

Qualitätsanforderungen an Embedded Software

Die Kunst der hieb- und stichfesten Qualitätsanforderungen

Thomas Batt, MicroConsult GmbH

0. Vorwort

Anforderungen zu erfassen und zu verwalten ist ein wesentlicher Schlüssel zum Projekterfolg. Die Embedded-Software-Funktionalität in Anforderungen zu beschreiben ist einfacher als die Qualitätsmerkmale. Dennoch sind die Qualitätsmerkmale zu erfassen, da sie sich am Ende NICHT hineintesten lassen. Je abstrakter die Qualitätsmerkmale sind, desto aufwendiger ist deren Erfassung. Mit genau dieser Herausforderung beschäftigt sich dieser Beitrag.

1. Qualitätsanforderungen und deren Mythen

Anforderungen sind in funktionale Anforderungen, Zusicherungen als Anforderungen und nichtfunktionale Anforderungen unterteilbar.

Funktionale Anforderungen

Funktionale Anforderungen beschreiben die nach außen hin sichtbaren Funktionalitäten der Embedded-Software.

Beispiel für eine funktionale Anforderung:

Die Kaffeevollautomat-Software muss fähig sein, sich zu aktualisieren.

Zusicherungen als Anforderungen

Zusicherungen (Constraints) sind allgemeine, globale Zusicherungen, die für das Produkt und/oder für die Produktentwicklung gelten. Auch die Projekttrandbedingungen gehören zu den Zusicherungen.

Unterarten von Zusicherungsanforderungen:

- **Technische Zusicherungen**

Technische Zusicherungen sind Anforderungen, die technische Vorgaben zur Realisierung festlegen, z.B. Bestimmung der Programmiersprache oder des Prozessors.

- **Projektbezogene Zusicherungen**

Projektbezogene Zusicherungen sind Anforderungen, die sich auf das gesamte Projekt beziehen, z.B. Einhaltung von gültigen Standards, Kosten (für Entwicklung, Produktion) und Daten, die sich in den Projektplänen widerspiegeln.

- **Lieferungsbezogenen Zusicherungen**

Lieferungsbezogene Zusicherungen sind Anforderungen, die sich auf die Systemlieferung beziehen, z.B. Lieferumfang, Liefervarianten, Lieferzeiten und Liefermengen.

- **Rechtliche und vertragliche Zusicherungen**

Rechtliche und vertragliche Zusicherungen sind Anforderungen, die die Abwicklung der Entwicklung betreffen, z.B. Kosten, Termin der Fertigstellung,

Zahlungsmeilensteine, Vertragsstrafen, Umgang mit neuen Anforderungen, Umgang mit Anforderungsänderungen und Eskalationspfade.

- **Prozessbezogene Zusicherungen**

Prozessbezogene Zusicherungen sind Anforderungen bezüglich des Entwicklungsprozesses (Vorgehensmodell), z.B. die Entwicklung muss basierend auf dem V-Modell XT durchgeführt werden.

Beispiel einer technischen Zusicherung als Anforderung:

Die Kaffeevollautomat-Software muss mit der UML (Unified Modeling Language) modelliert sein.

Nichtfunktionale Anforderungen

Mit dieser Gliederung reduzieren sich die nichtfunktionalen Anforderungen auf Qualitätsmerkmale. **Qualitätsmerkmale** lassen sich differenzieren in:

- Qualitätsmerkmale, welche **eine Funktionalität** anhand einer funktionalen Anforderung näher beschreiben (z.B. ein Zeitverbrauch)

Im diesem Fall ist klar, was bei der Implementierung zu tun ist, um den Zeitverbrauch einzuhalten. Der Zeitverbrauch lässt sich einfach testen, sofern eine konkrete Zeitangabe bestehend aus Wert und Einheit in der Qualitätsanforderung spezifiziert ist.

➔ **konkretes Qualitätsmerkmal**

- Qualitätsmerkmale, welche für die **gesamte Software** relevant sind (z.B. Wiederverwendbarkeit)

Im diesem Fall ist nicht eindeutig klar, was bei der Implementierung zu tun ist, um Wiederverwendbarkeit in der Software zu erreichen; Wiederverwendbarkeit ist nicht „einfach so“ testbar.

➔ **abstraktes Qualitätsmerkmal**

Dieses **Bewusstsein und Verständnis** über Qualitätsanforderungen sollte bei allen Mitarbeitern aus den Unternehmensbereichen Vertrieb, Marketing, Produktmanagement, Projektmanagement, Entwicklung, Test und Produktion vorhanden sein.

Fakten zu Qualität:

- Unterscheidung: Produkt- und Prozessqualität – beide beeinflussen sich gegenseitig
- Unterscheidung: Innere (Entwicklersicht) und äußere Qualität (Kundensicht)
- Qualität kann nicht später hineingetestet werden
- Qualitätsmerkmale müssen vor Entwicklungsbeginn bestimmt sein
- Manche Qualitätsmerkmale unterstützen sich gegenseitig, z.B. beeinflussen sich Änderbarkeit und Wiederverwendbarkeit gegenseitig positiv
- Manche Qualitätsmerkmale sind gegensätzlich, z.B. beeinflussen sich Performance und Zuverlässigkeit gegenseitig negativ
- Richtlinie: Vier aus sechs Qualitätsmerkmalen zur Realisierung auswählen

Schaffen Sie es, einen Formel-1-Rennwagen so zu realisieren, dass er als Drei-Liter-Auto und zum Containertransport geeignet ist?

Zu Software-Qualitätsmerkmalen sowie deren Spezifikation und Test existieren die folgenden **Standards**:

- ISO/IEC 9126-1:2001
Software Engineering – Product Quality – Part 1: Quality Model

Dieser Standard ist abgelöst durch:

- ISO/IEC 25010:2011
Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – System and Software Quality Models

Qualitätsmerkmal	Subqualitätsmerkmal
Funktionalität	Angemessenheit, Richtigkeit, Interoperabilität, Ordnungsmäßigkeit
Zuverlässigkeit	Reife, Fehlertoleranz, Wiederherstellbarkeit
Benutzbarkeit	Verständlichkeit, Erlernbarkeit, Bedienbarkeit
Effizienz	Zeitverhalten, Verbrauchsverhalten
Änderbarkeit	Analysierbarkeit, Modifizierbarkeit, Stabilität, Prüfbarkeit
Übertragbarkeit	Anpassbarkeit, Installierbarkeit, Konformität, Austauschbarkeit, Wiederverwendbarkeit
Sicherheit	Funktionale Sicherheit, Angriffssicherheit

Bild 1: Exemplarische Übersicht zu Qualitäts- und Subqualitätsmerkmalen

Bitte beachten Sie dazu immer die für Ihre Branche und Produkte gültigen Standards.

2. Erfassungsmethode für Qualitätsanforderungen

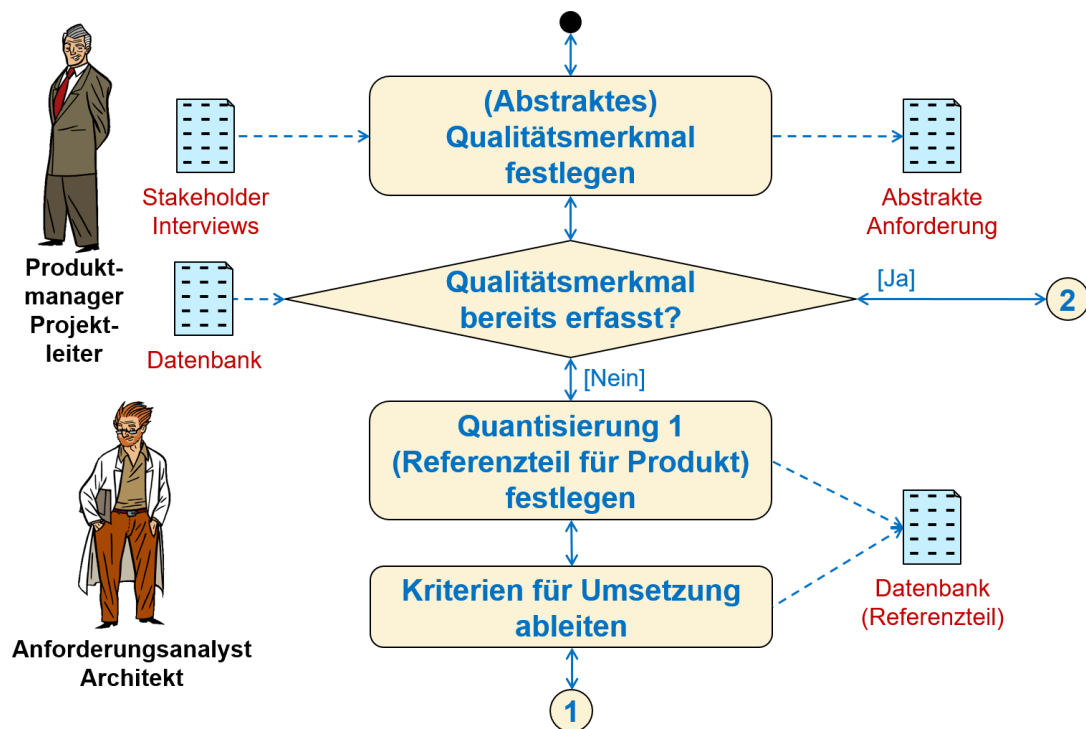


Bild 2-1: Erfassungsmethode für Qualitätsanforderungen (1. Teil)

Abstraktes Qualitätsmerkmal festlegen:

Zu jedem Qualitätsmerkmal formuliert der Anforderungsanalyst **erschließende Fragen** und zu jedem Subqualitätsmerkmal weiterführende Fragen. Daraus ergeben sich zunächst abstrakte Anforderungen. Er kann in der Datenbank prüfen, ob das Qualitätsmerkmal bereits quantisiert ist. Wenn ja, kann der Anforderungsanalyst direkt die konkrete Anforderung formulieren.

Qualitätsmerkmal bereits in der Datenbank erfasst?

Falls das Qualitätsmerkmal bereits in vorherigen Produkten spezifiziert wurde, können der Anforderungsanalyst direkt die konkrete Anforderung und das Testteam den konkreten Test-Case formulieren.

Quantisierung 1 (Referenzteil für Produkt) festlegen:

Der Anforderungsanalyst erarbeitet zusammen mit Fachexperten ein Quantisierungsschema für das Qualitätsmerkmal.

Kriterien für die Umsetzung ableiten:

Aus dem Quantisierungsschema leiten sich Kriterien und Fakten für die Umsetzung der Anforderung ab.

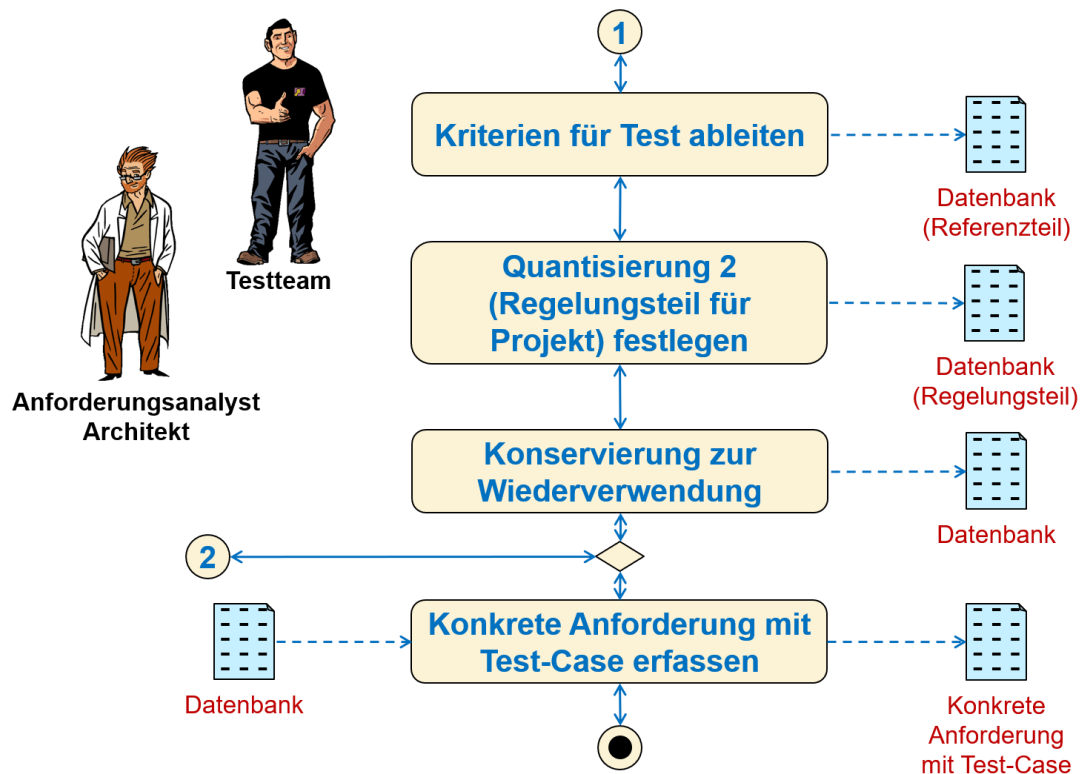


Bild 2-2: Erfassungsmethode für Qualitätsanforderungen (2. Teil)

Kriterien für den Test ableiten:

Aus dem Quantisierungsschema leiten sich Kriterien und Fakten für die Verifikation der Anforderung ab.

Quantisierung 2 (Regelungsteil) für Projekt festlegen:

Viele Quantisierungsschemata erlauben dem Anforderungsanalysten eine projektspezifische Anpassung von Werten und oder Zusammenhängen.

Konservierung zur Wiederverwendung:

Neu entwickelte Quantisierungsschemata zu Qualitätsmerkmalen pflegt der Anforderungsanalyst zur Wiederverwendung in die Datenbank ein.

Konkrete Anforderung mit Test-Case erfassen:

Auf Basis der bereits vorhandenen abstrakten Anforderung zusammen mit dem Quantisierungsschema für das betrachtete Qualitätsmerkmal erfasst der Anforderungsanalyst eine umsetzbare und nachweisbare Anforderung. Aus der Anforderung leitet das Testteam den passenden Test-Case ab.

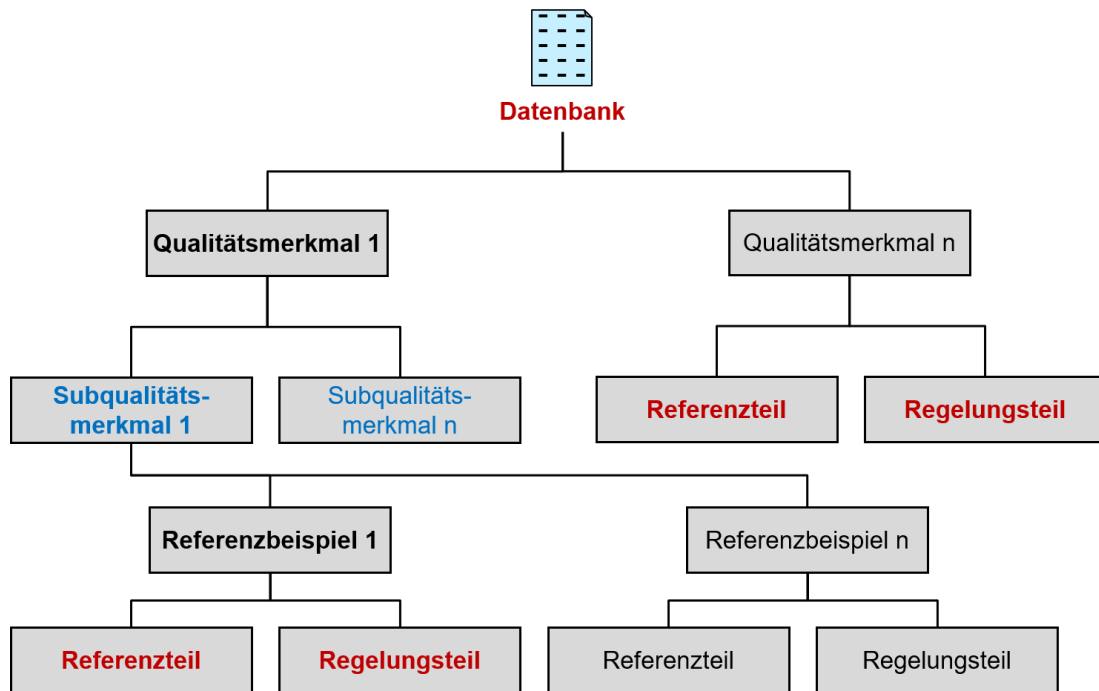


Bild 3: Datenbankaufbau für Qualitätsanforderungen

Die Referenzdatenbank ist möglichst generisch zu halten, damit sie den stetig wachsenden und immer wieder wechselnden Anforderungen aus Theorie und Praxis gerecht wird.

Allgemeiner Teil:

- Gültig und zu berücksichtigen bei der Erhebung aller Qualitätsmerkmale
- Allgemeine und merkmalsübergreifende Informationen und Anmerkungen
- Fachliche Ansprechpartner
- Weiterführende Literatur- und interne Linkhinweise

Referenzbeispiel:

Für die in der Datenbank enthaltenen Lösungen werden Referenzbeispiele erstellt. Zu jedem Referenzbeispiel gehören ein Referenzteil und ein Regelungsteil.

Referenzteil:

Der Referenzteil beinhaltet Kenngrößen und mögliche Wertebereiche zur Berechnung der entsprechenden Submerkmale.

Regelungsteil:

Im Regelungsteil wird der Lösungsweg zur konkreten Lösung anhand von Beispielen beschrieben. Am Ende des Regelungsteils kann die konkrete Anforderung für das entsprechende Merkmal stehen.

3. Erfassungsbeispiel

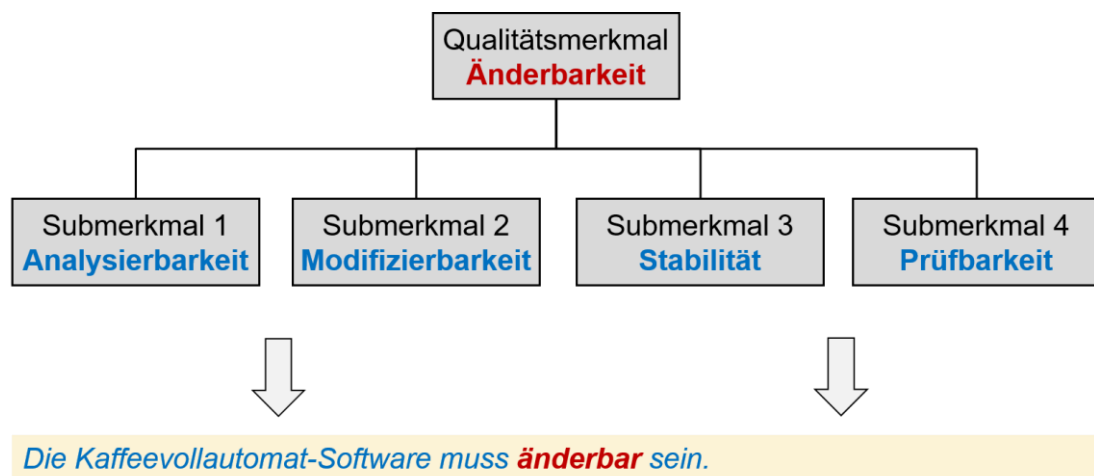


Bild 4: (Abstraktes) Qualitätsmerkmal festlegen

Das Qualitätsmerkmal „änderbar“ ist in diesem Beispiel auf die Software eines Kaffeevollautomaten angewandt, lässt sich aber auf beliebige Produkte übertragen.

Bezogen auf Bild 1 gelten für das Qualitätsmerkmal Änderbarkeit die Subqualitätsmerkmale Analysierbarkeit, Modifizierbarkeit, Stabilität und Prüfbarkeit.

Zu jedem Qualitätsmerkmal formuliert der Anforderungsanalyst **erschließende Fragen**:

Änderbarkeit

Wollen Sie Kenngrößen zu den Merkmalen definieren, die sich auf den Aufwand beziehen, der zur Durchführung vorgegebener Änderungen des zu entwickelnden Systems notwendig sind?

Zu jedem Subqualitätsmerkmal formuliert der Anforderungsanalyst **weiterführende Fragen**:

Analysierbarkeit

Wollen Sie Kenngrößen für die Merkmale definieren, die sich auf den Aufwand beziehen, um Mängel oder Ursachen von einem Versagen des zu erstellenden Systems zu diagnostizieren oder um änderungsbedürftige Teile zu bestimmen?

Modifizierbarkeit

Wollen Sie Kenngrößen für die Merkmale definieren, die sich auf den Aufwand beziehen, der zur Ausführung von Verbesserungen, zur Fehlerbeseitigung oder zur Anpassung an Umgebungsveränderungen notwendig ist?

Stabilität

Wollen Sie Kenngrößen für die Merkmale definieren, die sich auf den Aufwand beziehen, der zur Prüfung der geänderten Hardware/Software/... notwendig ist?

Prüfbarkeit

Wollen Sie Kenngrößen für die Merkmale definieren, die sich auf den Aufwand beziehen, der zur Prüfung der geänderten Hardware/Software/... notwendig ist?

Metrik	Erklärung	Wertebereich	
		Min	Max
NSTAT	Anzahl der ausführbaren Anweisungen	1	50
DCAL	Anzahl der direkt aufgerufenen Komponenten	0	7
VG	Komplexität einer Funktion nach McCabe (Cyclomatic Number)	1	15
UJMP	Anzahl der direkten Sprünge (z.B. goto)	0	0
NOUT	Anzahl der Austritte (z.B. return)	1	1
LEVL	Anzahl der Ebenen (Schachtelungstiefe + 1)	1	5
COMF	Anzahl der Kommentare dividiert durch die Anzahl der Anweisungen	0.2	1
VOCF	Vokabel-Frequenz nach Halstead (Indikator für Redundanz)	1	4
AGS	Durchschnittliche Länge einer Anweisung nach Halstead	1	10

Bild 5-1: Quantisierung 1 (Referenzteil für Produkt) festlegen: Softwaremetriken

Das folgende Beispiel basiert auf einer Veröffentlichung der DSF Deutsche Flugsicherung GmbH und Brandenburg Technische Universität Cottbus: „*Pieper: Definition von Software-Qualitätsmerkmalen*“ und „*Bennicke, Dörr: New Metric Application*“.

Die oben dargestellten Metriken sind ein Beispiel für einen Auszug aus der Referenzdatenbank zum Qualitätsmerkmal **Änderbarkeit**. Die Metriken und Grenzwerte können zur Bestimmung der Subqualitätsmerkmale **Analysierbarkeit**, **Modifizierbarkeit**, **Stabilität** und **Prüfbarkeit** herangezogen werden.

Analysierbarkeit	= VG*GW + NSTAT*GW + COMF*GW + AVGS*GW
Modifizierbarkeit	= AVGS*GW + LEVL*GW + UJMP*GW + VOCF*GW
Stabilität	= NOUT*GW + DCAL*GW + UJMP*GW + VOCF*GW
Prüfbarkeit	= LEVL*GW + UJMP*GW + VG*GW

Bild 5-2: Quantisierung 1 (Referenzteil für Produkt) festlegen: Zuordnung und Gewichtung

Nach Bestimmung der **Metriken** müssen diese den Submerkmalen **zugeordnet** werden:

Analysierbarkeit	=	$a + b + c + \dots$
Modifizierbarkeit	=	$a + b + c + \dots$
Stabilität	=	$a + b + c + \dots$
Prüfbarkeit	=	$a + b + c + \dots$

Nach Zuordnung der **Metriken** müssen diese **gewichtet** werden:

Analysierbarkeit	=	$a * x + b * x + c * x + \dots$
Modifizierbarkeit	=	$a * x + b * x + c * x + \dots$
Stabilität	=	$a * x + b * x + c * x + \dots$
Prüfbarkeit	=	$a * x + b * x + c * x + \dots$

Die Summe aller Gewichtungsfaktoren pro Subqualitätsmerkmal muss 100 ergeben.

Beim Errechnen der Änderbarkeit wird eine Metrik mit „1“ bewertet, wenn der Messwert innerhalb des Wertebereichs liegt. Ist der Wertebereich nicht erfüllt, wird die Metrik mit „0“ bewertet.

Für das konkrete System werden im Regelungsteil die Wertebereiche der einzelnen Metriken und deren Gewichtung angepasst.

Summe pro (Sub-) Qualitätsmerkmal	Erfüllungsgrad (Quantisierung)
100	3
67 – 99	2
34 – 66	1
0 – 33	0

Bild 5-3: Quantisierung 1 (Referenzteil für Produkt) festlegen:
Bewertungsschema für jedes (Sub-) Qualitätsmerkmal

Im Referenzteil ist auf Basis der berechenbaren Ergebnisse die **Quantisierung** in [Erfüllungsgrad] festgelegt. Beim **Subqualitätsmerkmal** ist die Bedeutung der Quantisierung eindeutig. Definition: Beim **Qualitätsmerkmal** muss sich der Grad der Erfüllung auf jedes Subqualitätsmerkmal übertragen. Ist also der Erfüllungsgrad 4 auf Qualitätsmerkmalebene gefordert, so müssen alle Subqualitätsmerkmale ebenfalls diesen Grad erfüllen.

Metrik	Erklärung	Wertebereich	
		Min	Max
NSTAT	Anzahl der ausführbaren Anweisungen	1	50
DCAL	Anzahl der direkt aufgerufenen Komponenten	0	7
VG	Komplexität einer Funktion nach McCabe (Cyclomatic Number)	1	15
UJMP	Anzahl der direkten Sprünge (z.B. goto)	0	0
NOUT	Anzahl der Austritte (z.B. return)	1	1
LEVL	Anzahl der Ebenen (Schachtelungstiefe + 1)	1	5
COMF	Anzahl der Kommentare dividiert durch die Anzahl der Anweisungen	0.2	1
VOCF	Vokabel-Frequenz nach Halstead (Indikator für Redundanz)	1	4
AGS	Durchschnittliche Länge einer Anweisung nach Halstead	1	10

Bild 6-1: Quantisierung 2 (Regelungsteil für Projekt) festlegen: Wertebereiche der Softwaremetriken definieren

Im Regelungsteil erfolgt die projektspezifische **Anpassung** der **Wertebereiche** der **Metriken**.

Analysierbarkeit	$= VG*25 + NSTAT*25 + COMF*25 + AVGS*25$
Modifizierbarkeit	$= AVGS*25 + LEVL*25 + UJMP*25 + VOCF*25$
Stabilität	$= NOUT*25 + DCAL*25 + UJMP*25 + VOCF*25$
Prüfbarkeit	$= LEVL*33 + UJMP*33 + VG*34$

Bild 6-2: Quantisierung 2 (Regelungsteil für Projekt) festlegen: Gewichtung der Softwaremetriken

Im Regelungsteil erfolgt die projektspezifische **Anpassung** der **Gewichtung** der **Metriken** für jedes Qualitätsmerkmal.

Summe pro (Sub-) Qualitätsmerkmal	Erfüllungsgrad (Quantisierung)
67 – 99	2

Bild 6-3: Quantisierung 2 (Regelungsteil für Projekt) festlegen: Quantisierung

In diesem Beispiel ist der Erfüllungsgrad für alle Subqualitätsmerkmale und das Qualitätsmerkmal identisch.

REQ-ID	Requirement
SW-REQ-00x	Die Kaffeevollautomat-Software muss mindestens mit dem Grad 2 änderbar sein.
TC-ID	Test-Case
SW-TC-00y	Testvorbedingungen: <ul style="list-style-type: none"> Der Release der Kaffeevollautomat-Software liegt vor.
	Testauslösendes Ereignis: <ul style="list-style-type: none"> Die Analyse bezüglich Änderbarkeit der Kaffeevollautomat-Software erfolgt.
	Testergebnis erwartet: <ul style="list-style-type: none"> Die Kaffeevollautomat-Software ist mit dem Grad 2 änderbar.

Bild 7: Konkrete Anforderung mit Test-Case erfassen

Auf Basis dieser konkreten Software-Anforderung weiß der Softwareentwickler, welche Metriken er wie bei der Implementierung zu beachten hat. Der Softwaretester kann die Metriken ermitteln und damit verifizieren, ob eine Änderbarkeit von mindestens 2 erreicht ist.

In der Anforderung und im Test-Case ist der Begriff „änderbar“ zum Referenz- und Regelungsteil in der Datenbank verlinkt.

4. Resümee

Schaffen Sie in Ihrem Unternehmen bei Ihren Mitarbeitern das Bewusstsein und Verständnis von Qualitätsanforderungen.

Der Aufwand für die Quantisierungen der abstrakten zu konkreten Qualitätsmerkmalen ist initial hoch. Sie profitieren davon aber in allen nachfolgenden Projekten.

5. Referenzierte und weiterführende Links

- [1] MicroConsult Download für diesen Beitrag - komplett und aktuell
<http://download.microconsult.net/ese2018/q-req.zip>
- [2] MicroConsult Download: Anforderungen – Eine Checkliste zur Reife
<http://download.microconsult.net/ese2017/req-checkliste.zip>
- [3] **MicroConsult Trainings:**
[Requirements Engineering und Management für Embedded-Systeme](#)
[Funktionale Sicherheit \(Safety\) von Elektronik und deren Software – Umsetzung nach ICE 61508 und ISO 26262](#)

Security: Sicherheit von Embedded-Systemen im Kontext der funktionalen Sicherheit

Usability Praxis-Seminar: Produkte benutzerfreundlich entwickeln

SysML: Modellbasierte Systemanalyse und Systemdesign mit der Systems Modeling Language

UML-Praxis-Workshop: Praktischer Einsatz modellbasierter Softwareentwicklung für Embedded- und Echtzeit-Systeme

www.microconsult.de

- [4] Fachliteratur:
Requirements Engineering und Management
SOPHIST GmbH, HANSER Verlag
Print-ISBN: 978-3-446-43893-4
E-Book-ISBN: 978-3-446-44313-6
<http://www.hanser-fachbuch.de>

6. Autor:



Dipl.-Ing. (FH) Thomas Batt ist gebürtiger Freiburger. Nach seiner Ausbildung als Radio- und Fernsichttechniker studierte er Nachrichtentechnik in Offenburg. Seit 1994 arbeitet er kontinuierlich in verschiedenen Branchen und Rollen im Bereich Embedded-/Real-Time Systementwicklung. 1999 wechselte Thomas Batt zur MicroConsult GmbH. Dort verantwortet er heute als zertifizierter Trainer und Coach die Themenbereiche Systems /Software Engineering für Embedded-/Real-Time-Systeme sowie Entwicklungsprozess-Beratung.

Kontakt:

t.batt@microconsult.de
www.microconsult.de