Fachhochschule Wiener Neustadt
Wirtschaft • Technik • Gesundheit • Sicherheit • Sport

Requirements Management und Modellierung - VO

Einheit 3 – WS 2013/2014

Dipl.-Ing. Mag. Dr. Michael Tesar
michael.tesar@fhwn.ac.at

Terminplan und LV-Inhalte

	Datum	Uhrzeit	Ein-heiten	Inhalt
VO – 01	Mo., 14.10.2013	17:30 – 21:00	4	Einführung in die Lehrveranstaltung. Einführung in das Requirements Engineering Ziele und Modelle, Arten von Anforderungen
VO – 02	Mo., 21.10.2013	17:30 – 21:00	4	Anforderungen ermitteln Rollen, Faktoren, Techniken Anforderungen formulieren Vorbereitungen zur guten Dokumentation
VO – 03	Mo., 28.10.2013	17:30 – 21:00	4	Anforderungen validieren Prüftechniken Qualitätsmetriken
VO – 04	Mi., 13.11.2013	17:30 – 21:00	4	Versionsmanagement Change- und Releasemanagement Wiederverwendung von Anforderungen Arbeiten in verteilten Projektteams
PR	**Mi., 27.11.2013**	**17:30 – 18:30**	**1**	Schriftliche Prüfung über die Inhalte der Vorlesung
UE – 01	Mi., 27.11.2013	19:00 – 21:00	2	1. Übungseinheit

12 Schritte im RE-Prozess (Ebert)

1. Identifizierung der verschiedenen Interessensgruppen
2. Definition der Vision und Zielsetzung
3. Ermittlung der Anforderungen
4. Strukturierte Spezifikation der Anforderungen
5. Beschreibung der Systemumgebung
6. Bestimmung des Lösungsraums

Vgl. Ebert, 2010, S. 157 ff

12 Schritte im RE-Prozess (Ebert)

7. Verhandlung und Beschreibung der Kundenkontakte
8. Vorläufige Analyse der Anforderungen
9. Klassifikation und Priorisierung der Anforderungen
10. Prüfung der Anforderungen
11. Modellierung des Problemraums
12. Modellierung und Beschreibung des Lösungsraums

Vgl. Ebert, 2010, S. 157 ff

Anforderungsspezifikationen nach IEEE

> Aus dem Jahre 1998

> Einfacher, klassischer Aufbau

> Ähnlichkeiten zu Zielschablonen

> Quelle:

IEEE Standard 830 – 1998

Recommended Practice for Software Requirements Specifications. IEEE, New York, USA, 1998

Aufbau einer Spezifikation nach IEEE

1. Einführung
2. Beschreibung
3. Spezifische Anforderungen

Anhänge

Index

IEEE 830: Einführung

1. Einführung
    1. Zweck
    2. Anwendungsbereich, Marktanforderungen
    3. Definitionen, Akronyme, Abkürzungen
    4. Referenzen
    5. Übersicht

IEEE 830: Beschreibung

2. Beschreibung
    1. Produktsicht (z.B. Systemschnittstellen, Benutzerschnittstellen, Hardware- und Software-Schnittstellen)
    2. Funktionen
    3. Benutzer, Profile
    4. Einschränkungen, Qualitätsanforderungen (z.B. Zuverlässigkeit, Sicherheit, Protokolle, Hardwareeinschränkungen, Algorithmen)
    5. Annahmen und Abhängigkeiten

IEEE 830: Spezifische Anforderungen

3. Spezifische Anforderungen (Struktur nicht vorgegeben)
   1. Funktionsdetails (z.B. Gültigkeitsprüfungen, externe Szenarios und Use Cases, spezifische Use Cases nach Benutzergruppen sortiert)
   2. System und Architektur (z.B. Spezifikation der Systemanforderungen, Datenbankdetails, Datenmodelle, Systemmodelle, Evolution des Systems)
   3. Standards
   4. Detaillierungen zu Qualitätsanforderungen

Informeller unstrukturierter Text

Beispiel

…

3.2. Benutzerfunktionen des Aufzugs

Die Stockwerkskonsole erlaubt den Ruf des Aufzugs in den Stockwerken. Die Konsole besteht aus zwei Ruftasten, mit denen die Auf- beziehungsweise Abwärtsrichtung signalisiert werden kann. Der Benutzer ruft den Aufzug durch das Drücken der Ruftaste. Die betätigte Richtungswahltaste wird beleuchtet, um ihren aktivierten Zustand zu kennzeichnen. Zwei Paare von Pfeilsymbolen in der Stockwerkskonsole zeigen dem Benutzer die Kabine und deren Fahrtrichtung, die aktuell auf sein Stockwerk zufährt.

…

Vgl. Ebert, 2010, S. 170

Strukturierte Spezifikation

Beispiel

…

3.2. Benutzerfunktionen des Aufzugs

3.2.1. Der Aufzug wird mit der Stockwerkskonsole in den Stockwerken gerufen. Die Auf- beziehungsweise Abwärtsrichtung wird durch zwei Ruftasten in der Konsole signalisiert. Der Aufzug wird durch das Drücken der Ruftaste gerufen. Die betätigte Richtungswahltaste wird beleuchtet, um ihren aktivierten Zustand zu kennzeichnen.

Begründung: Der Benutzer muss unmittelbar erkennen können, dass der Aufzug seinen Ruf angenommen hat.

Lösungsspezifikation: Kap. 5.3.

Systemarchitektur: Kap. 4.2.

Entwurf: ~/xyz/elevator/elevator-1000/design

…

Vgl. Ebert, 2010, S. 170

Strukturierte detaillierte Spezifikation

Beispiel

…

3.2.1. Aufzug rufen

3.2.1.1. Der Aufzug wird mit der Stockwerkskonsole in den Stockwerken gerufen. Die Auf- beziehungsweise Abwärtsrichtung wird durch zwei Ruftasten in der Konsole signalisiert. Der Aufzug wird durch das Drücken der Ruftaste gerufen. Die betätigte Richtungswahltaste wird beleuchtet, um ihren aktivierten Zustand zu kennzeichnen.

3.2.1.2. Die Abfolge der Schritte ist wie folgt:

3.2.1.2.1. Der Benutzer drückt die Ruftaste der gewünschten Richtung.
3.2.1.2.2. Die Aufzugssteuerung registriert den Fahrtwunsch.
3.2.1.2.3. Die gedrückte Ruftaste leuchtet als Bestätigung.

3.2.1.3. Das gleichzeitige Drücken beider Ruftasten wird als Fahrtwunsch in beide Richtungen interpretiert.

3.2.1.4. Wiederholtes Drücken der Ruftaste hat keinen Einfluss.

Begründung: Der Benutzer muss unmittelbar erkennen können, dass der Aufzug seinen Ruf angenommen hat.

Lösungsspezifikation: Kap. 5.3.

…

Vgl. Ebert, 2010, S. 171

Halbformale Spezifikation

Beispiel

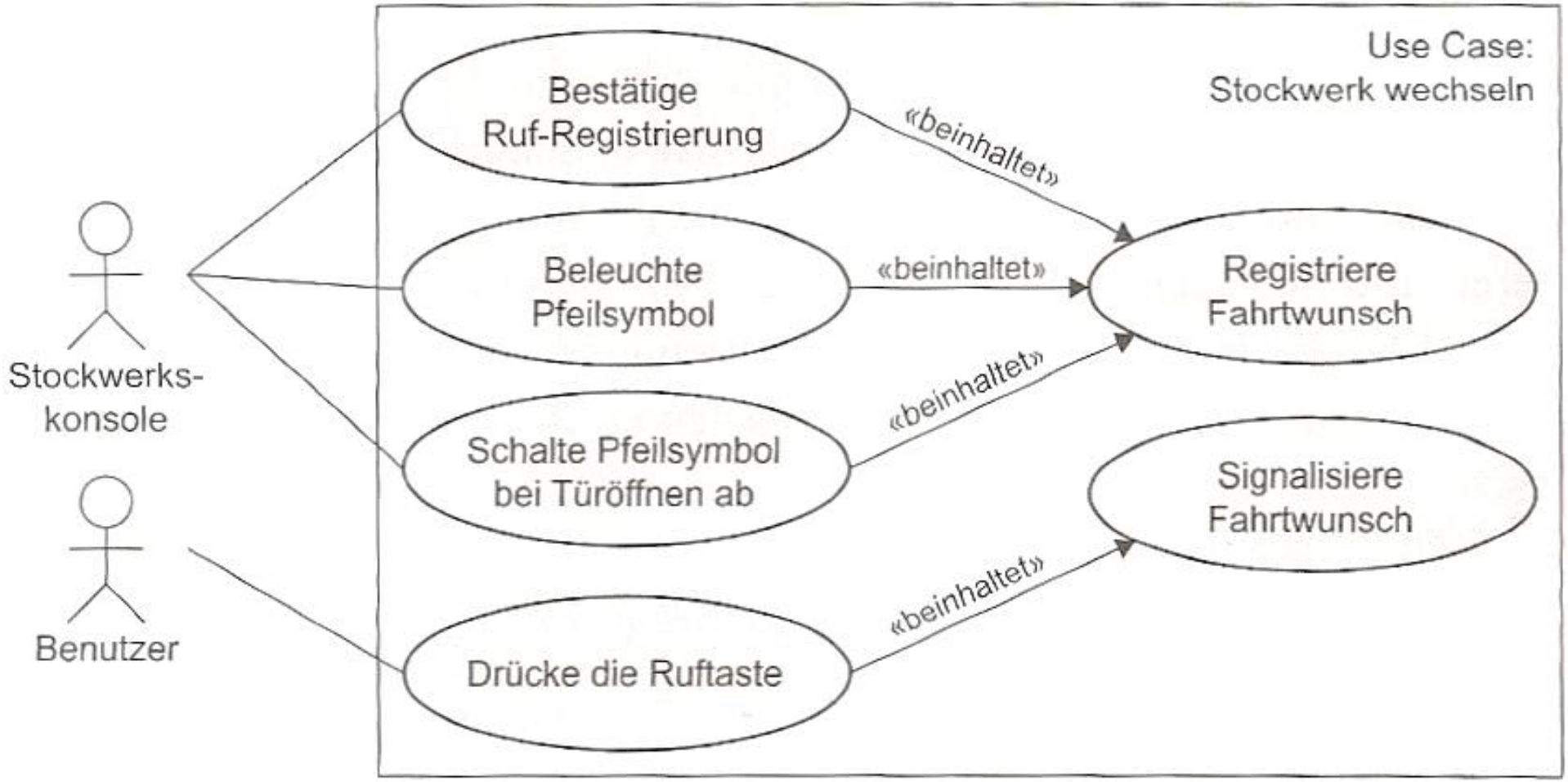
…

Spezifikation:	~/answering/anw-1000/design/3.2.1.2
Funktion:	Aufzug rufen
Beschreibung:	Der Aufzug wird durch das Drücken der Ruftastegerufen. Die betätigte Richtungswahltaste wird beleuchtet, um ihren aktivierten Zustand zu kennzeichnen.
Inputs:	Ruftaste.
Outputs:	Beleuchtung der Ruftaste.
Sequenz:	1. Der Benutzer drückt die Ruftaste der gewünschten Richtung. 2. Die Aufzugssteuerung registriert den Fahrtwunsch. 3. Die gedrückte Ruftaste leuchtet als Bestätigung.
Ausnahmen:	Im Feueralarmzustand erfolgt keine Bestätigung.
Vorbedingung:	Der Aufzug ist im aktiven Modus und hat das spezifische Stockwerk nicht registriert.
Nachbedingung:	Der Aufzug ist im aktiven Modus und hat das spezifische Stockwerk registriert.
Einschränkungen:	Das gleichzeitige Drücken beider Ruftasten wird als Fahrtwunsch in beide Richtungen interpretiert. Wiederholtes Drücken der Ruftaste hat keinen Einfluss.
Definition:	~/xyz/elevator/elevator-1000/design/24.11.

…

Vgl. Ebert, 2010, S. 172

Use-Case-Diagramm



Vgl. Ebert, 2010, S. 174

Formale Spezifikation

Beispiel

```
...
class ElevatorCall {
   // Stockwerktaste drücken und Aufzug rufen
   public void main (String args[]) {
      FloorButtonPoll {
         check.CallMode.valid () ;  // prüfe Vorbedingungen
         button = FloorButton.readFloorButton () ;
         while ( button = False && Alarm = False )
            // wait state
         ElevatorCall.initialize (Stockwerk) ;
         }
         ElevatorCall.Stockwerk = true ;
...
```

Vgl. Ebert, 2010, S. 175

Anforderungen formulieren II

Von der Anforderung…

… zur Spezifikation…

…zur Niederschrift.

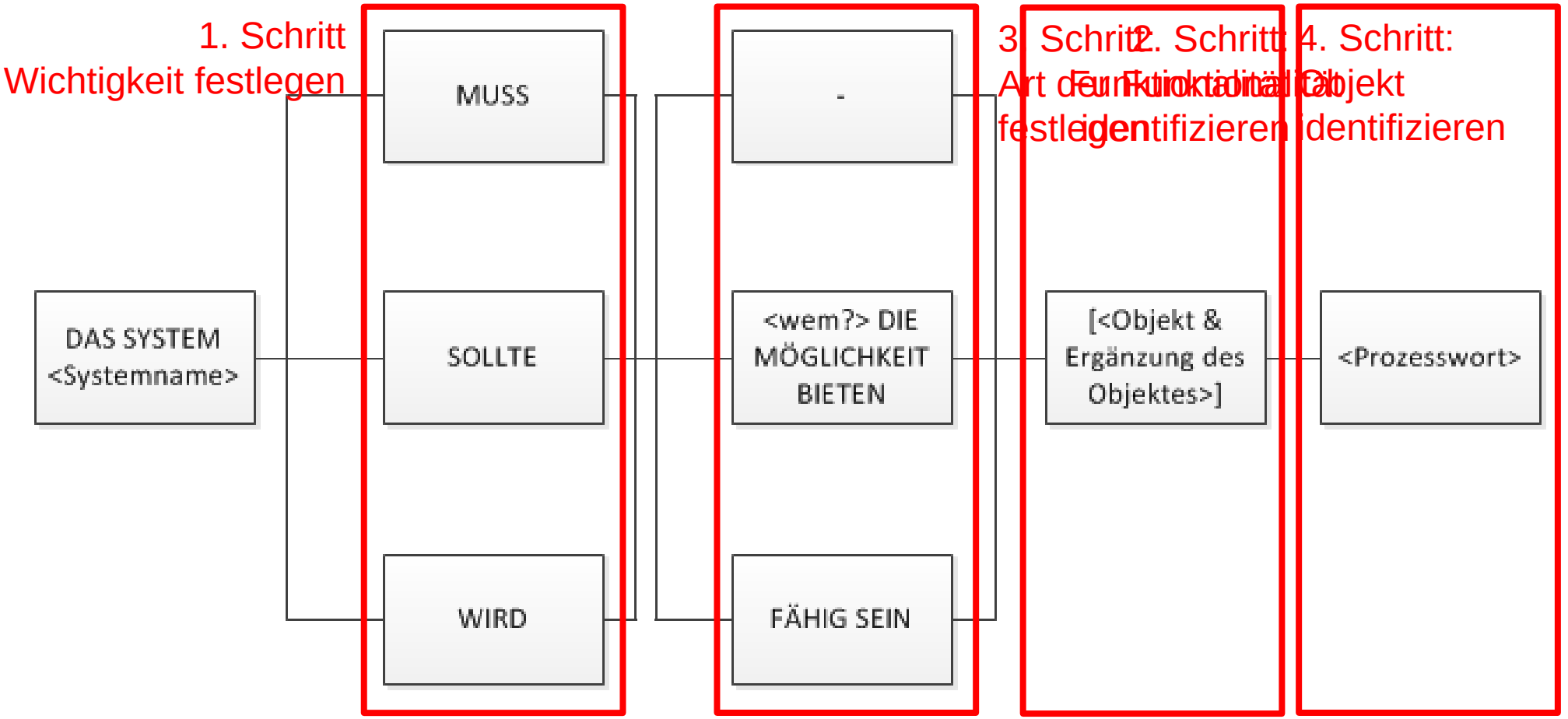
Anforderungsschablone nach Rupp

> *„Eine Anforderungsschablone ist ein Bauplan, der die Struktur eines einzelnen Anforderungssatzes festlegt.“*

1. Legen Sie fest, wie wichtig Ihnen die Anforderung ist.
2. Identifizieren Sie die geforderte Funktionalität
3. Legen Sie die Art der geforderten Funktionalität fest
4. Identifizieren Sie das Objekt und dessen Ergänzungen, für das die Funktionalität gefordert wird.

Vgl. Rupp, 2009 , S. 161 ff

Anforderungsschablone nach Rupp



Vgl. Rupp, 2009 , S. 161 ff

Unterschiedliche Systemaktivitäten

> Selbstständige Systemaktivität

DAS SYSTEM <Systemname> <muss|soll|wird> <Prozesswort>

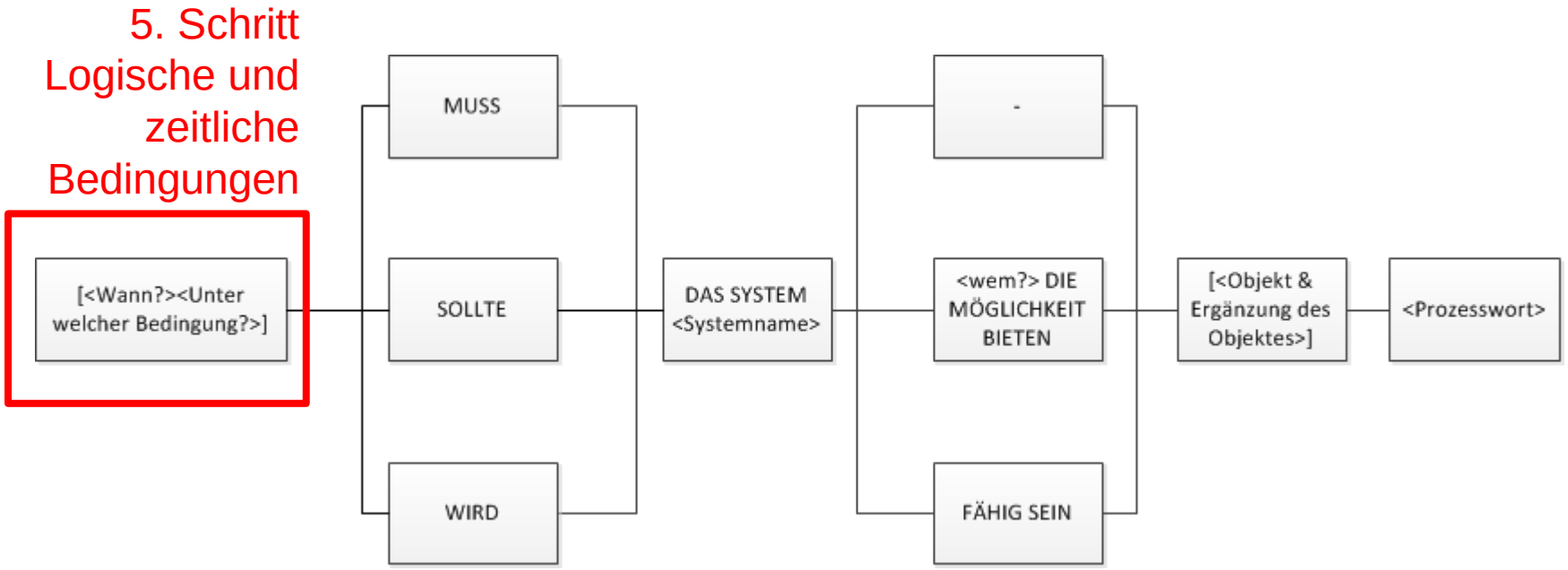
> Benutzerinteraktion

DAS SYSTEM <Systemname> <muss|soll|wird> <wem?> DIE MÖGLICHKEIT BIETEN <Prozesswort>

Vgl. Rupp, 2009 , S. 164

Anforderungsschablone nach Rupp II

5. Legen Sie die Bedingung fest, unter denen die geforderte Funktionalität durchgeführt wird.



Vgl. Rupp, 2009 , S. 166

Rechtliche Verbindlichkeiten

Rechtliche Verbindlichkeit	Schlüsselwort
Der Ausdruck *muss*, wird benutzt um verpflichtende Anforderungen zu definieren. • Die definierte Anforderung ist verpflichtend. • Die Erfüllung der Anforderung im Produkt ist verpflichtend. • Die Abnahme des Produktes kann verweigert werden, wenn einer *muss*-Anforderung nicht entsprochen wurde.	**muss**
Der Ausdruck *sollte*, wird benutzt um wünschenswerte Anforderungen zu definieren. Wünsche: • sind nicht verpflichtend. • müssen nicht erfüllt werden. • dienen der besseren Zusammenarbeit von Stakeholdern und Entwicklern. • erhöhen die Zufriedenheit der Stakeholder.	**sollte**
Der Ausdruck *wird*, wird benutzt, um Anforderungen zu definieren, die in der Zukunft integriert werden. • Zukünftige Anforderungen sind verpflichtend. • Sie helfen in der aktuellen Lösung Vorbereitungen zu treffen, um Zukünftiges später optimal zu integrieren.	**wird**

Vgl. Rupp, 2009 , S. 168

Verben und Prozesswörter

> Synonym
  - … ein Wort, das mit einem anderen oder mehreren anderen Wörtern derselben Sprache bedeutungsgleich ist.
  - z.B. Schicken, Übermitteln, Senden, Übertragen

> Homonym
  - … ein Wort, das in derselben Sprache mehrere unterschiedliche Bedeutungen besitzt.
  - z.B. Absetzen

Vgl. Rupp, 2009 , S. 170

Schablone: Prozessworte

Prozesswort	Semantische Definition des Prozesswortes	Synonyme
Speichern	Im Navigationssystem muss *speichern* den Prozess bedeuten, Informationen persistent aufzubewahren.	Sammeln, Aufbewahren
Auswählen	Im Navigationssystem muss *auswählen* den Prozess bedeuten, eines oder mehrere Elemente aus einer endlichen Menge von Elementen zu bestimmen.	Selektieren, Markieren
Anzeigen	Im Navigationssystem muss *anzeigen* den Prozess bedeuten, den Benutzern Informationen am Bildschirm darzustellen.	Darstellen, Ausgeben
Einfügen	Im Navigationssystem muss *einfügen* den Prozess bedeuten, neue Daten einzugeben oder vorhandene Daten zu überschreiben.	Eingeben, Einsetzen

Vgl. Rupp, 2009 , S. 170

Schablone: Glossar/Begriffsdefinition

Begriff	Semantische Definition des Begriffes	Synonyme
USB-Kabel	Im Navigationssystem muss *USB-Kabel* ein physisches Kabel bedeuten, das dem USB-Standard folgt und sowohl zur Datenübertragung als auch zur Stromversorgung geeignet ist.	Ladegerät, Datenkabel
Kunde	Im Navigationssystem muss *Kunde* eine natürliche Person bedeuten, die berechtigt ist, das Navigationsgerät zu nutzen.	Benutzer, Anwender
Persönliche Identifikations-nummer	Im Navigationssystem muss *Persönliche Identifikationsnummer* eine vierstellige Nummer, welche nur dem Kunden bekannt ist und zur Identifikation des Kunden benutzt wird, bedeuten.	Darstellen, Ausgeben

Vgl. Rupp, 2009 , S. 173

Schablone: Bedingungen

Bedingung	Logische Bedingung	Zeitliche Bedingung	Semantische Definition der Bedingung
FALLS	**X**	**-**	Das Wort *falls* bedeutet, dass das System eine Aufgabe bearbeitet, falls bestimmte logische Bedingungen wahr sind.
NACHDEM	**-**	**X**	Das Wort *nachdem* bedeutet, dass das System eine bereits gestartete Aufgabe erst vollständig abgearbeitet haben muss, bevor das System eine weitere Aufgabe beginnt.
SOBALD	**-**	**X**	Das Wort *sobald* bedeutet, dass das System eine bereits gestartete Aufgabe **nicht** vollständig abgearbeitet haben muss, bevor das System eine weitere Aufgabe beginnt.
SOLANGE	**-**	**X**	Das Wort *solange* bedeutet, dass das System eine bereits gestartete Aufgabe **nicht** vollständig abgearbeitet haben muss, bevor das System eine weitere Aufgabe beginnt. Die neu gestartete Aufgabe wird exakt solange ausgeführt, wie die vorher bereits gestartete Aufgabe andauert.

Vgl. Rupp, 2009 , S. 174

11 Regeln zur Formulierung

1. Schreiben Sie Szenarien in der *Gegenwartsform* (Präsens)
2. Schreiben Sie Szenarien in der *Aktivform*.
3. Formulieren Sie die Szenarien in der Form Subjekt-Prädikat-Objekt-Präpositionalgruppe (SPOPg).
4. *Vermeiden* Sie *Modalverben*.
5. *Trennen* Sie jede Interaktion deutlich von anderen Interaktionen.

können, sollen, wollen, müssen, mögen, dürfen

Vgl. Pohl, 2008, S. 151 ff

11 Regeln zur Formulierung

6. *Nummerieren* Sie die Szenarioschritte.
7. Nur eine *Interaktionsfolge* pro Szenario.
8. Beschreiben Sie Szenarien aus dem *Blickwinkel eines Außenstehenden*.
9. Benennen Sie explizit die *beteiligten Akteure*.
10. Benennen Sie das *Ziel* des Szenarios explizit.
11. Fokussieren Sie bei der Szenariobeschreibung auf die *Erfüllung bzw. Nichterfüllung* des Ziels.

Vgl. Pohl, 2008, S. 151 ff

Anforderungen validieren…

Lesen…

… verstehen…

…prüfen.

Häufige Fehler in Anforderungen I

> Anforderungen fehlen oder wurden falsch verstanden

> Anforderungen sind nach Änderungen inkonsistent

> Annahmen sind falsch

> Implementierung (wie?) wird beschrieben statt Anforderung (was?)

> Notationen falsch oder unzureichend genutzt

Vgl. Ebert, 2010, S. 236

Häufige Fehler in Anforderungen II

> Inkonsistente Terminologie oder Grammatik (Verbformen werden gemischt)

> Konjunktiv wurde eingesetzt

> Unklare Ausdrucksformen

> Schwammige oder nicht validierbare (messbare) Bezeichnungen

Vgl. Ebert, 2010, S. 236

Verifikation

„Die **Verifikation** prüft Ergebnisse hinsichtlich der Vorschriften zu Beginn der jeweiligen Phase (also, um die Dinge richtig zu tun).

Dazu werden die Anforderungen an den entsprechenden Prozessschritt, der dieses Arbeitsergebnis liefert, betrachtet und mit dem Ergebnis selbst verglichen."

Vgl. Ebert, 2010, S. 237

Validierung

„Die **Validierung** prüft die Ergebnisse der Phase daraufhin, ob sie die ursprünglichen (externen) Anforderungen erfüllen (also die richtigen Dinge tun).

Die Validierung ist eine externe Sicht und vergleicht Anforderungen an das Produkt mit den jeweils verfügbaren Ergebnissen."

Vgl. Ebert, 2010, S. 237

Qualitätskriterien für Anforderungen I

> Geschäftsnutzen
> Korrektheit
> Eindeutigkeit
> Verständlichkeit
> Vollständigkeit
> Konsistenz

Vgl. Ebert, 2010, S. 238

Qualitätskriterien für Anforderungen II

> Bewertbarkeit
> Prüfbarkeit
> Modifizierbarkeit
> Nachverfolgbarkeit
> Relevanz
> Realisierbarkeit

Vgl. Ebert, 2010, S. 239

„Nur wenn Ihnen die Qualität Ihrer Anforderungen bekannt ist, können Sie die Risiken für den weiteren Projektverlauf richtig abschätzen und Maßnahmen zielgerichtet einleiten."

Vlg. Rupp, S. 313

Anforderungen validieren…

Hilfsmittel,…

… Prüftechniken…

…und Qualitätsmetriken.

Standards und Vorlagen

> Einem Standard folgen (Vorlagen, Templates)

> Bei unterschiedlichen Anforderungsarten festlegen, welche Vorlagen genutzt werden

> Einhaltung ohne Ausnahmen

> Elektronische Überprüfung

- Nachverfolgbarkeit
- Verständlichkeit
- Formale Vollständigkeit

Vgl. Ebert

Reviews und Inspektionen

> Prüfen von Anforderungen auf formale Qualitätskriterien durch Checklisten (z.B. Fehlerlimit, Verständlichkeit)

> Planungswert: 1 bis 2 Fehler pro Seite Anforderungen

> Alle Szenarien und Abläufe durchspielen

> Verschiedene Perspektiven einsetzen

Vgl. Ebert

Missbrauchsszenarien

> Prüfen Sie Fälle und Situationen, die nicht eintreten dürfen.
> Analyse kritischer Anwendungsszenarien
> Was kann missverstanden werden?
> Wie würde ein Angreifer vorgehen?
> Welche Funktionen laden zum Missbrauch ein?
> Welche Fehler wären am schädlichsten?

Vgl. Ebert

Abnahmekriterien

> Spezifikation von konkreten Abnahmekriterien frühzeitig durchführen

> Sind keine Einschränkungen, sondern dienen der Unterstützung

> Werden auf jeden Fall in Testszenarien eingesetzt

> Dienen als Maßstab für die Abnahme durch den Kunden

Vgl. Ebert

Linguistische Analyse

> Verständlichkeit prüfen

> Formal

- Satzlänge
- Zahl der Substantive optimieren
- Zahl der Passivsätze
- Relativstrukturen reduzieren

> Formale Grammatiken, erlaubte

- Wörter
- Hilfsverben
- Modalstrukturen

Vgl. Ebert

Ermittlung und Quellen prüfen

> Ermittlungsverfahren prüfen

> Protokollanalyse

> „Problemrollen" in Ermittlungsverfahren identifizieren

- „Lautsprecher"
- „Gurus"
- Vorgesetzte

Vgl. Ebert

Glossar

> Annahmen zu Benutzerverhalten, Dokumenten etc. mit Glossar prüfen

> Unklarheiten klären und ggf. im Glossar „nachziehen“

> Glossar muss die einzige zentrale Anlaufstelle für Fachbegriffe sein

Vgl. Ebert

Analyse von Abhängigkeiten

> Gegenseitige Einwirkungen von Anforderungen oder Kombinationen von Anforderungen durch Abhängigkeitsprüfungen prüfen

> Quality Function Deployment
Abbildung von Anforderungen auf Systemeigenschaften bzw. Systemkomponenten

Vgl. Ebert

Dokumentation der Nachverfolgbarkeit

> Anforderungen abgebildet auf Lösungsbeschreibungen und Testfälle

> Jede Anforderung enthält Quelle und Erklärung, die zur Anforderung führte

> Qualitätsanforderungen und Randbedingungen wurden von denjenigen spezifiziert, die daran Interesse haben

Vgl. Ebert

Verbesserung der Klarheit

> Starten Sie nie Design / Entwicklung / Test mit unklaren Anforderungen

> Jede Frage muss geklärt werden!

> Keine eigenmächtigen Entscheidungen

> Eine *einzige* organisatorische Stelle, die alle Fragen klärt.

Vgl. Ebert

Benutzerdokumentation

> Frühzeitiges Erstellen der Benutzerdokumentation

> Ideale Prüfung einer Spezifikation

> Prüfung auf Vollständigkeit und Verständlichkeit sowie Korrektheit der Szenarien und Benutzungsfälle

> Berücksichtigung aller möglichen Benutzer

> AnfängerInnen vs. Fortgeschrittene

Vgl. Ebert

Wiederverwendung

> Sinnvoll bei Plattformen oder Produktlinien
> Anforderungen gelten auch im neuen Kontext?
> Bei externer Software Vorgaben der Komponenten als Einschränkungen spezifizieren
> Bei unsicherer Funktionalität rigorose Stresstests

Vgl. Ebert

Modelle

> Wesentliche Aspekte im Vordergrund
> Andere Aspekte im Hintergrund
> Zusammenhänge leicht erkennbar
> Modell dient zur Kommunikation zwischen Stakeholdern
> Modelle erlauben ein systematisches Durchgehen von Szenarien unter verschiedenen Perspektiven

Vgl. Ebert

Prototypen

> Bei vagen und ungenauen Anforderungen

> Funktionales Prototyping zur Identifikation von Antwortverhalten

> Technisches Prototyping hilft bei Klärung von offenen technischen Fragen

Vgl. Ebert

„Mittels Qualitätsmetriken können Sie die Qualität Ihrer Anforderungen oder Anforderungsdokumente für bestimmte Qualitätskriterien messen und eine objektive Aussage über das tatsächliche Qualitätsniveau erhalten."

Rupp, S.313

„Achten Sie darauf, dass Sie immer die Qualität solcher Aspekte messen, die für Ihr Projekt die größten Risiken darstellen. Dann können Sie die richtigen Entscheidungen […] optimal treffen."

Rupp, S.313

Definition: Qualitätsmetrik

*„Eine **Qualitätsmetrik** für Anforderungen oder Anforderungsdokumente ist eine Funktion, die eine qualitative Kenngröße einer Anforderung oder eines Anforderungsdokuments in einen Zahlenwert abbildet. Der berechnete Wert ist interpretierbar als der Erfüllungsgrad einer entsprechenden Qualitätseigenschaft der Prüfeinheit.“*

Vgl. Rupp, S. 314 in Anlehnung an den IEEE Standard for Software Quality Metrics, IEEE 1061 - 1998

Einsatzzwecke für Qualitätsmetriken

> Bewertung
  - Anforderungsqualität auf ein bestimmtes Qualitätsmerkmal beurteilen
  - Eindeutigkeit oder Vollständigkeit der Anforderungen bzw. Anforderungsdokumentation

> Vorhersage
  - Aus dem Messwert kann eine Vorhersage abgeleitet werden
  - Potenzielle Risiko für den weiteren Verlauf oder zu investierenden Aufwand

Vgl. Rupp

Zwei Kategorien von Qualitätsmetriken

> Inhaltsbasierte Qualitätsmetriken
  - Aussagen in Bezug auf sprachliche Formulierungen
  - Sowie Informationsgehalt

> Verwaltungsorientierte Qualtitätsmetriken
  - beziehen sich auf Weiterentwicklung und
  - Weiterverarbeitung

Vgl. Rupp

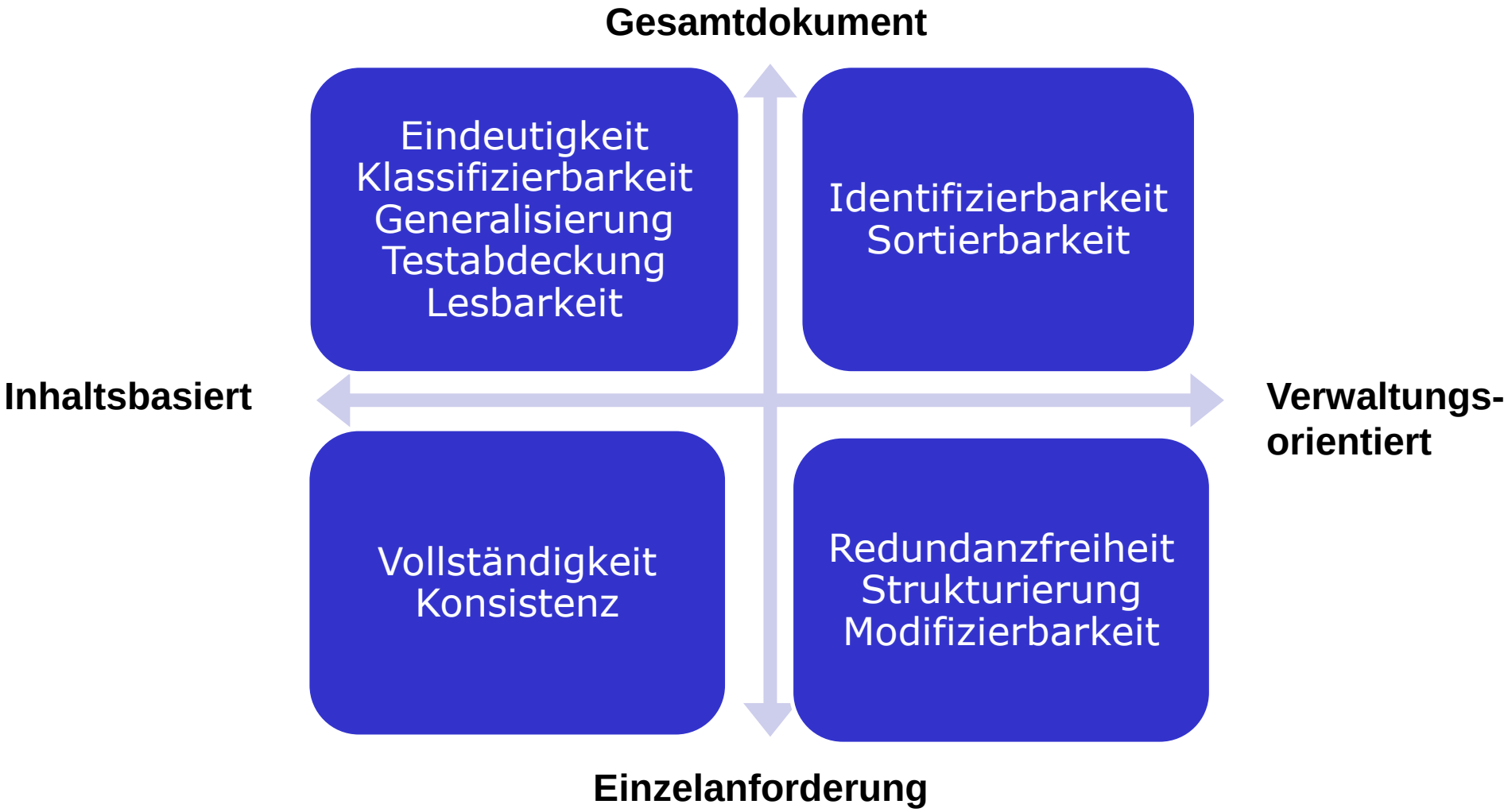
Zwei Ebenen der Qualitätsmetriken

> Ebene der Einzelanforderungen
  - Gültig für jede einzelne Anforderung / Anforderungsdokument
  - Ohne Kenntnis von Kontextinformationen einzusetzen

> Ebene der Gesamtspezifikation
  - Lassen sich nur mit Kenntnis
  - Und Verfügbarkeit der Gesamtspezifikation einsetzen

Vgl. Rupp

Die Qualitätsmetriken im Überblick



Vgl. Rupp

2 Qualitätsmetriken werden vorgestellt

> Inhaltsbasiert
  – Qualitätsmetrik „Eindeutigkeit“
> Verwaltungsorientiert
  – Qualitätsmetrik „Redundanzfreiheit“

Vgl. Rupp

Voraussetzungen

> Anzahl Anforderungssätze = Anzahl vermessener natürlichsprachlicher Sätze

> Erfüllungsgrad in Prozent: 0% … 100%

> Bei repräsentativer Stichprobe kann auf die Gesamtspezifikation „hochgerechnet" werden

Vgl. Rupp

„Eindeutigkeit“

> Messung Anteil der eindeutig formulierten Funktions- und/oder Eigenschaftsbeschreibungen

> Grad der Nicht-Interpretierbarkeit wird festgelegt

Vgl. Rupp

Formel „Eindeutigkeit“

$$Eindeutigkeit = \frac{u * PE + v * BPE + w * BE}{Anzahl\ Anforderungssätze} * 100$$

Gewichtungsfaktoren mit u + v+ w = 1
meist Schätz-/Erfahrungswerte

Prozessworteindeutigkeit (PE):
Sind alle in den Anforderungssätzen beschriebenen Prozessworte vollständig (eindeutig) definiert?

Bezugspunkteindeutigkeit (BPE):
Sind für alle Vergleiche oder Steigerungen, die in den Anforderungssätzen beschrieben werden, die jeweiligen Bezugspunkte explizit genannt?

Begriffseindeutigkeit (BE):
Sind alle Begriffe, die in den Anforderungssätzen verwendet werden, definiert?

Vgl. Rupp

Prozessworteindeutigkeit

> Ist der Anforderungssatz im Aktiv formuliert?

> Ist die Funktionalität durch ein Vollverb beschrieben (keine Nominalisierung, kein Funktionsverbgefüge, keine „schwammigen Prozessformulierung“)?

> Können alle wesentlichen W-Fragen zum Prozesswort durch die Informationen im Anforderungssatz beantwortet werden? (Was? Für wen oder was? Wann? …)

Vgl. Rupp

Bezugspunkteindeutigkeit

> Enthält der Anforderungssatz einen Vergleich bzw. eine Steigerung?

> Wenn ja, ist der Bezugspunkt für den Vergleich bzw. die Steigerung im Anforderungssatz spezifiziert?

Vgl. Rupp

Begriffseindeutigkeit

> Sind alle im Anforderungssatz verwendeten Substantive (OHNE Nominalisierungen) definiert und entsprechend der Definition verwendet? (im Glossar)

> Sind alle im Anforderungssatz verwendeten Abkürzungen definiert?

Vgl. Rupp

Bewertung

Für jeden Anforderungssatz A_i sind die PE, BPE und BE zu definieren und zwar:

$$PE(A_i), BPE(A_i), BE(A_i) = \begin{cases} 1 & \text{Wenn alle Fragen mit „JA“ beantwortet} \\ 0 & \text{sonst} \end{cases}$$

Vgl. Rupp

Gesamtbewertung

ergibt sich aus der Summe der Messwerte aller einzelnen Anforderungssätze

$$PE, BPE, BE = \sum_{i=1}^{n} PE(A_i), BPE(A_i), BE(A_i)$$

Vgl. Rupp

Beispiel

> A1: Während der Registrierung muss das Bibliothekssystem umfangreiche Prüfungen durchführen.

> A2: Das Bibliothekssystem muss dem Bibliothekar die Möglichkeit bieten, den Namen eines neu zu registrierenden Kunden einzugeben.

> A3: Falls die Plausibilitätsprüfung der eingegebenen Kundendaten erfolgreich war, kann das Bibliothekssystem die Kundendaten speichern.

Vgl. Rupp

Existierende Glossar enthält…

> Bibliothekssystem
> Bibliothekar
> Registrierung
> Plausibilitätsprüfung
> Kundendaten

Vgl. Rupp

Eindeutigkeits-Berechnung

	$PE(A_i)$	$BPE(A_i)$	$BE(A_i)$
A_1	0	0	1
A_2	1	1	0
A_3	1	0	1
PE, BPE, BE	2	1	2

$$Eindeutigkeit = \frac{0{,}4 * 2 + 0{,}2 * 1 + 0{,}4 * 2}{3} * 100 = 60\%$$

Vgl. Rupp

„Redundanzfreiheit“

> Misst den Anteil der redundanten Anforderungen im Anforderungsdokument
> Grad der Konsistenzstabilität im Falle von Änderungen
> Inkonsistenzen vorbeugen / vermeiden
> Gesamte Anforderungsdokument muss vorliegen
> Stichproben reichen im Regelfall aus

Vgl. Rupp

Redundanzfreiheit

$$Redundanzfreiheit = \left(1 - \frac{ArA}{Anzahl\ Anforderungssätze}\right) * 100$$

ArA:
Anzahl redundanter Anforderungssätze

Messgrößen und Formel

> Wird in diesem Anforderungssatz ein inhaltlich gleiches Verhalten wie in einem anderen Anforderungssatz beschrieben?

> Oder ist der Anforderungssatz genau identisch zu einem anderen Anforderungssatz?

> $ArA(A_i) = \begin{cases} 1 & \text{Wenn mind. eine Frage mit „JA“ beantwortet ist} \\ 0 & \text{sonst} \end{cases}$

> Die Originalanforderung ist mit 0 zu werten

> Die redundanten Anforderungen mit 1 bewerten

Beispiel

> A2: Das Bibliothekssystem soll dem Bibliothekar die Möglichkeit bieten, den Namen eines neu zu registrierenden Kunden eingeben.

> A3: Wenn die Plausibilitätsprüfung der eingegebenen Kundendaten erfolgreich war, kann das Bibliothekssystem die Kundendaten abspeichern.

> A5: Der Bibliothekar soll den Namen von neu zu registrierenden Kunden im Bibliothekssystem eingeben können.

Redundanzfreiheit-Berechnung

	$ArA(A_i)$
A_2	**0**
A_3	**0**
A_5	**1**
Summe	**1**

$$Redundanzfreiheit = \left(1 - \frac{1}{3}\right) * 100 = 66\%$$

Vgl. Rupp

Zusammenfassung

> Spezifikation

> Dokumentationsarten

- Prosa
- Szenarien

> Validierung

- Qualitätskriterien für Anforderungen
- Hilfsmittel und Prüftechniken
- Qualitätsmetriken

Ausblick

VO – 04	Mi., 13.11.2013	17:30 – 21:00	4	Versionsmanagement Change- und Releasemanagement Wiederverwendung von Anforderungen Arbeiten in verteilten Projektteams

Literatur

- Klaus Pohl; Requirements-Engineering: Grundlagen, Prinzipien, Techniken; dpunkt.verlag; 2008; 2. Auflage
- Christof Ebert; Systematisches Requirements Engineering: Anforderungen ermitteln, spezifizieren, analysieren und verwalten; dpunkt.verlag; 2010; 3. Auflage
- Tim Weilkiens; Systems Engineering mit SysML/UML: Modellierung, Analyse, Desig; dpunkt.verlag; 2008; 2. Auflage
- Chris Rupp; Requirements-Engineering und –Management: Professionelle, Iterative Anforderungsanalyse für die Praxis; Hanser Verlag; 2009; 5. Auflage
- Jutta Eckstein; Agile Softwareentwicklung mit verteilten Teams; dpunkt.verlag; 2009
- Uwe Vigenschow, Björn Schneider; Soft Skills für Softwareentwickler, Fragetechniken, Konfliktmanagement, Kommunikationstypen und –modelle; dpunkt.verlag; 2007
- Uwe Vigenschow,…; Soft Skills für IT-Führungskräfte und Projektleiter – Softwareentwickler führen und coachen, Hochleistungsteams aufbauen; dpunkt.verlag; 2009
- Marcus Grande; 100 Minuten für Anforderungsmanagement; Vieweg & Teubner, Springer Fachmedien; 2011
- McConnell S.; Aufwandschätzung bei Softwareprojekten; Microsoft Press Deutschland; 2006
- Klaus Pohl & Chris Rupp; Basiswissen Requirements Engineering; dpunkt.verlag; 3. korrigierte Auflage; 2011
- Helmut Balzert; Softwaretechnik: Basiskonzepte und Requirements Engineering; Spektrum Akademischer Verlag; 2009; 3. Auflage