 Vorstellung des Unternehmens

ThyssenKrupp-Presta

Die TKA-Presta Gruppe gehört zum Automotive Geschäftszweig des weltweit tätigen ThyssenKrupp-Konzerns. Unter der Führungsgesellschaft in Bochum werden bei ThyssenKrupp Automotive (TKA) 120 Fertigungsstandorte in 16 Ländern zu einem globalen Netzwerk zusammengefasst. Mehr als 38.500 Mitarbeiter tragen dazu bei, dass TKA Technologieführer bei Karosserieteilen, Motorkomponenten wie Kurbelwellen und gebauten Nockenwellen, Lenksäulen (elektrisch und mechanisch verstellbar) und Lenksystemen ist. Stark wachsend sind die zukunftssträchtigen Bereiche SystemEngineering und Mechatronik. Mit einem Umsatz von mehr als 6,3 Milliarden Euro im Geschäftsjahr 2003/2004 darf sich das Unternehmen zu den weltweit größten Automobilzulieferern zählen. Die TKA-Presta Gruppe ist Teil der Automotive-Sparte. Das Unternehmen wurde 1941 durch Oerlikon Bűhrle als Press- und Stanzwerk für Munitionskomponenten im Fürstentum Liechtenstein gegründet und ging 1991 aus der Übernahme von der ThyssenKrupp Automotive hervor. TKA-Presta ist in die drei Geschäftsbereiche Lenkungen, gebaute Nockenwellen und Massivumformung untergliedert. Im Bereich Lenkungen zählt TKA-Presta mit Produktionsstätten in Liechtenstein, Frankreich, Mexiko, USA, China, Deutschland und Brasilien zu den weltweit größten Herstellern von elektrisch oder mechanisch verstellbaren Lenksäulen. Zu den Kunden von TKA-Presta zählen namhafte Automobilhersteller wie beispielsweise Volkswagen, Audi, Ford, Volvo, BMW, DaimlerChrysler, Peugeot, Renault, Fiat sowie die Nischenhersteller Lamborghini, Ferrari und Maserati.

Das Unternehmen ist nach den Qualitätsnormen QS 9000, DIN ISO 9001, VDA 6.1 sowie der TS 16949 zertifiziert. TKA-Presta strebt in allen Bereichen die Null-Fehler-Qualität an (also auch bei der Software-Erstellung). Statistische Methoden bilden die Basis der Fehlerprävention

in der Entwicklung und werden in der Fertigung zur Prozessüberwachung genutzt.

Ich bin Softwareprojektleiter bei der TKA-Presta und zuständig für die Bereitstellung von notwendiger Software in der Produktion für BDE (Betriebsdatenerfassung) /MDE (Maschinendatenerfassung und QDE (Qualitätsdatenerfassung).

Darin ist beinhaltet,

- Software für die Steuerung der Maschinen,
- Erfassungsssoftware,
- Qualitäts-Auswertesoftware,
- Produktions-Reportingsoftware, usw.

Für die Realisation der Software ist ein Hauptlieferant mit 50MA zuständig und für kleinere Software-Lösungen/Programme wird regional ein Softwarepartner gesucht. Dabei liegt das Hauptaugenmerk bei der Vergabe von Softwareprojekten auf langfristigen Partnerschaften, weil die „Sichtweise“ der TKA-Presta erst eingeführt und dann gelebt werden muss.

5 Software

5.1 Definition Software

“Gesamtheit der Programme, die zum Betreiben und Steuern einer bestimmten (Klasse von) Hardware und der Erfüllung einer bestimmten (Klasse von) Informationsverarbeitungsaufgabe(n) benutzt wird” (Seibt 1972: 23).

“Immaterielles Produkt, das als geistiges Werk Informationen und Anweisungen enthält und keine Dienstleistung ist” (DIN ISO 2382-1: 15)

Zum Begriff Software werden neben dem eigentlichen Code und dem Programm desweiteren programmbegleitende Dokumentation (Anforderungsbeschreibungen, Spezifikationen, Testdaten, Testergebnisse, Abnahmeprotokolle, Benutzerhandbücher etc.) gezählt.

Im Vergleich mit anderen ingenieurtechnischen Prozessen liegt die Besonderheit der Software darin, dass es keinen Produktionsprozess im engeren Sinne gibt, in dem typgleiche Produkte „ready for use“ produziert werden.

Software ist zumeist ein Produkt-Prototyp, der evtl. durch Konfiguration und Installation zu einem gebrauchsfertigen Produkt entwickelt wird. Das Produkt selbst hat Werkzeugcharakter, nutzt sich aber nicht ab.

Folgende Kategorien werden unterschieden:

1) Standardsoftware

- hohe Stückzahl,
- große Einsatzbreite
- Parallelen zum Werkzeugmaschinenbau

2) Auftragssoftware (Sonderform: interne Auftragssoftware)

geringe Stückzahl,

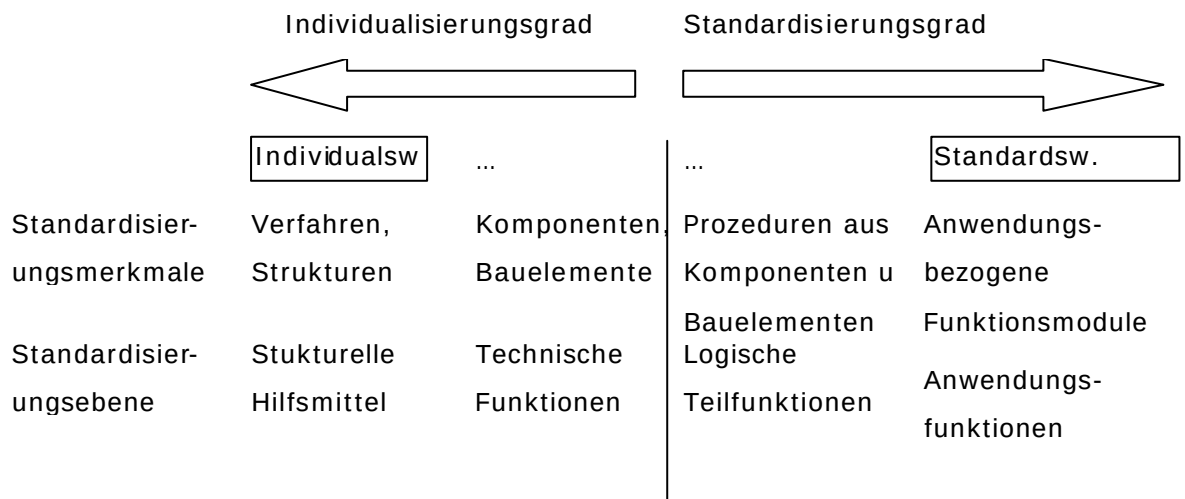
spezielle Einsatzbedingungen

Parallelen zum Anlagenbau

ing.-technischer Zugang handwerklicher Zugang

„Software ist als komplexes technisches Artefakt Materialisierung einer intellektuellen Leistung, welche ohne einen wohldefinierten Komplex von Kompetenzen weitgehend nutzlos ist „ (Uni Leipzig 2005: 15).

1. Kriterium der elementaren Wiederverwendung



2. Kriterium der Installationshäufigkeit

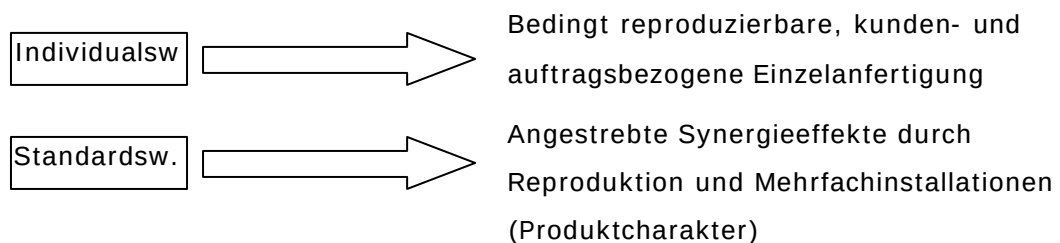


Abb. 5 Arten von Software (nach Standardisierungsgrad)

Quelle: Gerhardt, Tilmann (1992): Strategie und Struktur. 55. München

[eigene Bearbeitung]

Die eigene, mit Lieferanten entwickelte Software bei der TKA-Presta für die Produktion, sind, folgend der vorherigen Software-Klassifizierung, auftragsbezogene, individuelle Lösungen. Es ist die ständige Bestrebung, ausgehend von Prototypen, Software reproduziert einzusetzen, also zu standardisieren und Synergieeffekte zu nutzen, trotzdem sind viele Systeme individualisiert. Fokus bei jedem Einsatz von Software bei der TKA-Presta ist es die firmeninternen Anforderungen zu befriedigen, somit gibt es auch mit Herstellern von Standardsoftware, Projekte zur Adaptierung und Erweiterung ihrer Standardsoftware.

5.2 Softwarequalität

5.2.1 Versuch einer Definition von Qualität

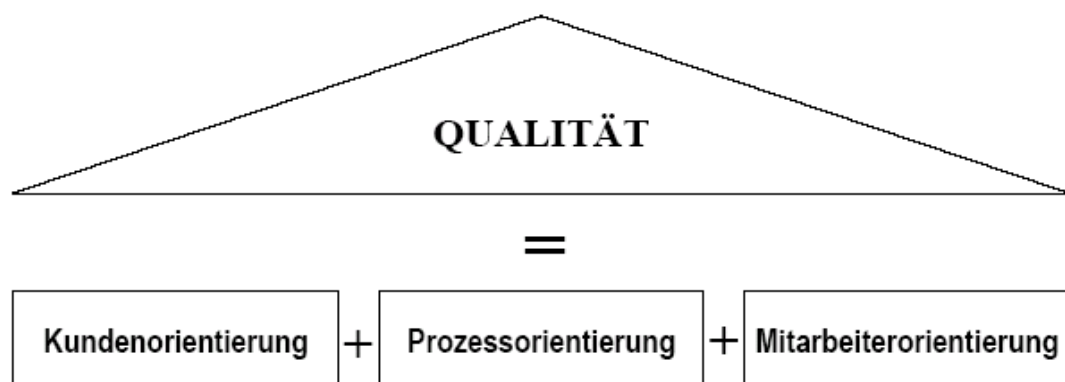


Abb. 6: Faktoren des Qualitätsbegriffs [eigene Abbildung]

„Qualität ist, wenn der Kunde zurückkommt, nicht das Produkt“
(Unbekannt)

Der Begriff Qualität ist abgeleitet aus dem lateinischen Wort *'qualitas'*: Beschaffenheit, Eigenschaft; Eignung eines Gutes für bestimmte Verwendungszwecke (Zweckeignung).

Es gibt viele Definitionen und Philosophien. Folgende Definitionen zeigen die unterschiedlichen Ansatzweisen:

„Die Gesamtheit von Merkmalen einer Einheit bezüglich ihrer Eignung, festgelegte und vorausgesetzte Erfordernisse zu erfüllen“(DIN ISO 8402 1995: 2.1).

„Qualität ist die Erfüllung von Anforderungen“(Crosby 1980: 20).

Die Norm DIN ISO 55350 definiert Qualität, „...als die Gesamtheit von Eigenschaften und Merkmalen eines Produktes oder einer Tätigkeit, die sich auf deren Eignung zur Erfüllung gegebener Erfordernisse bezieht.“

Bei der Qualität geht es nicht nur um die Sicht des Programmierers oder die aus einer abstrakten Sicht definierte Güte, sondern vielmehr und vor allem um die Eignung für den von einem Kunden vorgegebenen Zweck (Kundenorientierung).

„Unter dem Begriff "Qualität" werden alle Eigenschaften eines Gegenstandes oder einer Tätigkeit zusammengefasst, die in ihrer Summe das "Typische" und "Unverwechselbare" ausmachen“ (Aristoteles geb. 384 v.Chr.).

In der alltäglichen Praxis wird Qualität als (Ober-)Sammelbegriff "Zweckeignung" (fitness for use) für den Gebrauch verstanden. Einzelne Eigenschaften werden dabei durch gesetzliche Auflagen, durch Werbung oder im Verbraucherverhalten besonders hervorgehoben.

Haist schreibt, „Qualität ist vor allem eine Frage der inneren Einstellung eines jeden zu seiner Arbeit. Es kommt darauf an, Qualität nicht nur von anderen zu fordern, sondern sie selbst im eigenen Verantwortungsbereich zu leisten. Wir haben alle unsere „Kunden“ – externe wie interne -, die mit ihrer Arbeit auf unseren Arbeitsergebnissen aufbauen, deren

Anforderungen wir erfüllen müssen, und deren Zufriedenheit unser oberstes Ziel sein muss“ (Haist 1989: 25).

5.2.2 Definition Softwarequalität

„Gesamtheit von Eigenschaften eines Software-Produkts, die sich auf die Eignung zur Erfüllung gegebener Erfordernisse beziehen.“

(Schmitz 1983: 14; in Anlehnung an DIN 55 350).

Die IEEE-Norm für Softwarequalität (IEEE Std 729-1983) hebt die Erwartungen der Kunden hervor.

(1) The totality of features and characteristics of a software product that bear on its ability to satisfy given needs; for example, conform to specifications.

(2) The degree to which software possesses a desired combination of attributes.

(3) The degree to which a customer or user perceives that software meets his or her composite expectations.

(4) The composite characteristics of software that determine the degree to which the software in use will meet the expectations of the customer.

DeMarco schreibt, „*Softwarequalität zeichnet sich durch die Abwesenheit von Schäden aus*. Diese Definition schließt zwar einige der positiven Assoziationen aus, die oft durch das Wort Qualität ausgelöst werden – Eleganz, Grazie, Einfachheit – sie hat jedoch den Vorteil, messbar zu sein... Unterschiedliche Fehler haben unterschiedliche Kosten. Wenn Sie sorgfältig alle Fehler und deren Kosten betrachten, die für ihre Auffindung und Beseitigung anfallen, dann können Sie Qualität durch folgenden Ausdruck definieren:

$$\text{Qualität} = \frac{\text{? Fehlerdiagnose}}{\text{Programmumfang}}$$

„(DeMarco 1982: 337ff).

Softwarequalitätsmanagement ähnelt damit am meisten dem Entwicklungsprozess Qualitätsmanagement, mit der Besonderheit, dass das „Entwicklerteam“ (Entwurf, Installation, Wartung) im Vergleich zu klassischen ingenieur-technischen Prozessen deutlich weniger eng gekoppelt ist.

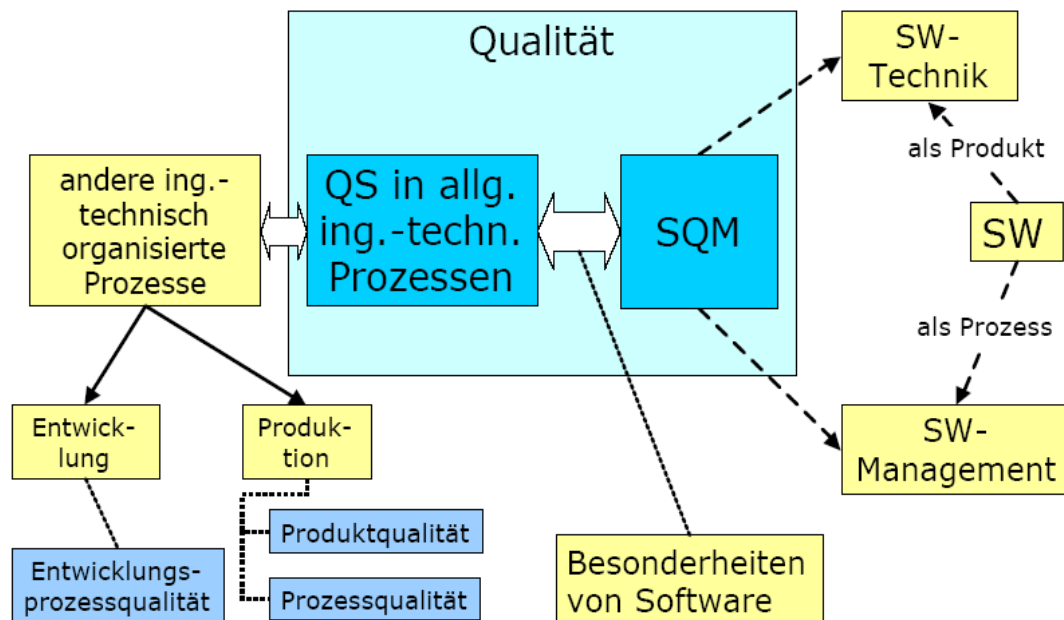


Abb. 7: Einordnung der Softwarequalität in die Qualitätswelt

Quelle: www.qm-infocenter.de, (Zugriff am 2004.12.20)

Softwarequalität steht im Spannungsfeld von

1. **Fehlerarmut**,
2. **zeitgerechter Erfüllung** der Kundenforderungen und der
3. **Funktionsvielfalt** der Software.

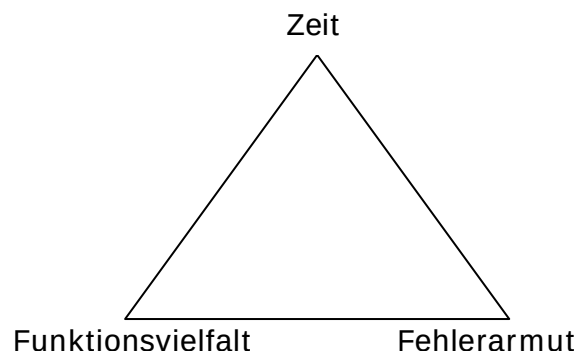


Abb. 8: Spannungsdreieck der Software

Quelle: Yourdon (1995): Good Enough Software. In: Guerrilla Programmer.

Z.B. umso schneller eine Software dem Kunden  bergeben wird, mit umso mehr Fehlern ist zu rechnen.

Fazit: Es ist sehr wichtig, dass man eine verst ndliche und einheitliche Definition des Qualit tsbegriffs hat, damit der Fokus aller T tigkeiten innerhalb der Prozesse auf diese ausgerichtet wird.

Ich w hle deshalb f r die Softwareentwicklung mit meinen Lieferanten in Anlehnung an die Norm DIN ISO 55350 folgende Definition:

Qualit t resultiert aus den Eigenschaften und Merkmalen des Softwareproduktes und der T tigkeiten der am Prozess beteiligten Personen, die alle auf die Erf llung von festgelegten und mitdenkend entstandenen Erfordernisse abzielen.

5.2.3 Bedeutung des Qualit tsbegriffs

Durch den h ufigen Gebrauch des Begriffs Qualit t wird er sehr unterschiedlich definiert und wahrgenommen. Einerseits spricht man von „**guter**“ oder „**schlechter**“ Qualit t, ein anderes mal ist der Begriff wertneutral, andererseits spricht man auch von einem Qualit tsprodukt und Qualit t hat damit eine positive Aussage. Der wertende Gebrauch des Begriffs Qualit t bestimmt heute seine Verwendung.

Es ist nicht sinnvoll, summarisch von guter oder schlechter Qualit t eines Softwareproduktes zu sprechen, bevor definiert ist, was das Produkt tun soll und welche Eigenschaften es aufweisen soll. Vom Verwendungszweck h ngt es ab, wie wichtig die einzelnen Eigenschaften sind und welche konkrete Auspr gung sie haben sollen (vgl. Fr hauf 2000: 18). Weiters ist zu unterscheiden, wie Qualit t objektiv bewertet werden kann, da jeder unterschiedliche Vorstellungen davon hat, weil z.B. ein Schreibfehler in der Benutzeroberfl che vom Anwender ganz anders wahrgenommen wird als vom Programmierer und ein Logikfehler vom

Anwender gar nicht bemerkt wird, aber vom Projektleiter als grundlegend einstuft wird.

5.3 Softwarequalitätsmodelle – Erfassung der Qualität

Um Qualität fass- und bewertbar zu machen wird Qualität im Allgemeinen durch ein Qualitätsmodell beschrieben. Ein Softwarequalitätsmodell operationalisiert den allgemeinen Qualitätsbegriff durch Ableitung von Unterbegriffen:

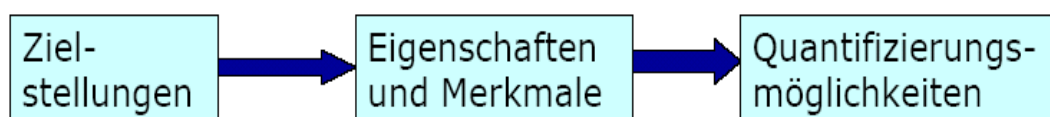
Qualitäts-Merkmalen und Qualitäts-Kriterien.

Qualitäts-Kriterien werden durch **Qualitäts-Indikatoren** erfasst.

Indikatoren bestehen aus einer quantitativen Skala und einer Methode, mit welcher der Wert bestimmt werden kann, den ein Indikator für ein bestimmtes Produkt oder eine bestimmte Tätigkeit aufweist.

Softwarequalität kann z.B. durch den FCM-Ansatz operationalisiert werden.

FCM-Ansatz (factor – criteria – metrics). Man unterscheidet den FCM als statischer Ansatz, der auf die Bewertung der Qualität eines vorhandenen Produkts ausgerichtet ist



und dem FCM – als dynamischer Ansatz, der auch das Qualitätsmanagement einschließt.

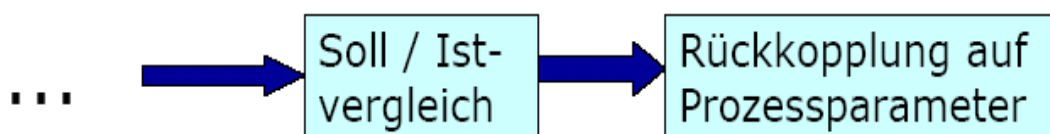


Abb. 9: FCM-Ansatz zur Operationalisierung der Qualität

Die bekanntesten Qualitätsmodelle sind jene von McCall (Wasserfallmodell), Boehm (Spiralmodell) und Willmer. Qualitätsmodelle fassen relevante Qualitätsmerkmale, sowie gegebenenfalls deren Teilmerkmale und Ausprägungen kontextbezogen zusammen. Der unklare Begriff Softwarequalität wird durch ihre strukturelle Zerlegungssystematik konkretisiert. Die am weitesten entwickelten Modelle, beispielsweise das von McCall, erlauben es, Qualitätsplanung und Qualitätsbewertungen durchzuführen.

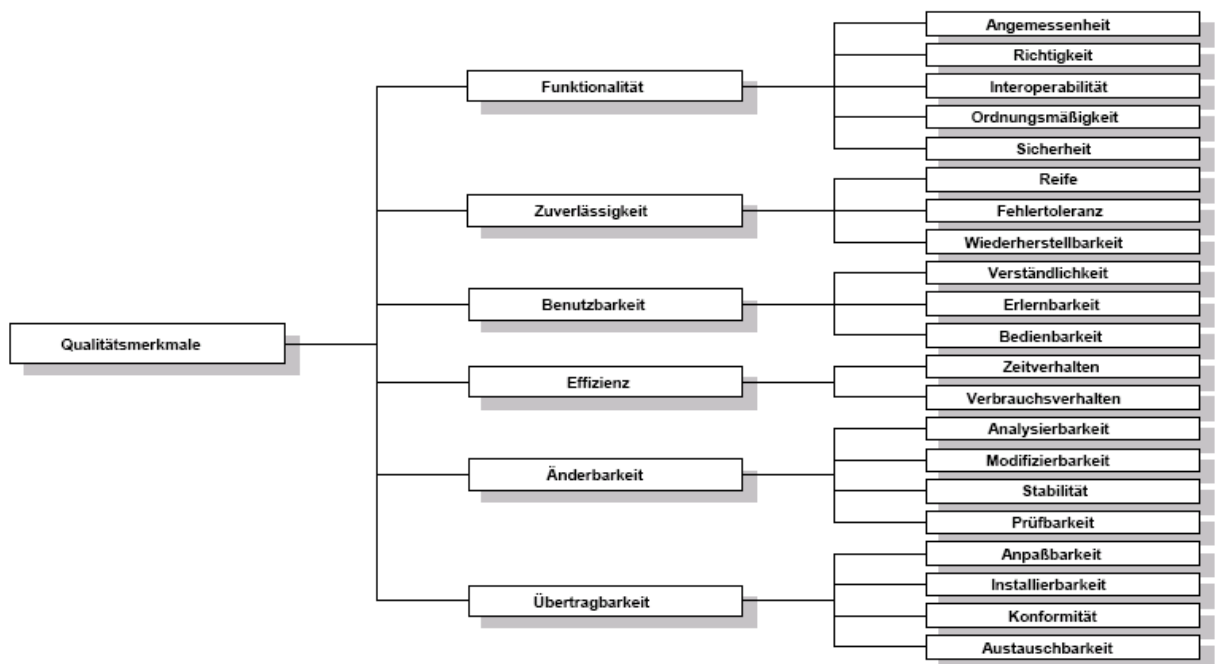


Abb. 10: Qualitätsmodell für Software

Quelle: DIN 66272 / ISO 9126

Ein Qualitätsmodell dient vorallem,

a) als Sammlung von

Entwicklungsleitlinien für den Auftragnehmer

Bewertungskriterien für den Auftraggeber

b) zur Beantwortung folgender Fragen:

Was versteht der Auftraggeber unter den einzelnen Qualitätsmerkmalen?

Wie gewichtet er die einzelnen Merkmale?

Wie will er die Ausprägung eines Qualitätsmerkmals überprüfen?

Lose Aufzählung weiterer Qualitätsmodelle ohne inhaltliche Detaillierung: Capability Maturity Model, Spiralmodell, ISO 900x Normen, Bootstrap Method, SPICE, TQM, u.a.m.

5.4 Operationalisierung der Qualität

Die Gesamtheit der Qualitätsmerkmale macht also die Qualität aus. Somit ist ein Qualitätsmerkmal ein fundamentaler Aspekt der Qualität und dient zur konkreten Festlegung und Beurteilung der Qualität. Ein Qualitätsmerkmal beschreibt eine Eigenschaft oder eine Klasse von Eigenschaften eines Gegenstandes (z.B. eines Softwareproduktes bzw. von Teilen eines Softwareproduktes). Beispiele für Qualitätsmerkmale sind Sicherheit, Zuverlässigkeit, Verfügbarkeit, Robustheit, Speicher- und Laufzeiteffizienz, Änderbarkeit, Portierbarkeit und Prüfbarkeit (vgl. Liggesmeyer 2002: 6).

Mittels des Qualitätsmaßes, der Qualitätskennzahl (metrics) kann ein Qualitätsmerkmal quantitativ erfasst und nominal bewertet werden, es kann somit die Ausprägung eines Qualitätsmerkmals bestimmt werden. Ein Qualitätsmaß ist eine objektive, messbare Größe, die in Bezug auf unterschiedliche Ausprägungen eines oder mehrerer Qualitätsmerkmale sensitiv reagiert. Z.B. ist das Qualitätsmaß für das Qualitätsmerkmal „Funktionalität“ die „Requirements Traceability“. Die Qualitätsmaßzahl gibt den erreichten Wert eines Qualitätsmaßes wieder. Unter dem Qualitätsziel wird dann folgerichtig der angestrebte Wert eines Qualitätsmaßes verstanden.

5.5 Software Fehler

Definition Fehler

„Jede Abweichung der tatsächlichen von der vorgesehenen bzw. explizit geforderten Ausprägung einer Eigenschaft eines Produktes der Softwareentwicklung“ (Liggesmeyer 2002: 31 ff).

„Ein Fehler oder Defekt (fault, defect) ist bei Software die statisch im Programmcode vorhandene Ursache eines Fehlverhaltens oder Ausfalls“ (Liggesmeyer 2002: 7).

Fehlerunterscheidung in der Software:

Fehler (error): ein falscher Wert wurde berechnet bzw. erarbeitet.

Defekt (defect): ein Wert wurde falsch berechnet.

Ausfall (failure): das Programm bricht ab bzw. führt zu einem unbeabsichtigten Zustand.

Störung (fault): das Programm setzt seine Bearbeitung nicht ordnungsgemäß fortsetzen.

Bei Fehlern ist zwischen einer **Abweichung**, der **Fehlerart** (Fehlerkategorie) und der **Fehlerbedeutung** zu unterscheiden.

Eine Abweichung von einer spezifizierten Eigenschaft bedeutet nicht immer einen Fehler (z.B. ein Anwender reklamiert, das Programm bringt nicht das gewünschte Resultat und er vermutet einen Fehler. Bei der Analyse stellt sich heraus, dass der Anwender das Programm falsch bedient hat → erkennen von Verbesserungspotenzial für Usability).

Fehler in der Software können in verschiedener Weise entstehen und haben unterschiedliche Auswirkungen.

Programmspezifikation:

Validationsfehler: der spezifizierte Algorithmus entspricht nicht den Anforderungen an das Programm.

Verifikationsfehler: die Anforderungen sind falsch umgesetzt worden.

Spezifikationsfehler: Das Problem wurde falsch/unvollständig in der Spezifikation abgebildet.

Programmdesign:

Verifikationsfehler: der spezifizierte Algorithmus ist falsch auf das jeweilige Programmiersprachparadigma abgebildet worden,

Entwurfsfehler: die Komponentenstruktur bzw. der Programmentwurf selbst sind unkorrekt abgeleitet worden.

Programmkodierungsfehler:

Editierungsfehler: die Programmiersprachelemente bzw. -konstrukte sind falsch ausgewählt worden.

Generierungsfehler: die (Makro-) Parameter für eine Quellcodegenerierung sind falsch oder der Generierungsalgorithmus selbst ist fehlerhaft.

Anpassungsfehler: die bereits vordefinierten Code-Artefakte sind falsch angepasst worden.

Übernahmefehler: von den bereits vorhandenen Code-Artefakten, wie z. B. Klassen, sind die falschen übernommen worden;

Programmarbeitungsfehler.

usw.

Einteilung der Fehler durch ihre Schwere/Bedeutung/Auswirkung:

Die Verwendbarkeit einer Software wird durch einen Fehler mehr oder minder stark beeinflusst. Deshalb werden die jeweils möglichen Fehler in verschiedene Klassen eingeteilt und gewichtet (Einteilung nach DIN 40 080):

Kritischer Fehler: Fehler, die für Benutzer und/oder Instandhalter u. U. lebens- bzw. gesundheitsgefährdende Situationen schaffen können oder die die Funktion wichtiger Anlagen, Maschinen, Vorgänge etc. stark stören oder gar verhindern. Z.B. wenn die Softwareverriegelung für Not-Aus bei Maschinen versagt (deshalb die Forderung nach mechanischen Absicherungen)

Hauptfehler: Nichtkritischer Fehler, der voraussichtlich zu einem Ausfall führt oder die Brauchbarkeit für den vorgesehenen Verwendungszweck wesentlich herabsetzt. Je nach Erfordernis können Hauptfehler in zwei Kategorien eingeteilt werden:

Hauptfehler A: Vollständige Beeinträchtigung der Brauchbarkeit, Ausfall oder Verlust.

Hauptfehler B: Teilweise Beeinträchtigung der Gebrauchstauglichkeit.

Nebenfehler: Fehler, der die Gebrauchstauglichkeit nicht wesentlich herabsetzt bzw. ein Abweichen von den geltenden Normen, das den Gebrauch oder den Betrieb der Einheit nur geringfügig beeinflusst.

Diese Einteilung ist einfach und wird praktisch immer in der Produktion angewandt. Nachteilig ist, dass die wirtschaftlichen Folgen des Fehlers nicht berücksichtigt werden. Durch Bildung von Fehlergewichtsklassen kann das präzisiert werden, dazu misst man den Aufwand um den Fehler zu beheben. Ein Maß für diesen Aufwand ist die Fehlergewichtung mit Wertziffern, deren Summe dann die Gewichtung des Fehlers in einer Qualitätskennzahl ergibt.

Aufwände, Schwierigkeiten bei der Fehlerfindung und Kosten im Zusammenhang mit Softwarefehlern aus praktischen Erfahrungen:

- Fehler werden häufig nicht unmittelbar (zum Zeitpunkt und am Ort ihrer Entstehung), sondern häufig erst später, anhand ihrer Symptome erkannt (z.B. falsche Ergebnisse / Ausgaben an den Benutzer).

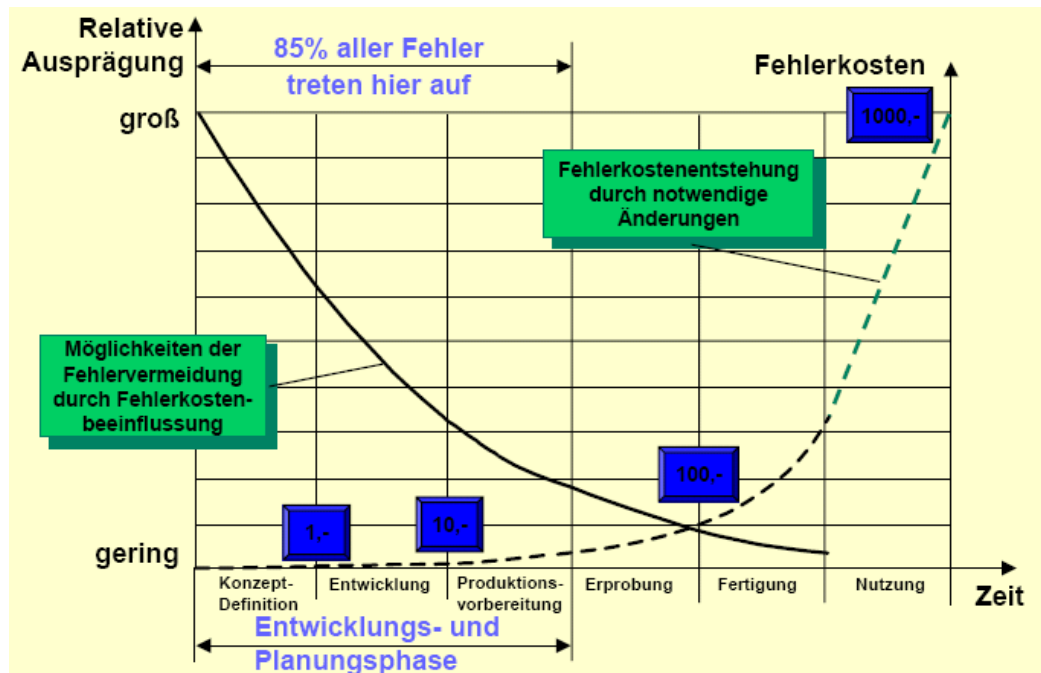


Abb. 11: Fehlerentstehungskosten

Quelle: Wallmüller, E. (1990): Software-Qualitätssicherung in der Praxis. München: Hanser.

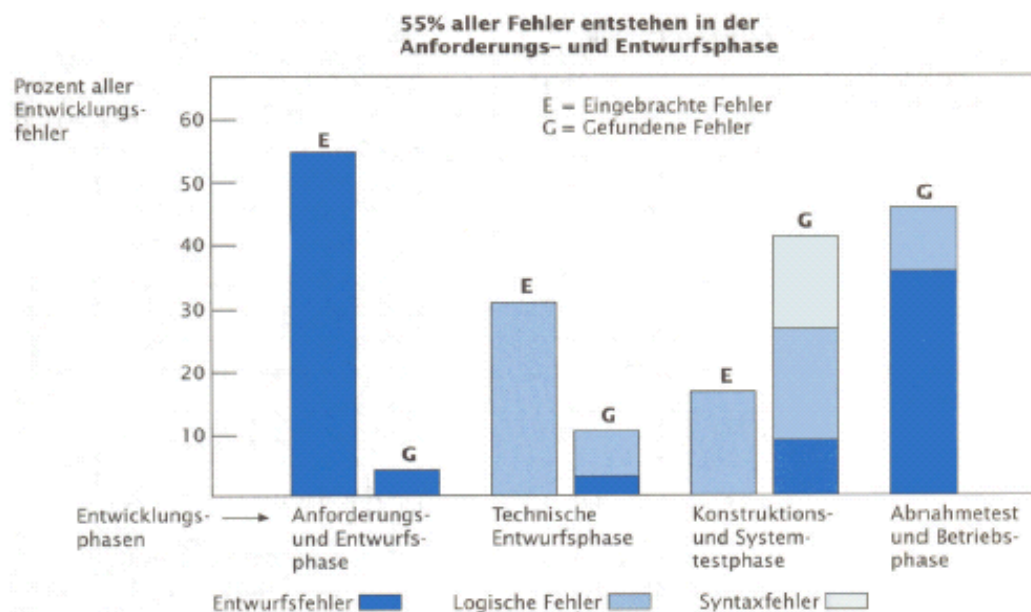


Abb. 12: Fehlerbeseitigungskosten

Quelle: Alberts, D.S. (1976): "The Economics of Software Quality Assurance," National Computer Conference. 433-441.

- Das Finden eines Fehlers macht ca. 80 %, das Beheben ca. 20 % des Gesamtaufwands der Fehlerentdeckung und -behebung aus.
(vgl. Wallmüller 1990: 167).
- Fehler lassen sich nicht vollständig ausschließen. Man muss aber an deren Minimierung arbeiten.
- Wie wichtig die rechtzeitige Erkennung ist, zeigt die Zehnerregel. Sie besagt, dass ein Fehler mit jeder Prozessphase, in der er später entdeckt wird, hinsichtlich der Kostenauswirkungen (Nacharbeit) um den Faktor 10 zunimmt. Damit soll verdeutlicht werden, dass frühe Phasen in der Softwareentwicklung besonders wichtig sind und unentdeckte Fehler in diesen Phasen enorme Kosten verursachen umso später sie entdeckt werden. Z.B. können aus Gewährleistungsansprüchen des Kunden rasch Millionen Euro teure Klagen resultieren.

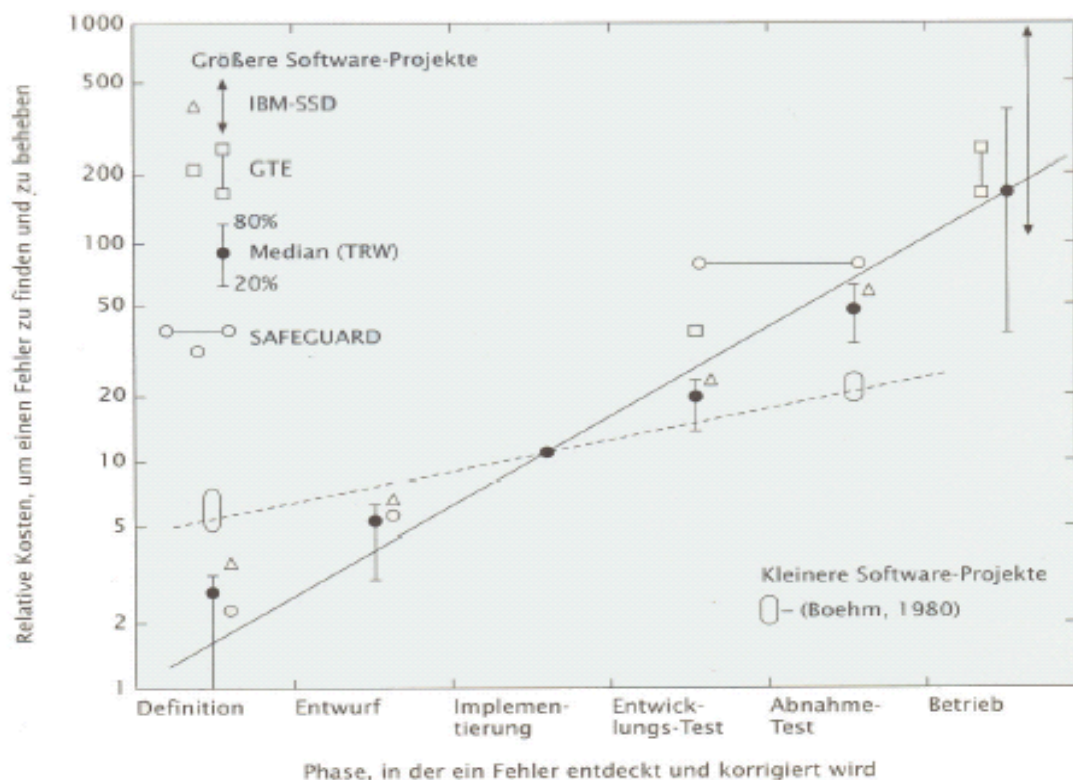


Abb. 13: Kosten einer verzögerten Fehlerentdeckung

Quelle: Boehm, B.W. (1981): Software Engineering Economics. Englewood Cliffs. Prentice-Hall.

6 Auswirkungen der Softwarequalität

6.1 Gründe für Mängel an Softwarequalität?

Nach Mizumo kommt es beim Entstehen von Fehlern zu einem Summationseffekt (Mizumo-Studie 1983: 34ff). Ein Projekt beginnt normalerweise mit der Erfassung, Analyse und Definition der Anforderungen. Diese Anforderungen werden in Form einer Anforderungsspezifikation festgehalten. Die Erfahrung zeigt, dass ein Teil dieser Spezifikation mit Fehlern behaftet ist. In der nächsten Phase wird der Entwurf erstellt. Ergebnis ist eine Systementwurfsdokumentation. Wiederum ist ein Teil des Entwurfs mit Fehlern behaftet, die im Entwurf entstanden sind, und ein dritter Teil des Entwurfs basiert auf der fehlerhaften Spezifikation. Der nächste Schritt ist die Programmierung. Ein Teil ist mit Fehlern behaftet, ein weiterer Teil der Programme basiert auf fehlerhaftem Entwurf, fehlerhafter Spezifikation und fehlerhaften Anforderungen. In der nachfolgenden Integrations- und Testphase ergibt sich folgende Situation: ein Teil der Programme funktioniert korrekt. Ein anderer Teil der Programme enthält korrigierbare Fehler, die auch korrigiert werden. Ein weiterer Teil enthält nicht korrigierbare Fehler und wiederum ein anderer enthält versteckte Fehler.

Der Summationseffekt der aufgetretenen Fehler bewirkte die mangelnde Qualität des Software-Produkts. Die gesamte Tragweite jedes einzelnen Fehlers kommt dadurch erst gegen Ende des Projekts zum Vorschein.

Dieser Effekt tritt auch bei anderen technischen Produktentwicklungen auf. Dort hingegen gibt es seit langem in der Entwicklung integrierte Zwischenprüfungen, so genannte Reviews, Inspektionen.

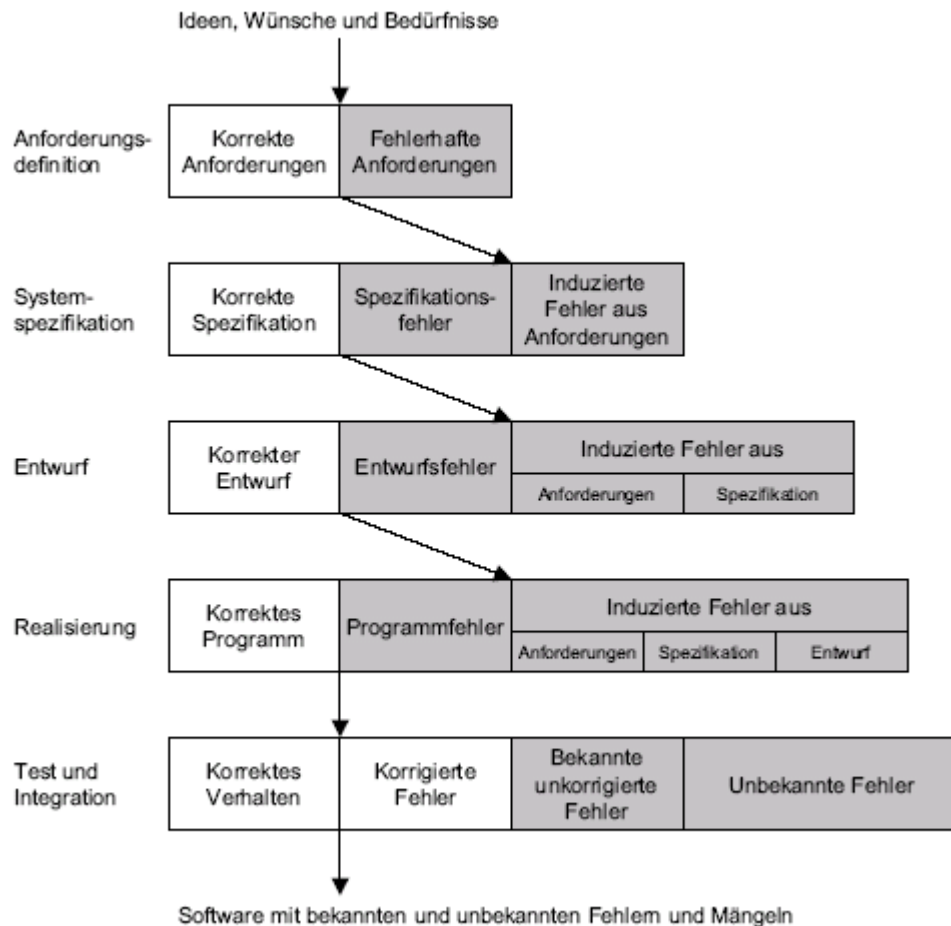


Abb. 14: Fehlersummutationsauswirkung

Quelle: in Anlehnung an Mizumo, Y. (1983): Software Quality Improvement. In IEEE. Computer. March 1983.

Diese Summation der Fehler im Entwicklungsprozess ist jedoch nicht der einzige Grund für mangelnde Softwarequalität.

1) Scheinbar geringer Änderungsaufwand, damit ist die Änderungsversuchung oder der Änderungsdruck groß.

Das normale Dilemma in der Software Entwicklung ist die Intransparenz für Aussenstehende, einerseits können Anpassungen/Adaptierungen oder Integration von neuen Kundenanforderungen im Verhältnis zum bisherigen Entwicklungsfortschritt vom Blickwinkel des Kunden/Projektleiters lange oder kurz dauern. Ausgehend von der „Änderungsfreundlichkeit“ der Software kann dies zu erheblichen

Argumentations-Problemen des Entwicklers führen. Oftmals werden Änderungen im Zuge der Integration oder Einführung der Software beim Kunden, ohne die sonst übliche Absprache mit dem Projektleiter und ohne das sonst übliche Durchlaufen der Vorgehensmodelle im Software Projekt durchgeführt.

In einem durchschnittlichen System bringen Verbesserungen jedoch eine ganze Menge Schaden mit sich (De Marco 1982: S 336).

2) Fehlen des visuellen Feedbacks

In "Metaphors we live by" (Lackoff 1980: 15ff) wird argumentiert, dass Sprache -und damit Denken- auf Erfahrungen, die jeder einzelne in der materiellen Welt macht, aufbaut. Der überwiegende Teil dieser Erfahrungen ist visueller Natur. Gerade in der Software Entwicklung, wo zu Beginn des Prozesses nur Ideen und immaterielle Abläufe existieren ist es besonders wichtig, Information visuell transparent zu machen, dabei wird einerseits der Kodier- und Dekodiervorgang, der bei verbaler oder schriftlicher Kommunikation notwendig ist, umgangen.

Vorteile visueller Darstellung:

- Möglichkeit der Darstellung mehrdimensionaler Information.
- Wesentlich höhere Informationsdichte möglich.
- In guter Auflösung lassen sich graphisch leicht mehrere tausend diskrete Werte in einem Schaubild darstellen.
- Graphische Information hat den Vorteil von subjektiver Neutralität, die bei Sprache und Schrift nicht gegeben ist.
- Visuelles Erinnerungsvermögen ist in der Regel deutlich besser, als das sprachliche.
- Kombinieren quantitativer und qualitativer Aussagen sehr gut möglich.

3) Ziel- und Definitionsunklarheit

Ein Ziel lässt sich nachweisbar nur dann erreichen, wenn es präzise definiert ist. Beste Qualität ist keine Definition. Softwarequalität ist

spezifisch. Nicht alle Eigenschaften sind für den Benutzer gleichermassen wichtig. Für eine bestimmte Software sind einige Eigenschaften besonders wichtig, andere wieder weniger oder sogar total irrelevant für das Produkt. Manche Eigenschaften stehen in negativer Wechselwirkung zueinander. Das Ziel ist die Entwicklung der Software mit der richtigen (vom Kunden geforderten) Qualität, dazu müssen die Qualitätsziele bestimmt werden. Mit dem Einsatz von konstruktiv vorausschauenden und von analytisch prüfenden Techniken müssen dementsprechend Maßnahmen eingeleitet werden. Dabei ist vor allem auf die Wirtschaftlichkeit zu achten. Deshalb gibt es auch verschiedenste Modelle und keine universell geeignete Lösung für dieses Problem.

Jede industrielle Softwareentwicklung muss als Motivation und Ziel haben, ein möglichst fehlerfreies Software-Produkt zu erstellen (vgl. Uni Leipzig 2002: 1ff).

6.2 Beispiele für mangelnde SW-Qualität

Einige spektakuläre Fälle

Venussonde fliegt 1979 am Ziel vorbei.

Ursache: Verwechslung eines Punktes mit einem Komma.

Jagdbomber drehen sich bei Testflügen 1983 über dem Äquator auf den 'Rücken'.

Ursache: Vorzeichenfehler.

Salomon Brothers, Inc. erleidet 1992 einen erheblichen Verlust.

Ursache: Angestellter erteilt einen Verkaufsauftrag über 11 Mio Aktien, statt über Aktien im Wert von USD 11 Mio. Ein zweiter Angestellter versäumt, den Auftrag zu prüfen; das Informationssystem macht zu spät auf einen möglichen Fehler aufmerksam.

Beim Start der Ariane-5 am 4. Juni 1996 explodiert die Rakete. Kosten des Ariane-5-Programms bis 1996: ca 8Mrd USD. Wert des zerstörten Satelliten: ca. 500Mio USD.

Ursache:

Während des Flugs läuft ein unnötiges Kalibrierungsprogramm für die Trägheitssensoren. Die gemessenen Werte der Ariane-5 überschreiten die in der Ariane-4-Software vorgesehenen Bereiche. Die (Ada-) Exception wird durch Anhalten des Steuerungscomputers behandelt, um auf ein zweites redundantes System umzuschalten. Im zweiten System tritt der gleiche Software-Fehler auf und wird identisch behandelt.



Abb. 15: Absturz der Ariane 5 am 4. Juni 1996

Quelle: http://de.wikipedia.org/wiki/Ariane_5 (Zugriff am 2004.12.20)

CHAOS Report

Im jährlich seit 1994 erscheinenden Chaos-Report wird der Erfolg von IT-Projekten aufgelistet.

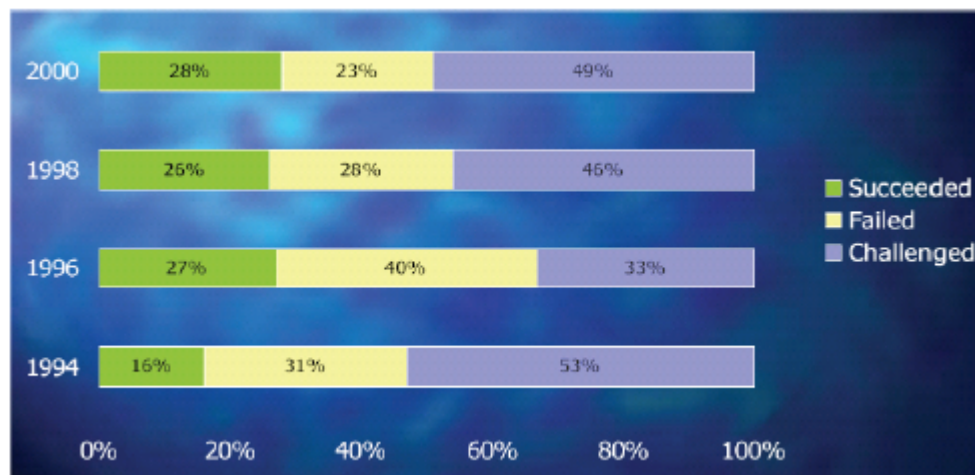


Abb. 16: Erfolgsstatistik von IT-Projekten

Quelle: CHAOS Report, Standisch Group International, Inc.

Es wurden seither weit über 100.000 Projekte in den USA untersucht
Herausgeber: Standish Group International, Inc. Der CHAOS Report ordnet IT-Projekte in drei Kategorien ein:

Successful: Projekt wurde innerhalb der vorgegebenen Zeit und Budget abgeschlossen. Projektergebnis ist im Einsatz und erfüllt alle Anforderungen.

Challenged: Projekt ist abgeschlossen. Projektergebnis ist im Einsatz. Zeit, Budget oder Leistung sind aber nicht im vorgegebenen Umfang.

Failed: Das Projekt wurde vorzeitig abgebrochen oder das Projektergebnis wurde nie eingesetzt.

Beispiele für schlechte Softwarequalität bei der TKA-Presta (Rüb 2005: 12ff):

- 4 Tagesproduktionen mit falschem Typ produziert. Durch einen SPS internen Fehler (falsche Parameterquittierung) wurden trotz korrekter Einstellung durch die jeweiligen Maschineneinrichter im Drei-Schichtbetrieb ca. 64.000 Lenkwellen zu Ausschuss (à 4CHF).
Ursache:
SPS übernimmt die Parameter unvollständig vom Leitsystem und meldet keinen Fehler, d.h. Teile der Anlage produzieren mit Einstellungen von anderen Typen.
- Aktuelle Q-Daten auf einem hoch gesicherten Datenserver wurden gelöscht. Durch einen immer noch nicht identifizierten Softwarefehler wurden ca. 80.000 Qualitätsdaten-Files (die zum Nachweis der Qualität das Herzstück der Produktion bilden) über Nacht gelöscht. Betroffen waren mehr als 10Mio Teile. Beim Rückspielen von Sicherungen kam es erneut zu Problemen, so dass eine manuelle Überprüfung notwendig war (Kostenpunkt: 1MA ca. 1 Woche + Ursachenfindung beim Softwarehersteller von 2MA ca. 2 Wochen + Ausfall des Systems für die Benutzer von 4 Tagen).
- Einsatz von „nichtanwenderfreundlicher“ Software verursacht kaum quantifizierbare Kosten durch ca. 20 % zeitlichem Mehraufwand in der Bedienung.

7 Anforderungsmerkmale von Softwarequalität (Ausprägung)

Die Norm DIN ISO 9126 definiert 6 Qualitätsmerkmale für die Qualität von Software-Produkten:

1. **Funktionalität:** Vorhandensein von Funktionen mit festgelegten Eigenschaften, welche die definierten Anforderungen erfüllen. Richtigkeit, Angemessenheit, Interoperabilität, Ordnungsmäßigkeit, Sicherheit
2. **Zuverlässigkeit:** Fähigkeit der Software, ihr Leistungsniveau unter festgelegten Bedingungen in einem festgelegtem Zeitraum zu entfalten. Fehlertoleranz, Wiederherstellbarkeit
3. **Benutzbarkeit:** Aufwand, der zur Einarbeitung und zur Benutzung erforderlich ist, und individuelle Beurteilung der Benutzbarkeit durch eine festgelegte oder vorausgesetzte Gruppe. Verständlichkeit, Erlernbarkeit, Bedienbarkeit
4. **Effizienz:** Verhältnis zwischen dem Leistungsniveau der Software und dem Umfang der eingesetzten Betriebsmittel unter festgelegten Bedingungen. Zeitverhalten, Verbrauchsverhalten
5. **Änderbarkeit:** Aufwand, der zur Durchführung vorgegebener Änderungen notwendig ist. Änderungen können Korrekturen, Verbesserungen oder Anpassungen an Änderungen der Umgebung, der Anforderungen und der funktionalen Spezifikationen einschließen. Analysierbarkeit, Modifizierbarkeit, Stabilität, Prüfbarkeit
6. **Übertragbarkeit:** Eignung der Software, von einer Umgebung in eine andere übertragen zu werden. Umgebung kann organisatorische Umgebung, Hardware- oder Softwareumgebung einschließen. Anpassbarkeit, Installierbarkeit, Konformität, Austauschbarkeit

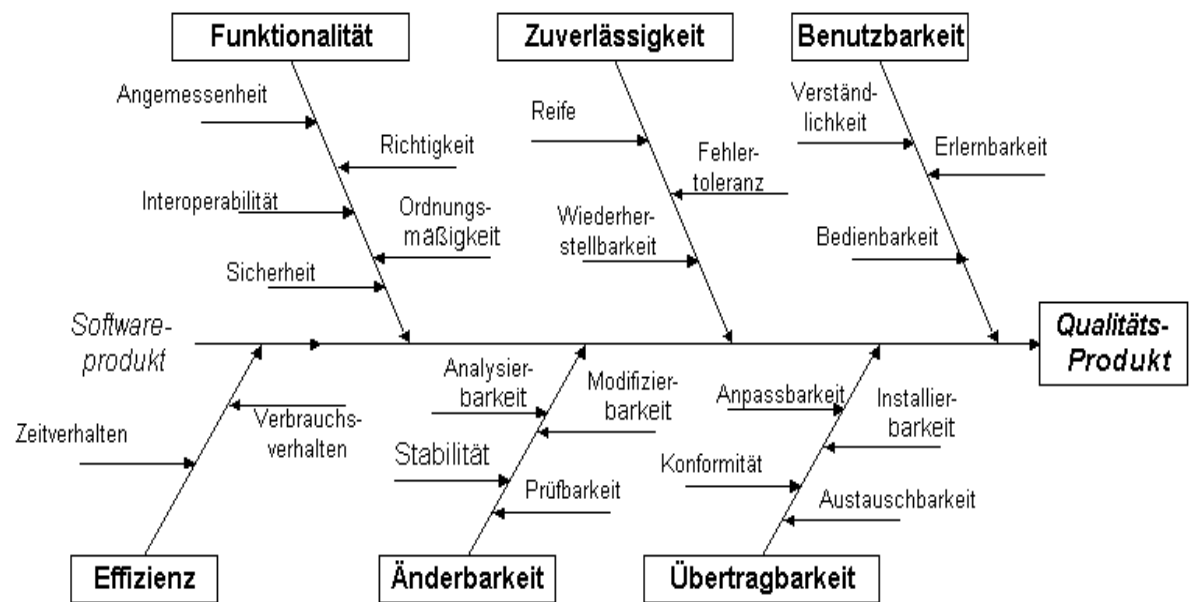


Abb. 17: Anforderungsmerkmale von Softwarequalität

Quelle: DIN ISO 9126

Will man nicht nur fertige Software-Produkte bewerten, sondern Qualität auch konstruktiv realisieren, muss ein Bewertungsansatz für den Entwicklungs- und Pflegeprozess aufgestellt werden. Das bedeutet, dass Merkmale und Bewertungsmaßstäbe auch für Zwischen-/Endprodukte und für deren Entwicklungsaktivitäten in allen Phasen des Software-Erstellungs-Prozesses benötigt werden. Wie schon erläutert beinhaltet eine Qualitätsbewertung immer einen Vergleich zwischen den aus den gegebenen Erfordernissen abgeleiteten Qualitätsvorgaben (Soll-Werte) und den tatsächlich erreichten Ausprägungen der Merkmale (Ist-Werte).

Zur Problematik der Bestimmung der Qualität eines Softwareproduktes stellt **Boehm** folgende drei Fragen (vgl. Boehm 1978: 34ff):

1. Definitionsproblem von Softwarequalität

Ist es überhaupt möglich, Definitionen der Eigenschaften und Merkmale eines Software- Produkts aufzustellen, die messbar sind und sich nicht überschneiden?

2. Qualitätsprüfungsproblem

Wie gut kann man die Qualität eines Software-Produkts bestimmen bzw. die Eigenschaften und Merkmale messen?

3. Problem der Qualitätslenkung

Wie kann man Informationen über die Qualität des Produkts zur Verbesserung des Produkts im Lifecycle verwenden?

Unter Punkt 1 wird zunächst einmal das Problem der Eindeutigkeit und Relevanz von Prozess- und Produktmerkmalen angesprochen., die wie oben angeführt durch die Norm ISO 0126 definiert werden.

In Punkt 2 wird die Vollständigkeit und Prägnanz der Qualitätsmerkmale hinterfragt, also wie einzelne Anforderungen mit einem vorgegebenen Instrumentarium formuliert werden können und in Punkt 3 wird Qualität als eine Steuerungsgröße für den Prozess gesehen. Entscheidend ist die Rückkopplung der Ergebnisse von Qualitätsprüfungen mit der Prozesssteuerung.

8 Vorgehen bei der Softwareentwicklung

Softwarequalität wird vor allem durch den Prozess bestimmt, in dem sie entwickelt wird und durch die Merkmale, die sie besitzt. Der Entwicklungsprozess muss zeitlich durch Phasen strukturiert werden, weil sich Software wie andere technische Produkte auch, nicht in einem Zuge entwickeln lassen kann. Zwischen der ersten Idee und der endgültigen Ablösung der Software durchläuft der Prozess verschiedene Phasen, Meilensteine bilden den Abschluss der vorigen Phase und sind die Entscheidungspunkte um in die folgende Phase einzutreten. Die Qualität muss in jeder dieser Phasen entstehen. Die Gesamtqualität eines Produktes setzt sich nun stufenweise aus der Qualität der Phasenergebnisse und der Erfüllung der Prozessanforderungen zusammen. Voraussetzungen für gute Prozess und Phasenqualität sind das Ausmaß des systematischen, methodischen Entwickelns (beispielsweise das Einhalten von Entwicklungsstandards), die Sorgfalt des Projektmanagements, die Qualifikation und der Ausbildungsstand der Mitarbeiter sowie die Beschaffenheit der eingesetzten Hilfsmittel.

Beispiel eines Vorgehensmodells:

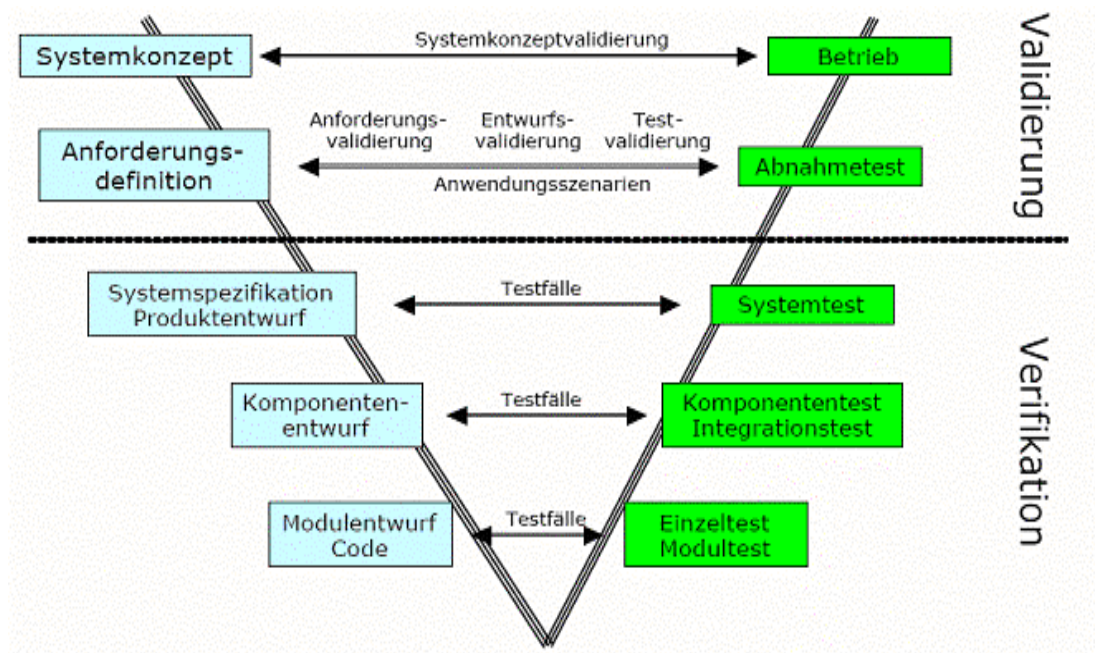


Abb. 18: V-Modell

Quelle: <http://www.kbst.bund.de/-/279/V-Modell.htm> (Zugriff am 2005.04.20)

Das V-Modell von KBSt (Koordinierungs und Beratungsstelle für Informationstechnik in der Bundesverwaltung) (vgl. KBSt 2005) erweitert das bekannte Wasserfallmodell um ein integriertes Qualitätssicherungssystem zur Durchführung des Qualitätsmanagements. In einem Phasenmodell sind neben der Prozessorientierung die Meilensteine das wesentlichste Merkmal. Um von einer Phase in eine nachgelagerte Phase zu gelangen, müssen Meilensteine erreicht werden, die die vorige Phase damit abschliessen. Das V-Modell legt genau fest wie Teilprodukte verifiziert und validiert werden müssen.

Unter **Verifikation** versteht man die Überprüfung auf Übereinstimmung zwischen Spezifikation und dem Produkt (Wird ein korrektes Produkt entwickelt?).

Unter **Validierung** wird die Überprüfung der Eignung eines Produkts hinsichtlich seines Einsatzzwecks (Wird das richtige Produkt entwickelt?) verstanden.

Das V-Modell ist ein Vorgehensmodell und gliedert sich in 4 Submodelle:

- System-Entwicklung (SE)
- Qualitätssicherung (QS)
- Konfigurationsmanagement (KM) und
- Projektmanagement (PM)

Vorteile von Vorgehensmodellen:

- Qualitätsverbesserung durch Standardisierung.
- Prüfungen und Wartung werden einfacher.
- Verbesserte vorausschauende Planung.
- Erzwingung der korrekten Abfolge der notwendigen Entwicklungsschritte.
- Integration und Beschleunigung von Aktivitäten, durch Zerlegung in Teilprozesse.
- Einsatz der Projektmitglieder nach persönlichen Eigenschaften.
- Vermeidung von Redundanz und Schleifen.

9 Maßnahmen um „gute“ Softwarequalität zu generieren

Wie jeder andere industrielle Produktions- und Entwicklungsprozess unterliegt die Softwareentwicklung dem Zwang zu hoher Produktivität. Dies wiederum bedingt eine Berücksichtigung der Ziele

1. Termintreue,
2. minimale Kosten und
3. zufriedenstellende Produktqualität.

Um diese Ziele zu erreichen, müssen gewisse Vorkehrungen getroffen werden. Folgende Maßnahmen sind zur Beeinflussung der Qualität gegenwärtig bekannt (vgl. ff TU Wien 2004):

- **konkrete und operationalisierbare Qualitätsmerkmale**

Beim Projektstart sind die wesentlichen Qualitätsmerkmale aus der Sicht des Auftraggebers und Benutzers zu definieren. Qualityfunctiondeployment kann zur Bestimmung und Findung der Merkmale eingesetzt werden. Qualityfunctiondeployment (QFD) ist eine in den 60er Jahren des vorigen Jahrhunderts entwickelte Methode, um die Kundenanforderungen zu gewichten und mittels des House of Quality Produktmerkmale zu definieren und davon technische Funktionen abzuleiten. Die Qualitätsmerkmale müssen möglichst konkret und quantifizierbar sind.

Es ist ebenso von Bedeutung, dass es gelingt, für einzelne Phasen konkrete Qualitätsmerkmale zu finden. Diese Merkmale können sowohl prozess- als auch produktorientiert sein. Es ist sinnvoll, für die Fertigstellung eines Pflichtenheftes bestimmte Qualitätsmerkmale von vornherein festzulegen und vorzugeben. Aus der Sicht der Prozessqualität ist es wichtig, für das Testen ausreichende Testzeit und Testressourcen bereitzustellen. Generell ist das rechtzeitige Bereitstellen ausreichender Ressourcen ein Indikator für gute Prozessqualität.

- **Projektkonforme Qualitätsplanung**

Der Verwendungszweck des geplanten und zu entwickelnden Software-Produktes muss eindeutig festgelegt werden, genauso wie die Lebensdauer dieses Produktes gesehen wird und wer die potenziellen Benutzer sind. Alle diese Faktoren beeinflussen die Ausprägung der Qualitätsanforderungen an ein Projekt und an ein Produkt wesentlich. Je nach Ausprägung dieser Qualitätsanforderungen sind geeignete Software-Engineering-Methoden und -Werkzeuge, aber auch Qualitätsmaßnahmen auszuwählen. Beispielsweise sind für ein Kundeninformationssystem einer Bank die Handhabbarkeit, die Verfügbarkeit und die Zuverlässigkeit wichtige Qualitätseigenschaften. Durch Ergonomie-Reviews sowie spezielle Last- und Stresstests kann die Ausprägung dieser Merkmale geprüft werden.

Bei der Qualitätsplanung sind auch die Projektrisiken zu berücksichtigen. Erfahrungen der Praxis zeigen, dass die Qualitätssicherungspläne bei Nichtberücksichtigung der Projektrisiken, wie beispielsweise Komplexität und Neuigkeitsgrad des Produkts, wertlos werden. Folgende Risikofaktoren haben Einfluss auf die Produkt- und Qualitätssicherungsplanung:

- Komplexität und Neuigkeitsgrad des Vorhabens,
- Anforderungsumfang,
- Mitarbeiterqualifikation,
- Projektgröße (Anzahl der Mitarbeiter, Aufwand an Personenjahren),
- Anforderungsungenauigkeit,

Bei der Projekt- und Qualitätsplanung sind jene Faktoren zunächst zu identifizieren, die ein potenzielles Risiko in sich bergen. Bei vorliegendem Risiko sind entsprechende Qualitätsmaßnahmen zu wählen und die Projektplanung (Planungshorizont, Ressourcen) ist zu verändern bzw. zu verfeinern.

- **Einfluss der Qualitätsprüfungsergebnisse in die Entwicklung**

Dieser Rückfluss ist ein wesentlicher Bestandteil für eine erfolgreiche Qualitätslenkung. Durch sie können Soll-Ist-Abweichungen des geplanten

Qualitätsniveaus erkannt werden. Das Prinzip dabei ist, dass Informationen, die bei Qualitätsprüfungen gewonnen wurden, zu entsprechenden Korrekturmaßnahmen im Entwicklungsprozess führen. Reviews (Inspektionen und Walkthroughs) und Audits (zur Prüfung des Entwicklungsprozesses oder des Managementprozesses) liefern die zur Korrektur des Entwicklungsprozesses notwendigen Informationen.

- **Gleiche Qualitätsprüfungen durch mehrere Personen**

Hierbei werden positive Effekte aus Gruppenprozessen, wie beispielsweise gemeinsame Analyse eines Dokuments, besser genutzt. Dass Menschen Fehler machen, ist eine Tatsache.

Neben den Prüfmethoden geht es hier um informelle Prüfungen. Davon spricht man dann, wenn ein Entwickler eines Zwischen- oder Endprodukts einer Phase seinen Kollegen bittet, sich ein von ihm erstelltes Dokument anzusehen und Korrektur zu lesen.

- **konstruktive Qualitätsmaßnahmen**

Bei der Qualitätsprüfung geht es darum, die vorhandene bzw. nicht vorhandene Qualität zu bestimmen. Ziel der konstruktiven Qualitätsmaßnahmen ist es, Fehler im Entwicklungsprozess zu vermeiden. Die Forderung lautet: Vermiedene Fehler müssen nicht entdeckt und behoben werden.

Die Produktqualität kann direkt durch geeignete vorbeugende Maßnahmen im Entwicklungsprozess verbessert und erhöht werden. Erfahrungen zeigen auch, dass durch vorbeugende Maßnahmen der Aufwand der Qualitätsprüfungen, z.B. des Testens, sich erheblich reduzieren lässt.

Ein weiterer positiver Aspekt dieses Prinzips ist, dass durch konstruktive Qualitätsmaßnahmen Qualitätsprüfungen erst ermöglicht werden. Beispielsweise kann durch eine geeignete Schnittstellenspezifikation für Module, in der Vorbedingungen, Nachbedingungen, der Effekt des Moduls und Fehlerbedingungen enthalten sind, das spezifikationsorientierte Testen wesentlich erleichtert werden. Konstruktive Qualitätsmaßnahmen

und analytische Qualitätsprüfungen ergänzen einander und bringen so einen effektiven Nutzen.

- **Frühzeitige Entdeckung und Behebung von Fehlern und Mängeln**

Wie im Kapitel 5.5 Software-Fehler beschrieben steigen die Kosten für die Entdeckung und Behebung eines Fehlers exponentiell mit der Dauer zwischen dem Zeitpunkt des Entstehens und dem Zeitpunkt der Entdeckung des Fehlers. Die Strategie muss also sein, Fehler möglichst früh zu erkennen und zu beheben. Durch dieses Prinzip ist auch eine verbesserte Qualitätslenkung gegeben, und der Verbreitungseffekt von Fehlern wird ebenfalls reduziert.

- **Qualitätsmanagement**

Qualitätsmanagement sollte in den gesamten Erstellungsprozess integriert sein. Das führt auch dazu, dass Qualitätsmaßnahmen organisatorisch mit den sonstigen Entwicklungsmaßnahmen geplant werden. Die Grundidee dabei ist, dass jede Entwicklungsaktivität aus einem konstruktiven und einem analysierenden/ prüfenden Teil besteht. Beide Teile schlagen sich in entsprechender Dokumentation nieder. Beispielsweise werden Anforderungen durch eine geeignete Anforderungsdefinitionsmethode erfasst und strukturiert. Durch eine informelle oder formelle Anforderungsprüfung werden anschließend die Anforderungsmängel beseitigt. Diese Prüfung wird auch dadurch erleichtert, dass durch den Einsatz einer Methode zur Anforderungsdefinition gleichzeitig Dokumentation entsteht (Anforderungsspezifikation). Durch dieses Prinzip wird auch das Qualitätsniveau zu jedem Zeitpunkt sichtbar. Insbesondere ist dies für die frühen Phasen wichtig. Es ist eine Voraussetzung für die realistische Beurteilung des Projektfortschritts und die Feststellung, ob die Qualitätsziele erreicht wurden.

- **Unvereinbarkeit des Programmierens mit der Qualitätsprüfung**

Viele der Softwareentwicklungsmaßnahmen sind konstruktiver und produktiver Natur, d. h. auch, dass der Entwickler Urheber vieler Arbeitsprodukte ist. Bei Qualitätsprüfungen ist die Situation eine andere, da es hier darum geht, Fehler und Mängel aufzudecken. Qualitätsprüfungen sollten sehr kritisch und sogar mit destruktiver Einstellung zum Prüfobjekt durchgeführt werden (vgl. DeMarco 1982: 374). Ein Entwickler kann aus diesem Grunde nicht seine eigenen Arbeitsergebnisse prüfen. Dieser prinzipielle Interessenskonflikt lässt sich nur durch eine unabhängige Qualitätsprüfung beheben. Ziel dabei ist, dass nur die Ist-Qualität festgestellt wird und die Fehlerbehebung in einem nachfolgenden Schritt durchgeführt wird.

- **Qualitätsmaßnahmen müssen bewertet werden**

Die eingesetzte Qualitätsmanagement-Organisation muss in bestimmten Zeitabständen mittels interner oder externer Qualitätsaudits geprüft werden. Externe Audits werden von externen Beratern durchgeführt, die das vorhandene (Qualitäts-) Managementsystem und die eingesetzten Qualitätsmaßnahmen überprüfen. Diese Audits führen in der Regel zu Korrekturen des Managementsystems und der eingesetzten Qualitätsmaßnahmen. Dadurch wird auch die Rentabilität der in ein Managementsystem getätigten Investition, gesteigert.

Erfahrungsgrundsätze die bei der Softwareentwicklung beachtet werden müssen (vgl. R. Dumke 2002):

nach Brooks: beim Entwicklungspersonal gibt es Produktivitätsunterschiede von bis zu 1:10

nach Moore: alle 18 Monate verdoppelt sich die Hardware-Leistung,
nach Jones: ein Entwickler schafft im Monat ca. 250 LOC,

nach Halstead: entwickelte (getestete) Programme haben bei ihrer Auslieferung noch ca. 3 Fehler pro KLOC,

nach Putnam: das Aufwandsverhältnis von einem entwickelten Programm zu einem Software-Produkt beträgt ca. 1:3,

nach Boehm: jeder Dollar bei der Softwareentwicklung kostet etwa zwei Dollar bei der Software-Wartung,

nach Zuse: qualitätsbewertende Maße können nur auf der Grundlage messtheoretischer Eigenschaften sinnvoll eingesetzt werden,

nach McCabe: die Entscheidungsanzahl in Programmen ist eines der grundlegendsten Komplexitätsmerkmale,

nach Symons: Software-Merkmale müssen klar unterscheidbar sein, um Umfangsschätzungen vergleichbar zu gestalten,

nach Pfleeger: formale Testmethoden für die Software zeigen nur die Korrektheit für ein formales Rechnermodell,

nach Humphrey: für die meisten Projektleiter zeigt sich heute die Situation für die Projekte: "out of control".

nach Dumke: die Halbwertszeit von COTS-Komponenten beträgt ca. 5 * tV, mit tV als durchschnittliche Zeit zwischen dem Versionswechsel der Komponentensoftware.

Qualitätsmaßnahmen lassen sich in zwei Bereiche unterteilen:

(vgl. Lenz 2003: 1ff) Konstruktive Qualitätsmanagementmaßnahmen und analytische Qualitätsmaßnahmen.

Konstruktive QM-Maßnahmen sorgen von vornherein für ein gewisses Maß an Qualität

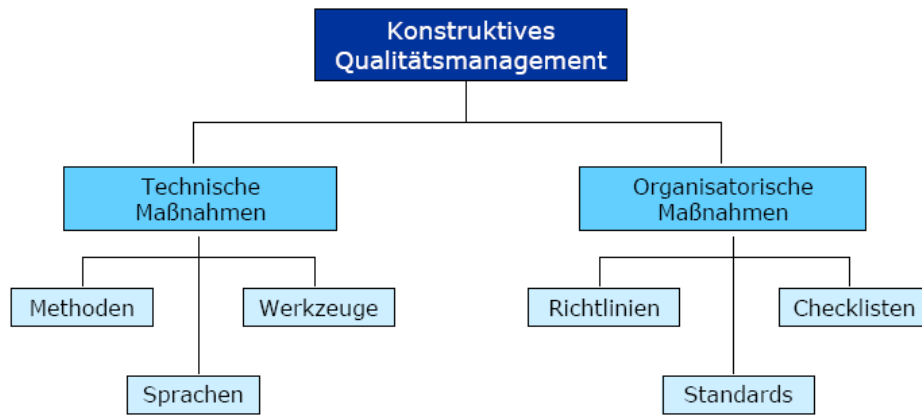


Abb. 19: Konstruktives QM

Quelle: Lenz, R. <http://www.2cool4u.ch> (Zugriff am 20.04.2005).

Analytische Qualitätsmaßnahmen dienen zur Qualitätssicherung, sie prüfen und messen das existierende Qualitätsniveau.

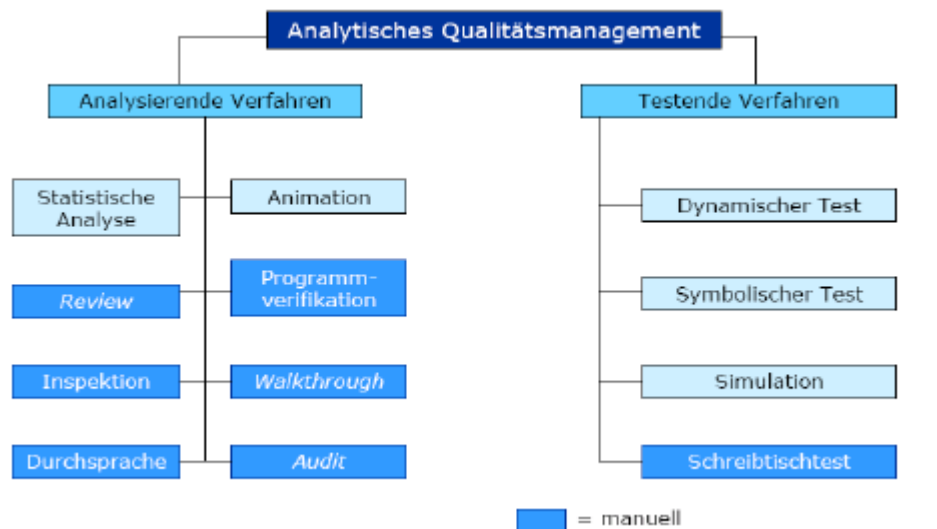


Abb. 20: Analytisches QM

Quelle: Lenz, R. <http://www.2cool4u.ch> (Zugriff am 20.04.2005).

Im nächsten Kapitel wird erläutert wie eine Wirtschaftlichkeits Beurteilung durchgeführt werden kann, danach werden die Kosten für Fehler bestimmt und aus dem abgeleiteten Potenzial werden ausgewählte Qualitätsmaßnahmen vorgestellt.

10 Wirtschaftlichkeitsanalyse

Kosten-Nutzenanalysen nehmen in der Betriebswirtschaft grundsätzlich Bezug auf das ökonomische Prinzip (Wirtschaftlichkeits-, Rationalprinzip), das Output zu Input eines Prozesses (Unternehmensaktivität) bewertet (vgl. Gutenberg 1975: 511).

Allgemein beschreibt Wirtschaftlichkeit das für ein bestimmtes Objekt (Software-Projekt) ermittelte Verhältnis aller aufgelaufenen Kosten zum resultierenden Nutzen.

Grundlegende Kennzahlen sind:

Return on Investment (ROI):

$\text{ROI} = \text{Nutzen (Ertrag - Aufwand)} / \text{Investitionskosten}$

$\rightarrow > 100 \%$

$\text{Amortisation} = \text{Nutzen (Ertrag - Aufwand)} / \text{Investitionskosten}$

$\rightarrow = 100 \%$

Wirtschaftliches Handeln beruht unter anderem auf 2 Grundsätzen

Maximalprinzip: Mit vorgegebenen Ressourcen das maximale Ergebnis erreichen.

Minimalprinzip: Das vorgegebene Ergebnis mit minimalem Ressourcenaufwand erreichen.

„Unter der Wirtschaftlichkeit des Qualitätsmanagements (Wirtschaftlichkeitsobjekt) wird eine unterschiedlich interpretierbare Ausprägung des Wirtschaftlichkeitsprinzips verstanden, die das Verhältnis zwischen Mitteln und Zielerreichung (Wirtschaftlichkeitskomponenten) im Rahmen des Qualitätsmanagements konkretisiert“ (Bruhn 1998: 99).

Um die Wirtschaftlichkeit einer Qualitätsmaßnahme beurteilen zu können, muss einerseits deren positive Wirtschaftlichkeitskomponente, resultierend in z.B. Mehr-Leistung, Ertrag, Einzahlungen oder Nutzen und

andererseits deren negative Wirtschaftlichkeitskomponente, die z.B. zu mehr Aufwand und Kosten führt, einander gegenübergestellt werden.

„Unter der qualitätsbezogenen Wirtschaftlichkeitsanalyse werden sämtliche Maßnahmen verstanden, die der Bewertung der Wirtschaftlichkeit des Qualitätsmanagements, d.h. des Verhältnisses seiner positiven und negativen finanziellen Konsequenzen, dienen“ (Bruhn 1998: 104).

Wirtschaftlichkeitsanalysen werden kategorisiert, ob sie mit einem **Bezugspunkt** verglichen werden oder dem **Zeitpunkt ihrer Durchführung**.

Hinsichtlich der Durchführung auf einen **Bezugspunkt** bezogen wird noch die **absolute Wirtschaftlichkeitsanalyse** und die **relative Wirtschaftlichkeitsanalyse** unterschieden. Die absolute Wirtschaftlichkeitsanalyse betrachtet die Wirtschaftlichkeit eines Qualitätsmanagements in Bezug zu einem Fixpunkt (meistens Null-Punkt). Eine Qualitätsmaßnahme wird dann als sinnvoll angesehen, wenn die Differenz zwischen Nutzen und Kosten größer Null ist. Die relative Wirtschaftlichkeitsanalyse vergleicht mehrere untersuchte, ökonomische Objekte miteinander.

Hinsichtlich des **Zeitpunkts der Durchführung**, muss unterschieden werden, ob die Wirtschaftlichkeitsanalyse vor (**präaktive qualitätsbezogene Wirtschaftlichkeitsanalyse**) oder nach (**postaktive qualitätsbezogene Wirtschaftlichkeitsanalyse**) dem unternehmerischen Handeln durchgeführt wurde (vgl. Bruhn 1999: 34ff).

Kriterien	Ausprägungen	Arten der Wirtschaftlichkeitsanalyse
Bezugspunkt	Fixpunkt	Absolute Wirtschaftlichkeitsanalyse
	Wirtschaftlichkeit eines realen Objektes	Relative Wirtschaftlichkeitsanalyse
Zeitpunkt	Vor Umsetzung des Qualitätsmanagements	Präaktive Wirtschaftlichkeitsanalyse
	Nach Umsetzung des Qualitätsmanagements	Postaktive Wirtschaftlichkeitsanalyse

Abb. 21: Arten der qualitätsbezogenen Wirtschaftlichkeitsanalyse

Quelle: Bruhn (1999): Kosten und Nutzen des Qualitätsmanagements. München: Hanser, S 34ff [eigene Bearbeitung]

Bruhn schlägt zur Durchführung einer Wirtschaftlichkeitsanalyse ein Phasenkonzept vor:

Phase 1: Die Qualitätsinvestition (Maßnahmenbündel) ist vor der Durchführung zu definieren.

Phase 2: Kategorienfestlegung der Wirtschaftlichkeitskomponenten, um Kennziffern zu bilden und Verbesserungspotenzial aufzuzeigen.

Phase 3: Erfassung der Ausprägungen der Wirtschaftlichkeitskomponenten.

Phase 4: Die Kosten- und Nutzenkategorien werden zur Bildung der Kennzahlen einander gegenübergestellt.

Phase 5: Durch die Beurteilung der Kennzahlen kann die Wirtschaftlichkeit des Qualitätsmanagements sichergestellt oder verbessert werden (vgl. Bruhn 1999: 41ff).



Abb. 22: Phasenkonzept der Wirtschaftlichkeitsanalyse

Quelle: Bruhn (1999): Kosten und Nutzen des Qualitätsmanagements. München: Hanser, S 43ff [eigene Bearbeitung]

Eine wesentliche Anforderung an ein Verfahren für die Wirtschaftlichkeitsbeurteilung stellt die Durchführbarkeit in der betrieblichen Praxis dar.

Hiermit sind hauptsächlich zwei Problemkreise angesprochen, die eng miteinander verknüpft sind:

- die Verfügbarkeit der Daten und der
- Aufwand zur Datensammlung.

Hierbei steht die Fragestellung im Vordergrund, ob die benötigten Informationen überhaupt zur Verfügung stehen bzw. mit welchem Aufwand sie beschafft werden können. Idealerweise sollte der zusätzlich vom Unternehmen zu leistende Aufwand zur Datenermittlung und - Datenauswertung so gering wie möglich sein.

Da z.B. der externe Nutzen bei der Abschätzung der Kosten für das zu untersuchende QDA-Projekt nicht vernünftig eruiert werden kann, wird auf eine zahlenmäßige Berücksichtigung bei der kommenden Wirtschaftlichkeitsanalyse verzichtet (folgend dem Prinzip: lieber keine Zahlen, als vom Ansatz her unwahre Zahlen). Der Nutzen für die nur

TKA-Presta intern verwendete Software QDA sollte im Hinblick auf die Argumentation bezüglich Verringerung der Fehlerkosten gegenüber dem Hersteller meiner Meinung nach nicht angeführt werden, da ja jeder Software-Hersteller seine Produktpreise natürlich auch unter Berücksichtigung des Kundennutzens des Programms kalkuliert und somit der Produktpreis den Nutzen für die TKA-Presta widerspiegelt. Produktpreise in der Software entstehen, anders als in der Produktion, nicht immer nur auf Grund der Herstellkosten und einem gleich bleibenden Deckungsbeitrag, sondern orientieren sich vor allem an Marktpreisen.

11 Qualitätskosten

Qualität kostet Geld; fehlende Qualität kostet mehr! (unbekannt)

Die Qualität eines Produktes ist nicht vom Produkt selbst zu trennen; sie entsteht mit ihm. Kosten für die Qualität sind nicht von den Herstellkosten zu trennen, sondern sind ein integrierter Teil dieser Kosten (Abgrenzung Qualitätskosten).

Bei der Erfassung der Qualitätskosten für eine Wirtschaftlichkeitsberechnung muss vor allem auch darauf Bedacht genommen werden, nicht einfach alle Kosten als Qualitätskosten auszuweisen. Es gilt jene Kosten zu identifizieren, die nicht generiert würden, wenn das Unternehmen keine Qualitätsstrategie verfolgen würde, und demnach keine konkreten Maßnahmen des Qualitätsmanagements einsetzen würde (vgl. Bruhn 1999: 102).

Mittels den so genannten Blind- und Fehlleistungen (stellen Verschwendung dar) der Qualitätskosten können Verbesserungspotenziale innerhalb des (Software-) Produktlebenszyklus aufgezeigt werden.

Dies ist auch der Ansatzpunkt für die nachfolgende Wirtschaftlichkeitsanalyse des QDA-Softwareprojekts, in der die Fehlleistungen, die aus Fehlerkosten und Kosten der Nacharbeit entstehen, erfasst werden.

In der Softwareentwicklung ist der Anteil der Qualitätskosten stark vom Gesamtentwicklungsaufwand des Projektes abhängig (vgl. Jones 1996: 142ff).

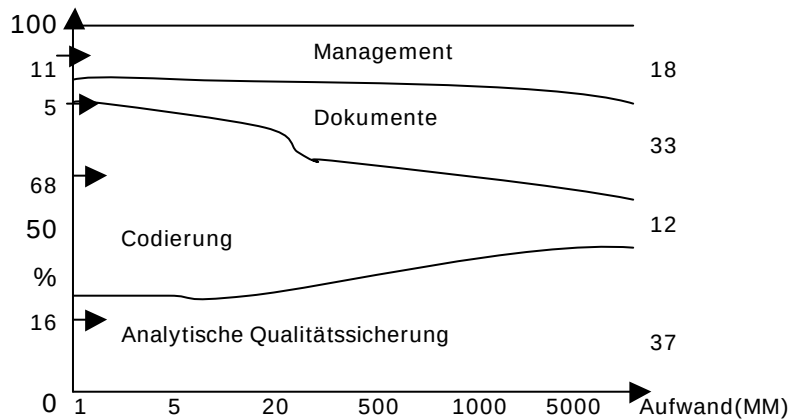


Abb. 23: Relative Anteile am Entwicklungsaufwand

Quelle: Liggesmeyer 2002: 28 [eigene Bearbeitung]

Für kleinere Projekte überwiegen die Programmierungskosten, bei großen IT-Projekten (mehrere 1000 Mannmonate) beträgt der Aufwand für die Qualitätssicherung schon ca. 37 %, während die Programmierung nur mehr 12 % ausmacht. Vor diesem Hintergrund ist es wichtig auch die Software-Projekte bei TKA-Presta einzuordnen. Für das QDA-Projekt beträgt der zeitliche Entwicklungsaufwand bis jetzt (2005.04) ca. 5500 Stunden. Für die beim Lieferanten installierte Qualitätssicherung wird ca. 1/3 der Projektzeit dafür veranschlagt, was nach Abb. 23 bei diesem zeitlichen Projektaufwand in etwa bestätigt werden kann. Aus der nachfolgend durchgeführten Fehlerkostenbetrachtung muss die Effizienz der durchgeführten Qualitätssicherungsmaßnahmen aber in Frage gestellt werden.

11.1 Entwicklung der Qualitätskosten

In den 50er Jahren begann man die Kosten für die Qualität aufzuzeichnen. Der Begriff Qualitätskosten wurde erstmalig im Jahre 1951 in Juran´s Quality Control Handbook als „Cost of Quality“ erwähnt (vgl. Juran 1988: 25).

Bis heute lässt sich die Entwicklung in 3 Phasen einteilen:

- 1) Tätigkeitsorientierte Analyse qualitäts-bezogener Kosten,
- 2) Wirkungsorientierte Analyse qualitäts-bezogener Kosten,
- 3) Investitionsbetrachtung des Qualitätsmanagements.

11.2 Tätigkeitsorientierte Analyse qualitäts-bezogener Kosten

Mitte der 50er Jahre begründete Feigenbaum die PAF-Kategorisierung (Prevention-Appraisal-Failure) (vgl. Feigenbaum 1985: 25ff). Sie erfasst die Kosten die aufgrund qualitätsbezogener Tätigkeiten im Unternehmen entstehen. Hierzu werden Tätigkeiten für die Fehlerprävention, bzw. der Fehlerverhütung, Tätigkeiten wegen Fehler- oder Ausfällen (Failure costs), sowie Tätigkeiten der Prüfung (Appraisal costs) systematisch erfasst und in die entsprechenden Qualitätskostengruppen aufgegliedert. Darauf aufbauend wurden später die Fehlerkosten häufig nach internen und externen Fehlerkosten unterschieden.

Dieser traditionellen Einteilung folgt die Norm **DIN 55350, Teil 11**:

- 1) Fehlerverhütungskosten (Präventionskosten),
- 2) Prüfkosten und
- 3) Interne und externe Fehlerkosten.

Aus Erfahrungen der industriellen Produktion ergibt sich eine Aufteilung der Kostenblöcke auf folgende Anteile.

Fehlerverhütungskosten	2 bis 15 %
Prüfkosten	60 bis 80 %
interne und externe Fehlerkosten	20 bis 30 %

(vgl. QM-Trends (2005)).

Fehlerverhütungskosten entstehen im Rahmen von Aktivitäten zur Vermeidung einer nicht anforderungsgerechten Qualität. Zu den Fehlerverhütungskosten (Präventionskosten) zählen beispielsweise Qualitätsplanung, Audits, Lieferantenbeurteilungen, Prüfplanung und Schulungen im Qualitätsbereich.

Unter **Prüfkosten** versteht man Qualitätsprüfungen im Sinne von Mess-, Evaluierungs-, Test- oder Auditaktivitäten. Beispiele für Prüfkosten sind u.a. Testen, Code-Reviews und Abnahmeprüfungen.

Kosten, die durch die mangelnde Übereinstimmung der Leistung mit gewissen Normen, Standards und/oder Anforderungsspezifikationen entstehen, sind die **internen/externen Fehlerkosten**. Als Beispiele für interne Fehlerkosten zählen mangelndes Projektkonzept, Fehlerbehebung und interne Korrekturen nach Testläufen. Zu den externen Fehlerkosten zählen Gewährleistungen, Kulanz und externe Fehlerbehebungen.

Nach dem Ansatz der tätigkeitsorientierten Analyse qualitätsbezogener Kosten führt eine wirtschaftliche Erstellung von Qualität zu einer Minimierung der gesamten qualitätsbezogenen Kosten (vgl. Feigenbaum 1985: 73ff). Somit ist aus Kostengesichtspunkten ein Qualitätsniveau bei < 100 % zu suchen (vgl. Abb. 22).

Bei einem Qualitätsniveau von Null Prozent nehmen die Präventions- und Prüfkosten ebenfalls einen Wert von Null an, da keine Aktivitäten zur Prüfung und Fehlerverhütung durchgeführt werden. Bei steigendem Qualitätsniveau erfährt der Kostenverlauf der Präventions- und Prüfkosten nach Modellannahme einen exponentiellen Verlauf. Zur Erreichung eines Qualitätsniveaus von 100 % müssten folglich die Präventions- und Prüfkosten mit einem Wert gegen unendlich in Kauf

genommen werden. Im Umkehrschluss dazu sind die Fehlerkosten bei einem Qualitätsniveau von Null Prozent sehr hoch. Bei einem Vollkommenheitsgrad von 100 % würde ein Wert Null erreicht werden. Zu beachten ist jedoch, dass auch dieser Fall nie eintreten kann, da ein völliges Vermeiden von Fehlern nicht realisierbar ist, und die Kosten wie dargestellt ins Unendliche anwachen würden. Es werden also Fehler toleriert, d.h. Kundenanforderungen werden/müssen nicht zu 100 % erfüllt sein, weil das als unwirtschaftlich angesehen wird (vgl. Abb. 24).

Kritische Würdigung: Dass Fehler nicht zu 100 % eliminiert werden können ist nachvollziehbar. D.h. aber nicht, dass Kundenforderungen trotzdem nicht zu 100 % erfüllt werden können. Vielmehr wird unter dem Qualitätsbegriff eine Interaktion mit dem Kunden verstanden, sodass die Qualitätsanforderungen und somit die Qualitätsmerkmale kritisch hinterfragt werden müssen (QFD), damit eine Erfüllung der Kundenanforderungen erreicht werden kann (genauso wie Management by Objectives ohne SMARTe Ziele nutzlos ist).



Abb. 24: Tätigkeitsorientiertes Modell qualitätsbezogener Kosten

Quelle: Q-Berlin (2005).

11.3 Wirkungsorientierte Analyse qualitätsbezogener Kosten

Die tätigkeitsorientierte Analyse wird als sehr „technik-bezogen“ angesehen, weil Qualitätskosten nur als zusätzliche Kosten verstanden werden und positive Investitionen zur Qualitätsverbesserung mit negativen Ausgaben zur Fehlerbeseitigung vermischt werden. Bei der Entstehung der tätigkeitsorientierten Analyse war das Vermeiden von Fehlern zentrales Ziel des Qualitätsmanagements. Mitte der 80er Jahre des letzten Jahrhunderts wurde daher die wirkungsorientierte Analyse qualitätsbezogener Kosten entwickelt, um eine bessere Beurteilung der Qualitätskosten vorzunehmen. Das wirkungsorientierte Modell baut auf dem tätigkeitsorientierten Modell qualitätsbezogener Kosten auf. Die folgende Grafik stellt die Schnittstellen zwischen den beiden Modellen dar.

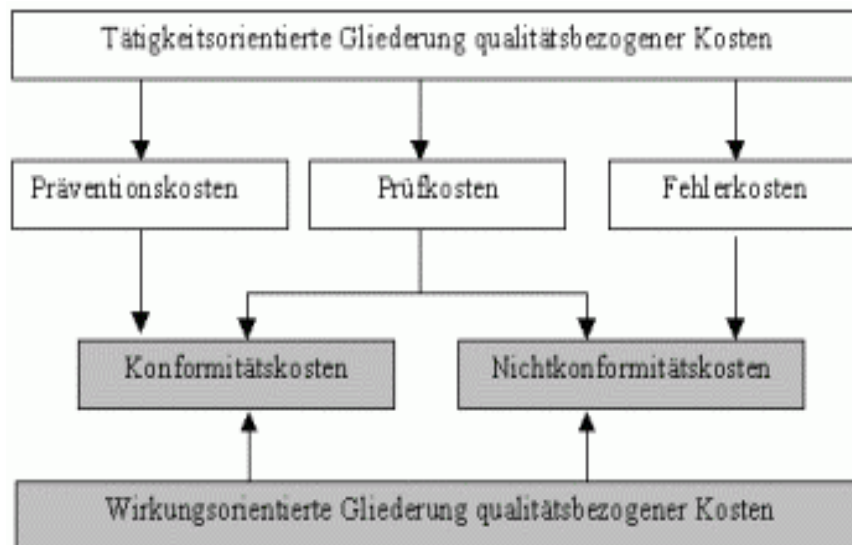


Abb. 25: Tätigkeits- vs. wirkungsorientierte Kostengliederung

Quelle: Bruhn 1999: 51 (in Anlehnung an Wildemann 1995).

Die qualitätsbezogenen Kosten werden in

- **Konformitätskosten** (Cost of conformance“) und
- **Nichtkonformitätskosten** (Abweichungskosten, Fehlleistungsaufwand, „Cost of Non-Conformance“) eingeteilt (vgl. Wildemann 1995: 762ff).

Konformitätskosten sind jene notwendigen Ausgaben zur Erfüllung der Kundenanforderungen. Hauptbestandteil der Konformitätskosten sind demnach die Präventions- und Prüfkosten des PAF-Schemas.

Im Gegensatz dazu können die **Nichtkonformitätskosten** bei einer Opportunitätskostenbetrachtung als der entgangene Nutzen aufgefasst werden. Sie stellen also eine Verschwendung von Ressourcen dar, die bei der Abweichung von den Qualitätsanforderungen des Kunden entstehen. Die Fehler- und Prüfkosten (extra Prüfungen bei Fehlern) aus dem PAF-Schema würden analog die Nichtkonformitätskosten bilden. Da Konformitätskosten und Nichtkonformitätskosten also gegensätzlich sind, ist eine Kosten-Nutzenbetrachtung so nicht möglich.

Erklärung: Entweder führt eine Qualitätsmaßnahme zur Konformität mit den dazugehörigen Kosten und Nutzen oder sie ist Nicht-Konform.

Nichtkonformitätskosten sind im Gegensatz zu den Konformitätskosten **nicht bekannt, nicht planbar** und nur **schätzbar**.

Abb. 26 beschreibt die Zusammenhänge zwischen Konformitäts-, Nichtkonformitäts- und qualitätsbezogenen Gesamtkosten. Die Nichtkonformitätskostenkurve, die annähernd der Fehlerkosten nach der klassischen Kosteneinteilung entspricht, hat einen fallenden Verlauf. Bei 100 % Vollkommenheit schneidet sie die Abszisse. Die Konformitätskostenkurve steigt exponentiell an. Sie strebt bei einem Vollkommenheitsgrad von 100 % nicht gegen unendlich. Die qualitätsbezogenen Gesamtkosten haben deshalb bei 100 % ebenfalls einen endlichen Wert.

Aus diesem Sachverhalt geht hervor, dass das optimale Qualitätsniveau dort liegt, wo keine Fehler entstehen, dadurch werden die Kundenerwartungen auch zu 100 % erfüllt.

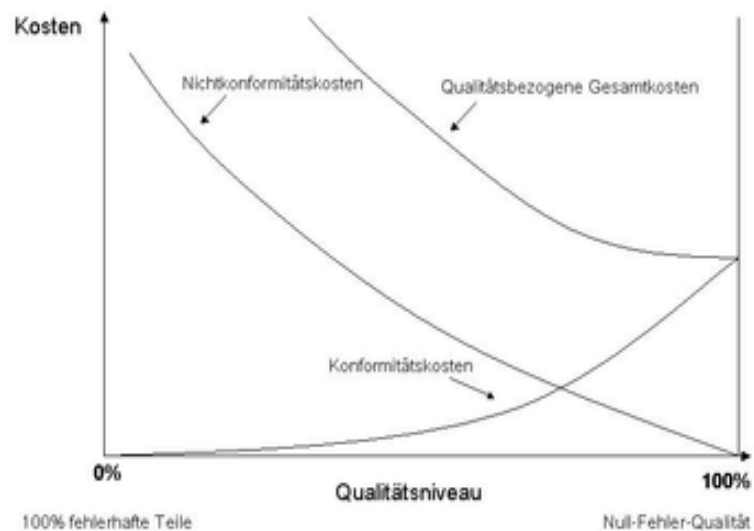


Abb. 26: Wirkungsorientiertes Modell qualitätsbezogener Kosten

Quelle: Rust, Zahorik, Keinigham (1994): Return on Quality. Measuring the Financial Impact of Your Company's Quest for Quality. Chicago.

11.4 Fehlerkostenrechnung

Die Fehlerkostenrechnung entstand daher, da weder die Zusammenfassung der Fehler- und Qualitätssicherungskosten noch die Zusammenfassung von Konformitäts- und Nichtkonformitätskosten sinnvoll ist (vgl. Seghezzi 1997: 185ff).

Die Fehlerkostenrechnung gliedert sich in zwei Dimensionen:

Nach dem **Ort der Fehlerentstehung** werden interne und externe Kosten differenziert

Nach den **Fehlerkonsequenzen** werden direkte und indirekte Fehlerkosten und fehlerbedingte Opportunitätskosten unterschieden.

Fehlerkonsequenz Ort der Fehlerentstehung	direkte Fehlerkosten	indirekte Fehlerkosten	Fehlerbedingte Opportunitätskosten
Interne Fehlerkosten	-Nacharbeitskosten -Ausschusskosten Kosten zur Korrektur einer Dienstleistung	-Aufwand zur Fehlerbehandlung -Mehraufwand bei der Fehlerbehebung (z.B. Feiertagsarbeit)	-Verlust verärgerter Mitarbeiter -Kosten für erhöhte Lagerhaltung bei fehleranfälligen Produkten
Externe Fehlerkosten	-Wertminderung aufgrund von Fehlern Kosten aufgrund vertraglicher Haftung Kosten aufgrund gesetzlicher Haftung	-Kosten wegen fehlerbedingt verspäteter Lieferung (z.B. verspätete Zahlung) -Service- und Ersatzleistungen im Rahmen der Kulanz	-Verlust künftiger Aufträge aufgrund von Fehlern in vorausgegangenen Lieferungen -Verlust verärgerter Kunden -Mehraufwand für Werbung u.ä. aufgrund von Qualitätseinbrüchen

Abb. 27: Erweiterte Gliederung der Fehlerkosten nach der Fehlerkostenrechnung

Quelle: Bruhn (1999): 53 [eigene Bearbeitung]

11.5 Zusammenfassung

Den beiden erstgenannten Konzepten ist gemein, dass sie vergangenheitsorientiert sind und somit eine proaktive Wirtschaftlichkeitsanalyse nicht durchführbar ist. Deshalb führt eine Quantifizierung der qualitätsbezogenen Kosten nur zur Identifizierung und Verdeutlichung von qualitätsbezogenen Verbesserungspotenzialen, weniger zur wirkungsgerechten Planung und Kontrolle entsprechender Verbesserungsmaßnahmen. Auch bei der wirkungsorientierten Analyse entsprechen die Nichtkonformitätskosten nur einem Teil des Nutzens des Qualitätsmanagements, da kundenverhaltensbezogene Wirkungen, wie die Erhöhung der Kauffrequenz, Kundenbindung nicht berücksichtigt werden. Immerhin können die Nichtkonformitätskosten als entgangener Nutzen interpretiert werden. Deshalb bringt die wirkungsorientierte Analyse zum Ausdruck, dass Konformitätskosten Investitionen in die Schaffung und Erhaltung einer den Kundenanforderungen entsprechenden Qualität dienen. Die Fehlerkostenanalyse bietet den Vorteil der klaren Zuordenbarkeit der Fehlerkosten zu den einzelnen Fehlerkostenkategorien. Durch die Einteilung in direkte und indirekte Fehlerkosten, wird eine erhöhte Erfassbarkeit der Fehlerkosten möglich. Diese Gründe sind auch ausschlaggebend, dass für die wirtschaftliche Ist-Betrachtung des QDA-Projektes diesem Konzept der Vorzug eingeräumt wird.

12 Nutzen von Qualitätsmaßnahmen

Bei effizienten Qualitätsmaßnahmen fallen nicht nur Kosten an, sondern es ergibt sich auch ein Nutzen. Ein Nutzen von Qualitätsmanagement/Qualitätsmaßnahmen wird durch das Maß der Zielerreichung durch qualitätsbezogene Aktivitäten ausgedrückt. Der Nutzen kann hier, anders als bei Sachinvestitionen nicht direkt aus dem Erlös vom Verkauf von Produkten angegeben werden, sondern durch die Verbesserung ökonomischer Zielgrößen in Form von Erlössteigerungen, z.B. durch erhöhte Kundenzufriedenheit und (Fehler-) Kostensenkungen.

Die Folgen von Qualitätsmaßnahmen sind oft vielfältig und komplex, daher ist es generell schwierig, ihre Wirtschaftlichkeit zu beurteilen. Eine allgemein anwendbare Methode zur Wirtschaftlichkeitsbeurteilung existiert daher nicht. Es muss stets im Einzelfall geprüft werden, welche Methode der spezifischen Situation am besten gerecht wird. Analysen der Wirtschaftlichkeit von Maßnahmen des Qualitätsmanagements erfordern eine Erfassung und Bewertung der zu erwartenden bzw. bereits realisierten positiven wie negativen Auswirkungen.

Grundsätzlich können beim Aufdecken und Bewerten dieser unternehmensspezifischen Wirkungen und Effekte folgende Merkmale unterschieden werden:

1) qualitätsbezogene Perspektiven

2) Realisierung der Wirkungen des Qualitätsmanagements

12.1 Nutzenarten des Qualitätsmanagements

Bei der qualitätsbezogenen Perspektive werden Zeiträume miteinander verglichen, zum einen jener, in der die Qualitätsmaßnahme noch nicht eingeführt wurde (Aktivitätsabsenz) und der Zeitraum nach der Ein-/Durchführung der Qualitätsmaßnahme (Aktivitätspräsenz). Wenn der Nutzen in der Aktivitätspräsenz den in der Aktivitätsabsenz übersteigt so liegt ein Nutzen des Qualitätsmanagements vor. Die Realisierung der nutzenrelevanten Werte können in eine Liquiditäts- und eine Opportunitätskomponente eingeteilt werden (vgl. Bruhn 1998: 201f).

Situation Realisierung	Aktivitätsabsenz	Aktivitätspräsenz
	Kosten	Erlöse
Liquiditätskomponente	Beispiele Zusätzliche Kosten aufgrund von: <ul style="list-style-type: none"> - nachträglichen Fehlerkorrekturen - Nacharbeit - Garantiezahlungen 	Beispiele Zusätzliche Erlöse durch <ul style="list-style-type: none"> - erhöhte Kundenbindung - Cross Buying - weitere Projekte
Opportunitätskomponente	Opportunitätskosten Beispiele Entgangene Erlöse aufgrund von <ul style="list-style-type: none"> - Kundenabwanderungen - Einschränkung der zur Verfügung stehenden Geldmitteln - Preiseinbußen 	Opportunitätserlöse Beispiele Vermiedene Kosten aufgrund von <ul style="list-style-type: none"> - verringerten Fehlerkorrekturen - weniger Abstimmungsaufwand - weniger Garantiezahlungen

Abb. 28: Nutzenarten des Qualitätsmanagements

Quelle: Bruhn 1999: 75

In der Aktivitätsabsenz treten Kosten in Form zusätzlicher Kosten aufgrund z.B. nachträglicher Fehlerkorrekturen auf, Kosten die in Aktivitätspräsenz nicht mehr auftreten sollen und Opportunitätskosten in Form von einem Nutzenentgang, wie z.B. durch Preiseinbußen (wegen

verminderter Qualität, Kosten wegen Nacharbeit, Gewährleistungszahlungen).

Durchgeführte Qualitätsmaßnahmen führen in der Aktivitätspräsenz zu

- Erlösen, z.B. zu erhöhte Kundenbindung. Und
- Opportunitätserlösen, wie z.B. vermiedene Kosten aufgrund verringerter Fehlerkorrektur.

Der Betrag dieser Erlöse entspricht der Höhe der Opportunitätskosten in der Aktivitätsabsenz. Der Betrag der Opportunitätserlöse entspricht den Kosten in der Aktivitätsabsenz. Opportunitätserlöse sind Kosten, die anfallen würden, wenn die Qualitätsmaßnahme nicht durchgeführt worden wäre.

12.2 Nutzenkategorien des Qualitätsmanagements

Weiters lässt sich anhand der Nutzenkategorien des Qualitätsmanagements **interner** und **externer** Nutzen unterscheiden.

Interne qualitätsbezogene Nutzen haben eine Verbesserung der Leistungserstellung zum Gegenstand und führen meistens zu Kostensenkungen (vgl. Bruhn 1999: 76ff).

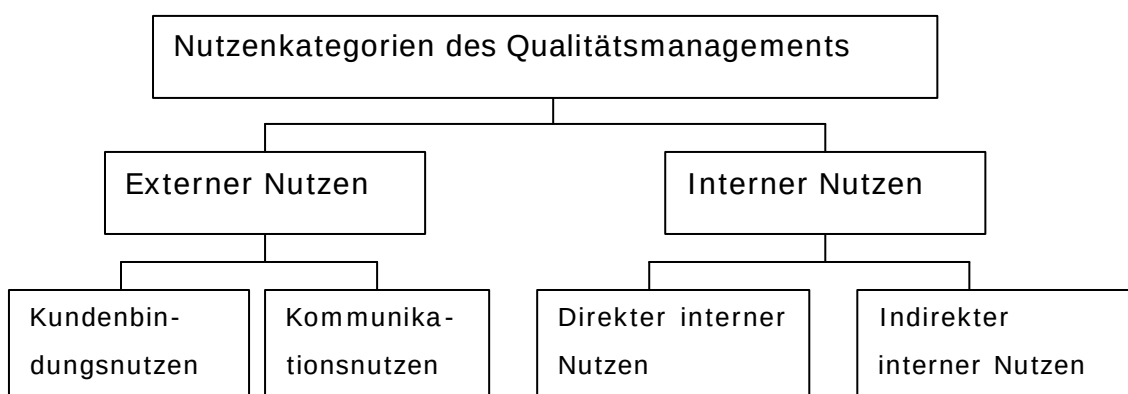


Abb. 29: Nutzenkategorien des Qualitätsmanagements

Quelle: Bruhn 1999: 77 [eigene Bearbeitung]

Der **interne Nutzen** unterscheidet zwischen **direktem** Nutzen und **indirektem** Nutzen.

Der direkte Nutzen führt zu Kosteneinsparungen und kann auf Kostenstellen und Kostenträger direkt zugerechnet werden, während der indirekte Nutzen nicht eindeutig den Kostenstellen, Kostenträgerstellen zu gerechnet werden kann.

Der **externe Nutzen** kann anhand des Kundenverhaltens festgemacht werden. Dazu zählen

- **Kundenbindungsnutzen** und
- **Kommunikationsnutzen.**

„Unter **Kundenbindungsnutzen** werden sämtliche psychologischen Bewusstseinsprozesse bzw. beobachtbaren Verhaltensweisen eines Kunden verstanden, in denen sich die Erhaltung bzw. Intensivierung seiner Beziehung zum Unternehmen manifestiert“ (Bruhn 1998: 212).

Der Kundenbindungsnutzen wird von der Kundenzufriedenheit beeinflusst, jedoch stehen sie in keiner linearen Beziehung. So können beispielsweise aufgrund von rechtlichen Bindungen oder fehlenden Alternativen auch unzufriedene Kunden eine hohe Kundenbindung aufweisen. Hingegen kann ein schlechtes Image trotz hoher Zufriedenheit zur Kundenabwanderung führen (vgl. Homburg 1998: 56ff).

„Unter der Mund-zu-Mund **Kommunikation** wird die in unterschiedlichem Maße zweckorientierte Übermittlung von unternehmens- oder leistungsspezifischen Informationen und Bedeutungsinhalten durch Kunden eines Unternehmens mit der Folge der Beeinflussung von Meinungen, Einstellungen, Erwartungen und Verhaltenweisen der Adressaten verstanden“ (Bruhn 1998: 228).

Eine Steigerung der Kundenzufriedenheit führt zu einer positiven Mund-zu-Mund-Kommunikation, während die Erhöhung der Kundenunzufriedenheit zu einer negativen Mund-zu-Mund-Kommunikation führt (vgl. Anderson 1994: 55).

Die Relevanz des Kommunikationsnutzens für das Unternehmen, ist von seiner Branchenzugehörigkeit abhängig. Im Falle einer Lieferkette, wie sie zwischen der TKA-Presta und seinen Software-Lieferanten besteht, ist wegen Konkurrenzaspekten weniger Mund-zu-Mund-Kommunikation vorhanden. Durch die meistens enge Bindung der Lieferanten mit TKA-Presta entstehen so Partnerschaften, die weniger anfällig gegenüber negativer Mund-zu-Mund-Kommunikation sind.

13 Wirtschaftlichkeitsmodell des Qualitätsmanagements

Um das optimale Aktivitätsniveau des Qualitätsmanagements bestimmen zu können, ist die Entwicklung von Kosten und Nutzen zu betrachten. Die **Kostenanalyse** dient wie beschrieben zur Bestimmung eines optimalen Qualitätsniveaus, und so wird sie als Bezugsgröße bei der Grenzkostenbetrachtung gewählt. Aus Untersuchungen ergibt sich der folgende Verlauf der Kostenkurve wie in Abb. 30 gezeigt.

Intensive Qualitätsanstrengungen ziehen ein höheres Aktivitätsniveau nach sich, der Kurvenverlauf wird durch Erfahrungswerte und „Economies of scope“ des Qualitätsmanagements beeinflusst.

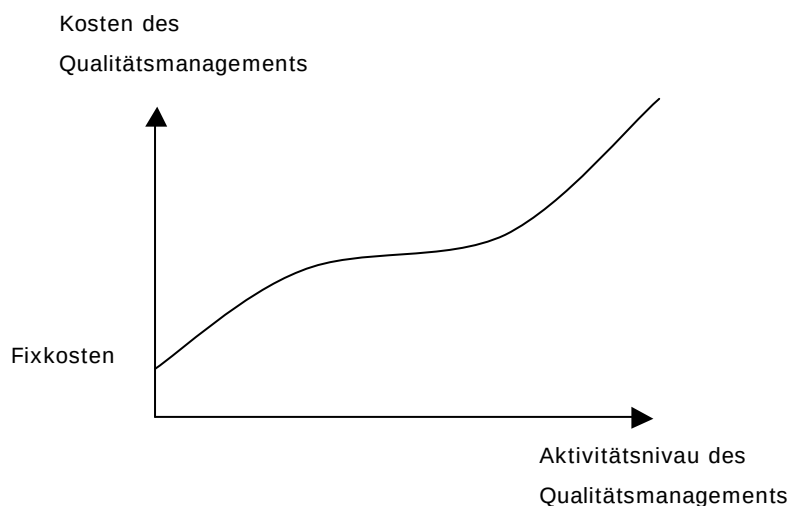


Abb. 30: Kostenkurve des Qualitätsmanagements

Quelle: Bruhn 1999: 89 [eigene Bearbeitung]

Qualitätsbezogene Lerneffekte resultieren aus der technologische Entwicklung und „economies of scale“, z.B. gewinnen die Mitarbeiter mit zunehmender Erfahrung durch eine qualitätsorientierten Denkhaltung an Kreativität beim Finden von Problemlösungen für die Fragestellungen des Kunden.

„Economies of scope“ werden durch das Abstimmen der verschiedenen Aktivitäten des Unternehmens auf ein bestimmtes Ziel hin erreicht, z.B. ist es notwendig die Aktivitäten der Mitarbeiter auf verschiedenen Hierarchiestufen abzustimmen und zu integrieren.

Die Kurve in Abb. 30 sagt aus, dass bei niedrigem Aktivitätsniveau des Qualitätsmanagements zunehmende Grenzkosten anfallen.

Z.B. erfordert die Einführung von Qualitätsmaßnahmen einen hohen Koordinationsaufwand zwischen den beteiligten Abteilungen.

Im Bereich eines mittleren Aktivitätsniveaus können Erfahrungskurveneffekte realisiert werden, dadurch nehmen die Grenzkosten ab.

Bei der **Nutzenentwicklung** ist die Entwicklung des Nutzens in Abhängigkeit des qualitätsbezogenen Aktivitätsniveaus zu analysieren. Die Kurve hat einen degressiven Verlauf. Sie sagt aus, dass bei geringer Qualitätsmanagementaktivität, eine kleine Änderung der Aktivität, eine große Nutzenänderung nach sich zieht.

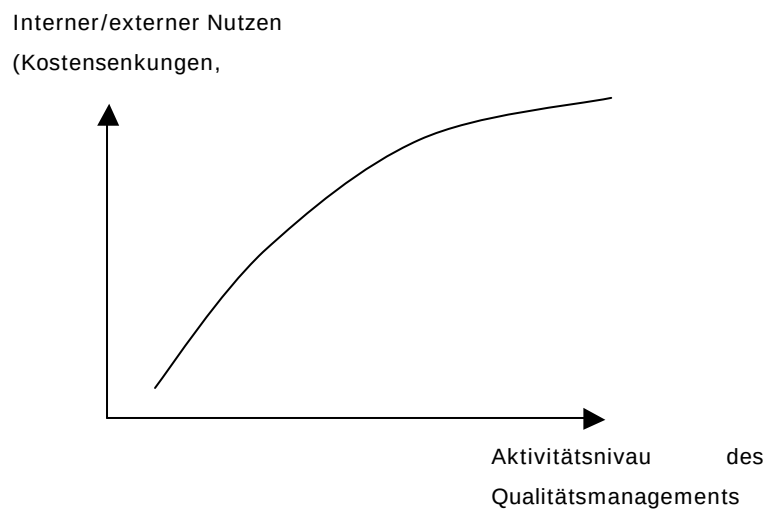


Abb. 31: Zusammenhang zwischen Aktivitätsniveau und Nutzen

Quelle: Bruhn 1999: 92 [eigene Bearbeitung]

Wird die Kostenkurve der Nutzenkurve gegenüber gestellt, so kann eine Gewinnfunktion des Qualitätsmanagements abgeleitet werden.

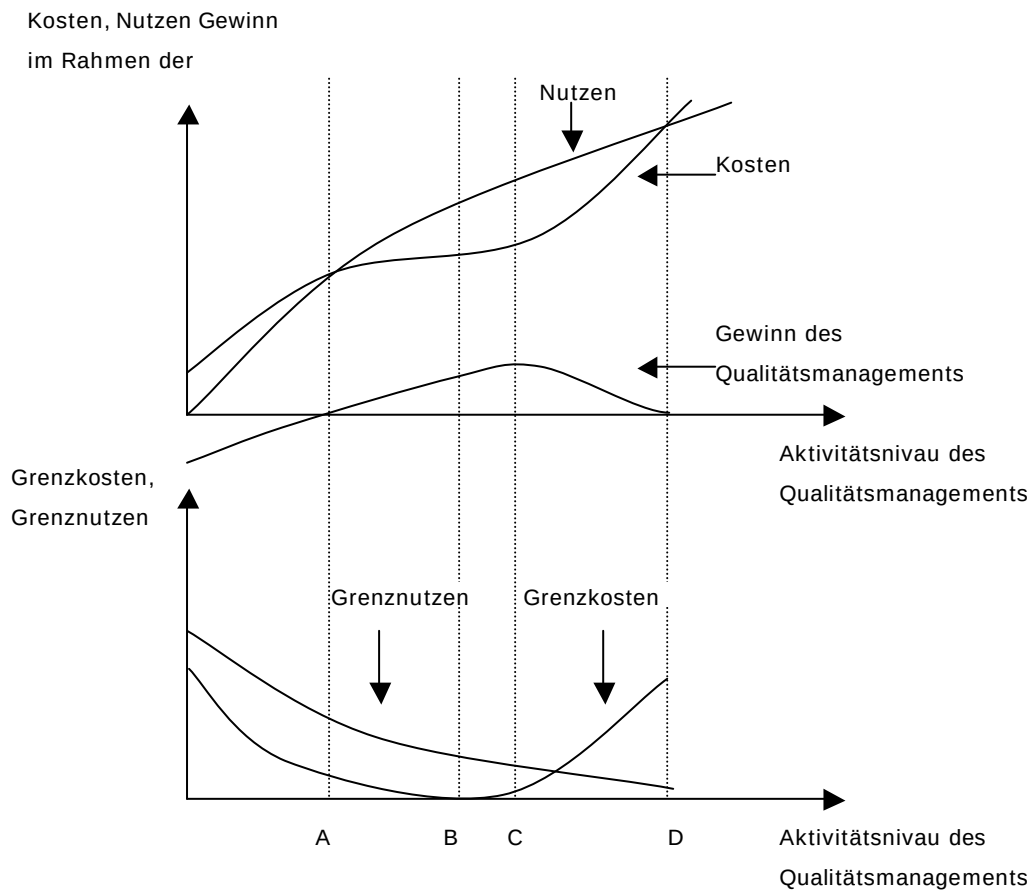


Abb. 32: Optimales Aktivitätsniveau des Qualitätsmanagements

Quelle: Bruhn 1999: 97 [eigene Bearbeitung]

In Abb. 32 bildet die Differenz zwischen Kosten- und Nutzenkurve die Gewinnkurve. Im Bereich eines geringen Aktivitätsniveaus des Qualitätsmanagement liegen die Kosten über dem Nutzen. D.h. die Qualitätsmaßnahmen sind verlustreich.

Ab dem Punkt A wird ein Qualitätsgewinn erzielt. Wegen abnehmenden Grenznutzen und der zunehmenden Grenzkosten erreicht der Gewinn im Schnittpunkt C sein Optimum (Gewinnkurve des Qualitätsmanagements in oberer Abb. 32). Ab diesem Punkt nimmt die Profitabilität des Qualitätsmanagements wieder ab. Der ermittelte Verlauf der Gewinnfunktion des Qualitätsmanagements impliziert die Existenz eines

optimalen Aktivitätsniveaus des Qualitätsmanagements, so dass eine qualitätsbezogene Intensivierung nicht zwingenderweise, positive finanzielle Konsequenzen nach sich zieht. Wegen der aufgezeigten Existenz eines optimalen Aktivitätsniveaus des Qualitätsmanagements, sollten Wirtschaftlichkeitsgesichtspunkte bei der Gestaltung des Qualitätsmanagements stets berücksichtigt werden.

14 Wirtschaftlichkeitsanalyse des QDA-Softwareprojekts

14.1 Vorstellung des Projektes

QDA steht für Quality Data Analysis und ist ein Softwaretool um Qualitäts-Daten zu suchen und die aufbereitete Ergebnismenge an eine weitere, nach geschalteten, statistischen Auswertesoftware zu übergeben.

Bei der Fertigung von Produkten ist eine zentrale Forderung des Qualitätsmanagements, dass die Qualität der Produkte laufend gemessen, kontrolliert und analysiert werden muss.

Die Norm ISO 9000:2000 schreibt vor, dass Qualitätsdaten aufgezeichnet werden müssen und dass der Lenkungsprozess von Qualitätsaufzeichnungen dokumentiert sein muss (Dokumentationspflicht sämtlicher Unternehmensprozesse).

Bei der TKA-Presta werden zu diesem Zweck sämtliche Messwerte der zu überwachenden Merkmale (Einteilung nach: kritisches Merkmal /Hauptmerkmal, etc. siehe Kapitel 5.5 Software-Fehler), mittels eines Erfassungssystems auf jeder Produktionsmaschine mitprotokolliert. Die Q-Daten werden dazu im QDAS[®] Datenformat geschrieben. Diese Q-Daten sind die Grundlage für jede Qualitätsanalyse bei der TKA-Presta. Nur über diese Aufzeichnungen kann die Gewährleistungspflicht erfüllt werden.

QDA wird seit 2003 mit unserem Software-Hauptlieferant entwickelt. Vom gleichen Lieferanten wird ebenfalls die Leitsystem- und Erfassungssoftware, welche auf den meisten Produktionsmaschinen im Lenkungsbereich eingesetzt wird, entwickelt. Der Lieferant ist ein reines Softwareunternehmen mit ca. 50 MA und ist eine Tochtergesellschaft eines großen österreichischen Maschinenherstellers, welcher für die TKA-Presta Produktionsmaschinen aufbaut. Aus dieser Verbindung ist entsprechendes Know-how, über die Fertigungsprozesse bei der TKA-

Presta ausreichend, vorhanden. Somit ist der Lieferant bestens mit den Anforderungen von der industriellen Lenkungsproduktion vertraut.

QDA Software Beschreibung:

QDAUpload processes quality-data of single machines and saves those on a central file-server and indexes the data for fast retrieval. The index-data is therefore stored in a central database.

QDASelect uses the index-data, copies respectively aggregates the source-quality-data into a temporary dfd/dfx-filepair and starts qs-STAT with these files for analysis.

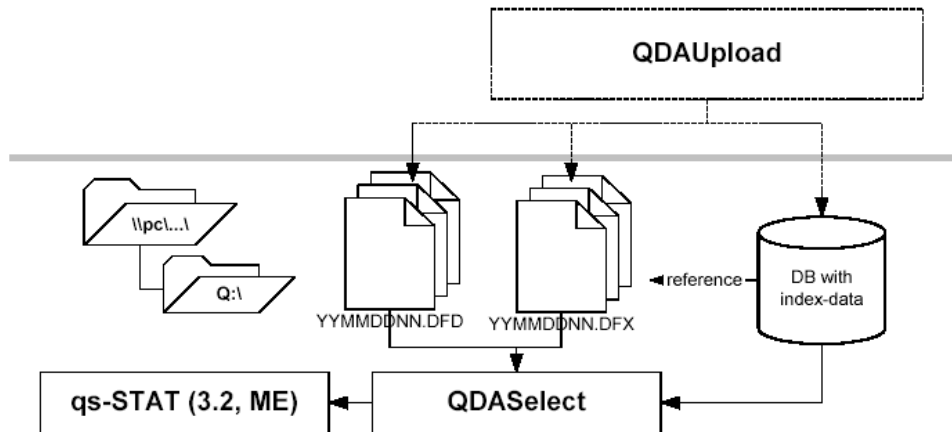


Abb. 33: Funktionsprinzip und Datenfluss der QDA-Software

Quelle: QDASelect_EN_Installation Manual V1.0 (2005): 5

QDA besteht aus 2 Applikationen:

- **QDAUpload** und
- **QDASelect**

QDAUpload ist auf jedem Produktionserfassungssystem installiert und validiert alle Q-Daten auf Abweichungen zum QDAS[®] Datenformat. Im zweiten Schritt werden sämtliche Q-Daten, die auf dem Produktionserfassungssystem-Rechner liegen, auf einen zentralen Fileserver hochgeladen und gleichzeitig in eine Datenbank indexiert (erfasst).

QDASelect ist ein Auswertetool f r alle Mitarbeiter in der TKA-Presta, die Qualit tsanalysen (Qualit tspr fungen, Prozessverbesserungen, usw.) mit den Q-Daten durchf hren (z.B. Q-Verantwortliche, Prozesstechniker, Steuerungstechniker, Qualit tsleiter, Qualit tspr fplaner bis hin zu den Maschinenbedienern, also 100 % der Belegschaft).

QDASelect bietet eine Filter-/Abfragefunktion, mit der die Q-Daten sehr einfach gesucht und für die Analyse aufbereitet werden können.

Beispielhafte Abfrage: „zeige mir alle Q-Daten der Schicht z an der Maschine y vom Typ x vom Auftrag w, die NIO waren“ oder „zeige mir alle Q-Daten des Typs x des letzten Jahres“.

QDASelect findet die zu den Filterkriterien passenden Q-Daten auf dem Server. Somit kann ein einfaches und schnelles Auffinden von Q-Daten gewährleistet werden.

Dieses Projekt soll beispielhaft für alle anderen Softwareprojekte bei der TKA-Presta stehen. Es wurde nicht ausgewählt, weil hier eine überdurchschnittliche Fehlerhäufung vorliegt, vielmehr ist es durch den definierten Einsatzbereich, Funktionsumfang und bestimmten Betrachtungszeitraum einfacher eingrenz und somit kalkulierbar.

14.2 Erfassung der Fehlerkosten des QDA-Projekts

In diesem Abschnitt sollen die Kosten aufgezeigt werden, die durch mangelnde Qualität bei der Umsetzung des Softwareprojektes QDA bei der TKA-Presta entstanden sind. Dabei wird der objektiv eingetretene Schaden ermittelt, also nicht explizit aus Sicht der TKA-Presta oder des Softwarelieferanten. Ziel ist es, jene Kosten zu eruieren, die durch von der TKA-Presta aufgezeigten Produktfehler entstanden sind.

14.3 Untersuchung der Fehler

Da die Projektumsetzung wie in vielen Softwareprojekten von einigen Rückschlägen und Fehlerkorrekturen gekennzeichnet ist, möchte ich durch die Aufstellung der verursachten Schäden Argumente schaffen, um

mit dem Softwarelieferanten einige anstehende Qualitätsmaßnahmen umzusetzen.

Ein Auszug aus der Auflistung, der von mir an den Lieferanten gemeldeten Fehler, ist in Tab. 1 dargestellt. Diese Fehler sind also nach Auslieferung durch den Lieferanten aufgetreten und sind den externen Fehlerkosten hinzuzurechnen.

QDA Probleme bei TKA		
Datum:	Problem	Lösung
08.03.2005	M2673 Hierarchie, es existieren PC und PC Verzeichnisse unter Mxx	wird in der nächsten Beta umgesetzt QDAUpload wird auf der Anlage deaktiviert, und die Q-Daten per Batchfile gehandelt.
10.03.2005	M2575: QDAUpload handelt die Daten nicht aufs Q. Netzwerkprobleme?	
...	...	...

Tab. 1: QDA Fehler bei TKA-Presta

Quelle: Auszug aus AUF_QDA_Errors_RueMar_200308.xls [eigene Auflistung]

Die Fehleraufzeichnung seitens TKA-Presta führe ich pro Eintrag mit Datum und Fehlerbeschreibung. Wenn nach eigener Abschätzung, der Fehler nicht kritisch ist und damit nicht sofort behoben werden muss, wird der Fehler zusätzlich beim Lieferant in ein Logbuch eingetragen und dort mit einer Prioritätszahl, sowie „implementieren in Version“ klassifiziert. Kritische Fehler werden beim Lieferanten im Supportcenter erfasst und so schnell wie möglich abgearbeitet (ASAP: as soon as possible).

Bei der Auflistung sind jene Fehler/Anforderungen ausgenommen, die eine neue Funktion in der Software bilden würden. Die Trennung zwischen „neuer Anforderung“ und einer Anforderung, die schon immer bestand, welche aber bei der Projektentwurfsphase nicht berücksichtigt wurde (Mangels Kenntnis aus unvollständiger Kundenbefragung?), wird hier nicht durchgeführt und nicht als Fehler gewertet.

CRID (AMS intern)	Lieferant	Produkt	Version	Anforderung	Kategorie	Beschreibung	Zu Implem- ent. in Versio	Bewert- ung	Pri- o	Bearbeitun- g durch	Wann	Status	Erfasst durch	Erfass Datum
	AMS	QDA Select	101Build01 1alpha	GUI	qs-STAT	Folgende Änderungen sollen bzgl. Der Benutzerführung in qs-STAT erfolgen: - Merkmalsauswahldialog bei Report-Aufruf ausblenden - "Speichern unter"-Dialog bei Uebergabe eines neuen Files ausblenden. - Windows-Anmeldung muss funktionieren (funktioniert im Normalmodus jedoch nicht bei Anmeldung über Interface)	V1.01bet a3	K	1	AMS	V1.01	O	RueMar	16.03.2005
	AMS	QDA Select	101Build01 1alpha	GUI	COM-Interface	qs-STAT bleibt auch dann geöffnet, wenn QDASelect beendet wird damit ein Verlust von Änderungen/erstellten Berichten vermieden wird.	V1.01bet a3	K	1	AMS	V1.01	O	RueMar	16.03.2005
	AMS	QDA Select	101Build01 1alpha	GUI	Files	MaxTempQsStatFileAge-Property erweitern damit die Möglichkeit besteht sofort nach erfolgreicher Übergabe der Daten an qs-STAT, alle temporären Dateien zu löschen (bis auf die aktuelle).	V1.02	F	1	AMS	V1.02	O	RueMar	16.03.2005
	AMS	QDA Select	101Build01 1alpha	GUI	Files	Dialog: Links und Rechts Bündigkeit der Daten --> weiter Einrücken, damit mehr Abstand zum Rand entsteht, damit wird die Lesbarkeit verbessert. Z.B. beim Event-Auswahl Dialog	V1.02	F	1	AMS	V1.02	O	RueMar	16.03.2005

Abb. 34: QDA Fehler Protokoll beim Lieferanten

Quelle: Auszug aus AUF_AMS_QDA-DevelopmentSheet.xls (Zugriff am 17.03.2005)

Klassifizierung der Fehler:

Alle aufgetretenen Fehler müssen klassifiziert werden, wegen:

- der unterschiedlichen Schwere der Fehler
- der Häufigkeit des Auftretens
- der Anzahl der Fehler, die in die gleiche Kategorie eingeteilt werden können
- dem Ort des Auftretens (QDASelect, QDAUpload, usw.)

Tab. 2 stellt eine Zusammenfassung und Klassifizierung aller aufgezeichneten Fehler von AUF_QDA_Errors_RueMar_200308.xls (vgl. Auszug in Tab.1) dar.

Zeitraum der Untersuchung 2003.07 - 2005.03

Fehlerart	Softwareteil	Anzahl der Fehler	Häufigkeit %	Fehlerart	Häufigkeit der Auswirkung
Spezifikationsfehler, Pflichtenheftfehler	QDA	5	3%	K	10
Abweichungsfehler von der Spezifikation	QDA	6	4%	K	10
Codierungsfehler	QDA	12	8%	K	4
Installationsprobleme	QDA	6	4%		2
Kopierfunktion	QDAUpload	9	6%	K	2
Validierungsfehler	QDAUpload	14	10%	K	9
Service	QDAUpload	11	8%	H	1
Datenbankprobleme	QDAUpload	10	7%	K	5
Unkorrekte Q-Daten	QDAUpload	8	6%		1
Grafische Oberfläche	QDASelect	42	29%		8
Funktionen der Dialoge	QDASelect	13	9%	H	1
Falsche Ergebnisse	QDASelect	8	6%	K	1
Summe	QDA	29	20%		
Summe	QDAUpload	52	36%		
Summe	QDASelect	63	44%		
Summe gesamt		144			
Summe Kritischer Fehler		64			

Tab. 2: QDA Fehlerklassifizierung bei TKA-Presta

Quelle: [eigene Auflistung]

Legende:

Softwareteil: QDA, davon sind QDASelect und QDAUpload betroffen

Anzahl der Fehler: Wieviele (aus meiner Sicht) unterschiedliche Fehler sind aufgetreten

Fehlerart:

K = Kritisch, Software kann nicht eingesetzt werden, sofortige

Reaktion notwendig

H = Hoch, Software kann nur eingeschränkt betrieben werden

Häufigkeit der Auswirkung: Wie oft tritt die Fehlerart beim Einsatz bei einer Skala von 1-10 auf, z.B. wie oft bleibt der Service während dem Betrieb stehen

→ 1 = ganz selten (dies kann auch Einfluss auf die Reproduktion eines Fehlers haben, da es mehr Zeit kostet sich Tests zu überlegen, damit die Häufigkeit zu nimmt)

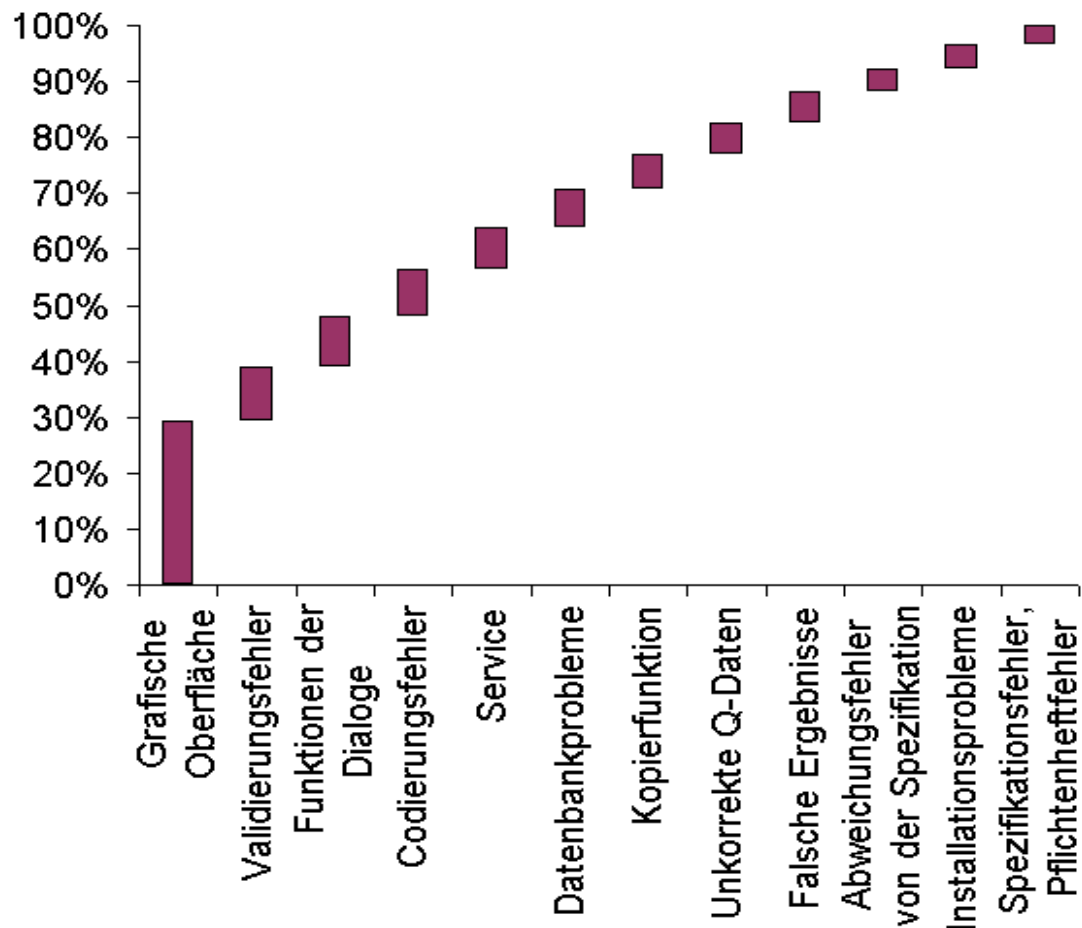


Abb. 35: Paretodiagramm der QDA-Fehleranzahl

Quelle: [eigene Auflistung]

Aus dieser Aufstellung geht hervor, dass die meisten Fehler an der grafischen Oberfläche auftraten und beim Software-Lieferant beanstandet wurden. Dies sagt aber noch nichts über die Kostenbedeutung dieser Fehler aus, deshalb müssen diese Fehler mittels Kostensätzen quantifiziert werden.

14.4 Aufstellung der Fehlerkosten

Zeitraum der Untersuchung 2003.07 - 2005.03

Kosten der TKA-Presta

Die Fehler die bei der Installation und im Betrieb der QDA-Software bei TKA-Presta auftreten müssen von mir analysiert werden (qualitätsbezogene Fehlerprüfkosten). Wenn eine Fehlfunktion des Programms vorliegt, leite ich eine Fehlerbeschreibung an den Software-Lieferanten weiter. Diese Aufwendungen sind mit Kosten verbunden und können folgendermaßen eingeteilt werden:

Unkritischer und Kritischer Fehler

Bei allen Fehlern fallen folgende Tätigkeiten an:

- Analyse des Fehlers
- Beurteilen der Schwere des Fehlers

Die Tätigkeit „Analyse des Fehlers“ beinhaltet die Einordnung des Fehlers, durch eine kurze Analyse, z.B. ob der Fehler wegen falscher Bedienung, wegen der Konfiguration oder wegen des Programms entsteht. Es werden nur Aufwände für Programmfehler berücksichtigt, da die anderen Reklamationen direkt behoben werden können. Fehler aufgrund der Bedienung können als Verbesserungspotenzial der Usability betrachtet werden. Sie werden noch nicht aufgezeichnet. Diese Tätigkeiten können den direkten externen Fehlerkosten zugerechnet werden.

Durchzuführende Tätigkeiten bei einem **unkritischen Fehler**:

- Behebung der Ursache und
- Eintrag in der Fehlerliste.

Durchzuführende Tätigkeiten bei einem **kritischen Fehler**

- Weiterleiten der Fehlerbeschreibung an den Support des Lieferanten.
- Fehlerursache finden,
- Erarbeitung einer Lösung mit dem Lieferanten,
- Einteilung des Fehlers (Programmfehler, Spezifikationsfehler, etc)
- Eintrag in AUF_AMS_QDA-DevelopmentSheet.xls (vgl. Abb. 35)
- Regelung der Kostenverantwortung,
- Terminabstimmung der Fehlerbehebung,
- Prüfung des behobenen Fehlers,
- Rolloutplanung,
- Installation/Update der Software bei der TKA-Presta,
- Information der Anwender über Systemkorrektur (Ausfall des Systems wegen Wartung).

Die Tätigkeit „Regelung der Kostenverantwortung“ muss näher erklärt werden. Darunter verstehe ich die Verhandlung mit dem Lieferanten darüber, in welcher Version der Fehler behoben wird. Z.B. bedeutet der Eintrag in Abb. 34 in Spalte „Wann“ bei einer Versionserhöhung, dass dieser Punkt in der nächsten Angebotsstellung durch den Lieferanten als zusätzliche Anforderung verrechnet wird. Zwischenversionen des Produktes, wie z.B. „Alpha“, „Beta“ werden nicht von der TKA-Presta bezahlt und stellen somit Kosten für den Lieferanten dar.

Zur Vereinfachung werden bei allen Tätigkeiten der TKA-Presta nur die Personalbelegungszeiten berücksichtigt. Telefonkosten, Computerkosten, etc. berücksichtige ich hier nicht.

Tätigkeit	Zeitaufwand	Kostensatz/h	Fehleranzahl	Kosten
Fehlerprotokollierung	0,5h	EUR 90	144	6.480
Fehleranalyse	0,5h	EUR 90	144	6.480
Fehlerdokumentation/ Meldung Support	1h	EUR 90	144	12.960
Summe	2h			EUR 25.920
Kosten / Fehler				EUR 180

Tab. 3: QDA Fehlerkosten bei TKA-Presta

Quelle: [eigene Auflistung]

Nach der Behebung des Fehlers (Nacharbeit, Fehlerkorrektur) durch den Software-Lieferanten fallen noch zusätzliche Kosten an, die wie folgt beschrieben werden können.

Bei kritischen Fehlern fallen nach der Auslieferung einer neuen Version, durch den Software-Lieferanten, Kosten für TKA-Presta interne Tests und Installationskosten an (bis 2005.04 wurden 8 verschiedene Rollouts (Installationen) der QDA-Software am Standort Eschen der TKA-Presta durchgeführt) (vgl. Tab. 4). Ziel des Abnahme-Tests ist es, das Programm vor allem in jenen Funktionen auf Korrektheit zu prüfen, in dem der kritische Fehler aufgetreten ist.

Interner Test	
Zeitaufwand	2h
Kostensatz/h	EUR 90
Fehleranzahl	64
Kosten	EUR 11.520
Installation SW (8 Versionen)	
Zeitaufwand/Rollout	8h
Kostensatz/h	EUR 90
Anzahl Update	8
Kosten	EUR 6.300
Gesamtkosten TKA-Presta	EUR 43.740
Kosten / unkritischer Fehler	EUR 180
Kosten / kritischer Fehler	EUR 458

Tab. 4: QDA Rollout Aufwände bei TKA-Presta

Quelle: [eigene Auflistung]

Aufwand des Lieferanten

Der Lieferant erfasst die Zeiten der Programmierer für die Behebung der externen Fehler. In der Kostenstellenrechnung wird für die Behebung externer Fehler von QDA 510h ausgewiesen. Es wird keine Unterscheidung kritischer/unkritischer Fehler gemacht, auch nicht der Fehlerart. Deshalb ist es nicht möglich für die einzelnen Fehler explizit einen Kostensatz anzugeben. Die Beurteilung des Verbesserungspotenzials eines Fehlers muss über seine Anzahl und kritisch/unkritisch Klassifizierung erfolgen. Ein unkritischer Fehler wird ausschließlich über den Support abgewickelt, die Testarbeit übernimmt der Programmierer selber. Als Zeitaufwand für einen kritischen oder unkritischen Fehler wird 1h veranschlagt. Bei einem kritischen Fehler benötigt die Testabteilung pro Fehler 2h Testzeit.

Programmieraufwände	
Zeitaufwand	510h
Kostensatz/h	EUR 90
Kosten	EUR 45.900

Support	
Zeitaufwand	1h
Kostensatz/h	EUR 90
Fehleranzahl	144
Kosten	EUR 12.960

Testcenter	
Zeitaufwand	1h
Kostensatz/h	EUR 90
Fehleranzahl	64
Kosten	EUR 5.760
Gesamtkosten Lieferant	EUR 64.620
Kosten / Fehler	EUR 448

Tab. 5: interne Kosten des Lieferanten bei externen Fehlern

Quelle: [eigene Auflistung]

Aufgrund mangelnder Qualität ergeben sich die Gesamtfehlerkosten des QDA-Projekts, aus den Fehlerkosten der TKA-Presta und den internen Fehlerkosten des Lieferanten. Die Gesamtfehlerkosten für den Zeitraum vom 2003.07 bis 2005.03 betragen **EUR 108.360**.

Ein Vergleich der angefallenen Gesamtfehlerkosten mit den Gesamtkosten für die Software drückt die Kosten besser aus:

Vergleich der Fehlerkosten mit den Softwarekosten	
Softwarekosten (Entwicklungsbeitrag von 35%)	EUR 95.000
Gesamterstellungskosten Software	EUR 271.429
Gesamtfehlerkosten extern	EUR 108.360
Anteil Gesamtfehlerkosten an Gesamterstellungskosten	39,9%
Fehlerkosten TKA-Presta	EUR 43.740
Anteil Fehlerkosten TKA-Presta an Gesamterstellungskosten	16,1%
Anteil Fehlerkosten TKA-Presta an Gesamtfehlerkosten	40,4%
Fehlerkosten Lieferant	EUR 64.620
Anteil Fehlerkosten Lieferant an Gesamterstellungskosten	23,8%
Anteil Fehlerkosten Lieferant an Gesamtfehlerkosten	59,6%
Gesamtkosten / kritischer Fehler	EUR 71.820
Gesamtkosten / unkritischer Fehler	EUR 36.540

Tab. 6: Anteil der Fehlerkosten an Softwaregesamtkosten

Quelle: [eigene Auflistung]

TKA-Presta zahlt für das QDA-Projekt einen Entwicklungsbeitrag von 35 %. Somit ergibt sich für die Gesamterstellungskosten für die QDA-Software ein Betrag von EUR 271.429. Die Kosten die dem Lieferanten aus Fehlern entstanden sind betragen 23,8 % von seinen Gesamterlösen aus dem QDA-Projekt. Die Tab. 6 zeigt deutlich, dass TKA-Presta 40,4 % der Gesamtfehlerkosten tragen muss, oder anders ausgedrückt, der Lieferant hat 16,1 % der Gesamtfehlerkosten an den Gesamterstellungskosten der Software auf den Kunden abgewälzt.

Die Gesamtfehlerkosten entsprechen bei einem Stundensatz von EUR 90/h und 168h/MM mehr als 7MM.

Fehlerkosten in Mannmonaten (MM)	
Gesamtfehlerkosten extern	108.360
Kostensatz/h	90
h/MM	168
Gesamtfehlerkosten in MM	7,17

Tab. 7: Fehlerkosten in MM

Quelle: [eigene Auflistung]

14.5 Notwendigkeit der Durchführung von Qualitätsmaßnahmen

Ausgehend von den berechneten Gesamtfehlerkosten aus Kapitel 14.4 des QDA-Projekts sollen nun Qualitätsmaßnahmen vorgeschlagen werden um zukünftig die Softwarequalität zu verbessern. Die Gesamtfehlerkosten stellen eine Verschwendung und Ineffizienz dar. Sie können als Potenzial aufgefasst werden um Qualitätsmaßnahmen durchzuführen. Dieses Potenzial kann auch als Nutzen (Opportunitätserlös) gesehen werden, dann nämlich, wenn die Durchführung von Qualitätsmaßnahmen bewirkt, dass Fehlerkosten zukünftig reduziert und verringert werden können.

Wie bereits erwähnt, wäre es aber unökonomisch von der „Null-Fehler-Prinzip“-Forderung auszugehen 100 % fehlerfreie Software zu erstellen. Wie in Kapitel 5.5 nachgewiesen sind jene Fehler, die erst beim Kunden auftreten, am teuersten. Deshalb müssen Maßnahmen definiert werden, um die externen Fehlerkosten drastisch zu senken.

Da die QDA-Software schon im Einsatz ist, werden nur mehr solche Maßnahmen Auswirkungen haben, die durch die Erweiterung der Software auch greifen. Alle vorgeschlagenen Maßnahmen betreffen den gesamten Software-Erstellungsprozess, damit die Tätigkeiten besser abgestimmt sind und bei weiteren Projekten positiven Einfluss auf die Softwarequalität und somit einen Reduktionseffekt der Qualitätskosten haben. Die hier nun vorgeschlagenen Qualitätsmaßnahmen müssen im

Sinne des modernen Qualitätsmanagements als Investition gesehen werden und nicht als zusätzliche Kosten. Deshalb erachte ich es als besonders wichtig, die handelnden Personen des Software-Lieferanten, für die Notwendigkeit der Umsetzung qualitätsverbessernder Maßnahmen zu gewinnen.

14.6 Vorschläge und Kostenabschätzung für Qualitätsmaßnahmen im QDA-Projekt

Kerngedanke:

„Der Lieferant ist verpflichtet ein Produkt zurück zuhalten, bis:

- die geforderten Prüfungen abgeschlossen sind und
- die Ergebnisse die Qualitätsanforderungen erfüllen sowie
- die zugehörigen Daten und Dokumente verfügbar und genehmigt sind“

(DIN EN ISO 9001:2000. Dez. 2000).

Betrachtet man die Kosten für die Installation, den Rollout einer Version wird schnell klar, dass das Ziel sein muss, Software erst dann auszuliefern, wenn alle Tests erfolgreich abgeschlossen wurden. Das heißt eine Woche Verzögerung kann dann akzeptiert werden, wenn dadurch Fehler reduziert werden.

14.6.1 Allgemeine Forderungen

a) Einführung eines Vorgehensmodells zur Softwareentwicklung

Der Softwareerstellungsprozess soll sich wenn möglich an dem neuen V-XT Vorgehensmodell V1.0.3 orientieren. Darin werden die Prozesse und Meilensteine für die Softwareerstellung beschrieben. Die Dokumentvorlagen sind, wenn nicht anders von der TKA-Presta gefordert, zu verwenden.

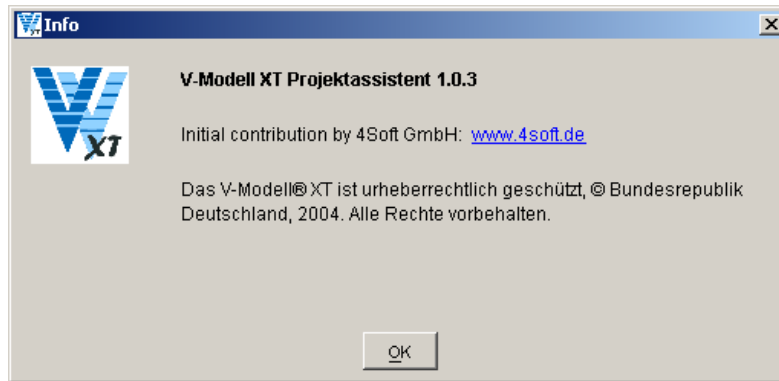


Abb. 36: Neue V-Modell XT von der deutschen Bundesregierung

Quelle: Infoscreen des V-Modell XT Projektassistenten V1.03. [eigene Abbildung]

Die Verwendung des V-XT Modells wird allen Softwarelieferanten empfohlen, damit eine vereinfachte, schnellere und kostengünstigere Projektabwicklung möglich ist. Die anfallenden Kosten zur Implementierung des V-XT Modells werden bei der Kostenbewertung des QDA-Projekts nicht berücksichtigt, da diese Maßnahme sehr langfristig gesehen werden muss und alle Softwareprojekte davon betroffen sind. Untersuchungen in der Praxis haben gezeigt, dass bei der Komplexität der heutigen Softwareentwicklung, ohne ein entsprechendes Vorgehensmodell keine effiziente Projektabwicklung mehr möglich ist.

b) Maßnahmen für die Prüfung

Info: Hauptbetroffen von diesen Maßnahmen ist die **Testabteilung**.

1) Erstellen einer Verfahrensanweisung für die Prüfung:

Als erster Punkt muss eine Verfahrensanweisung, wie die Prüfung zu erfolgen hat, erstellt werden. Die Verfahrensanweisung muss folgende Punkte umfassen:

- Prüfspezifikationen und Prüfprozeduren müssen dokumentiert sein,
- Aufstellung der Prüfungsaktivitäten, die folgende Punkte beinhalten
 - o die Festlegung der Prüfkriterien für Dokumente, Prozesse oder Systemelemente,

- o die Prüfung von Dokumenten, Prozessen oder Systemelementen,
- o die Festlegung von Prüffällen sowie
- o die Erstellung der Nachweisakte.

Kosten für die einmalige Erstellung der Verfahrensanweisung: ca. 5MT

2) Führen einer Fehlerprotokollliste:

Die Fehlerliste wird von der Testabteilung des Software-Lieferanten bereits seit geraumer Zeit geführt. Es muss nur mehr der „online“ Zugriff für den Kunden über eine Webapplikation möglich sein.

Kosten: EUR 0. Es ergibt sich kein Aufwand, da die Liste bereits geführt wird. Die Liste muss in das Online-Portal des Lieferanten eingepflegt werden.

3) Systematische Testaktivität:

Jeder Softwaretest muss von der Testabteilung in einem Testprotokoll dokumentiert werden. Die Vorlage für das Testprotokoll bildet der Testplan.

Der Testplan muss folgende Angabe enthalten:

- Testplan ID
- Zu testende Komponenten
- Zu testende Funktionen
- Nicht zu testende Funktionen
- Vorgehensweise
- Pass / Fail Kriterien
- Produkte
- Testtätigkeiten
- Testumgebung, Zuständigkeiten, Personal, Zeitplan, Risiken und Risikomanagement , Genehmigungen

Das Testprotokoll ist vom Testleiter auszufüllen und bei der Lieferung der Software an den Kunden zu übergeben.

1. Projektdaten

Datum:	2005.03.20
Projektbezeichnung	QDA
Version:	
Testerverantwortlicher:	Xxx
Tester 1	Xxx
Tester x	Xxx

iO. n.i.O entfällt

☒ ☐ ☐ K-Felder (Anzahl, Länge, Inhalt, Datentyp) nach TV1831

☒ ☐ ☐ K1000 werden durch

...

Abb. 37: Beispielvorlage für Testprotokoll

Quelle: [eigene Abbildung]

Forderung an die Erstellung von Testspezifikationen:

In der Testspezifikation ist festgehalten, wie eine Softwarekomponente getestet werden soll.

Folgende Punkte müssen in der Testspezifikation enthalten sein:

- Testspezifikation ID
- Zu testende Funktionen
- Testverfahren
- Testfälle
- Pass / Fail Kriterien

QDAUpload V1.01beta15

Testplan 23: Q-Datenvalidierung

Prűfbeginn: xxxx

Prűfdauer: xxx

Nr	Testfall	Testdaten	erwartetes Ergebnis	Testergebnis
01	Reihenfolge K1xxx und K2xxx abwechselnd	K0100 K1002 K2022 K1086	Errormeldung: „Abweichung von Datenformat“	Errormeldung: „Abweichung von Datenformat“

...

Abb. 38: Beispielvorlage für Testspezifikation

Quelle: [eigene Abbildung]

Danach muss das Testergebnis in das Testprotokoll übertragen werden.

Aufwand: Erstellung der Vorlagen 1MT/Vorlage, Zeitaufwand zum Ausfüllen der Vorlagen: 1h/Test

5) Systemtest durch Vollzeit-Tester (existiert bereits beim Lieferant)

6) Test und Entwicklung sind getrennt (klarere Trennung der beiden Aktivitäten ist erforderlich)

7) Testwiederholung bei Softwareänderung (wird bereits gemacht)

Bei jeder Softwareänderung müssen jene Tests wiederholt werden, die zwingend vorgeschrieben sind.

8) Tests werden aufbewahrt (muss in der Verfahrensanweisung dokumentiert werden)

9) Standardisierte Tests plus angepasste Tests aufgrund der Sourcecode-Dokumentation.

Aufwand: 1MT/Test

10) Test-Dokumente werden inspiziert und dem Kunden bei Auslieferung der Software übergeben

11) Aufzeichnung der Testkosten

12) Messung der Codeüberdeckungsanzahl (**Aufwand pro Modul:** 0,5h)

Diese Forderungen werden auch in der Praxis als grundlegend angesehen, damit Softwarequalität überhaupt erstellt werden kann.

14.6.2 Forderung von Maßnahmen aufgrund der Fehlerkosten-Untersuchung des QDA-Projekts

a) Grafische Oberfläche

Abgeleitet aus dem kostenmäßig unbewerteten Paretodigramm (vgl. Abb. 35) fällt auf, dass 29 % aller externen Fehler bei der grafischen Oberfläche anfallen. Wie weiter oben festgestellt, sind Fehler in der grafischen Oberfläche meist unkritisch. Sie betreffen z.B. die Anordnung von Buttons, Rechtschreibfehler, „unübliches Verhalten“, falsche Sortierung von Listboxen, etc.

Kommentar:

Aus Kundensicht dürfen Fehler in dieser Häufung und Art nicht vorkommen. Diese müssen vom Programmierer und der Testabteilung des Lieferanten vor Auslieferung an den Kunden behoben sein. Durch das Einführen von spezifizierten Tests sollten diese Fehler stark reduziert werden können.

Ziel:

Das Ziel ist die Reduktion der Fehler an der grafischen Oberfläche um 80 % von der aktuellen Fehleranzahl.

Maßnahmen:

Für den unter Kapitel 14.6.1 Allgemeinen Forderungen aufgeführten Testablauf müssen Testpläne erstellt werden, in denen die einzelnen Dialogelemente auf Handling, Aussehen, Funktion und Sprache getestet werden.

Das Aussehen der Oberfläche muss mit dem Kunden eingehender spezifiziert werden. Dies kann am Effektivsten mit Prototypen erreicht

werden. Da der Kunde meistens auch noch keine konkrete Vorstellung vom Produkt besitzt, ist es wichtig möglichst früh Anschauungs-Prototypen zur Verfügung zu haben

Deshalb die Forderung, dass dem Kunden gratis ein erster Prototyp vor dem Review 3, nach dem V-XT Vorgehensmodells, durch den Lieferanten zur Verfügung stehen soll.

Durch die Prototypen kann folgendes spezifiziert werden:

Aufteilung der Bildschirmoberfläche

- Aussehen der Dialoge
- Dialogabhängigkeiten
- Dialoginhalte
- Interaktion mit Benutzer
- Eingabeformate
- Eingabe Werte, usw.

Messung:

Die Messung erfolgt anhand der Spezifikation nach der Phase 3 Systemspezifikation (Produktentwurf) des V-XT Vorgehensmodells)

Aufwand:

Der Testaufwand wird pro Dialog auf ca. 2h geschätzt.

b) Fehler in der Validierungsfunktion der Q-Daten

Ebenfalls ein gewichtiger Punkt, welcher hohe Fehlerkosten verursacht sind Validierungsfehler. Damit sind Fehler gemeint, die im QDAUpload auftreten und das Überprüfen der Q-Daten betreffen.

Dabei wurden folgende Ursachen gefunden:

Erfassungssystem schreibt die Q-Daten, QDA meldet beim Validieren Fehler oder „bleibt hängen“. Einerseits, weil das Erfassungssystem sehr breit konfiguriert werden kann andererseits, weil in der Praxis mehr Fälle auftreten als ursprünglich angenommen.

Kommentar:

Es drängt sich die Frage auf, da das Erfassungssystem vom gleichen Lieferanten stammt, dass die Funktionslogik wie Q-Daten geschrieben werden, hundertprozentig bekannt sein sollte, anderenfalls wären die Tests beim Lieferanten unvollständig, da sie nicht alle Fälle abdecken würden. Da das Validieren eine sehr zentrale Funktion vom QDAUpload darstellt ist die Fehleranzahl deutlich zu verringern. In der Praxis hat sich herausgestellt, dass einige Erfassungssysteme an den Produktionsmaschinen nicht hundertprozentig QDAS® Datenformat konforme Q-Daten schreibt. Die Validierungsfunktion hat den Sinn solche fehlerhaften Systeme einzugrenzen, damit die Q-Datenqualität gezielt gefördert werden kann. Q-Daten können im Falle einer Gewährleistung sehr wichtig sein, deshalb ist es wesentlich, dass Abweichungen vom QDAS® Datenformat früh erkannt werden.

Ziel:

Ziel ist die Reduktion der Fehler bei der Validierung um 90 %.

Maßnahmen (vgl. Liggesmeyer 2002: 474ff):

Analytische Maßnahmen:

Es wird angeregt, dass standardisierte Testpläne erstellt werden müssen, um die Struktur des QDAS® Datenformats abzudecken. Zu Testzwecken sollen möglichst viele unterschiedliche Q-Daten, von der TKA-Presta dem Lieferanten bereitgestellt werden.

Forderungen an Tests des Programmcodes:

Alle Module, werden mittels einer Code-Inspektion überprüft.

Alle geforderten Funktionen pro Modul sind mindestens einmal zu testen (Funktionsabdeckung).

Module müssen soweit getestet sein, dass sich eine Anweisungsüberdeckung (C0) von mindestens 95 % ergibt.

Konstruktive Maßnahmen:

Mechanismen für das Error-Handling sollen möglichst früh implementiert werden, denn dadurch können Fehlersuchzeiten entscheidend reduziert werden, wie sie bei der Nacharbeit von Q-Daten Validierungsfehlern aufgetreten sind.

Weiters muss ein besonderes Augenmerk darauf gelegt werden, dass testbare Module codiert werden. Dazu eignet sich am Besten eine defensive Programmierung. Untersuchungen zeigen, dass durch dieses Verhalten die Implementierungsphase entscheidend verkürzt werden kann.

Messung der Zielerreichung:

Anzahl der Fehler nach Auslieferung (ab Beta Versionen).

Aufwand:

Der Programmieraufwand sollte nur unwesentlich steigen. Er erfordert aber eine größere Aufmerksamkeit und Gewissenhaftigkeit.

c) Abweichungsfehler von der Spezifikation

Das Projekt hat einen Umfang von über 5500 Programmierstunden, trotzdem sind noch immer Abweichungen des Softwareverhaltens von der Spezifikation feststellbar, z.B. „Ausführen von Suchabfragen innerhalb x Sekunden“.

Kommentar:

Diese Fehler können vom Lieferanten durch konsequente Ausrichtung auf die Kundenforderungen minimiert werden bzw. bei ständiger Kommunikation mit dem Kunden eine andere technische Lösung gefunden werden. Das Augenmerk und Bewusstsein des Lieferanten muss auf die Erfüllung der Kundenanforderungen gelegt werden bzw. Maßnahmen bei Zielabweichung eingeleitet werden (z.B. Review, etc).

Ziel:

Das Ziel ist die Reduktion der Abweichungsfehler um 40 %.

Maßnahmen:

Konstruktive Maßnahmen:

Bei der Erstellung des Pflichtenheftes muss eine engere Abstimmung zwischen Kunden und Lieferant erfolgen, damit die Wichtigkeit und zeitliche Priorisierung der Qualitätsmerkmale in einem gemeinsamen entwickelten Systementwurf eindeutig festgelegt werden. Einführung von wöchentlichen Design-Reviews nach Projektkickoff, bei der gemeinsam der Systementwurf erarbeitet und validiert wird.

Statistische Analysen von Projekten ergeben, dass mit Reviews und Inspektionen ca. 60 bis 70 % der Fehler im Prüfobjekt gefunden werden und Entwicklungs-Fehlerkosten um bis zu 75 % gesenkt werden können. Das führt zu 20 % Nettoeinsparungen über den gesamten Softwarelebenszyklus (vgl. Liggesmeyer 2002: 472ff).

Analytische Maßnahmen:

Es müssen Testpläne erstellt werden, die jede Anforderung aus dem Pflichtenheft abdecken. Danach ist zu prüfen ob das Objekt mit der Spezifikation übereinstimmt, vollständig ist und die Funktionen korrekt implementiert wurden. Überfüllung ist ebenso wenig sinnvoll wie das Fehlen von Anforderungen, deshalb ist zu prüfen, dass in dem SW-Produkt auch nur die spezifizierten und geforderten Funktionen enthalten sind (vgl. Wallmüller 1998: 155).

Messung der Zielerreichung:

Anzahl der Abweichungsfehler nach Auslieferung (ab Beta Versionen) aufgrund des Pflichtenheftes.

Aufwand:

Der Reviewaufwand für Dokumente kann bis zu 20 % des Erstellungsaufwandes der Software betragen. Der Reviewaufwand um Code zu verifizieren wird mit bis zu 15 % des Erstellungsaufwandes der Software angegeben.

14.6.3 Wirtschaftlichkeit der Maßnahmen

Nun ist der Aufwand für die Qualitätsmaßnahmen dem Nutzen durch Einsparung gegenüber zu stellen, um die Wirtschaftlichkeit der Maßnahmen zu prüfen.

Der Aufwand für die geforderten Maßnahmen wurde bereits angeführt. Die Maßnahmen werden in Tab.8 zusammengefasst und monetär bewertet.

Aufwand durch Qualitätsmaßnahmen

Maßnahme	Aufwand(Zeit)	Aufwand (Kosten) EUR
Einführung V-XT Vorgehensmodell		
Erstellung Verfahrensanweisung Prüfung	5 MT	
Fehlerprotokollliste führen		0
Erstellung Testplan/spezifikationvorlage (ca. 15 Tests á 1MT)	15MT	10.800
Ausfüllen des Testplans (15 Tests)	2MT	1.440
Angepasste Testdurchführung (15 Tests)	15MT	10.800
Messung Codeüberdeckungszahl (C0)	1MT	720
Grafik: Prüfen Dialoge (20 Dialoge á 2h)	5MT	3.600
Validierung: SourceCode		0
Reviewaufwand für Dokumente (kann nur geschätzt werden)	10MT	7.200
Summe	53MT	34.560
Kostensatz/h		90
h/MT (MT=Manntage)	8	

Tab. 8: Kosten der Qualitätsmaßnahmen

Quelle: [eigene Auflistung]

Einsparungen (Erlöse) aus den Qualitätsmaßnahmen:

Maßnahme	Einsparung	Anzahl Fehler vor Durchführung der Maßnahmen	Anzahl Fehler nach Durchführung der Maßnahmen	Fehlerart	Kosten/Fehler [EUR]	Kosten vor Durchführung [EUR]	Kosten nach Durchführung [EUR]
Grafische Oberfläche	-80%	42	8		628	26.376	5.275
Validierungsfehler	-90%	14	1	K	906	12.684	1.268
Funktionen der Dialoge	-80%	13	3		628	8.164	1.633
Spezifikationsfehler, Pflichtenheftfehler	-40%	5	3	K	906	4.530	2.718
Abweichungsfehler von der Spezifikation	-40%	6	4	K	906	5.436	3.262
Summe		80	19			57.190	14.156
Erlöspotenzial (Kosten vor Durchführung - Kosten nach Durchführung)							43.034

Tab. 9: Erlöspotenzial aus den Qualitätsmaßnahmen im QDA-Projekt

Quelle: [eigene Auflistung]

Die Einsparungen durch die Qualitätsmaßnahmen aus Tab. 9 können als Nutzen aufgefasst werden (Opportunitätserlöse, wegen geringerer Fehlerkosten).

Um die Wirtschaftlichkeit zu bestimmen ist die Differenz aus

$$\begin{aligned} & \text{Erlöse aus Tab. 9 (Einsparungen)} \\ & \underline{\% \quad \text{Kosten aus Tab. 8 (Kosten der Durchführung)}} \\ & = \quad \text{Gewinn} \\ & \text{zu berechnen.} \end{aligned}$$

Damit kann bei Durchführung der Qualitätsmaßnahmen mit einem Kostenvorteil (Opportunitätsgewinn) von **EUR 8.474** erzielt werden.

Wirtschaftlichkeit der Qualitätsmaßnahmen	EUR
Einsparungen aus Qualitätsmaßnahmen	43.034
Kosten der Qualitätsmaßnahmen	34.560
Opportunitäts Gewinn	8.474

Tab. 10: opportunitäts Gewinn durch Qualitätsmaßnahmen im QDA-Projekt

Quelle: [eigene Auflistung]

14.6.4 Fazit

Durch den resultierenden positiven Gewinn ist die Durchführung der Qualitätsmaßnahmen als Sinnvoll zu erachten. Es ist zu Beachten, dass durch die Einführung der Maßnahmen beim Lieferanten auch die anderen aufgelisteten Fehlern reduziert werden können, diese sind in der Einsparungsrechnung noch nicht berücksichtigt. Ich leite deshalb ab, dass bei konsequenter Durchführung der Qualitätsmaßnahmen der oben angeführte mindestens erreicht werden kann.

15 Zusammenfassung

Die Arbeit verfolgte als erstes Ziel die Begrifflichkeit der Softwarequalität anhand der Literatur darzustellen. Erkenntnis war, dass sich die Auffassung von Qualität in der recht jungen Softwarewelt erst allmählich vereinheitlichen muss.

Nach dem die Qualität durch Operationalisierung fassbarer gemacht wurde, wurden Maßnahmen und Methoden erläutert, um „gute“ Softwarequalität zu schaffen. Dabei ist zu betonen, dass neben den Rahmenbedingungen vor allem ein definiertes Vorgehensmodell Softwarequalität erst ökonomisch möglich macht.

Um Abzuschätzen, ob Qualitätsmaßnahmen ökonomisch sein können, wurde in der Arbeit der aktuelle Stand der Wirtschaftlichkeitsanalyse anhand der Literatur erläutert. Nach der theoretischen Darstellung wurde eine Wirtschaftlichkeitsanalyse des QDA-Projekts bei der TKA-Presta vorgenommen. Dabei konnte erfolgreich gezeigt werden, dass ausgesuchte Qualitätmaßnahmen sehr wohl kostensenkende Auswirkungen haben.

Meine persönliche Erfolgskriterienliste um Softwarequalität zu erreichen:

„Qualität muss geschaffen werden, sie kann nicht hineingeprüft werden“
(Unbekannt)

- 1) Endanwender müssen in der Entwurfs- u. Designphase eng eingebunden werden
- 2) Unterstützung durch die Mitarbeiter
- 3) Zu Beginn müssen die Anforderungen eindeutig festgelegt werden, trotzdem sollen sie z.B. bei Reviews immer wieder kritisch hinterfragt werden.
- 4) Einstellung zu Fehlern: Alles beim ersten Mal richtig machen. Wenn Fehler aufgetreten sind, werden sie sofort behoben.
- 5) Kompetente Mitarbeiter
- 6) Konstruktive Qualitätsmaßnahmen kommen vor analytischen Qualitätsmaßnahmen
- 7) Ein Test muss destruktiv sein, somit kann der Programmierer nicht der Testleiter sein.
- 8) Qualität kommt vor dem Terminplan.

Literaturverzeichnis

Albach, H. (1988): „Kosten, Transaktionen und externe Effekte im betrieblichen Rechnungswesen.“ In: Zeitschrift für Betriebswirtschaft, 58. Jg. (1988), Nr. 11, S. 1143-1170.

Balzert, H. (1998): Lehrbuch der Software-Technik. Software-Management. Software-Qualitätssicherung. Unternehmensmodellierung. Heidelberg; Berlin: Spektrum.

Bartsch-Beuerlein, S. (2000): Qualitätsmanagement in IT-Projekten. Planung, Organisation, Umsetzung. München: Hanser.

Basili, V. R.; D. M. Weiss (1984): "A Methodology for Collecting Valid Software Engineering Data." In: IEEE Transactions on Software Engineering. Nr. 6 (1984), S. 728-738.

Bellin, G.; D. Stelzer (1995): Softwarequalitätsmanagement gemäß ISO 9000. Ergebnisse einer empirischen Untersuchung zertifizierter Qualitätsmanagementsysteme. Studien zur Systementwicklung des Lehrstuhls für Wirtschaftsinformatik der Universität zu Köln. Bd. 7. Köln: Uni Köln

Boehm, B. W. (1976): "Software-Engineering." In: IEEE. Transactions on Computers, Vol. C-25.

Bröhl, A. P.; W. Dröschel (1993): Das V-Modell. Der Standard für die Softwareentwicklung mit Praxisleitfaden. München: Oldenbourg.

Bruhn, M. (1998): Wirtschaftlichkeit des Qualitätsmanagements. Berlin: Springer.

Bruhn, M.; D. Georgi (1999): Kosten und Nutzen des Qualitätsmanagements. Grundlagen Methoden, Fallbeispiele. München: Hanser.

Bundschuh, M.; A. Fabry (2004): Aufwandschätzung von IT-Projekten. 2. überarb. u. erw. Aufl. Himmer; Augsburg: mitp.

Crosby, P. B. (1980): Quality is Free. The Art of Making Quality Certain. New York: McGraw -Hill.

DeMarco, T. (1982): Controlling Software Projects. New York: Yourdon Press.

DeMarco, T. (1997): Warum ist Software so teuer? Und andere Rätsel des Informationszeitalters. München; Wien: Hanser.

DeMarco, T. (2004): Was man nicht messen kann, kann man nicht kontrollieren. 2. Aufl. mitp.

Deutsche Gesellschaft für Qualität e. V. (DGQ) (Hrsg.) (1995): Begriffe zum Qualitätsmanagement, DGQ-Schrift, Nr. 11-04, 6. Aufl. Frankfurt am Main.

DIN, EN, ISO (Hrsg.) (1995) : Qualitätsmanagement. Begriffe. DIN EN ISO 8402. Berlin: Normenausschuss Qualitätsmanagement, Statistik und Zertifizierungsgrundlagen (NQSZ) im DIN Deutsches Institut für Normung e.V.

DIN, EN, ISO (Hrsg.) (1984):Datenverarbeitung. Begriffe. DIN ISO 2382-1 Teil 1: Grundlegende Begriffe. Berlin: Normenausschuss Qualitätsmanagement, Statistik und Zertifizierungsgrundlagen (NQSZ) im DIN Deutsches Institut für Normung e.V.

DIN, EN, ISO (Hrsg.) (1995): Qualitätsmanagement. Begriffe. DIN ISO 8402. Normenausschuss Qualitätsmanagement, Statistik und Zertifizierungsgrundlagen (NQSZ) im DIN Deutsches Institut für Normung e.V.

DIN EN ISO 9000-2000: (2000): Leitfaden zur Auswahl und Anwendung. Normenausschuss Qualitätsmanagement, Statistik und Zertifizierungsgrundlagen (NQSZ) im DIN Deutsches Institut für Normung e.V.

DIN, EN, ISO (Hrsg.) (2000): Qualitätsmanagementsysteme. Modell zur Qualitätssicherung / QM-Darlegung in Produktion, Montage und Wartung. DIN EN ISO 9002-2000. Berlin. Normenausschuss Qualitätsmanagement, Statistik und Zertifizierungsgrundlagen (NQSZ) im DIN Deutsches Institut für Normung e.V.

DIN, EN, ISO (Hrsg.) (2000): Qualitätsmanagementsysteme. Modell zur Qualitätssicherung / QM-Darlegung bei der Endprüfung. DIN EN ISO 9003-2000. Berlin. Normenausschuss Qualitätsmanagement, Statistik und Zertifizierungsgrundlagen (NQSZ) im DIN Deutsches Institut für Normung e.V.

DIN, EN, ISO (Hrsg.) (1987): Qualitätsmanagement und Elemente eines Qualitätsmanagementsystems. Teil 1: Leitfaden.
DIN55350/ DIN 55 350, Teil 17 – Begriffe der Qualitätsprüfungsarten.

DIN, EN, ISO (Hrsg.) (1991): DIN ISO 9126/ DIN ISO 9126 – Informationstechnik – Beurteilen von Softwareprodukten, Qualitätsmerkmale und Leitfaden zu deren Verwendung. Normenausschuss Qualitätsmanagement, Statistik und Zertifizierungsgrundlagen (NQSZ) im DIN Deutsches Institut für Normung e.V.

Feigenbaum, A. V. (1986): Total Quality Control. 3. Aufl. New York: McGraw-Hill.

Fröhling, O.; A. Wullenkord (1991): „Qualitätskostenmanagement als Herausforderung an das Controlling.“ In: Kostenrechnungspraxis. 35. Jg. (1991), Nr. 4, S. 171-178.

Glaap, W. (1993): ISO 9000 leichtgemacht. Praktische Hinweise und Hilfen zur Entwicklung und Einführung von QS-Systemen. München; Wien: Hanser.

Halfmann, M. (2005):
http://lehre/ss02/proseminar/ausarbeitungen/marc_halfmann.pdf
(Zugriff am 15.03.2005).

Haist, F.; H. Fromm (1989): Qualität im Unternehmen. Prinzipien - Methoden – Techniken. München: Hanser.

Haist, F.; H. Fromm (1991): Qualität im Unternehmen. Prinzipien-Methoden-Techniken. 2. durchg. Aufl. München; Wien: Hanser.

Hammerschall, U. (2005): www.forumbb.de/files/Ulrike_Hammerschall_V-Modell_XT.pdf. Online im Internet: URL: <http://www.forumbb.de> (Zugriff am: 14.03.2005).

Jones, C. (1996): Applied Software Measurement: Assuring Productivity and Quality. Mcgraw-Hill 2nd Edition.

Juran J. M. (Hrsg.) (1988): Quality Control Handbook. 4. Aufl. New York.
KBst (2005): www.kbst.bund.de/-,279/V-Modell.htm. V-Modell XT 1.01.
Online im Internet: URL: <http://www.kbst.bund.de> (Zugriff am: 15.03.2005)

Lakoff, G.; M Johnson (1980): Metaphors We Live By, University of Chicago Press (Trd).

Lenz, R. (2003):

http://www.2cool4u.ch/business_it/qualitaet_testsoftware/qualitaet_testssoftware.htm. Qualitätssicherung und Testen von Software. Online im Internet: URL: <http://www.2cool4u.ch> (Zugriff am 20.04.2005).

Liggesmeyer, P. (1990): Modultest und Modulverifikation. State of the Art. Mannheim; Wien; Zürich: Spektrum

Liggesmeyer, P. (2002): Software-Qualität. Testen, Analysieren und Verifizieren von Software. Heidelberg; Berlin: Spektrum.

Mizumo, Y. (1983): Software Quality Improvement, in: IEEE. Computer, S. 23ff.

Q-Berlin (2005): <http://www.qi-bb.de/228.0.html>. Online im Internet: URL: <http://www.qi-bb.de> (Zugriff am 12.04.2005).

QM-Trends (2005): <http://www.qm-trends.de/fb0812.htm>.
Rechnergestützte Methoden zur prozessorientierten Erfassung qualitätsbezogener Kosten. Online im Internet: URL: <http://www.qm-trends.de> (Zugriff am 12.04.2005).

Rombach, H. D.; M. V. Zelkowitz (1995): "Directions in Software Process Research." In: Advances in Computers. Vol. 41. S. 1-63.

Rüb M. (2005): Statusreport zu Krisensitzung. Eschen. TKA-Presta internes Dokument.

Seghezzi H. D.; J. R. Hansen (Hrsg.) (1993): Qualitätsstrategien. Anforderungen an das Management der Zukunft. München: Hanser.

Seghezzi, H. D. (1997): Integriertes Qualitätsmanagement. Das St. Galler Konzept. München; Wien: Hanser.

Seibt, D. (1972): Organisation von Softwaresystemen. Betriebswirtschaftlich-organisatorische Analyse der Softwareentwicklung. Wiesbaden: Gabler.

Technische Universität Wien (2004): qse.ifs.tuwien.ac.at/courses/F_A_QM/ws0405/QMVU_intro_SB_20041007.pdf. Online im Internet: URL: <http://> (Zugriff am 20.04.2005).

Universität Leipzig (2005): http://8ais.informatik.uni-leipzig.de/studium/vorlesungen/2005_ss/sqm/2005s_sqm_v_01.pdf. Online im Internet: URL: <http://www.8ais.informatik.uni-leipzig.de> (Zugriff am 15.03.2005).

Universität Tübingen (2005): www-ti.informatik.uni-tuebingen.de/deutsch/. Online im Internet: URL: www-ti.informatik.uni-tuebingen.de (Zugriff am: 14.03.2005).

Wallmüller, E. (1990): Software-Qualitätssicherung in der Praxis. München: Hanser.

Wildemann, H. (1996): „Qualitätsorganisation neu gestalten. Die Transaktionskostentheorie hilft, Qualitätsaufgaben optimal zu organisieren.“ In: Zeitschrift für Qualität und Zuverlässigkeit (QZ), 41. Jg. (1996) Nr. 12, S. 1393-1400.

Anhang

Folgen von Missverständnissen

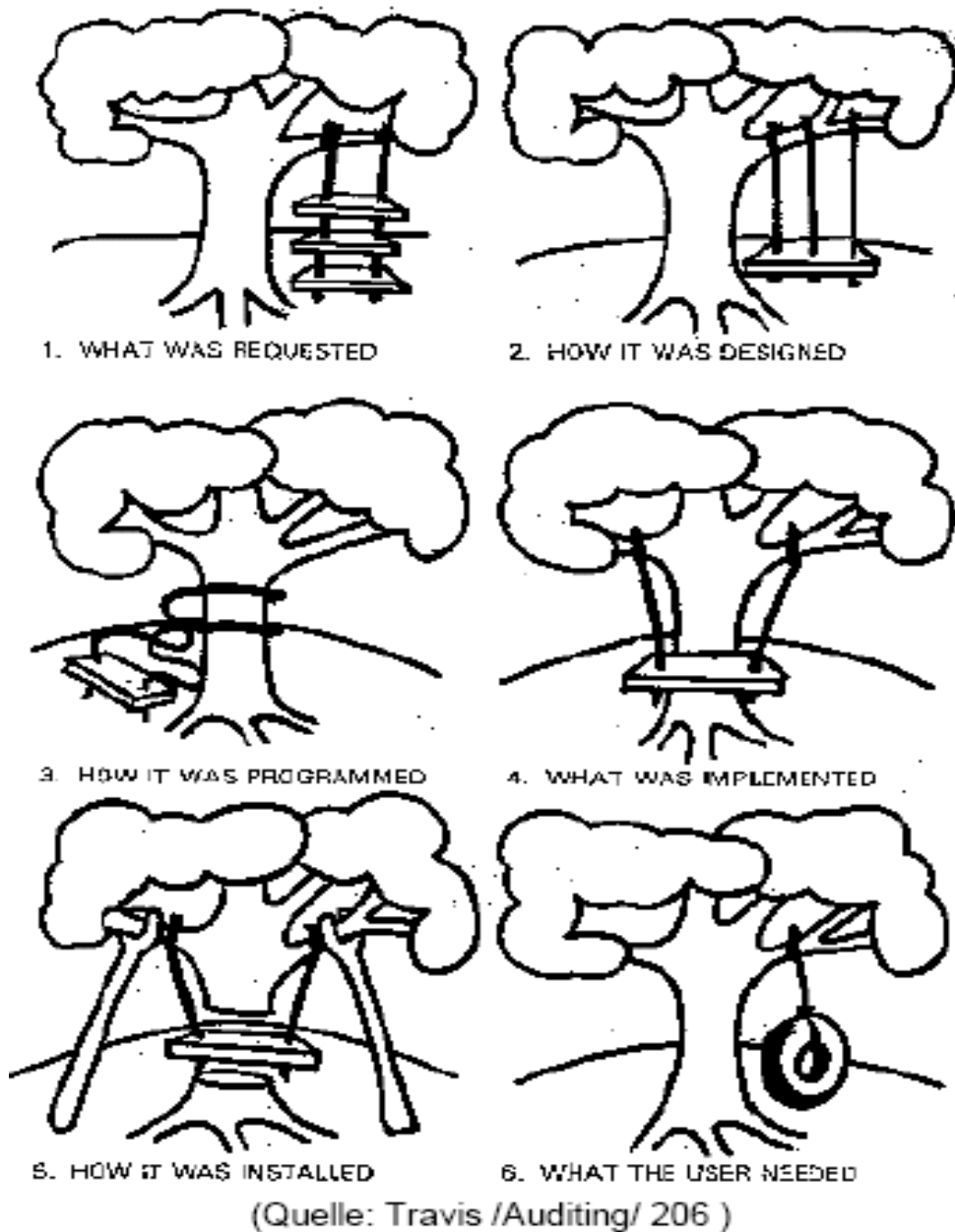


Abb. 39: Folgen von Mißverständnissen

Quelle: Travis (1987): Auditing the Development of Computer Systems. CIMA (Chartered Institute of Management Accountant).