FSS.

Vom Requirement zum Testfall

Ein langer und steiniger Weg, oder?

Mirko Drobietz
02.09.2011

Unternehmensprofil
FSS CONSULTING GMBH

Unternehmen

- Gründung im Jahr 1993 in Hannover
- Beratungsunternehmen mit Schwerpunkt Finanzindustrie

Kernkompetenzen

- Anwendungsentwicklung
- Test- und Anforderungsmanagement
- Projektmanagement
- Prozessorganisation

Standorte

- Hannover (Hauptsitz)
- Berlin
- Hamburg
- Münster
- Stuttgart

Michael Stratmann

Roland Schaper

Marc Gehrke

Stefan Gebhardt
(Berlin)

Dirk Schimmelmann
(Hamburg)

Unternehmensprofil
FSS CONSULTING GMBH

FSS.

Berater

- Unsere Berater sind in ihren Fachgebieten zertifiziert
  - Oracle / Sun certified Java Programmer / Architect
  - IBM certified Developer (Lotus Notes /Domino)
  - ISTQB certified Tester / Testmanager
  - IREB certified Professionals for Requirements Engineering
  - Project Management Professional (pmp)
  - ITIL-Servicemanager
- Ausgeprägte soziale Kompetenz
- Jahrelange Projekterfahrung
- Zugriff auf Erfahrung anderer FSS-Berater durch hausinternes Wissensmanagement (FSS.portal)

Unternehmen

- Sanftes und stetiges Wachstum auf hohem Niveau
- Enge und langfristige Kundenbindung durch qualitativ hochwertige Arbeit
- Inhabergeführt

Agenda

FSS.

- Motivation zum Vortrag
- Rollen und Charaktere in einem Softwareentwicklungsprojekt
- Anforderungserhebung leicht(er) gemacht
- Dokumentation für Anforderungen mit Blick auf den Test
- Findung der Teststrategie und Testfälle
- Raum für Diskussion

Was ist …

Testen:

~~Der Prozess, ein Programm auf systematische Weise auszuführen, um Fehler zu finden.~~ [MYR91]

Prozess, der aus allen Aktivitäten des Lebenszyklus besteht, die sich mit der **Planung**, **Vorbereitung** und **Bewertung** eines **SW-Produkts** und der **dazugehörigen Arbeitsergebnisse** befasst. [IMB07]

Anforderungsmanagement [SOP09]:

Das Requirements Engineering ist ein **kooperativer**, **iterativer**, **inkrementeller** Prozess, dessen Ziel es ist zu gewährleisten, dass

1. alle **relevanten Anforderungen bekannt** und in dem erforderlichen Detailgrad **verstanden** sind,
2. die involvierten Stakeholder eine **ausreichende Übereinstimmung** über die bekannten Anforderungen erzielen,
3. alle Anforderungen **konform** zu den Dokumentationsvorschriften **dokumentiert** bzw. konform zu den Spezifikationsvorschriften **spezifiziert** sind.

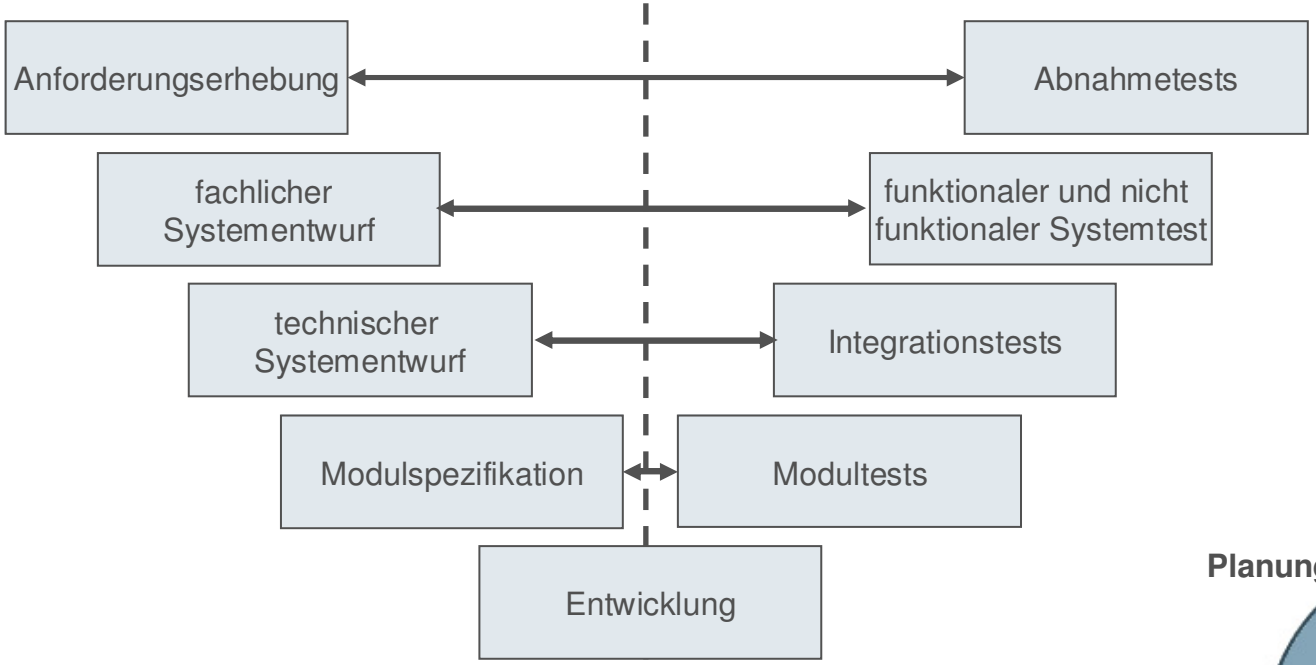
Warum beides kombinieren?

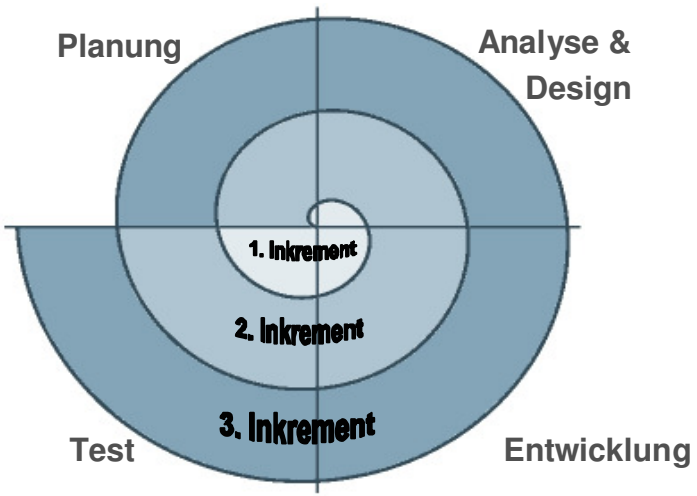
Eine Anforderungserhebung kann ohne systematischen Test nicht validiert werden.

und

Testen ohne strukturierte Anforderungserhebung ist wesentlich teurer und ineffektiver.

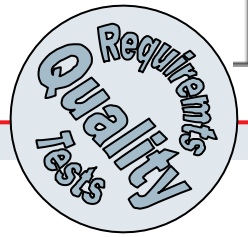
Welches ist aus Sicht der QS das richtige Vorgehensmodell für Softwareentwicklung?





Das hängt ganz vom Problem, der Branche und dem Team ab.

Die grundsätzlichen Probleme in der Softwareentwicklung

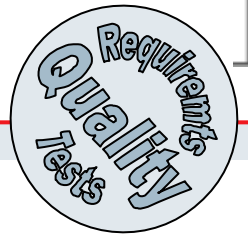


FSS.

… existieren in beiden Paradigmen

- **das Vorgehensmodell wird oft unabhängig vom Problem und dem Team ausgewählt**
- **Kommunikation im Team und an die Stakeholder wird vernachlässigt**
- **Konflikte werden nicht aufgelöst**
- **Anforderungserhebung wird nicht systematisch durchgeführt**
- **Tests und Dokumentation werden eingespart**

Agenda



FSS.

Rollen und Charaktere

in einem Softwareentwicklungsprojekt

Teamzusammenstellung - Rollen

Warum sind Tester meist unbeliebt?

Auszug unterschiedlicher Rollen

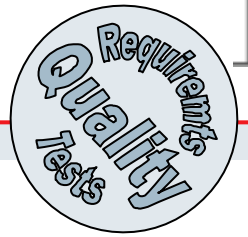
konstruktiv

▪ Projektleiter	glaubt an den Projekterfolg
▪ Requirements Engineer	glaubt an die Möglichkeit das neue System zu konstruieren
▪ Architekt	hat Spaß am Aufbau des notwendigen Komponenten
▪ Entwickler	erschafft das System

▪ Testmanager & Tester	müssen daran glauben, dass das System nicht funktioniert

destruktiv

Teamzusammenstellung - Charaktere

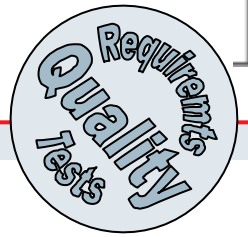


FSS.

Unterschiedliche Charaktere (nach Belbin) [BEL93]

Rolle	Rollenbeitrag	zulässige Schwäche
Neuerer / Erfinder	bringt neue Ideen ein	oft gedankenverloren
Wegbereiter / Weichensteller	entwickelt Kontakte	oft zu optimistisch
Koordinator / Integrator	fördert Entscheidungsprozesse	kann als manipulierend empfunden werden
Macher	hat Mut, Hindernisse zu überwinden	ungeduldig, neigt zu Provokation
Beobachter	untersucht Vorschläge auf Machbarkeit	mangelnde Fähigkeit zur Inspiration
Teamarbeiter / Mitspieler	verbessert Kommunikation, baut Reibungsverluste ab	unentschlossen in kritischen Situationen
Umsetzer	setzt Pläne in die Tat um	unflexibel
Perfektionist	vermeidet Fehler, stellt optimale Ergebnisse sicher	überängstlich, delegiert ungern
Spezialist	liefert Fachwissen und Information	verliert sich oft in technischen Details

Das Erfolgsrezept in der Teamzusammenstellung



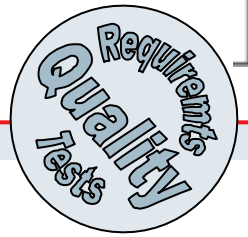
FSS.

Die richtige Mischung an Rollen und Charakteren fördert den Projekterfolg.

gegenseitige Toleranz und Verständnis kennzeichnen ein gutes Entwicklungsteam

- jede Rolle wird für den Projekterfolg benötigt, ist also auch gleich wichtig
- die destruktive Arbeitsweise des Testers (im Sinne, beweisen das die Software fehlerhaft ist) ist gewollt und für seine Tätigkeit notwendig
- ein Team, das einige Charaktere nicht besetzt oder eine deutliche Ballung einzelner Charaktere aufweist, wird im Projektverlauf oft vor zusätzliche Herausforderungen gestellt
- eine Teamzusammenstellung mit unterschiedlichen Charakteren wird besonders in der Startphase des Projektes zusätzlichen Aufwand für Teambuilding und Konfliktlösung verursachen, es lohnt sich aber auf Dauer

Agenda

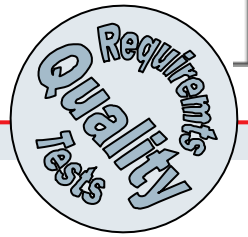


FSS.

Anforderungserhebung leicht(er) gemacht

und die Dokumentation

Anforderungsarten



FSS.

[SOP09]

funktionale Anforderungen

- Strukturanforderungen
- Funktionsanforderungen
- Verhaltensanforderungen

Qualitätsanforderungen

- Änderbarkeit
- Zuverlässigkeit
- Verbrauchsverhalten
- Portabilität
- Benutzbarkeit
- Zeitverhalten
- Mengengerüst
- Sicherheit

Randbedingungen

➔ können nicht beeinflusst werden und schränken die Systementwicklung ein

Qualitätsanforderungen und Randbedingungen sind größtenteils vom Fachbereich festzulegen.

Eine Anforderungsdefinition, die auf einige dieser Aspekte verzichtet ist sehr wahrscheinlich unvollständig

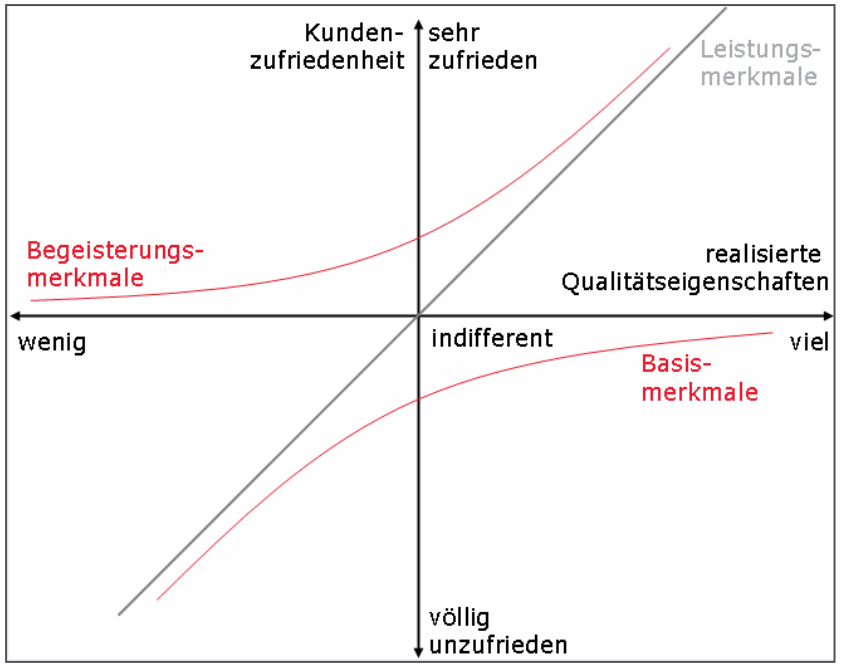
Qualitätskriterien für Anforderungen

Qualitätskriterien nach IEEE 830-1998 (Software Requirements Specification)

- Bewertet
- Eindeutigkeit
- Korrekt
- Konsistent
- Prüfbar
- Verfolgbarkeit (Tracebility)
- Vollständig

Diese Liste an Qualitätskriterien eignet sich gut als kurze Checkliste.

Die (unklare) Stimme des Kunden (Kano-Modell) [HÖL07,SOP09]



Das menschliche Wissen ist in drei Ebenen angesiedelt, die bei der Anforderungserhebung zu berücksichtigen sind.

Basisfaktoren

- selbstverständliche Systemmerkmale
- unbewusstes Wissen
- Beobachtungstechniken o. dokumentenzentrierte Techniken

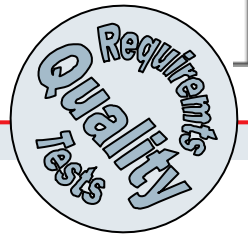
Leistungsfaktoren

- explizit geforderte Systemmerkmale
- bewusstes Wissen
- Befragungstechniken o. dokumentenzentrierte Techniken

Begeisterungsfaktoren

- sind dem Stakeholder nicht bewusst und treten erst bei der Benutzung auf
- unterbewusstes Wissen
- Kreativitätstechniken

Sprache ist nicht eindeutig



… wird aber oft dafür gehalten oder zumindest

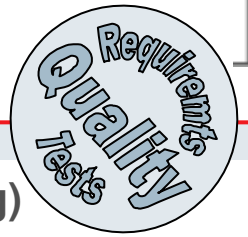
wird gerne wörtlich interpretiert.

Natürlichsprachliche Defekte (Auszug aus [SOP09])

- unvollständig spezifizierte Bedingungen (bspw. wenn…dann ohne sonst, im Falle von)
  → Tilgung
- unvollständig spezifizierte Prozesswörter (bspw. anzeigen, eingeben, drucken)
  → Tilgung
- Substantive ohne Bezugsindex (bspw. der Anwender, die Meldung, die Funktion)
  → Generalisierung
- Universalquantoren
  (bspw. alle, jeder, immer)
  → Generalisierung

Ein gewisser Grad an Prosa-Anteil in den Dokumentationen ist meist nicht zu umgehen (und wegen eines einheitlichen Verständnisses im Team meist auch sinnvoll).

Anforderungsdokumenation in Modellen [SOP09]

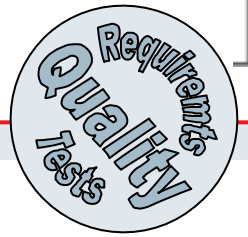


FSS.

- **Ziele (wie stehen die Ziele und Teilziele in Zusammenhang)**
  - Und-Oder-Bäume
- **Szenarien (Use-Cases)**
  - Use-Case-Diagramme
  - Use-Case-Spezifikation (zusätzliche Beschreibung zum Use-Case)
- **(System-)Anforderungen**
  - **Strukturperspektive**
    - ERM-Diagramme
    - UML-Klassendiagramme
  - **Funktionsperspektive**
    - Datenflussdiagramme
    - UML-Aktivitätendiagramme
  - **Verhaltensperspektive**
    - UML-Zustandsdiagramme

Modelle sind nützlich, um Aspekte eines Systems formal zu beschreiben. Sie sind aber kein Allheilmittel.

Good-Practice-Ansätze für das Anforderungsmanagement



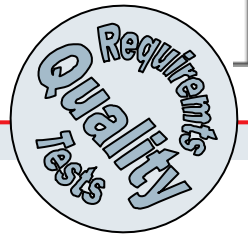
FSS.

Konflikte in den Anforderungen sollten so früh wie möglich erkannt und wenn möglich aufgelöst werden.

Für die eindeutige Terminologie ein Glossar anlegen … und pflegen.

Richtig verstandenes Prototyping eignet sich gut zur Anforderungserhebung oberflächenbasierter Systeme.

Agenda



FSS.

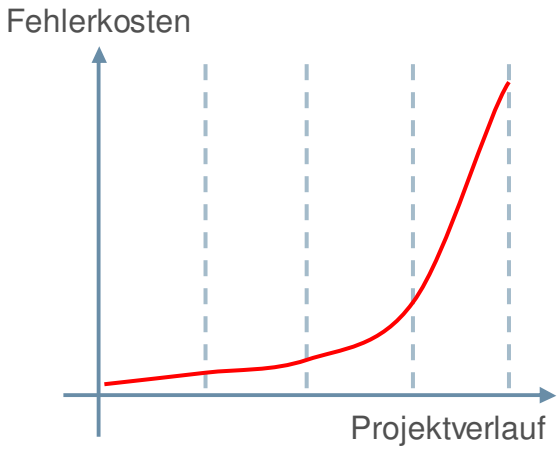
Findung der Teststrategie und Testfälle

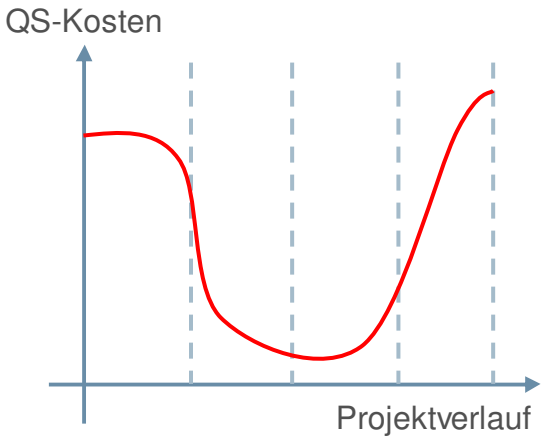
Prinzipien des Testens

Allgemeine Prinzipien des Testens

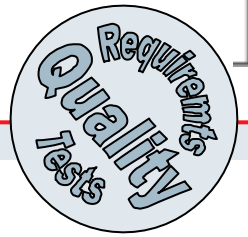
- Testen zeigt die Anwesenheit von Fehlern
- es lässt sich keine Nachweis für das Fehlen von Fehlerursachen erbringen
- Vollständiges Testen ist bis auf Trivialfälle nicht möglich, Tests sind immer Stichproben und der Testaufwand ist nach Risiko und Prioritäten zu steuern

Fehler kosten - QS aber auch





Testkonzept

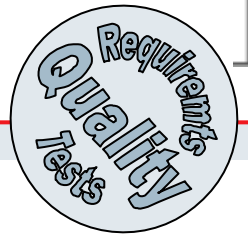


FSS.

Ein Testkonzept sollte mindestens die folgende Aspekte abdecken:

Abgrenzungen	Was wird nicht getestet (bspw. Last-, Stress- und Performancetests)?
Teststufen / Testvorgehen	bspw. Modul- , Integration- , System- und Abnahmetests
Testobjekte	bspw. Dokumente, Schnittstellen, Oberflächen
Priorisierung von Testobjekten	erforderlich um im Test selbständige Entscheidungen fällen zu können!
Testeintritts- und Testendekriterien	sehr nützlich um in Stresszeiten unnötigen Diskussionen aus dem Weg zu gehen.
Kriterien zur Fehlerklassifikation	möglichst nicht mehr als 4-5 Stufen, diese sollten aber auch für „Testlaien“ klar voneinander abgrenzbar sein
Kriterien für Testunterbrechung	Hilfreich zur Aufwandreduktion
Benötigte Testinfrastruktur, Ressourcen und Zeitplanung	Besonders in größeren Unternehmen und Projekten zwingend erforderlich
Intervall und Form Teststatusberichte	Intervall und Form der Berichte festlegen um einem subjektiven Empfinden bei unangenehmen Botschaften entgegenzuwirken.

Testfallbildung



FSS.

Die am meisten benutzten (aber leider nicht unbedingt systematischen) Testentwurfstechniken sind erfahrungsbasiert (Error Guessing, Exploratives Testen).

Vor der Bildung systematischer Testfälle steht die Frage nach Black- oder White-Box-Testing … und dann die Nutzung der entsprechenden Entwurfstechniken.

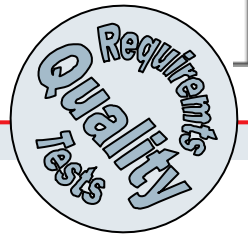
Black-Box-Testentwurfstechniken

- Äquivalenzklassenbildung
- Grenzwertanalyse
- Entscheidungstabelle
- zustandsbezogener Test
- Anwendungsfallbasierter Test

White-Box-Testentwurfstechniken

- Anweisungsüberdeckung
- Zweigüberdeckung
- Bedingungsüberdeckung einfach
- Bedingungsüberdeckung mehrfach
- Pfadabdeckung

Testfalldokumentation



FSS.

Eine Testfallbeschreibung sollte mindestens die folgende Aspekte abdecken (angelehnt an [IMB10]):

Test-ID	Eindeutige und beständige Nummer
Quelle	Referenz zum Requirement
Priorität	sollte angelehnt an der Priorität der Testobjekte gestaltet werden
Testsystem / Release	wichtig zur Nachverfolgung
Beschreibung	Beschreibung des Testziels / der zu testenden Systemeigenschaft
Vorbedingungen	notwendiger Status des Testobjektes
Testschritte	Aktionen im Testfall
Testdaten	Liste der Eingabedaten
Sollreaktion	erwartete Ergebnisse

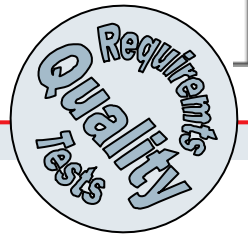
Detaillierungsgrad der Dokumentation für Anforderungen und Testfälle

Wann ist eine Dokumentation von Anforderungen und Testfällen ausreichend?

Wenn die Stakeholder des Projektes diese nachvollziehen können (also z.B. auch Wirtschaftsprüfer, Management oder eine interne QS-Abteilung).

Ein zu geringer Detaillierungsgrad erhöht das Risiko fehlerhafter Ausführung in Entwicklung und Test, ein zu hoher Detaillierungsgrad demotiviert die Beteiligten, lenkt von wesentlichen Aspekten des Systems ab und erhöht unnötig den Aufwand.

Good-Practice-Ansätze für die Testdurchführung



FSS.

Wann ist ein Test nicht sinnvoll?

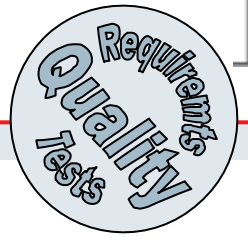
Test ist nicht sinnvoll

- wenn keine neuen Informationen gewonnen werden
- wenn gewonnene Informationen nicht erfasst bzw. genutzt werden
- wenn kein Risiko besteht, nicht zu testen

Die Tester und der Testmanager sollten möglichst objektiv anhand der festgelegten Teststrategie die Testfälle abarbeiten und Ihre Dokumentation anfertigen.

Die Erfassung gefundener Fehler sollte durch eine zentrale Instanz (bspw. den Testmanager) regelmäßig gegen die Konventionen des Testkonzeptes validiert werden.

Fazit



FSS.

Für viele Probleme der Anforderungserhebung und des Testens gibt es bereits geeignete und vielfach erprobte Werkzeuge.

Damit ist der Weg genau so steinig, wie wir unsere Projekte planen…

Vielen Dank für die Aufmerksamkeit!

Fragen?

Mirko Drobietz

FSS CONSULTING GMBH

Appelstraße 20 • 30167 Hannover

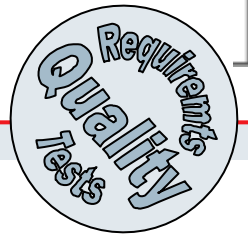
+ 49 - (0)172 - 54 00 378

mirko.drobietz@fss.de

Tel. + 49 - (0)511 - 16 99 66 66

Fax + 49 - (0)511 - 16 99 66 99

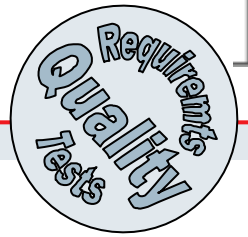
Zusammenfassung der Tipps 1/3



FSS.

- **Vorgehensmodell abhängig von Problem, der Branche und dem Team auswählen**
  Man kann in beiden Paradigmen erfolgreich Software entwickeln … oder einen Misserfolg erfahren.
- **die Chance des Projekterfolges kann durch die richtige Mischung an Rollen und Charakteren gesteigert werden**
  Eine einseitige Besetzung oder die Ausstattung der Rollen mit den falschen Charakteren wird den Projekterfolgt meist behindern.
- **Qualitätsanforderungen und auch Randbedingungen sind größtenteils ebenfalls vom Fachbereich festzulegen**
  Daher ist es geschickt, sich von dem Begriff der Nicht-Funktionalen Anforderungen zu trennen, da so oft beim Fachbereich der Eindruck entsteht, hierfür wäre der technische Bereich zuständig.
- **eine Anforderungsdefinition, die auf einige Anforderungsaspekte verzichtet ist sehr wahrscheinlich unvollständig**
  Es lohnt sich meistens, hier vor Implementierungsbeginn noch einmal nachzuforschen.
- **Liste für Qualitätskriterien an Anforderungen gemäß IEEE 830 als kurze Checkliste verwenden, wenn keine vorgegebene Struktur für das Projekt existiert**
- **menschliches Wissen ist in drei Ebenen angesiedelt, die bei der Anforderungserhebung zu berücksichtigen sind**
  Es ist nicht möglich, alle Ebenen mit den gleichen Erhebungstechniken zu erreichen.
- **natürlichsprachliche Defekte in Prosa-Dokumentation vermeiden**
  Aufgrund der aufgezeigten Probleme sollten die Prosa-Dokumentationen durch eindeutigere Beschreibungen (bspw. Formeln, Metacoding oder Modelle) ergänzt werden und auf natürlichsprachliche Defekte überprüft werden.
- **Modelle in der Dokumentation verwenden, wenn sie im Kontext der Stakeholder geeignet sind**
  In manchen Fällen ist die Festlegung von Satzschablonen oder strukturierten Textblöcken zielführender.

Zusammenfassung der Tipps 2/3

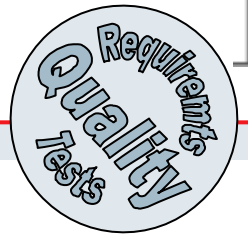


- **Konflikte in den Anforderungen sollten so früh wie möglich erkannt und wenn möglich aufgelöst werden**
  Meistens funktioniert die Erkennung gut (z.B. durch das Wissen im Team oder Stakeholderanalysen), aber es mangelt an der Auflösung oder zumindest Strategien mit den Konflikten umzugehen. Wichtig ist dabei auch das Bewusstsein, dass Konflikte je nach Typus unterschiedlich aufgelöst werden müssen (bspw. Sach- , Interessen- , Werte- , Beziehungs- oder Strukturkonflikte).
- **für die eindeutige Terminologie ein Glossar anlegen und pflegen**
  Oft wird ein Glossar mit Begriffen begonnen die jedem klar sind, in Frage gestellt und nach kurzer Zeit nicht weiterverfolgt. Statt dessen sollten die wirklich wichtigen Begriffe erläutert werden und in der Dokumentation auf Synonyme möglichst verzichtet werden.
- **Prototyping richtig verstanden eignet sich gut zur Anforderungserhebung oberflächenbasierter Systeme**
  Aber…

  - ein Prototyp muss mit wenig Aufwand entstehen (oft genügt eine Zeichnung einer Oberfläche) denn

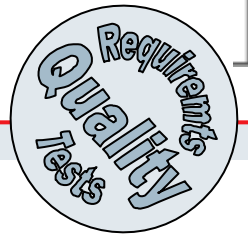
  - ein Prototyp kann auch obsolet und damit nicht weiterverfolgt werden
- **Fehler- oder QS-Kosten nicht isoliert betrachten sondern in Kombination**
  Um diesem Paradigma Rechnung zu tragen, sollten für die Einhaltung der QS-Maßnahmen pragmatische anstelle von dogmatischen Ansätzen genutzt werden. Um dieses durchführen zu können ist eine Priorisierung der Anforderungen und Testobjekte notwendig.
- **Bildung systematischer Testfälle durch Nutzung der entsprechenden Entwurfstechniken für Black- oder White-Box-Testing**
  Oft werden die systematischen Ansätze zum Testfallentwurf nicht benutzt. Sie verringern aber das Risiko nicht abgedeckter Systemaspekte deutlich.

Zusammenfassung der Tipps 3/3



- **Testdokumentation möglichst objektiv und ohne Wertung anfertigen**
  Jede Form von Spekulation oder Wertung verstärkt den rollenbasierten destruktiven Eindruck und erschwert die Arbeit des Testteams.
- **Erfassung gefundener Fehler regelmäßig durch eine zentrale Instanz validieren**
  Besonders zum Entwicklungsende hin neigen alle Fehler dazu, einsatzverhindernd zu werden…

Literatur



FSS.

[MYR91] Myers, G.J., Methodisches Testen von Programmen, 4. Auflage Oldenbourg Verlag, München/Wien, 1991

[IMB07] Imbus AG, Definition laut Lehrplan, http://www.imbus.de/glossar/, 2007

[IMB10] Imbus AG, Unterlagen zur Zertifizierungsschulung CTFL

[SOP09] Pohl/Rupp, Basiswissen Requirements Engineering, dpunkt Verlag, 2009

[BEL93] Belbin, R. Meredith, Team Roles At Work. Oxford. Butterworh Heinemann, 1993

[HÄL07] Hölzing, Die Kano-Theorie der Kundenzufriedenheitsmessung, Gabler Verlag, 2007

weitere Literaturvorschläge:

Chris Rupp & die SOPHISTen Requirements-Engineering und -Management, 5. Auflage, Hanser Verlag, 2009