
Best Practices im Requirements Engineering

Erhebung und Spezifikation (Nicht-)funktionaler Anforderungen

Michael Eisenbarth

Requirements und Usability Engineering Department

Michael.eisenbarth@iese.fraunhofer.de

Best Practices im Requirements Engineering

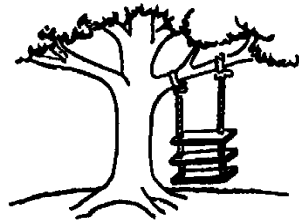
Agenda

- Motivation Requirements Engineering
- Funktionale Anforderungen
- Nicht-funktionale Anforderungen
- Anforderungsmanagement

Seite 2/X

Best Practices im Requirements Engineering

Kommunikation ist alles



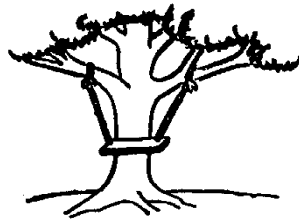
Das ist der Vorschlag
des Projektponsors



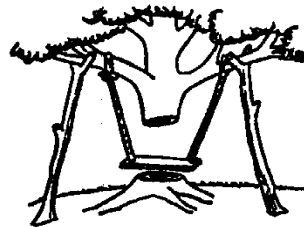
So ist es in der
Projektanfrage spezifiziert



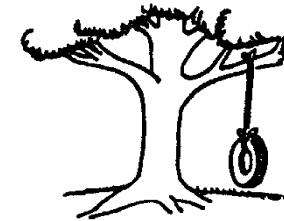
Das ist der Entwurf des
Senioranalysten



So wurde es von den
Programmierern erstellt



So wurde es beim
Benutzer installiert

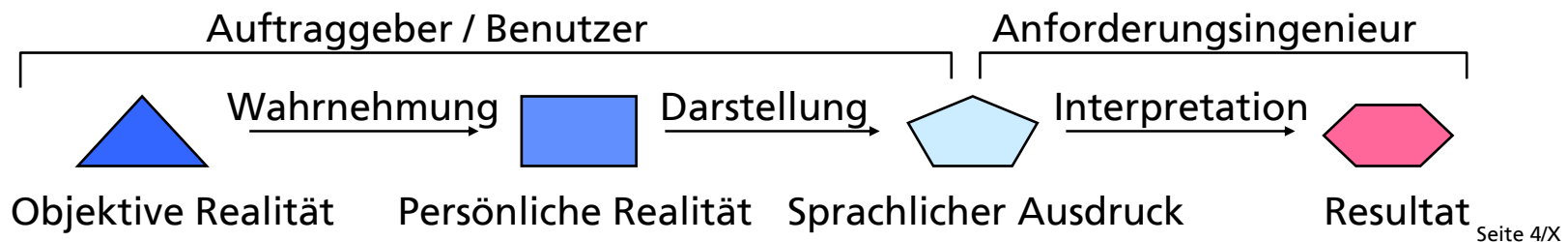


Und das wollte
der Benutzer eigentlich!

Seite 3/X

Ursachen von Problemen

- „The hardest single part of building a software system is deciding precisely what to build. [...]“ [Brooks, 1987]
- **Inhärente Probleme:**
 - Komplexität von Systemen
 - Menschliche Kommunikation
 - Sich permanent ändernde Anforderungen

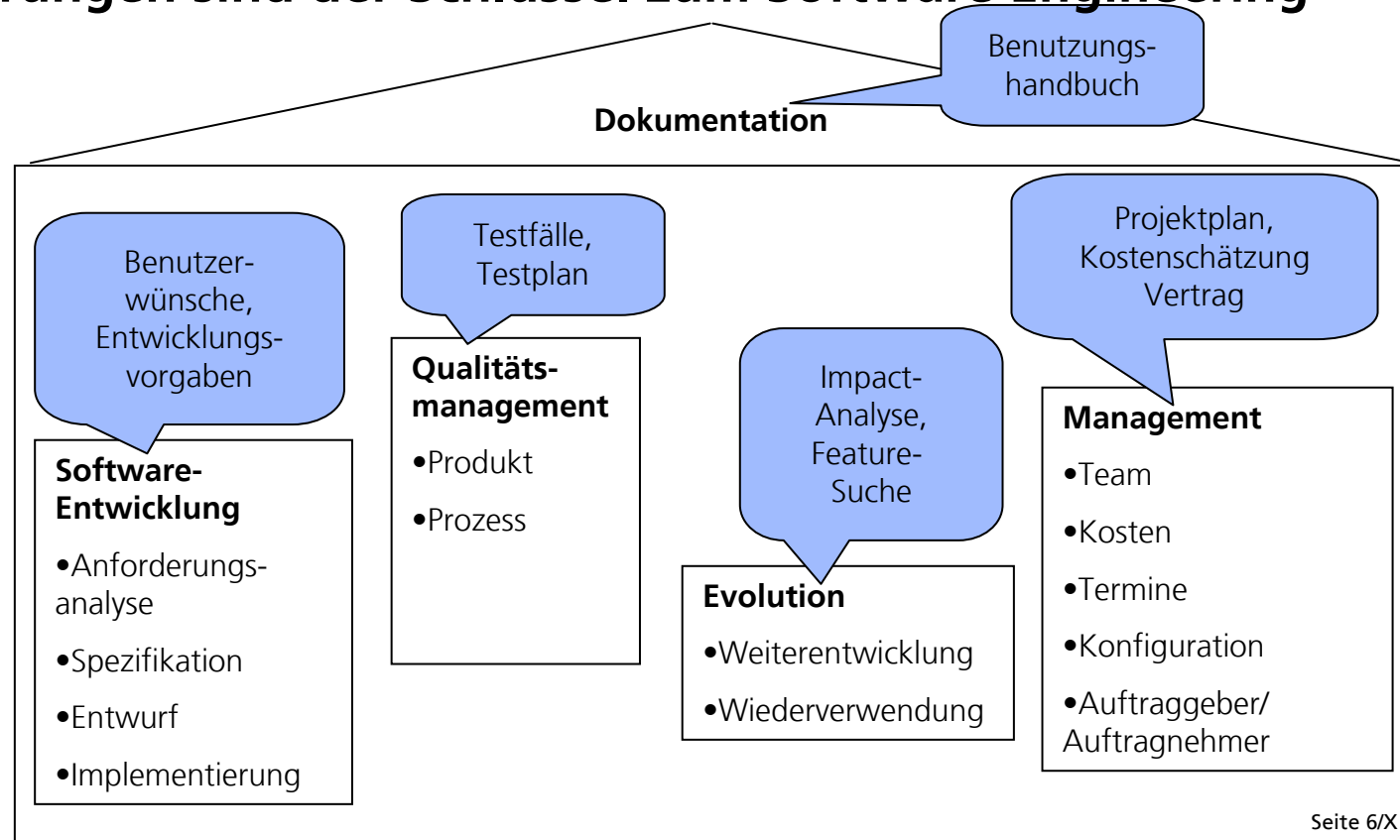


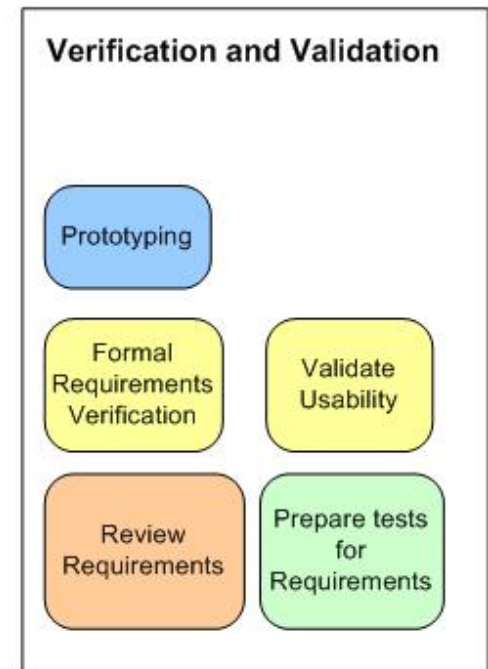
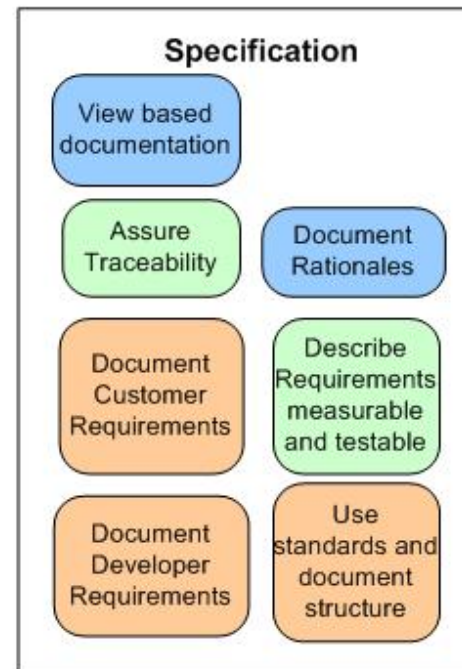
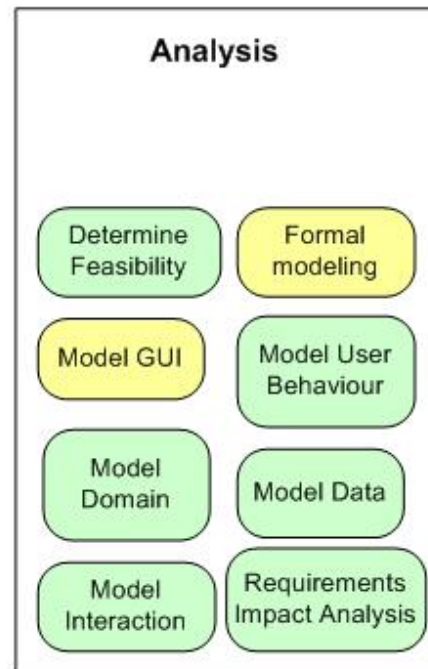
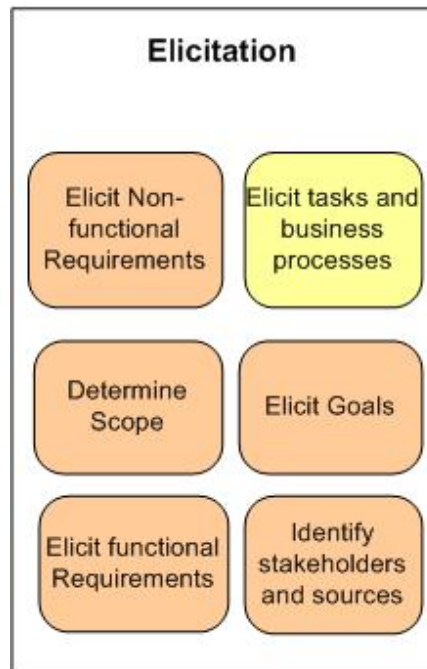
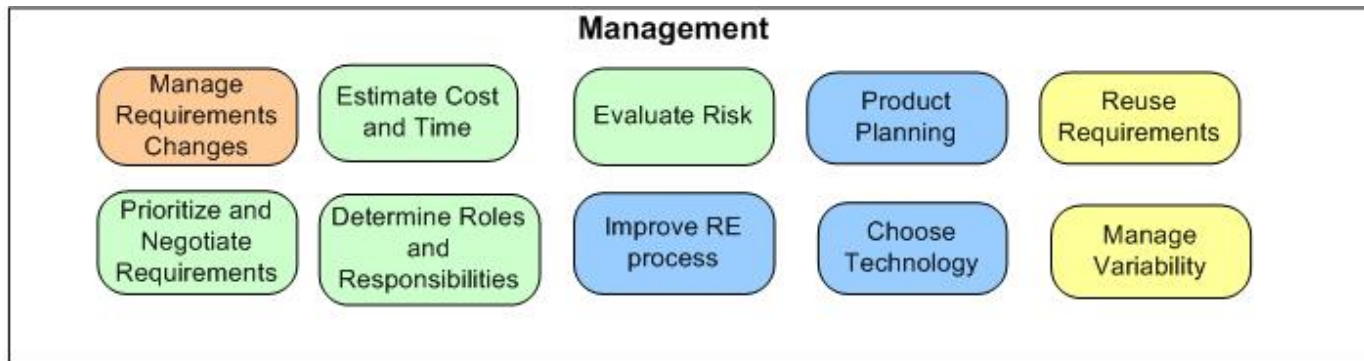
Vermeidbarkeit von Problemen mit systematischem RE

- **Vermeidbare Probleme:**
 - Nichtbeachtung von Benutzergruppen/ Projektbeteiligten
 - Unklare Anforderungen
 - Mangelnde Dokumentation/Verwaltung
 - Mangelnde Abstimmung System- und Softwareentwicklungsprozess
 - Keine Trennung von Benutzer- und Entwickleranforderungen
 - Vermischung mit Entwurfsentscheidungen
 - Unklare Verantwortlichkeiten bei der Realisierung
 - Mangelnder Bezug zum Testen

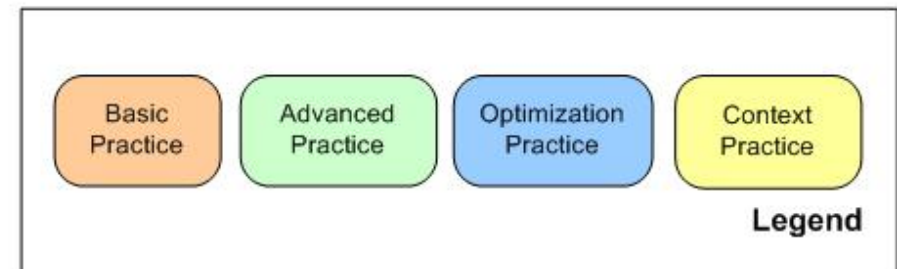
Seite 5/X

Anforderungen sind der Schlüssel zum Software Engineering



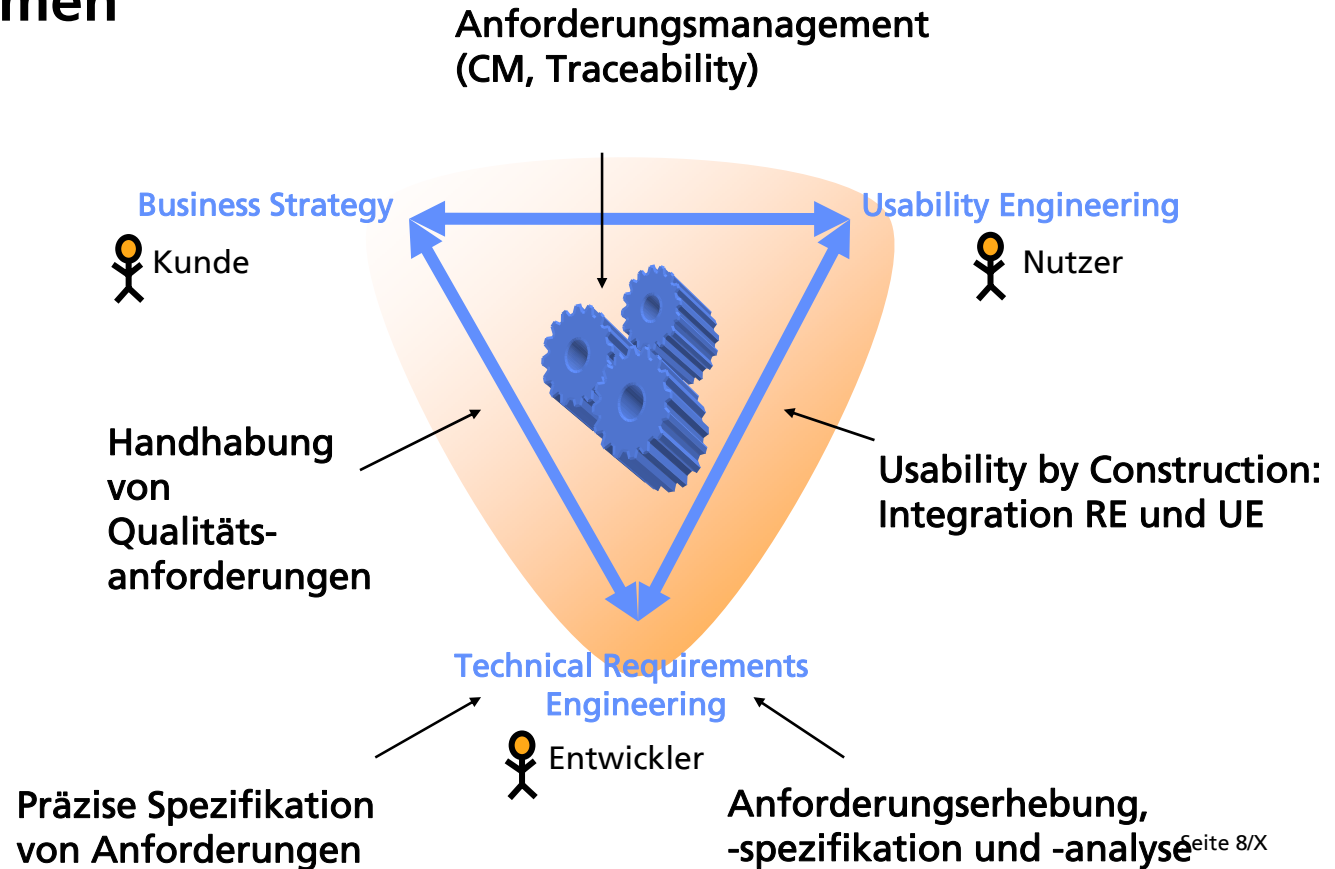


Best Practices in Requirements Engineering



Best Practices im Requirements Engineering

RUE Kernthemen



Best Practices im Requirements Engineering

Agenda

- Motivation Requirements Engineering
- **Funktionale Anforderungen**
- Nicht-funktionale Anforderungen
- Anforderungsmanagement

Seite 9/X

Best Practices im Requirements Engineering

Lastenheft (Anforderungsdefinition) vs. Pflichtenheft (Softwarespezifikation)

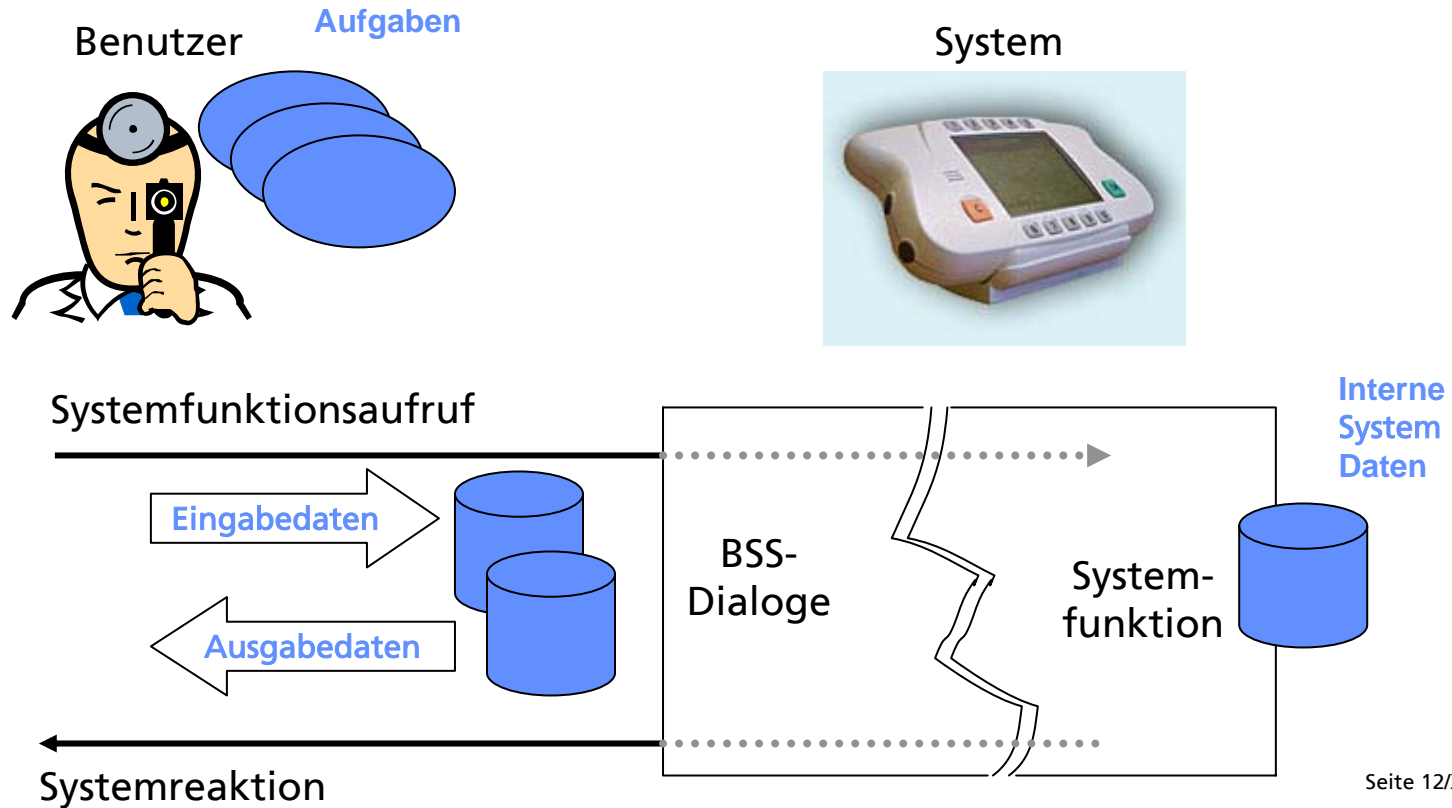
- Zusammenstellung aller Anforderungen des Auftraggebers hinsichtlich Liefer- und Leistungsumfang (VDI/VDE 3694)
- beschreibt die Anforderungen aus Anwendersicht
- beschreibt WAS und WOFÜR etwas zu erstellen ist
- wird oft vom Auftraggeber erstellt
- dient als Ausschreibungs-, Angebots- und Vertragsgrundlage
- Beschreibung der Realisierung aller Anforderungen des Lastenhefts (VDI/VDE 3694)
- enthält / detailliert das Lastenheft
- beschreibt die Vorgaben für die Entwicklung
- beschreibt WIE und WOMIT Anforderungen zu realisieren sind
- wird i.d.R. nach Auftragsteilung vom Auftragnehmer erstellt
- wird vom Auftraggeber genehmigt und dient dann als verbindliche Vereinbarung für die Realisierung

Seite 10/X

Standards

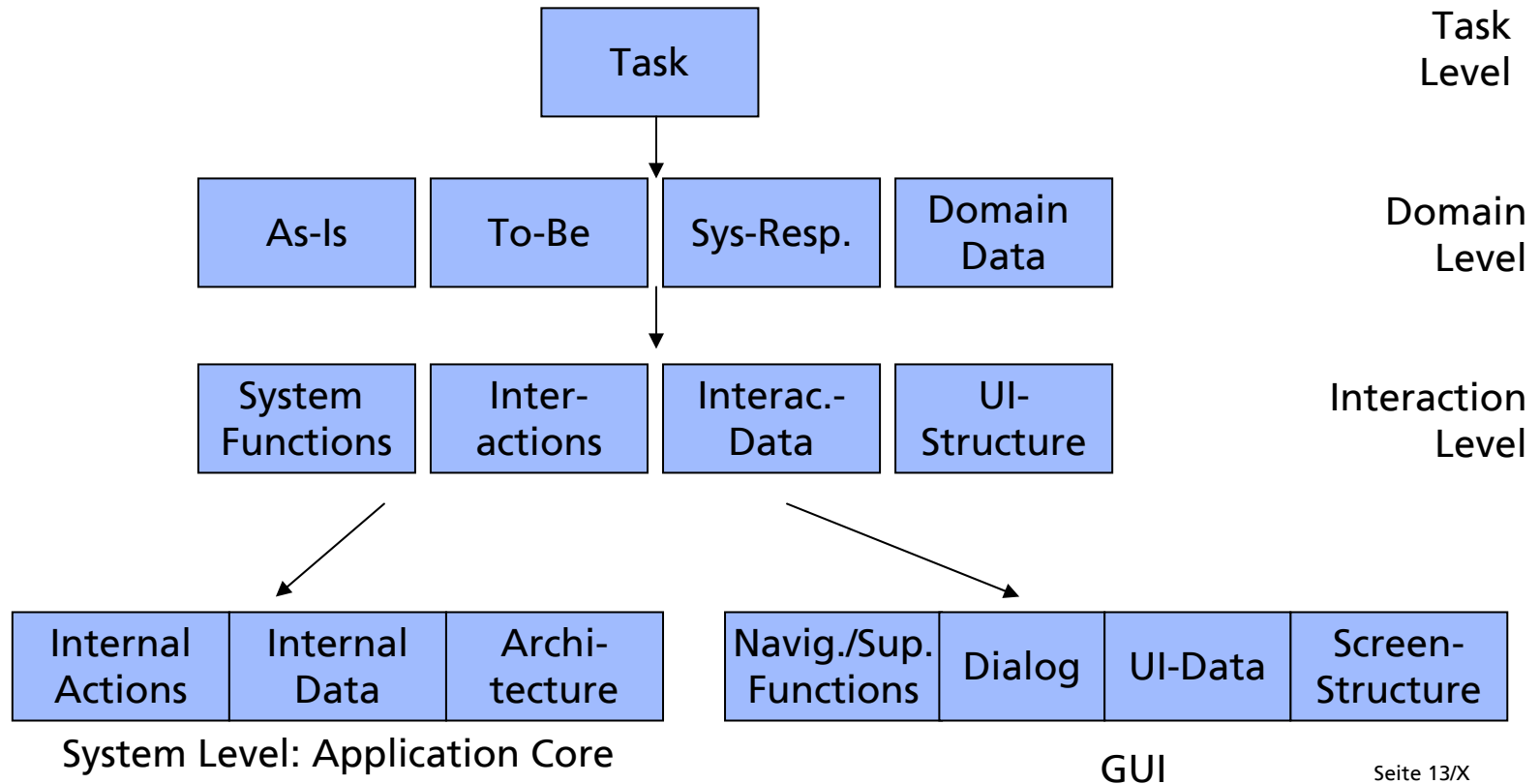
- **Standards bieten ein Mittel zur Strukturierung von Anforderungsdokumenten**
 - Besserer Überblick
 - Bessere Lesbarkeit
 - Insgesamt verbesserte Verständlichkeit
- **Standards geben Hinweise darauf, was eine Anforderungsspezifikation beinhalten sollte**
 - Sie bieten ein Mittel, um die Vollständigkeit zu erhöhen
 - Sie sind eine gute Ausgangsbasis, um sich ein Template für die Anforderungsspezifikation zu definieren
- **Standards geben nicht an, WIE verschiedene Teile zu spezifizieren sind bzw. WIE man die Eigenschaften eines guten Dokuments gewährleisten kann**
 - Keine Unterstützung bezüglich der Wahl der Notation zur Spezifizierung eines bestimmten Bereichs
 - Keine Unterstützung dafür, wie man bspw. Vollständigkeit oder Verfolgbarkeit erreichen kann

Aufgabenorientierte Anforderungsspezifikation



Seite 12/X

Welche Entscheidungsarten gibt es?



Beispiel für eine Persona im Umfeld Medizinsauger

Profil

Name:
Prof. Dr. med. Ziak

Alter: 50



Arbeitsumfeld: Eigene HNO-Praxis, Belegbetten, denkt über Zweitpraxis nach. Nicht allzu groß, schlicht, spärlicher Untersuchungsraum. Geht gelegentlich zum operieren in eine Uniklinik. Am liebsten Equipment mobil mitnehmen können.

Produktkenntnisse: Nur das nötigste. Hatte Einweisung. Hat Fortbildungsveranstaltungen besucht.

Patienten: Viele Sänger und Schauspieler. Würgreize bei Kontakt mit Mediastrob. Privatpatienten bevorzugt.

Häufigste Tätigkeiten (mit dem Produkt): Stimmlippendiagnose, Untersuchungen des Kehlkopfes, mit Videoarchivierung.

Wichtigste Tätigkeiten: Früherkennung von Kehlkopf-Karzinomen; Wiederherstellung von Stimmfunktion.

Seltene Tätigkeiten: Stimmanalyse;

Typische Hindernisse: Wenig bis gar keine PC-Kenntnisse. Mangelnde Praxis beim stroboskopieren. Kabel und Fußschalter-Wirrwarr. Rentabilität

Familiäres: Frau managed die Praxis; 2 Kinder, Tochter studiert und der Sohn soll eines Tages die Praxis übernehmen, hat allerdings ganz andere Pläne...

Sonstiges: Ist auf Expansion und Einfluß aus. Hat einen guten Steuerberater. Lebt in Saarbrücken.

Kerneigenschaften:

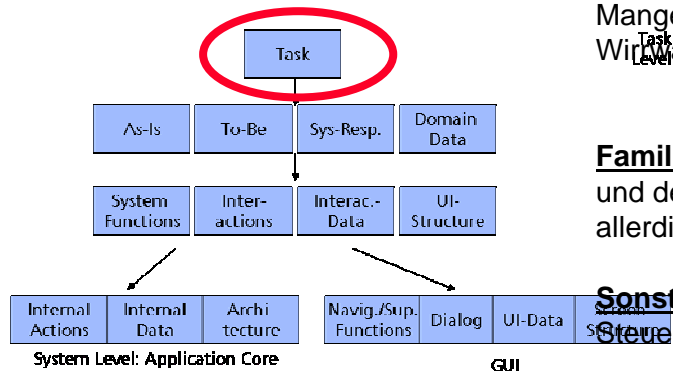
- Dominanz, Einfluß
- Unentschlossen, skeptisch
- sachlich, kühl und kompetent
- Schnäppchenjäger

Kernziele:

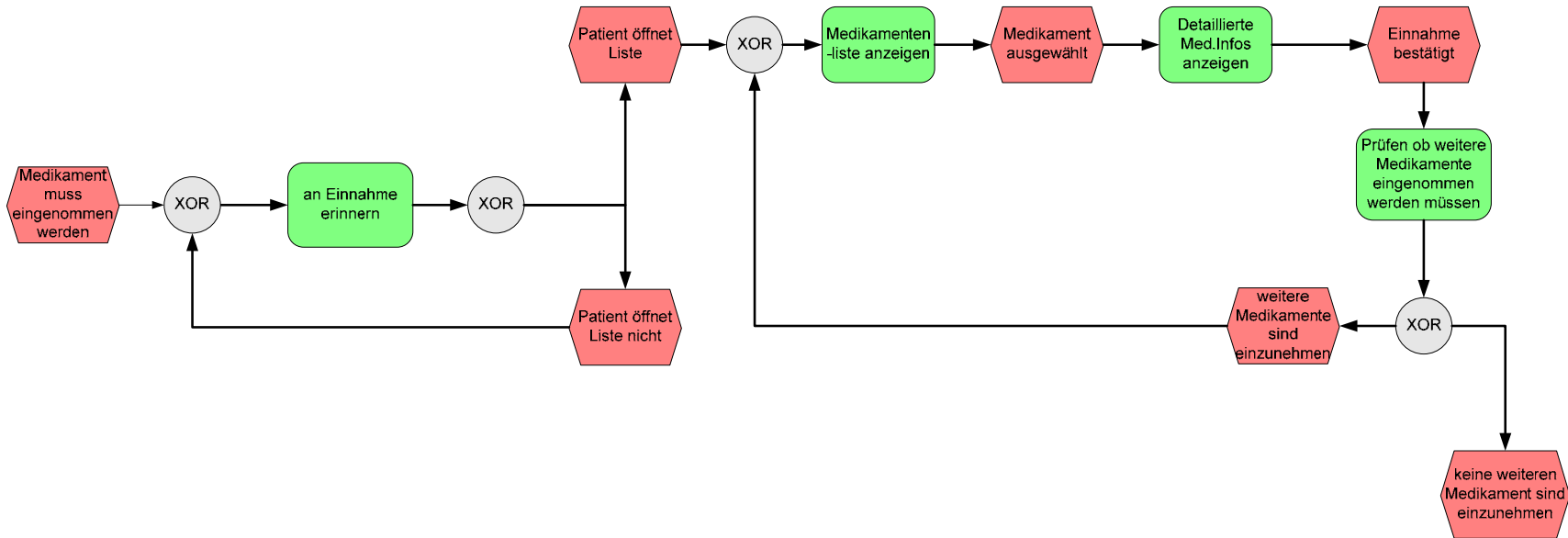
- Reputation
- Privatversichertes Klientel
- Geld verdienen

Motto:

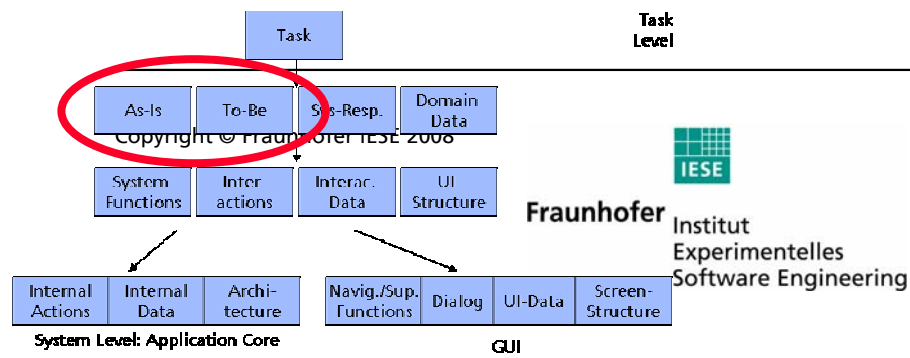
„Arbeiten und Geld verdienen“



Beispiel einer Aufgabenbeschreibung in EPK Notation

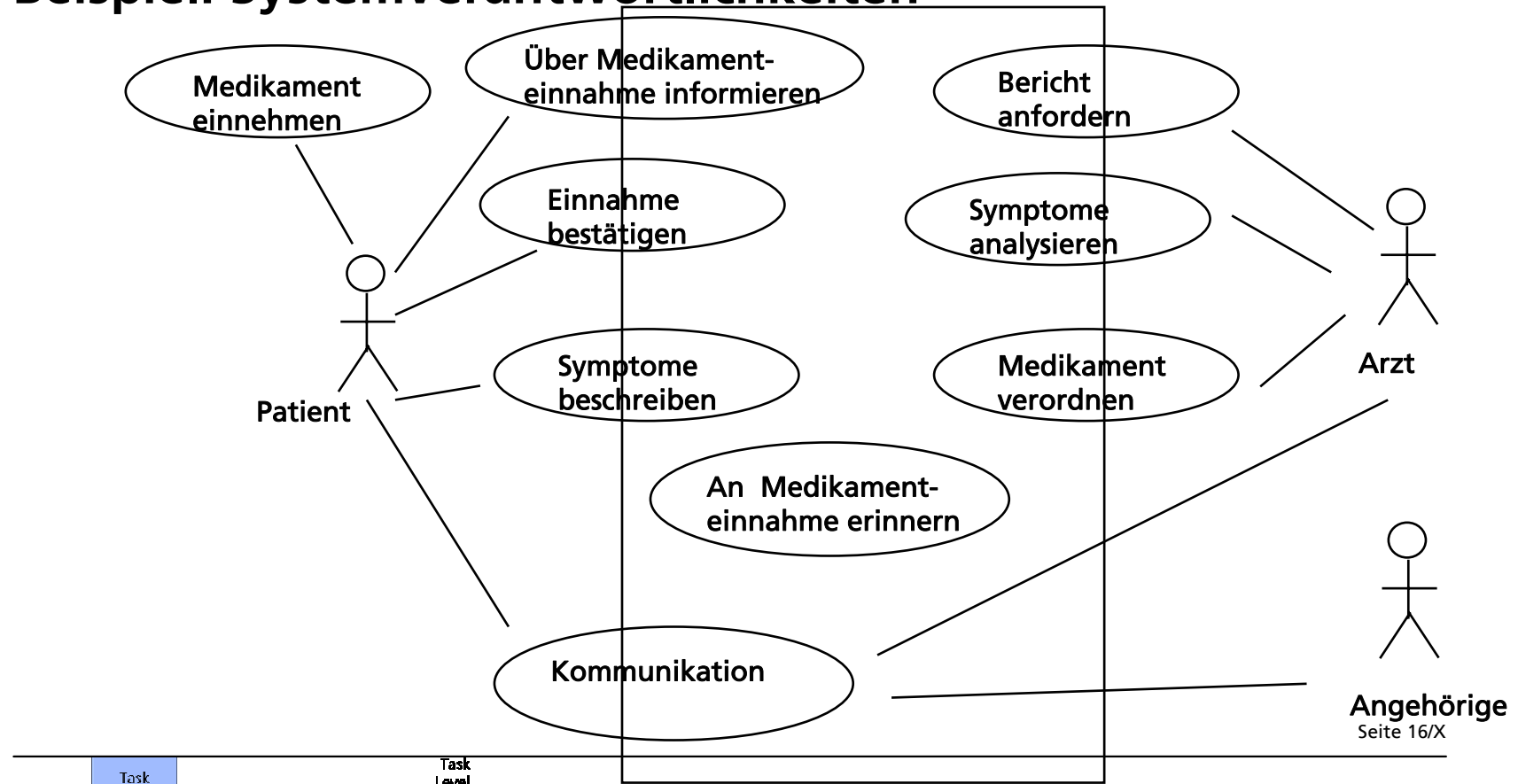


Seite 15/X

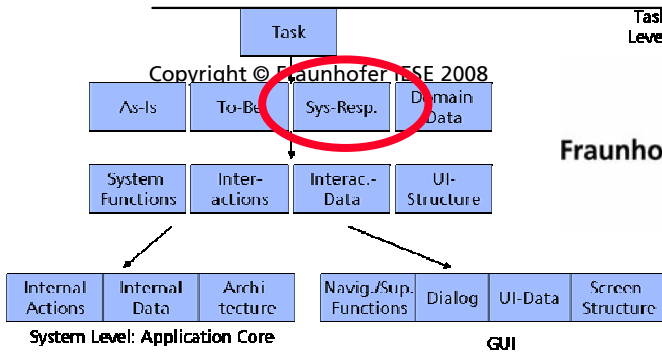


Q-Day
Kaiserslautern, 23.04.2008

Beispiel: Systemverantwortlichkeiten



Seite 16/X



Copyright © Fraunhofer IESE 2008



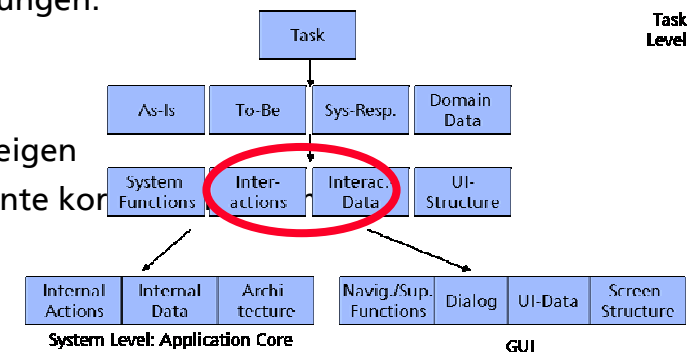
Fraunhofer Institut Experimentelles Software Engineering

Q-Day
Kaiserslautern, 23.04.2008

Best Practices im Requirements Engineering

Use-Case: Über Medikamenteneinnahme informieren

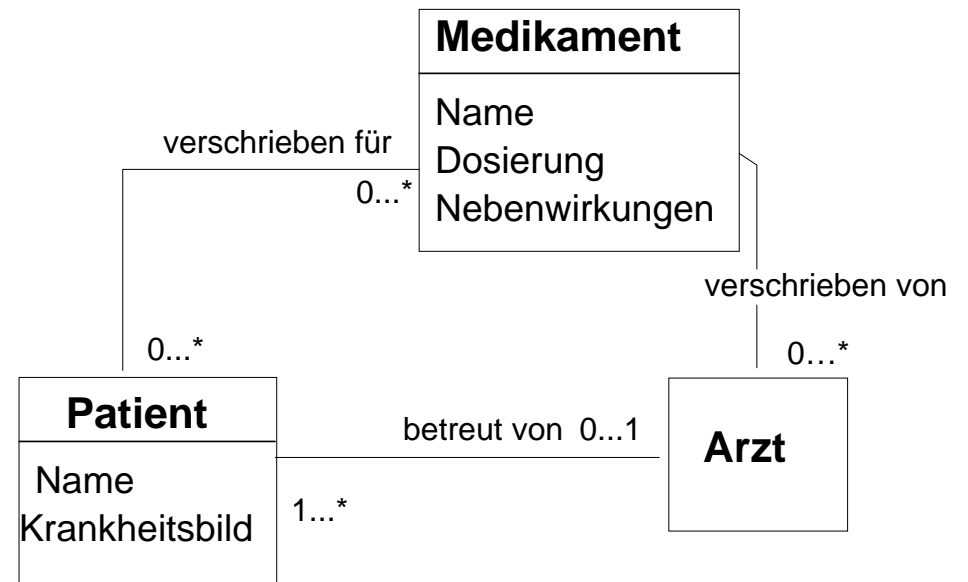
- **Name:** UC1: Über Medikamenteneinnahme informieren
- **Aktor:** Patient
- **Ziel:** Korrekte Einnahme eines verschriebenen Medikaments sicherstellen.
- **Vorbedingungen:** Medikamente wurden verschrieben.
- **Beschreibung des Ablaufs:**
 1. *Aktor* ruft Medikamentenübersicht auf.
 2. *Das System* zeigt Liste der einzunehmenden Medikamente an.
 3. *Der Aktor* wählt aus der Liste ein Medikament aus.
 4. *Das System* zeigt medikamentenspezifische Informationen wie Art der Dosierung, Anleitung zur Einnahme sowie Nebenwirkungen an.
 5. Nach Einnahme des Medikamentes bestätigt *der Aktor* die Einnahme [weitere Medikamente sind einzunehmen] [Medikamentvorrat reicht nur noch für 3 Tage].
- **Regeln:** Verschriebenes Medikament muss verfügbar sein.
- **Ausnahmefälle:** [weitere Medikamente sind einzunehmen]: Das System zeigt Liste der noch einzunehmenden Medikamente an. Weiter mit Schritt 3.
- **Qualitätsanforderungen:** Informationen müssen für den Patienten gut lesbar (Großschrift) und übersichtlich angezeigt werden.
- **Daten:** Name des Medikaments, Dosierungsanleitung, Nebenwirkungen.
- **Systemfunktionen:**
 - SF1: Einzunehmende Medikamente anzeigen
 - SF2: Medikament-spezifische Informationen anzeigen
- **Nachbedingungen:** Der Patient hat die verschriebenen Medikamente kor



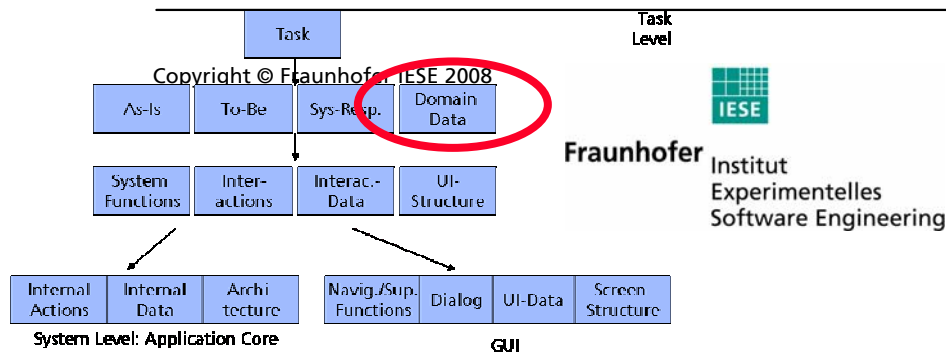
Beispiel: Glossar und Anwendungsdatenmodell

Ein **Medikament** hat einen Namen und eine Dosierungsanleitung. Ist für keinen, einen oder mehrere Patienten verschrieben.

Ein **Patient** wird von höchstens einem Arzt betreut. Er hat ein spezifisches Krankheitsbild und ist zur Einnahme von keinem, einem oder mehrerer Medikamenten aufgefordert.



Seite 18/X

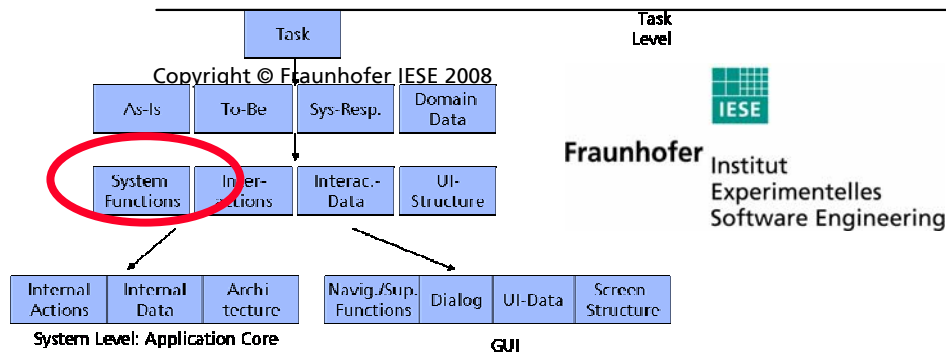


Q-Day
Kaiserslautern, 23.04.2008

Wie beschreibe ich eine Systemfunktion?

Name	Kurzbezeichnung der Funktion
Eingangsdaten	Welche Daten verarbeitet die Funktion?
Ausgangsdaten	Welche Daten erzeugt oder verändert die Funktion?
Beschreibung	Wie soll das Ergebnis normalerweise berechnet werden?
Ausnahmefälle	Welche Ausnahmen gibt es? Was soll dann passieren?
Regeln	Komplexe funktionale oder kausale Zusammenhänge bei der Berechnung
Qualitätsanforderungen	Welche übergreifenden Eigenschaften sind wichtig?
Vorbedingungen	Zustand von System und Umgebung aus Sicht des Aktors <i>bevor</i> die Funktion ausgeführt wird
Nachbedingung	Zustand des Systems aus Sicht des Aktors <i>nachdem</i> die Funktion erfolgreich beendet ist

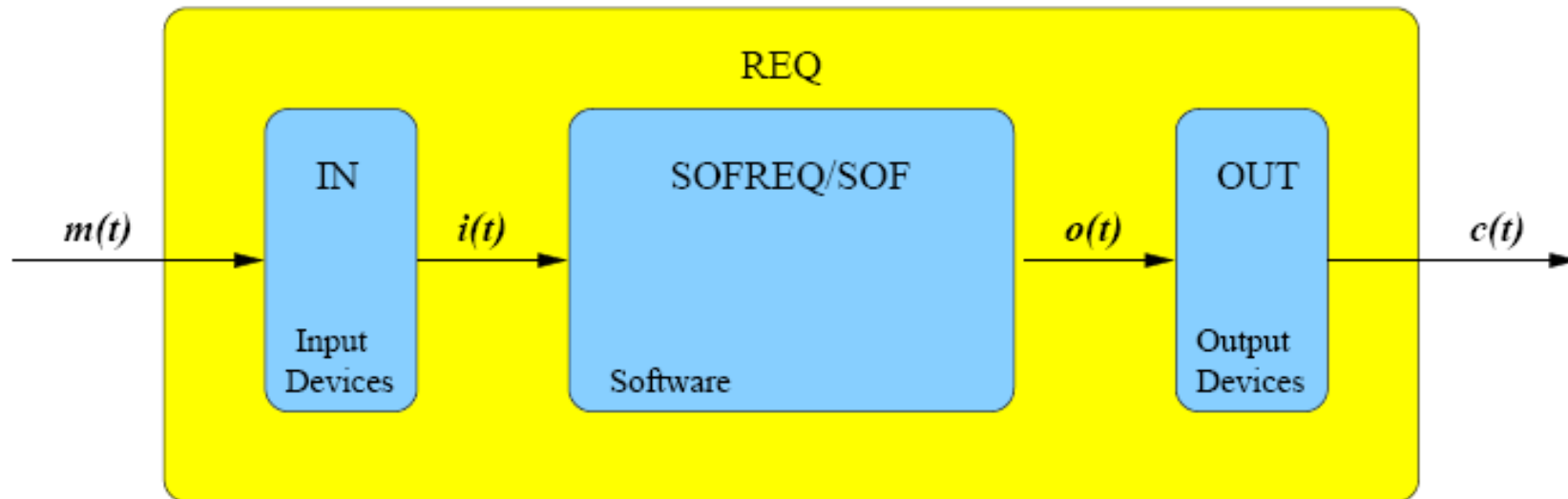
Seite 19/X



Q-Day
Kaiserslautern, 23.04.2008

Exkurs: Das 4 Variablen-Modell

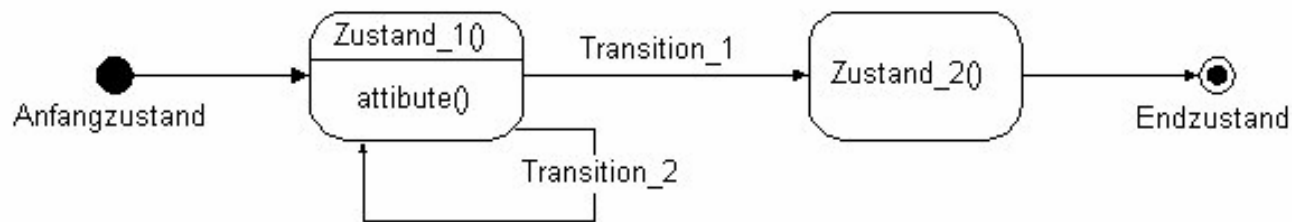
System Design



Seite 20/X

UML-Zustandsdiagramm

- Visualisiert verschiedene Zustände eines Objektes, sowie Funktionen, die zu Zustandsänderungen eines Objektes führen
- Beschreibt eine hypothetische Maschine (endlichen Automat), der sich zu jedem Zeitpunkt in einer Menge endlicher Zustände befindet.
- Besteht aus:
 - einem Anfangszustand
 - einer endlichen Menge von Zuständen
 - einer endlichen Menge von Ereignissen
 - einer endlichen Anzahl von Transitionen (Übergänge)
 - einem oder mehreren Endzuständen



Seite 21/X

Copyright

Best Practices im Requirements Engineering

Agenda

- Motivation Requirements Engineering
- Funktionale Anforderungen
- **Nicht-funktionale Anforderungen**
- Anforderungsmanagement

Seite 22/X

**Das ESP muss
sicher sein.**

**Die Kommunikation muss über
einen sicheren Kanal erfolgen.**

**Das System soll leicht
wartbar sein.**

**Das System soll einfach zu
bedienen sein.**

**Der Zulieferer muss Erfahrung
mit der Erstellung von
zuverlässigen Systemen haben.**

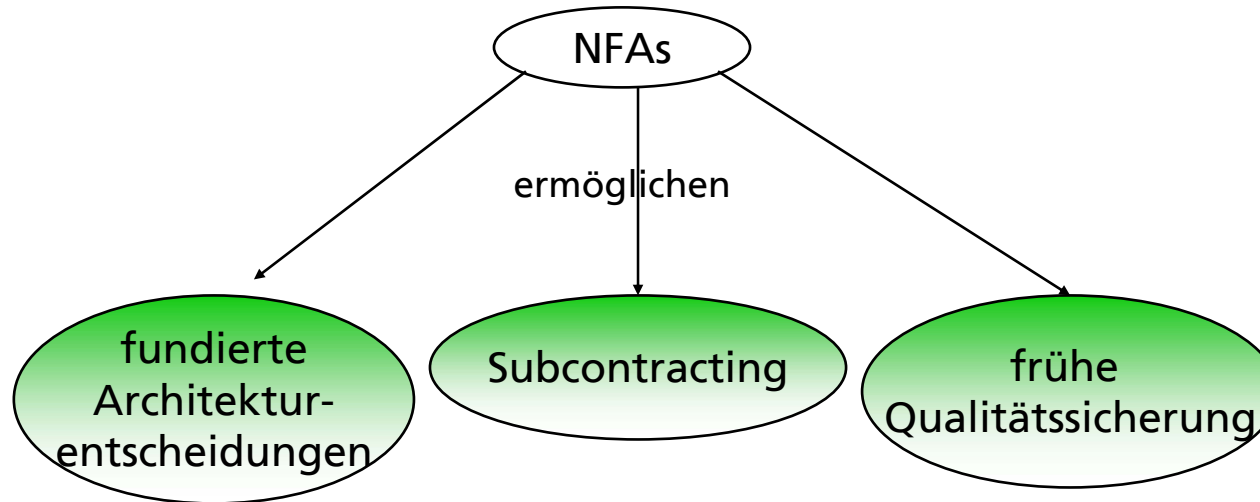
**Schatz, Du solltest mehr
im Haushalt helfen.**

Qualitätsattribute nach ISO 9126

- **Übertragbarkeit**
 - Anpassbarkeit
 - Installierbarkeit
 - Konformität zu Standards
 - Austauschbarkeit
- **Zuverlässigkeit**
 - Reife
 - Fehlertoleranz
 - Wiederherstellbarkeit
- **Benutzbarkeit**
 - Verständlichkeit
 - Erlernbarkeit
 - Bedienbarkeit
- **Effizienz**
 - Zeitverhalten
 - Verbrauchsverhalten
- **Änderbarkeit**
 - Analysierbarkeit
 - Modifizierbarkeit
 - Stabilität
 - Prüfbarkeit
- **Funktionalität**
 - Angemessenheit
 - Sicherheit
 - Genauigkeit der Berechnung
 - Interoperabilität
 - Konformität zu Standards

Seite 24/X

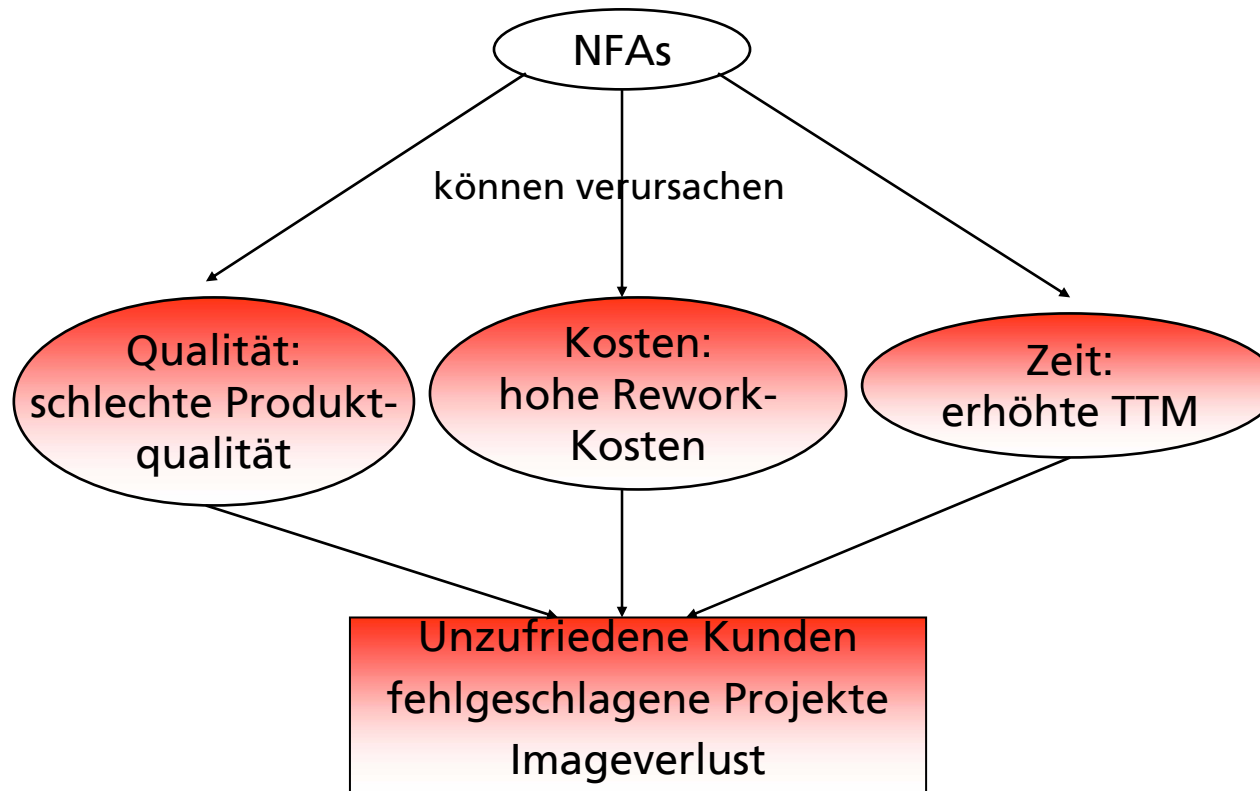
NFAs sind essentiell in der Software- & Systementwicklung!



NFAs können ein wesentliches Differenzierungsmerkmal gegenüber dem Wettbewerb darstellen!

Seite 25/X

Typische Folgen der Vernachlässigung von NFAs



Seite 26/X

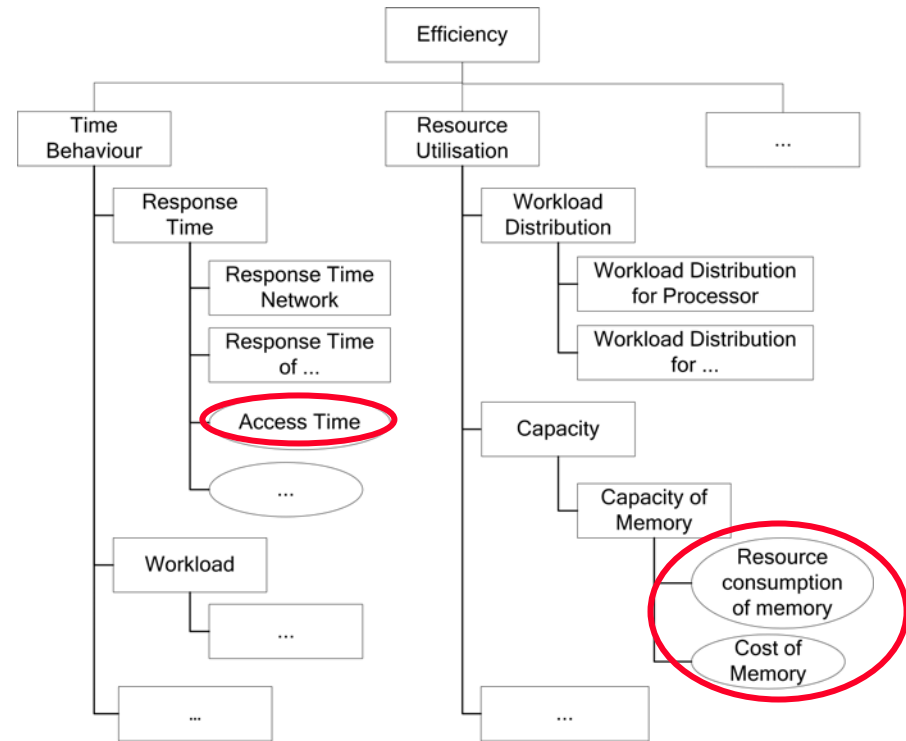
NFAs in der Praxis und damit verbundene Probleme

- Häufig natürlichsprachliche Dokumente, gerade bei der Unterbeauftragung
- Teilweise Einsatz strukturierter Sprache (Pattern, Use Cases)
- Nutzung von Lasten- / Pflichtenheft Templates
- Separate Kapitel für NFA
- Wenig dokumentierte NFAs
- Standards schlagen einige Qualitätsattribute (Kategorien von NFAs) vor
- Unsicherheit in der Handhabung (gibt es eine Methode für alle NFAs?)
- Hoher Aufwand Erhebung und Spezifikation notwendig?
- Unvollständige Menge an NFAs
- Inkonsistente und / oder konfliktbehaftete NFAs
- Nicht validierbare / messbare NFAs

Seite 27/X

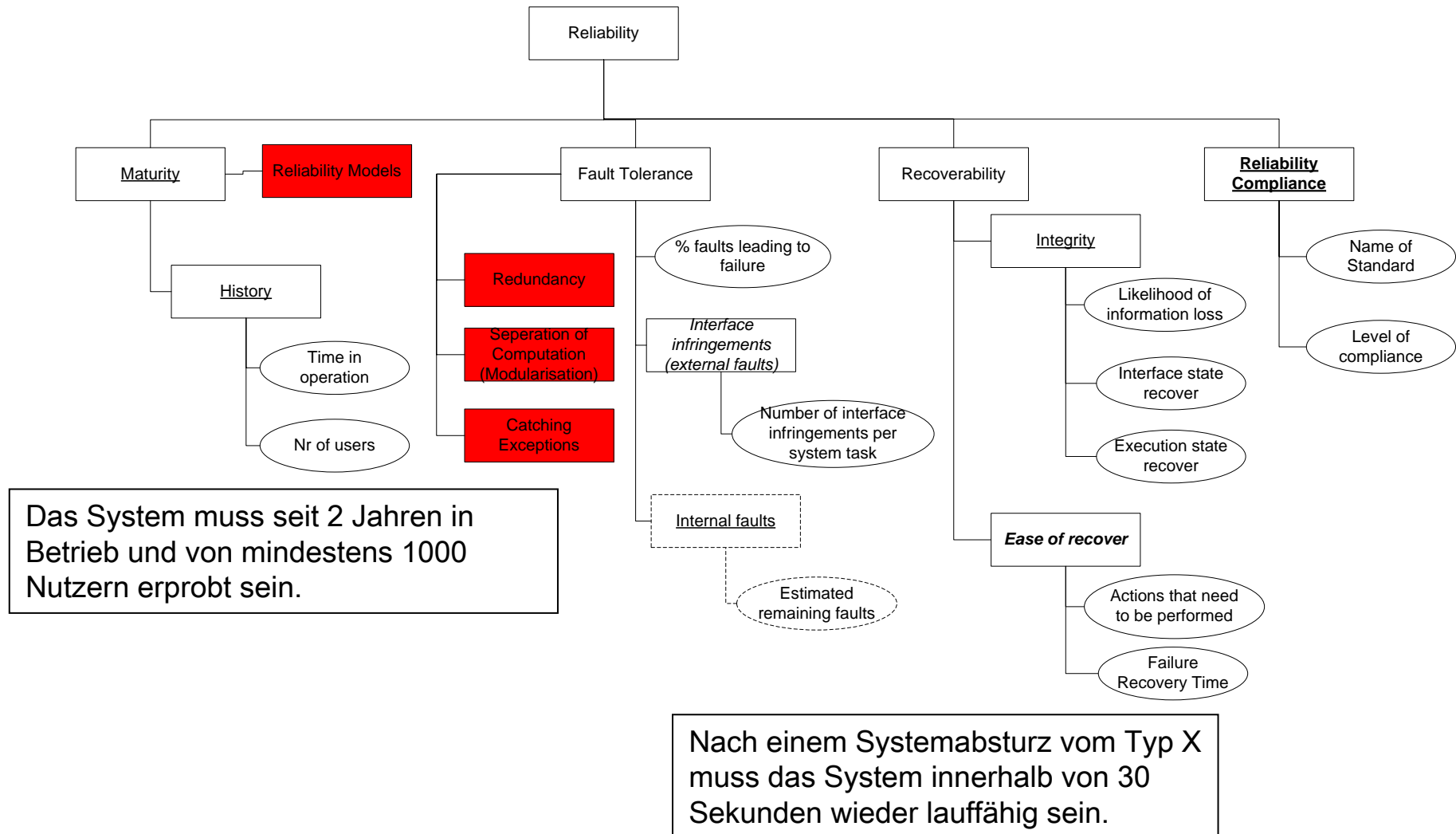
Best Practice: Spezifiziere messbare NFAs!

- Nichtfunktionale Anforderungen müssen über Metriken ausgedrückt werden (>90%)
- Metriken können in Qualitätsmodellen an die unterste Ebene der Qualitätsattribute angehängt werden
- Spezifizieren von messbaren Anforderungen erhöht die Vorhersagefähigkeit der Qualität von Endprodukten in frühen Phasen



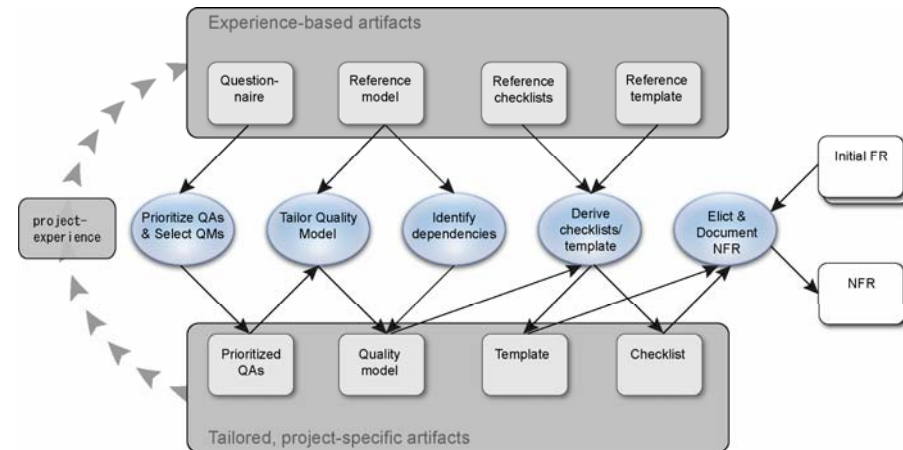
Best Practices im Requirements Engineering

Ausschnitt aus einem Zuverlässigkeits-QM



Best Practice: Etabliere einen systematischen Prozess!

- NFAs sollten durch einen systematischen Prozess ermittelt, spezifiziert und analysiert werden
- Prozess sollte durch Hilfsmittel gestützt effizient ablaufen können (NFA Ermittlung erfordert häufig Mitarbeit durch Kunden)
- Systematik ermöglicht Wiederholbarkeit, Vollständigkeit und Anwendbarkeit durch Nicht-Experten

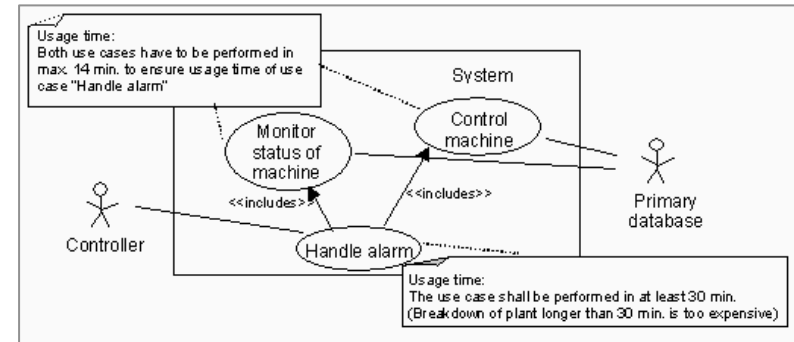


Seite 30/X

Best Practice: Spezifiziere NFAs in Templates & nah zu Funktionalen Anforderungen!

- NFAs sollten in den verwendeten Lastenheft-Templates verankert sein
- Dokumentation der NFA möglichst nahe an den betroffenen funktionalen Anforderungen
- Prominente Position der NFA führt zu gesteigerter Aufmerksamkeit
- Gute Struktur vermeidet Redundanzen und verbessert die Verfolgbarkeit

→ besseres Verständnis, bessere Beachtung, bessere Struktur des Dokuments



UseCase	Handle alarm
Actors	Controller
Intent	Actor removes a warning send by a certain machine
Preconditions	Use Case "Boot up system"
Flow of events	<ol style="list-style-type: none"> 1. System requests alarm messages from the first database regularly 2. System shows alarm and where the alarm was produced. [Exception: Multiple alarms] 3. Actor acknowledges alarm to let others know he/she is going to take care of it. [Exception: Another actor has acknowledged the alarm] 4. Actor moves to the machine following the path displayed by the system. (-> use case "monitor status of machine") 5. During the walk, actor monitors the status of this machine by the system. (-> use case "control machine") 6. Actor removes the problem by controlling the machine (-> use case "control machine") 7. System sends control data to first database.
Exceptions	<ol style="list-style-type: none"> 1.1 Multiple alarms: System opens a window for each alarm message. 3.1 Another user has acknowledged the alarm: System removes alarm message from screen and shows acknowledgement.
Rules	The alarm warning will always have the highest priority.
NFRs	Response time (assumed times): <ol style="list-style-type: none"> 1. every 5 sec. 2. at least in 5 sec. 3. just one click 4. at least 15 min. (worst case) 5. -> usage time of use case "monitor status of machine" 6. -> usage time of use case "control machine" 7. at least 5 sec.

Seite 31/X

Best Practice: Nutze Checklisten zur Erhebung!

- Erhebung der konkreten NFAs sollte durch Checklisten unterstützt werden
- Die Checklisten stellen die Brücke zwischen Qualitätsattributen und der Erhebung dar
- Nutzung von NFA Beispielen zur Illustration
- Checklistenenerstellung ist durch Werkzeugunterstützung automatisierbar

1 **Elicitation of Throughput NFRs** (*FRs-> UC description -> NFRs -> throughput // NFRs->efficiency ->throughput*)

For each network element in the system architecture:

- Go through each Use Case: Identify the UC-steps and exceptions involving data transportation on this component. Think of an **average** scenario of the Use Case

- How much data has to be transported by this component?

Specify the throughput NFRs on the network element.

- Go through each Use Case: Identify the UC-steps and exceptions involving data transportation on this component. Think of a **maximum** usage of the Use Case

- How much data has to be transported by this component?

Specify the throughput NFRs on the network element.

7. **Elicitation of boot / start up time requirements** (*NFRs->efficiency ->start/boot time*)

- For each component that has to be booted / started, specify the boot-time / startup-time of the HW and SW.

Best Practices im Requirements Engineering

Agenda

- Motivation Requirements Engineering
- Funktionale Anforderungen
- Nicht-funktionale Anforderungen
- Anforderungsmanagement

Seite 33/X

Anforderungsmanagement

Anforderungsmanagement ist wichtig um:

- die Zusammenarbeit von verschiedenen Rollen zu unterstützen und
- um das Wissen über Anforderungen und Gestaltungsentscheidungen über die Lebenszeit des Systems aktuell zu halten.

Wichtige Aktivitäten

- Attributierung der Anforderungen (z.B. Priorität, Risiko, Klassen, Aufwand)
- Verfolgbarkeit innerhalb und zwischen Entwicklungsdokumenten
- Prozesse zum Umgang mit Änderungen

Hauptprobleme

- Dokumentationsdisziplin und Motivation der Beteiligten
- Definition eines effektiven aber gleichzeitig effizienten AM-Ansatzes

Nutzung von RM Tools

- Dokumentenrepository
- Änderungssupport
- Organisation von Informationen
- Sichtenbildung für verschiedene Zwecke
- Suchmöglichkeiten
- Traceabilityunterstützung (Querverweise)

→ Wenn Sie das falsch tun, hilft Ihnen das Tool das Falsche schneller zu erledigen!

RM Tools vs. Standardsoftware (Word, Excel, etc.)

RM Tools

- Teuer
- Training nötig

- Multi-user fähig
- Spezieller Support für RM Aufgaben
 - Change Mgmt,
Traceability
- Integration mit anderen Entwicklungswerkzeugen

Standard-Software

- Limitierte Unterstützung für RM spezifische Aufgaben
- Anpassungen nötig
- Keine Integration mit Entwicklungswerkzeugen

- Günstig (meist installiert)
- Selten Training nötig

Vielen Dank für Ihre Aufmerksamkeit

- Michael Eisenbarth
- phone:+49 631 6800 2181
- Michael.eisenbarth@iese.fraunhofer.de