3. Anforderungs-management

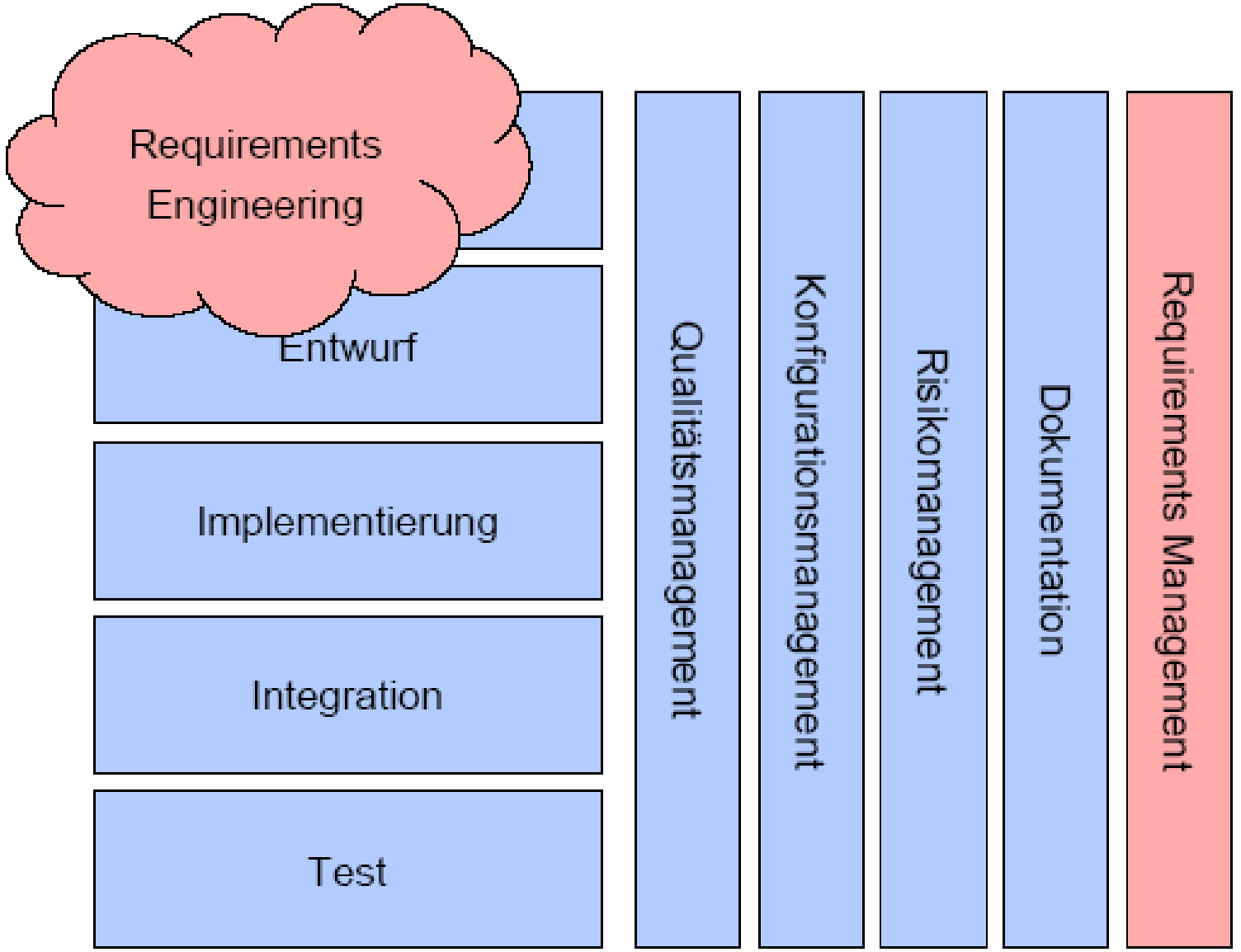
Dr.-Ing. Clemens Reichmann

Clemens.Reichmann@vector.com
Tel. 0721 / 91430-200

Institut für Technik der Informationsverarbeitung
Fakultät für Elektrotechnik & Informationstechnik
Universität Karlsruhe (TH)

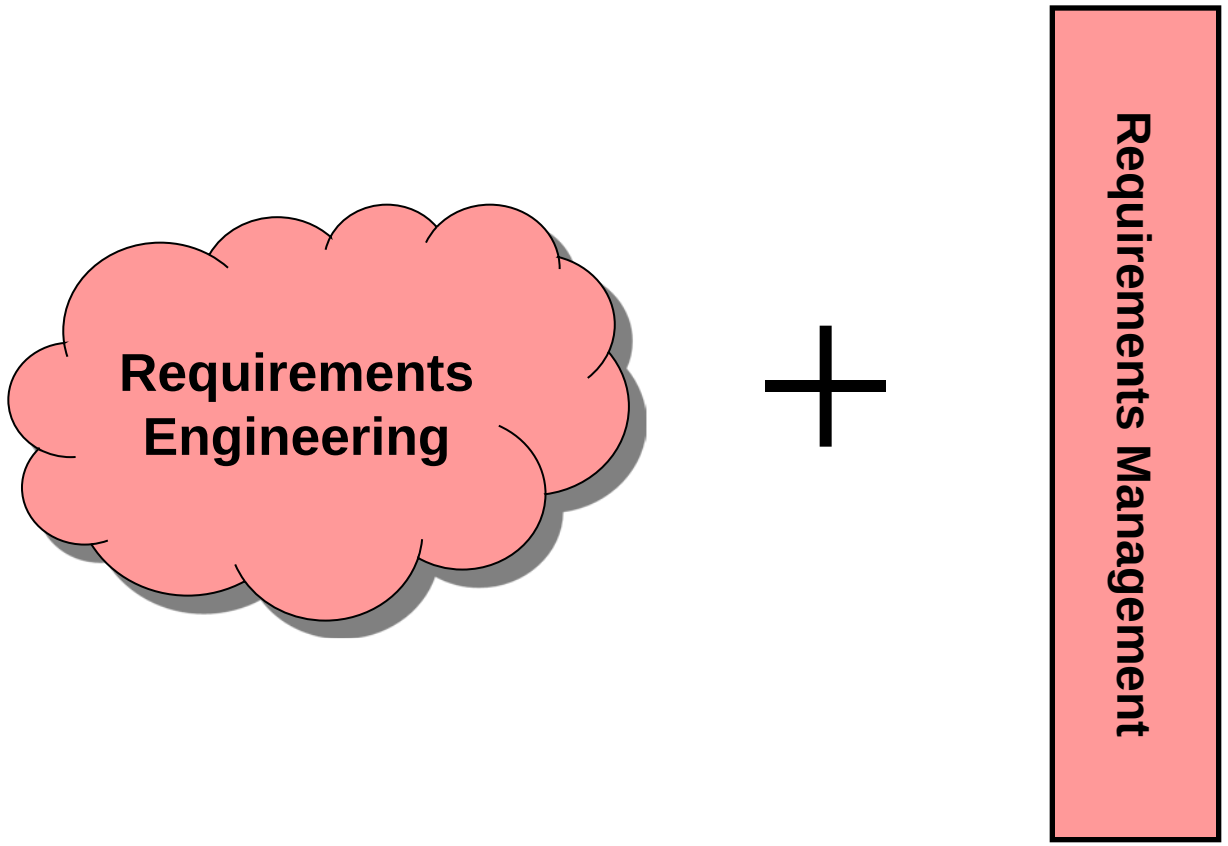
Inhalt 3. Anforderungsmanagement

3.1 Requirements Engineering vs. Management

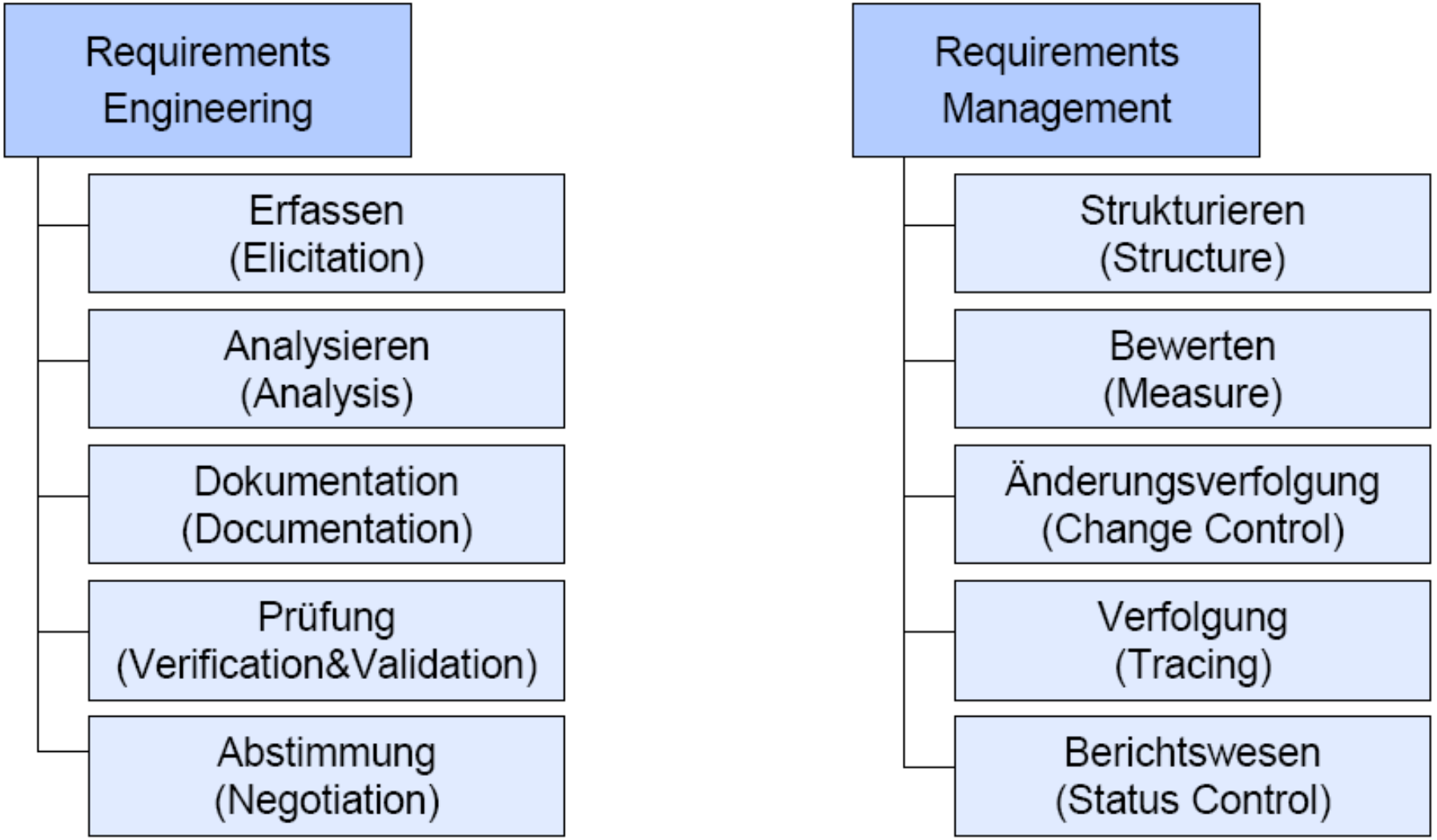


Begriffe

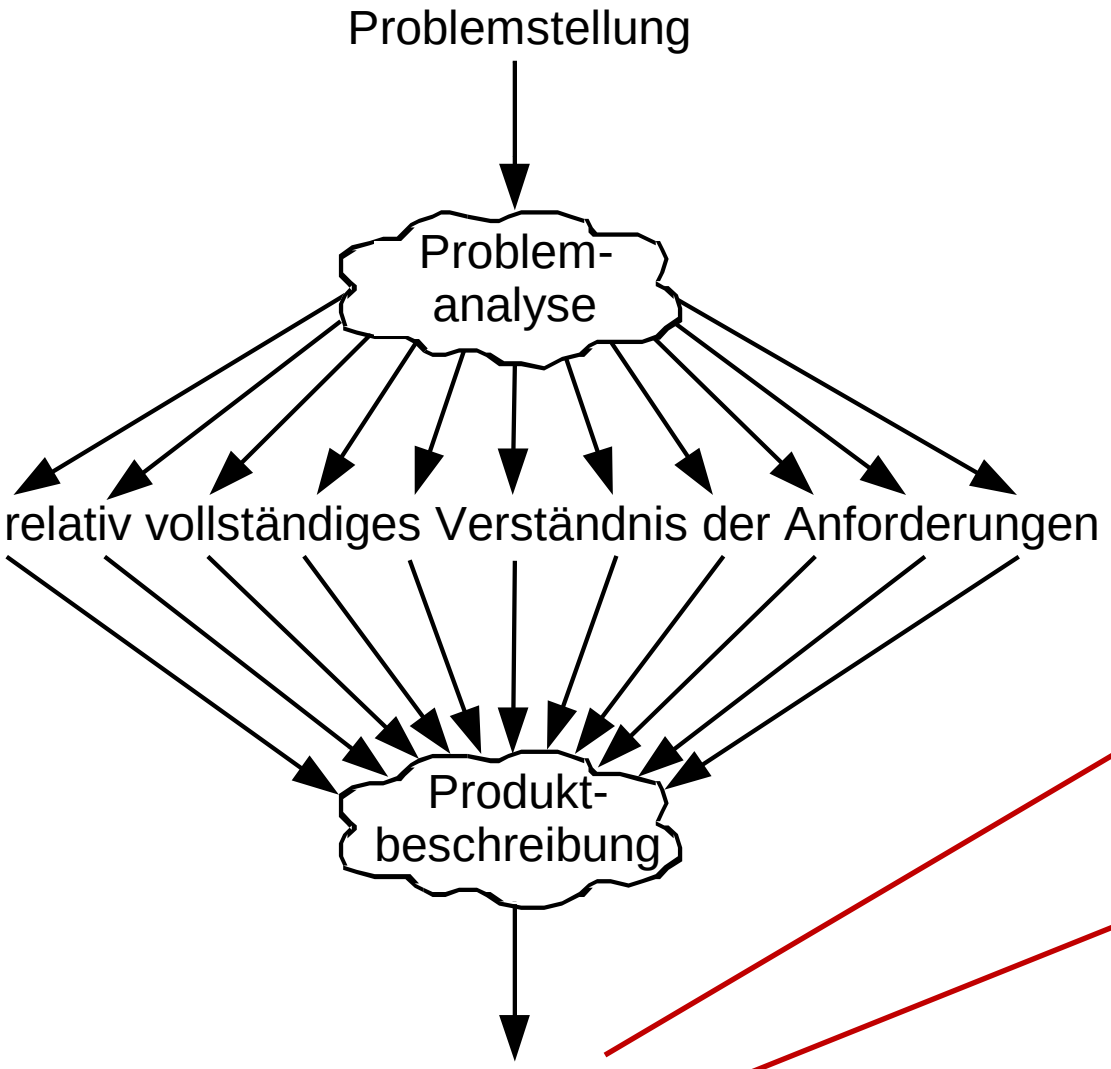
- Anforderungsmanagement als Oberbegriff (Schienmann) für
  - Requirements Engineering
  - Requirements Management

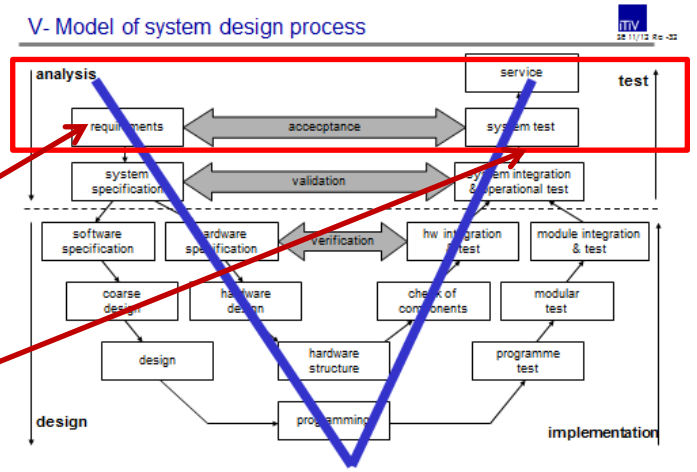


Wesentliche Bestandteile



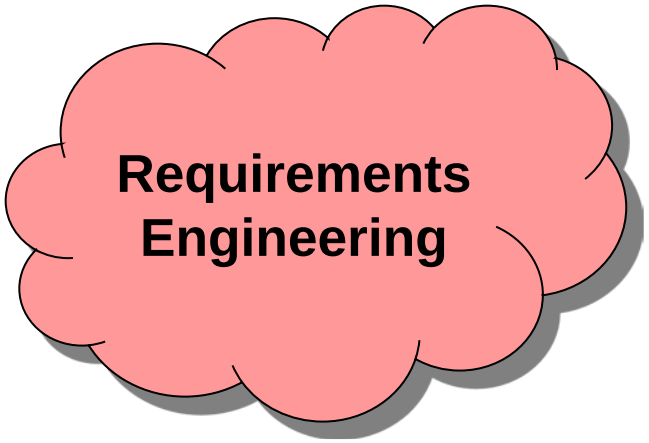
Umfeld





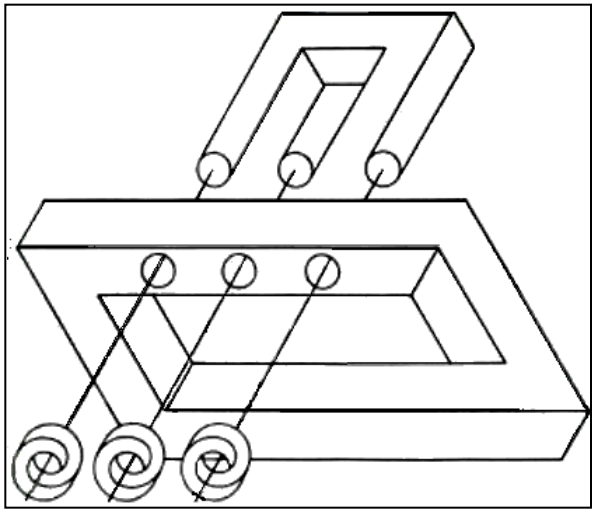
konsistente und vollständige Anforderungsdefinition aus der die Testfälle für die Abnahmetests abgeleitet werden können.

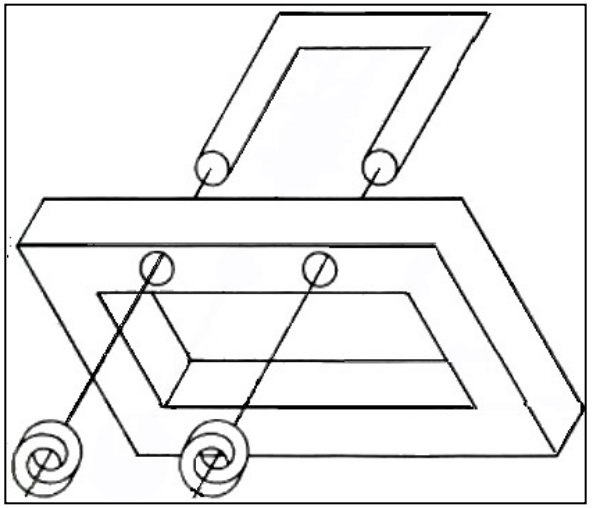
Anforderungsmanagement
→ requirements engineering

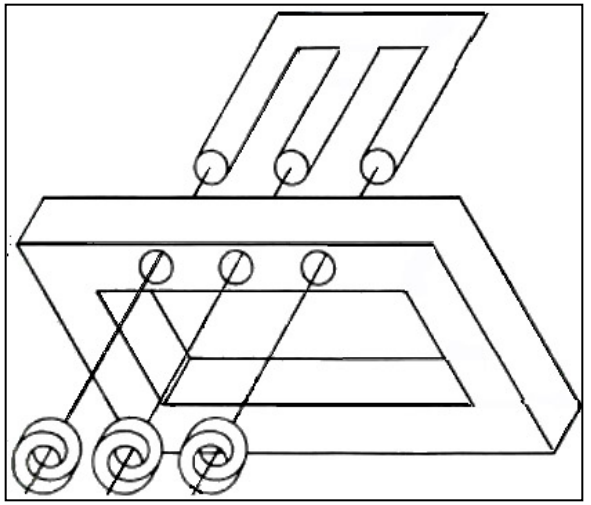


3.2.1Begriff: Anforderung

- Anforderung
  - Beschreibung der Dienste und Einschränkungen des Systems sind Anforderungen







Gewünschte Eigenschaft einer Anforderung

- Eine Anforderung sollte sein (10):
  - Korrekt
  - Eindeutig
  - Vollständig
  - Konsistent
  - Machbar
  - Verstehbar
  - Verifizierbar
  - Notwendig
  - Traceable up to test
  - Unabhängig von Lösung

Gewünschte Eigenschaft einer Anforderung

- Terminologie:
  - muss/must (constrain by law)
  - soll/shall (necessary requirement)
  - sollte/should (recommended requirement)
  - wird/will (future requirement)

▪ **5.1.2.3.2 - WHAT2971 Math Function¶**

▪ Requirement Description¶

<FDESC>The behavior of this ASCET block shall be defined by a ESDL code. The ESDL files shall be located in the ./lib/esdl/ folder. The file name shall be equal to the SL block type.¶
¶
Note: the SL block contains 15 functions. Not all shall be supported.</FDESC>¶

▪ Other information¶

Owner: clreichm	Version: 4.1	Priority: Prio C - If possible¶
Effort [MD]: 0	2	0¶

Responsible(s):¶

aquintos(clreichm)¶

Reference(s):¶

¶

Trace(s):¶

¶

▪ Validation hints/procedure¶

<VDESC>same as WHAT2775</VDESC>¶

Spezifikation der Anforderungen

- Beschreibung von **WAS** (nicht WIE) zu implementieren ist
  - Nicht im Lösungsraum

- Probleme dabei:
  - Architektur wird oft implizit mit Strukturierung der Anforderungen vorgenommen
  - Zusammenwirken mit anderen Systemen: Implementierungseinschränkungen
    - Z.B. Zielsprache C++, Verwendung von OSEK-OS ("**O**ffene **S**ysteme und deren Schnittstellen für die **E**lektronik im **K**raftfahrzeug„ - Betriebssystem)
  - Architektur kann als Anforderung vorgegeben sein
    - Programmierung von redundanten Systemen wegen Zuverlässigkeit

3.2.2 Kategorisierung von Anforderungen

- Typen:
  - Funktionale Anforderungen
  - Nichtfunktionale Anforderungen
  - Problembereichsanforderungen (Domäne, Fachlichkeit)

Problem:

→ künstliche Unterscheidung: nicht klar trennbar (nichtfunktionale Anforderungen können „funktionalisiert“ werden)

Funktionale Anforderungen

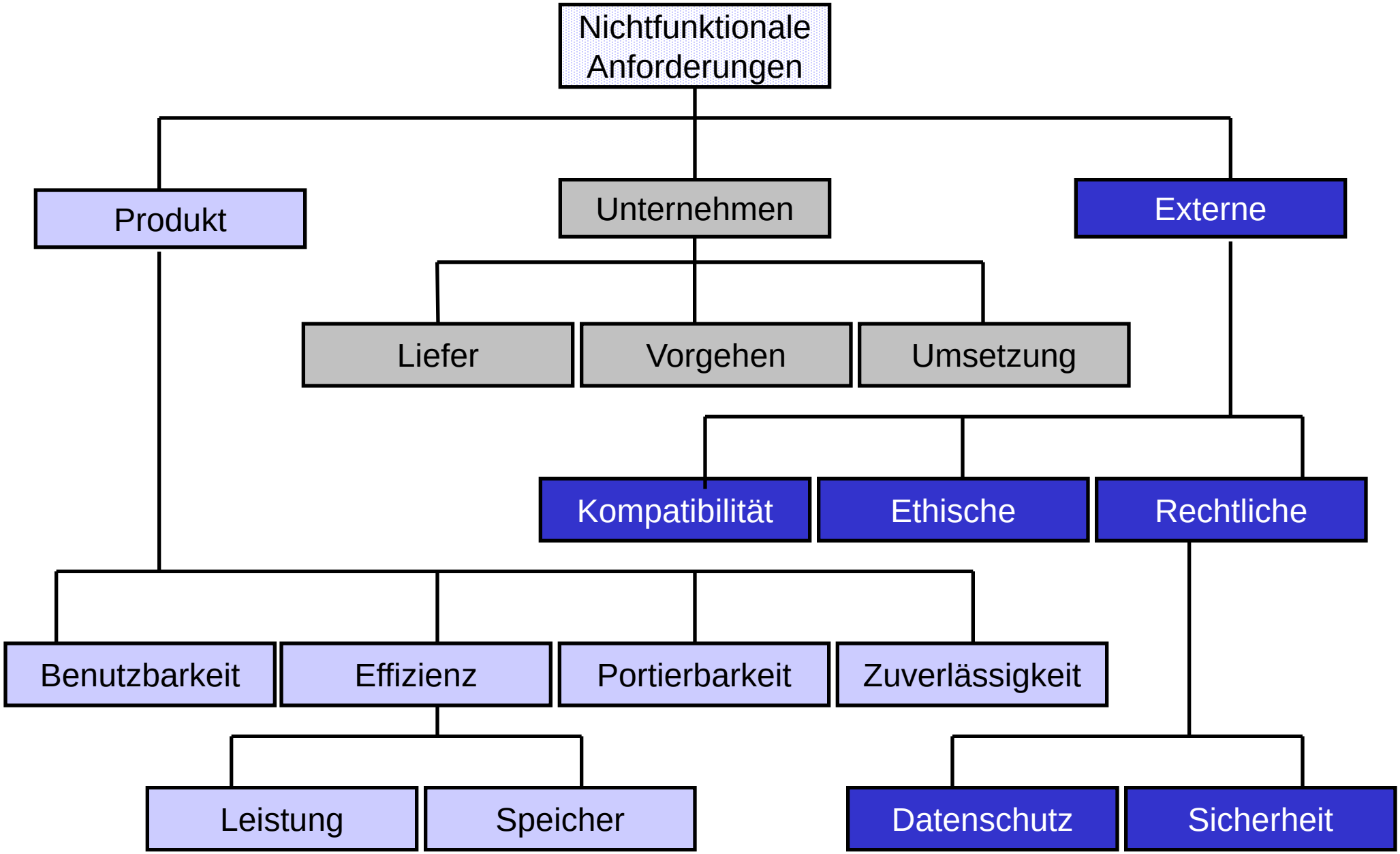
- Funktionale Anforderungen
  - Aussagen zu Diensten, die das System leisten soll
  - Reaktion des Systems auf Eingaben
  - Verhalten in bestimmten Situationen
  - Was das System nicht tun soll
- Funktionale Benutzeranforderung
  - Häufig sehr allgemein
- Funktionale Systemanforderung
  - Detaillierte Spezifikation
  - Häufige Fehlerquelle: Mehrdeutigkeit führt zu Implementierung, die Kunde nicht will
  - Ausführbare Spezifikation z.B. MATLAB/Simulink
    Blockdiagramme formalisieren grafisch Anforderung

Nichtfunktionale Anforderungen

- Beschränkungen
  - Sollte qualitativ sein (Metriken)
  - Sollte testbar sein (oft schwer überprüfbar)
- Globale Sichtweise
  - Wichtiger als funktionale Anforderungen
  - Da bei Nichteinhaltung System unbrauchbar ist

- Beispiele:
  - Zeitbeschränkungen
  - Entwicklungsprozess
  - einzuhaltende Standards
  - Zuverlässigkeit
  - Reaktionszeit
  - Speicherbedarf
  - Leistungsfähigkeit von E/A Geräten
  - Datendarstellung an Systemschnittstellen

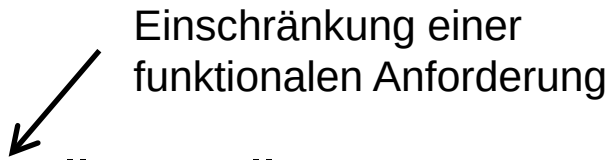
Arten von nichtfunktionalen Anforderungen



Problembereichsanforderungen

- Problembereichsanforderungen
  - Problembereich des Systems, gibt Charakteristik wieder
  - Sind funktionale/nichtfunktionale Anforderungen

- Beispiel (Bibliotheksystem)
  - 1. Es sollte eine Standardbenutzerschnittstelle zu allen Datenbanken geben, die auf dem Z39.50 Standard basiert
  - 2. Aus urheberrechtlichen Gründen müssen einige Dokumente direkt bei der Ankunft gelöscht werden. Abhängig von den Anforderungen des Benutzers sollen diese Dokumente entweder lokal auf dem Systemserver ausgedruckt, um manuell vom Benutzer verschickt zu werden, oder sie sollen an einen Netzwerkdrucker weitergeleitet werden.

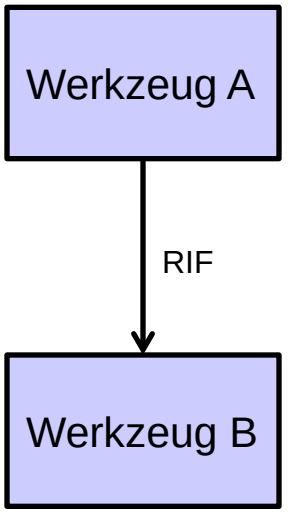


Anforderungen: Beschreibungsformen

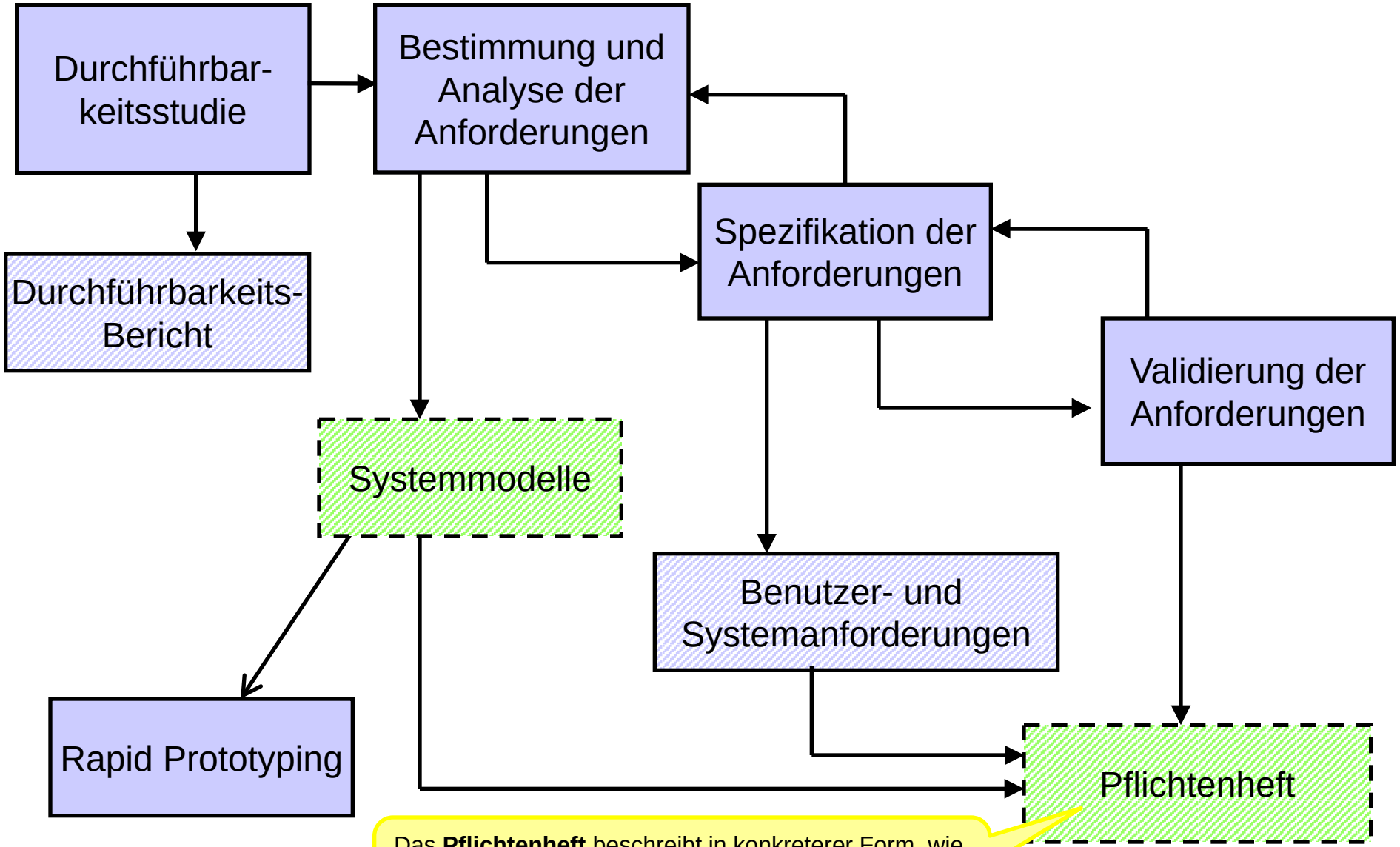
- Natürliche Sprache
  - Mangel an Genauigkeit
  - Verwirrung bei Anforderungen (Typen können nicht unterschieden werden)
  - Verschmelzung von Anforderungen
  - Gleicher Sachverhalt kann unterschiedlich ausgedrückt werden (überflexibel)
  - Nicht einfach modularisierbar → Gruppierung
- Sprachgebrauch
  - Verbindliche Anforderungen: sollen (shall)
  - Wünschenswerte Anforderungen: sollten (should)
  - Rechtliche/Ethnische Vorgaben: muss (must)
  - Zukunftsvisionen: wird (may)

Verwendung von CASE Werkzeugen

- Standardisierung des Formats (RIF, ReqIF: requirement interchange format)
- Strukturierung
- Informationen in einer Anforderung
  - Beschreibung mit Hervorhebungen (fett, kursiv)
  - Attribute
  - Beziehungen (benötigt)
  - Versionierung, Benutzermanagement
  - Eingaben
  - Ausgaben
  - Vorbedingung
  - Nachbedingung
  - Invarianten
- Glossar
  - Begriffsdefinition, z.B. Hervorhebung durch Farbe



3.2.3 Prozess der Anforderungsanalyse



Prozess der Anforderungsanalyse (Durchführbarkeitsstudie)

- Durchführbarkeitsstudie
  - Bedürfnisse mit
    - Software Technologie erfüllbar?
    - Hardware Technologie erfüllbar?
  - Kosten im Rahmen?
  - Ziel: Entscheidung, ob detaillierte Analyse sinnvoll ist

Prozess der Anforderungsanalyse (Analyse)

- Bestimmung und Analyse der Anforderungen
  - Ableiten aus
    - Beobachtung bestehender Systeme
    - Diskussion mit potentiellen Benutzern und Käufern
    - Aufgabenanalyse
  - Systemmodelle
    - CASE Tools
  - Prototypen
    - Tools für funktionale Prototypen
      - MATLAB/Simulink, MATLAB/Stateflow (The Mathworks)
      - Statemate/Rhapsody (IBM)
      - ASCET (ETAS)
      - …
  - Ziel: System verstehen

Analysetechniken

- Viewpoint
  - Blickwinkel auf das System
  - Rollen der Benutzer
- Szenarien
  - UML Message Sequence Charts (MSC)
  - Strukturierung in Anwendungsfälle (UML Use Case)
- Ethnografie
  - Gesellschaftliches Umfeld
  - Wirtschaftliches Umfeld
  - Analytiker taucht in Umfeld des zu erstellenden Systems ein

Viewpoint attribute		Attribute Description
Name		This identifies the viewpoint.
Focus		This shows the definition of the perspective taken by the viewpoint.
Sources		This shows a list of sources of the viewpoint's requirements.
Actors		This shows a list of actors associated with the viewpoint.
NFRs		This shows a list of the NFRs applied to the viewpoint.
Use Cases		This shows a list of functionalities relevant to the viewpoint.
Aggregation		This shows a list of aggregations where the viewpoint is involved, illustrated in the viewpoint diagram and a specific template.
Association		This shows a list of associations where the viewpoint is involved, illustrated in the viewpoint diagram and a specific template.
Gen/ Spec	Sub VP	This shows a list of sub viewpoints, illustrated in the viewpoint diagram.
	Super VP	This shows a list of super viewpoints, illustrated in the viewpoint diagram.
History		This keeps the alterations of the viewpoint information through time.

Spezifikation der Anforderungen (requirements document)

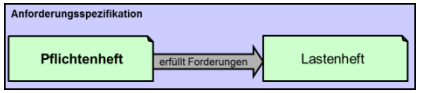
- Ziel: Dokument, das aus Informationen der Analyse Anforderungen spezifiziert und strukturiert
- Pflichtenheft

  - Produktdefinition, Produktspezifikation
  - Offizielle Aufstellung: was wird vom SW Entwickler erwartet
  - Pflichtenheft enthält 2 Typen von Anforderungen
    - Benutzeranforderungen: (BENUTZER/USER)
      - Abstrakte Beschreibung der Systemanforderungen
      - Kunde braucht grobe Aufstellung
    - Systemanforderungen: (WAS/WHAT)
      - Detaillierte Auflistung der Funktionen
      - Systementwickler braucht detaillierte Systemspezifikation

Systemanforderungen

- Systemanforderungen:
  - Anforderungen als Ganzes erkennen
  - Beinhaltet Beratung mit dem Kunden und Benutzer
  - Gesamtziele des Systems
  - Abstrakte Funktionale Anforderungen
  - Systemeigenschaften
    - Nicht funktionale, sichtbare Eigenschaften des Systems
    - Z.B. Verfügbarkeit, Leistungsfähigkeit, Sicherheit
    - Globale Auswirkungen
  - Eigenschaften, die das System nicht aufweisen darf
- Schwierigkeiten
  - Anforderungen für komplexe Probleme lassen sich vor dem Auftreten des Problems nicht endgültig spezifizieren
  - Implizite Anforderungen werden häufig bei der Spezifikation vergessen

Beispiel: Benutzer- und Systemanforderung



- Benutzeranforderung (im Lastenheft)
  1. Die Software soll über Mittel zur Darstellung externer, von anderen Werkzeugen erzeugter, Dateien verfügen und auf sie zugreifen können.
- Systemanforderungen (im Pflichtenheft)
  1.1 Der Benutzer sollte über Möglichkeiten zur Definition externer Dateitypen verfügen
  1.2 Jeder externe Dateityp kann eine damit verknüpfte Anwendung besitzen, mit der die Datei bearbeitet wird
  1.3 Jeder externe Dateityp kann als bestimmtes Symbol auf dem Bildschirm des Anwenders dargestellt werden
  1.4 Es sollten Möglichkeiten für die Definition des Symbols für externe Dateitypen durch den Benutzer bereitgestellt werden
  1.5 Wenn ein Benutzer ein Symbol auswählt, das eine externe Datei repräsentiert, so soll die durch dieses Symbol dargestellte Datei mit der Anwendung geöffnet werden, die mit dem entsprechenden externen Dateityp verknüpft ist.

Spezifikation der Anforderungen (requirements document)

- Struktur eines Pflichtenheftes nach IEEE 830-1993:
  - Vorwort
    - Leserschaft
    - Versionsgeschichte
    - Warum neue Version, Zusammenfassung der Veränderungen
  - Einleitung
    - Notwendigkeit des Systems
    - Zusammenfassung der Funktion
    - Wie Zusammenarbeit mit Umwelt
    - Zusammenhang zu wirtschaftlichen und strategischen Zielen des Unternehmens des Auftraggebers
  - Begriffe
  - Benutzeranforderungen
    - Überblick über erwartete Systemarchitektur
    - Verteilung der Funktionen
    - Wiederverwendete Komponenten

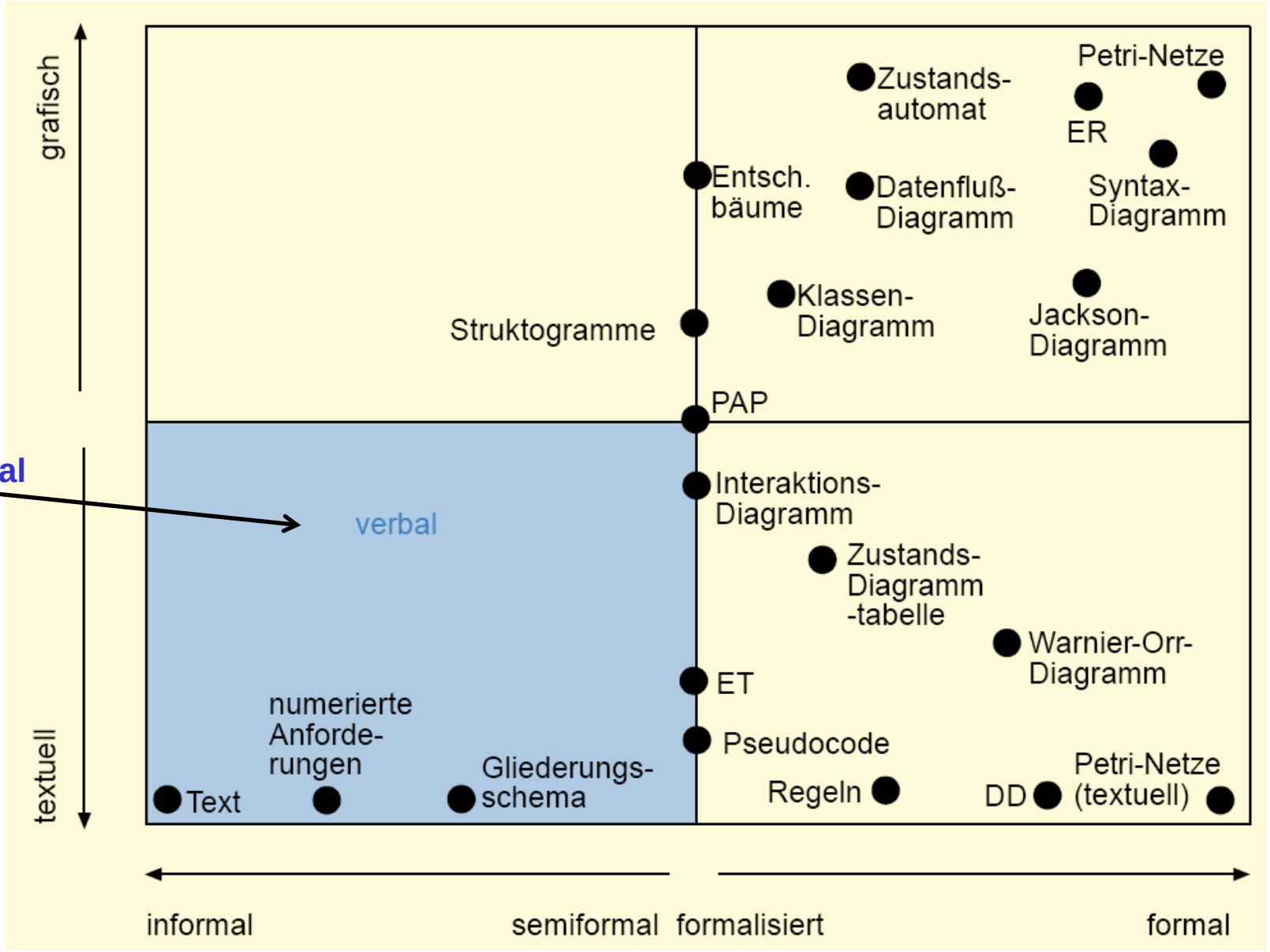
Spezifikation der Anforderungen (requirements document)

- Struktur eines Pflichtenheftes nach IEEE 830-1993 (cont.):
  - Systemarchitektur
  - Systemanforderungen
  - Systemmodelle
  - Systementwicklung
    - Grundsätzliche Voraussetzungen
    - Erwartete Veränderungen aufgrund
      - HW Entwicklung
      - Benutzeranforderungsänderungen
  - Anhänge
    - Minimale/maximale Hardwarekonfiguration
    - Datenbankanforderungen
  - Index

Prozess der Anforderungsanalyse (Validierung)

- Validierung der Anforderungen
  - Sind Anforderungen realistisch?
  - Sind Anforderungen konsistent?
  - Sind Anforderungen vollständig?
  - Ziel: Erkennen von Fehlern im Anforderungsdokument
- Vorgehen
  - Durchführung von Reviews
  - Validierung mit Rapid Prototyping
    - Ausgangspunkt: Modelle
    - Automatische Codegenerierung
    - Rapidprotyping Hardware

Beschreibungsmittel (siehe Vorlesung SSE)



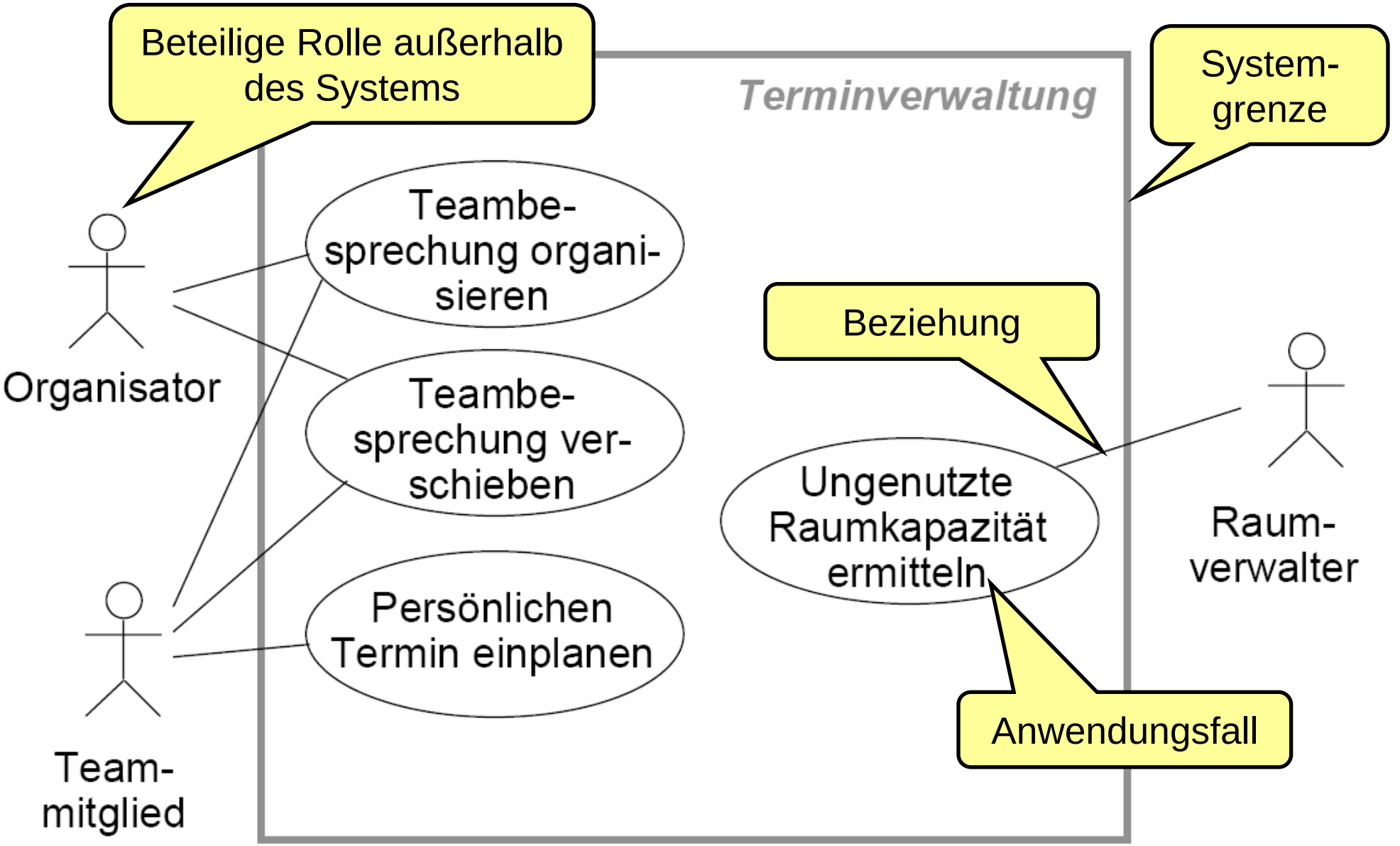
Anforderungsmodelle und ihre Diagramme

- Kontextmodell (SART, UML Klassendiagramm,…)
- Anwendungsfalldiagramm (UML use case, …)
- Verhaltensmodelle
  - Datenfluss z.B. MATLAB/Simulink®, ASCET® (ETAS)
  - Zustand z.B. MATLAB/Stateflow®, Statemate® (Telelogic / IBM)
- Datenmodelle
- Objektmodelle (UML z.B. ARTiSAN®, Enterprise Architect)
  - Vererbung
  - Objektaggregation
  - Objektverhalten

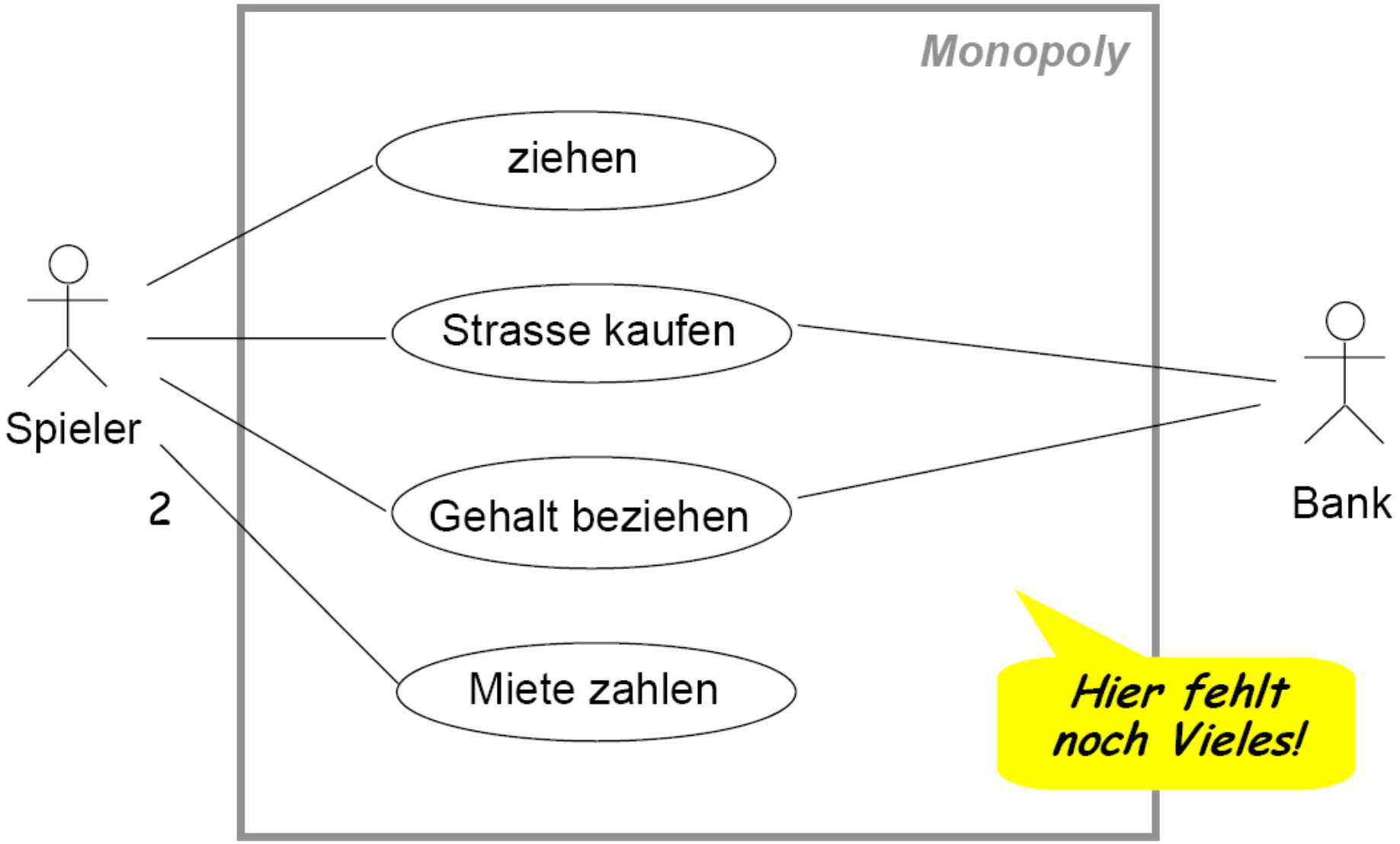
Dynamische Diagrammarten

- Bei der Problemanalyse beschreibt man in einem ***Use-Case-Diagramm*** (deutsch: Anwendungsfalldiagramm) die wichtigsten Leistungen des zu erstellenden Systems.
- Für die **Darstellung der Abläufe** bei der Durchführung eines Use Case gibt es folgende, komplementäre Diagrammarten mit unterschiedlichen Darstellungsschwerpunkten:
  - ***Sequenzdiagramme*** betonen die zeitliche Abfolge.
  - ***Kollaborationsdiagramme*** betonen die an Interaktionen beteiligten Objekte, ihr Zusammenspiel und den Datenfluss.
  - ***Aktivitätsdiagramme*** stellen Kontrollflüsse dar.
  - ***Zustandsdiagramme*** betonen unterschiedliche Verhaltensweisen in unterschiedlichen Zuständen im Objekt-Lebenszyklus.
- ***OCL*** (object constraint language) dient der **Spezifikation** von Zuständen & Operationen.

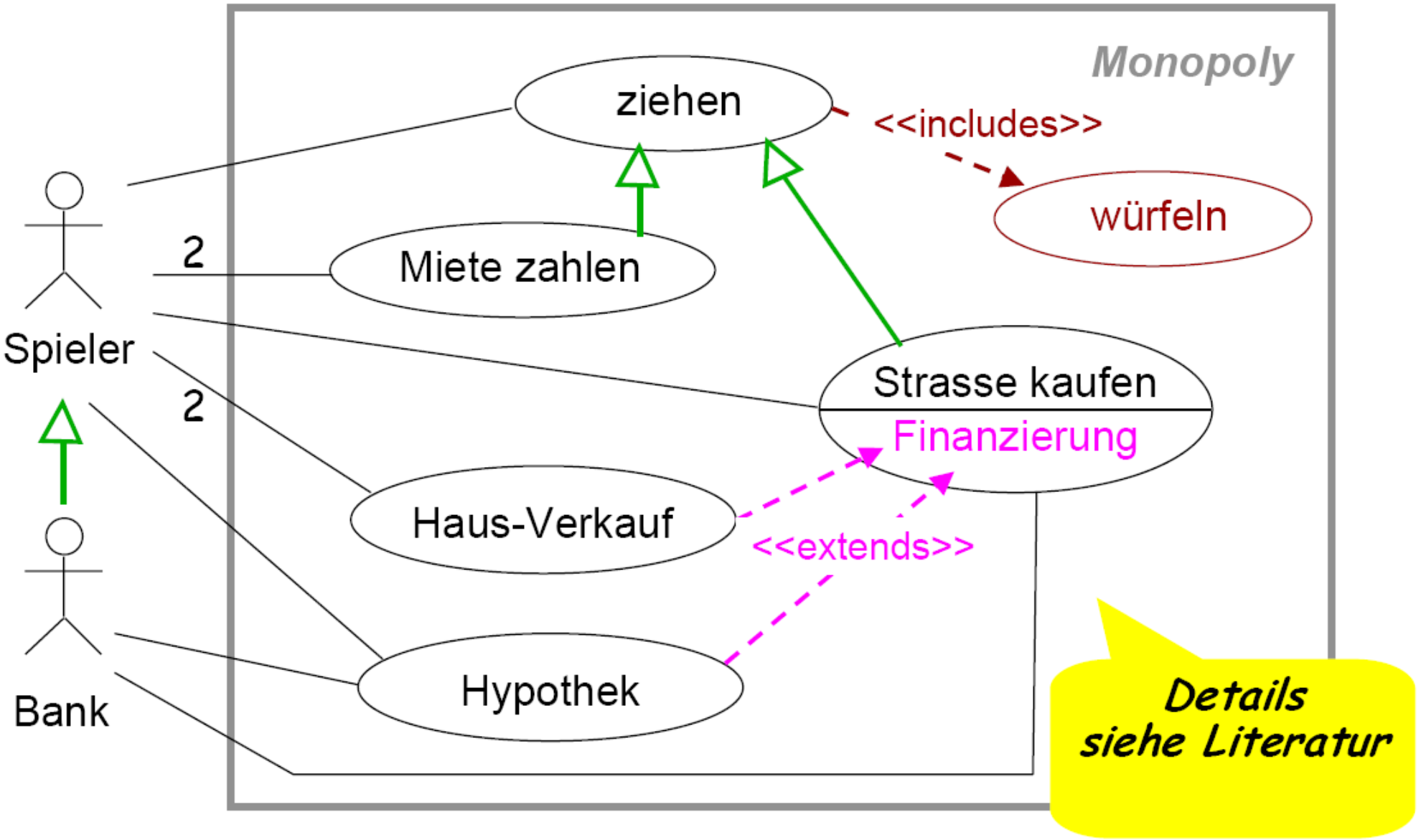
Use-Case-Diagramm zur Terminverwaltung



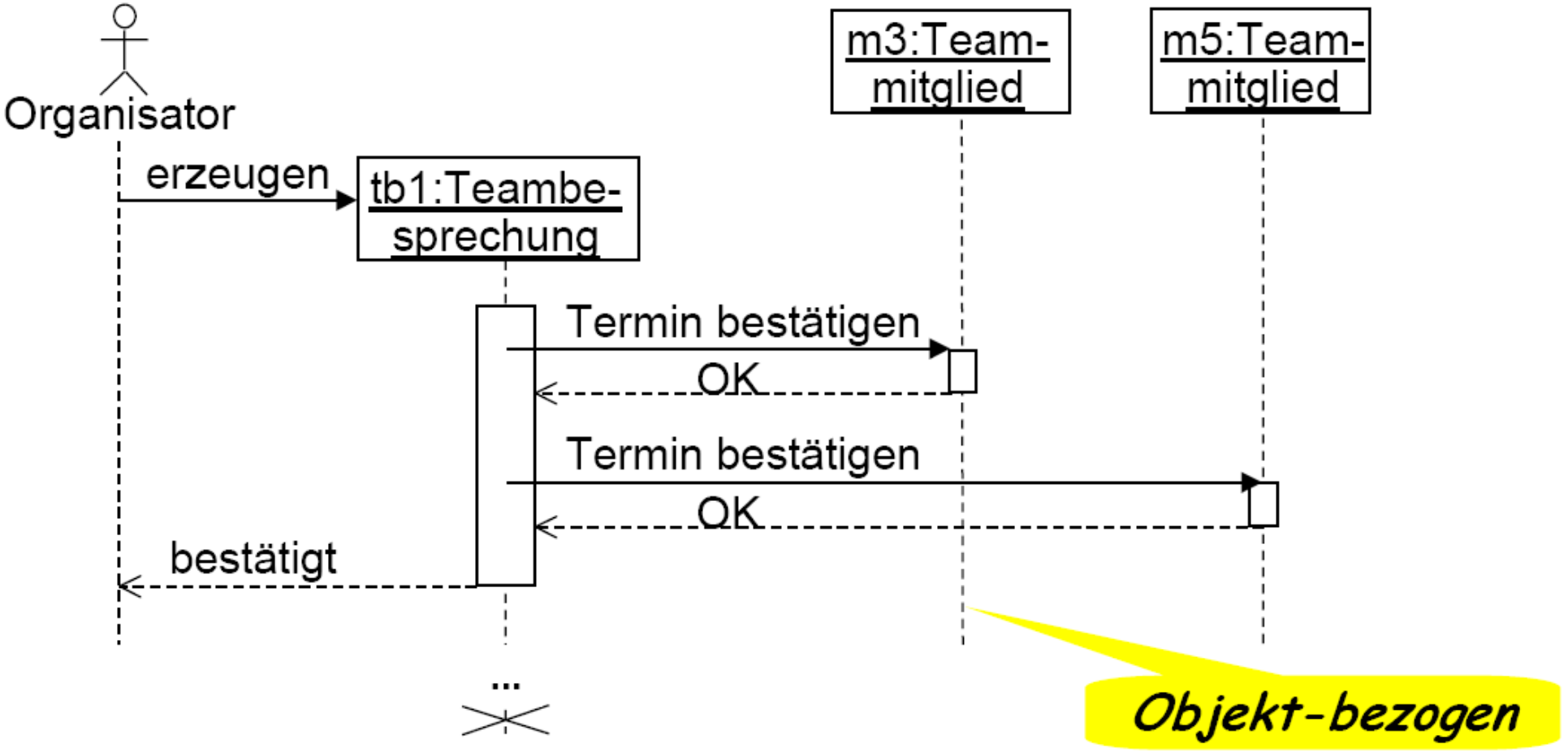
Use-Case-Diagramm zum Monopoly-Spiel



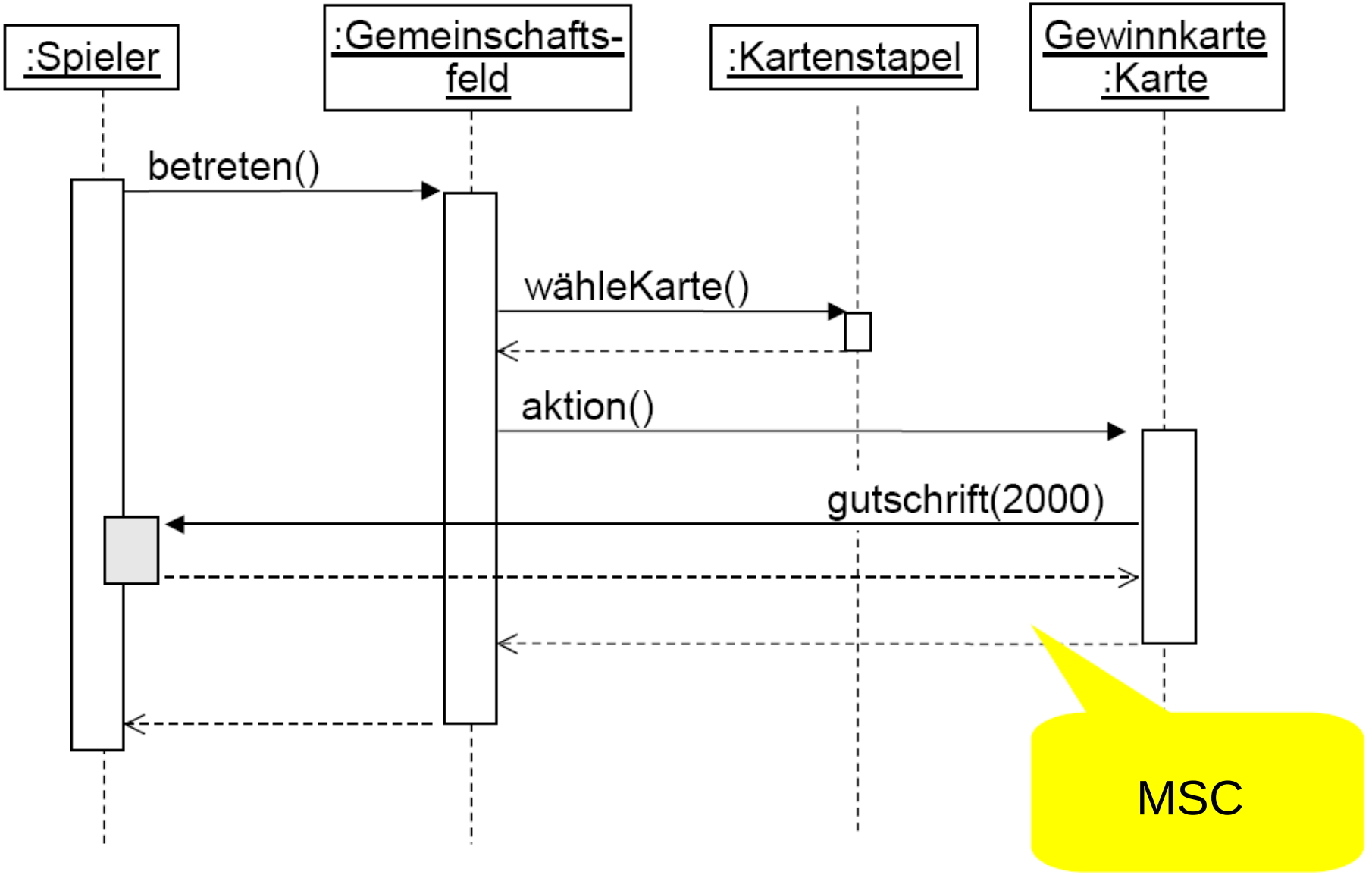
Etwas detaillierteres Use-Case-Diagramm



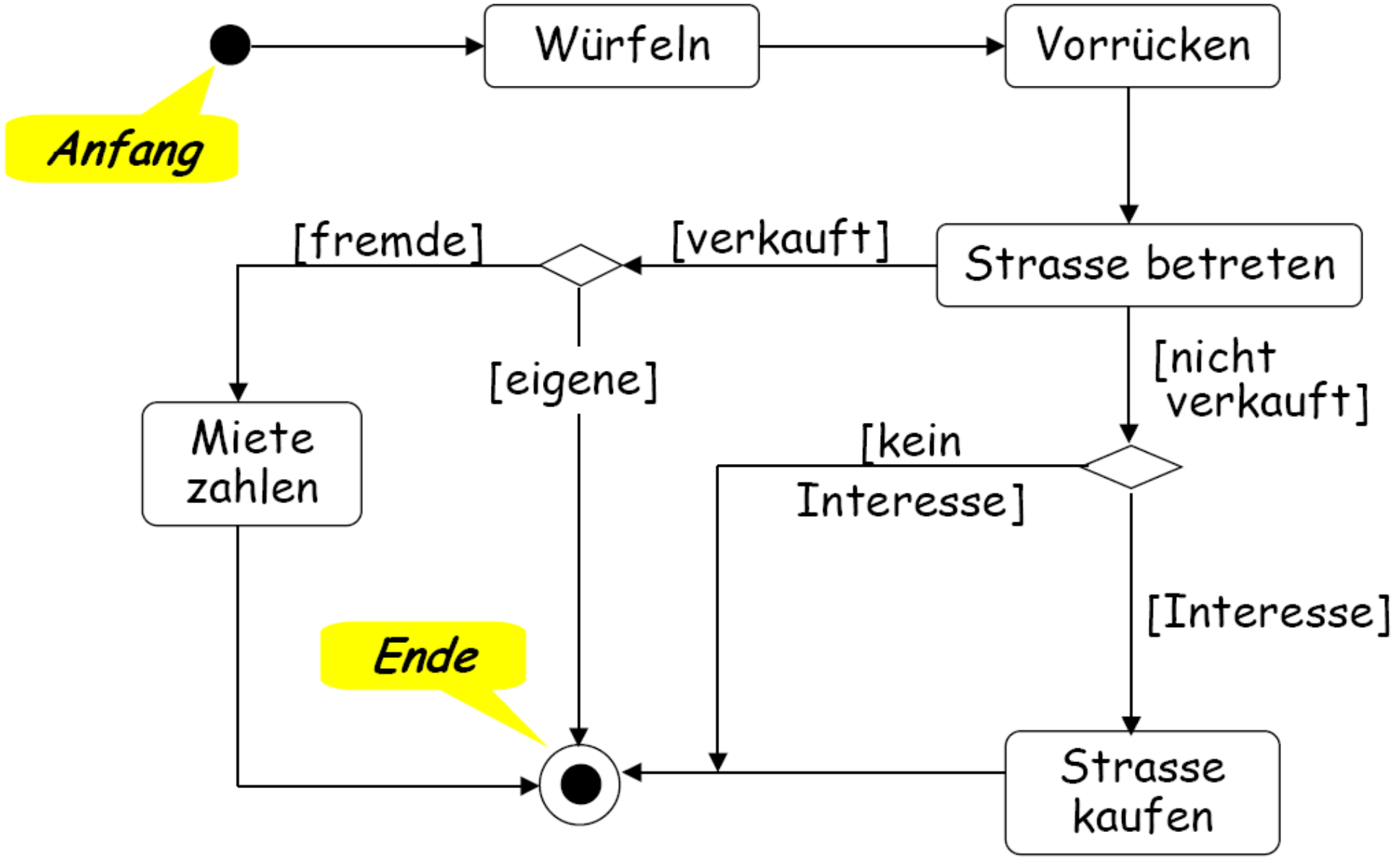
Sequenzdiagramme für Szenarien



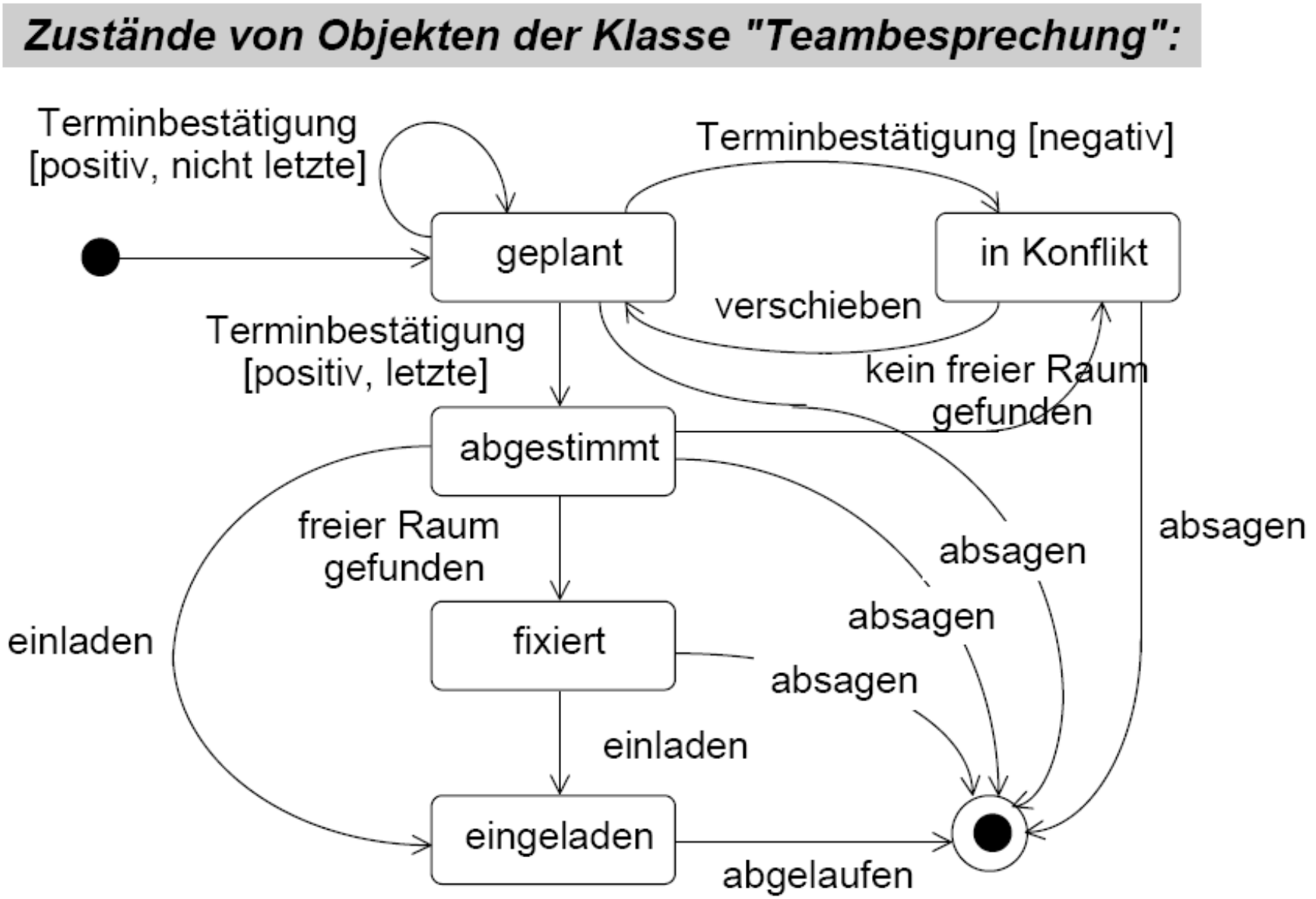
Sequenzdiagramm zu Monopoly-Zug



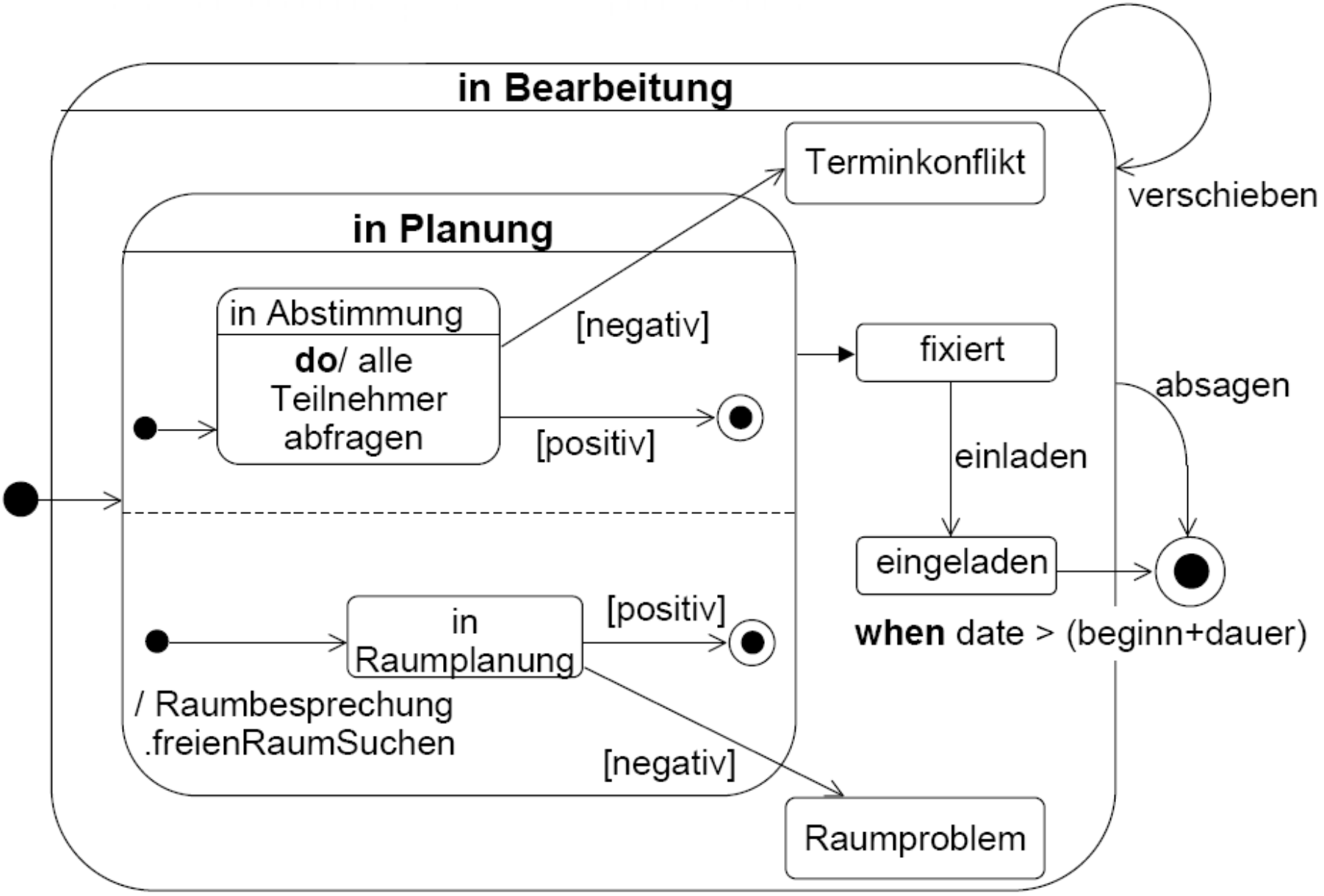
Aktivitätsdiagramm zu Monopoly-Zug



Ein Zustandsdiagramm mit Bedingungen



Zustandsdiagramm „mit Allem“



Fragen ?

Anforderungsmanagement

→ requirements management

Requirements Management

Was ist Anforderungsmanagement (AM)?

- Anforderungsmanagement umfasst Prozesse, die notwendig sind, um
  - Anforderungen und dazugehörige Informationen für verschiedene Rollen aufzubereiten und
  - diese konsistent zu ändern
- Anforderungsmanagement muss während der gesamten Entwicklung berücksichtigt werden

3.3.1 AM beantwortet Fragen verschiedener Rollen

- Projektmanager → siehe Projektmanagement (Kapitel 4)
  - Was ist der aktuelle Entwicklungsstand des Projekts?
  - Wie aufwendig ist eine Anforderungsänderung?
  - Wer ist für eine Anforderung zuständig?

- Kunde
  - Was ist der Ursprung einer Anforderung?
  - Welche Testfälle zeigen, dass das zu entwickelnde System die Anforderungen erfüllt?
  - Werden alle Anforderungen vom entwickelten System erfüllt?
  - Was kostet eine gewünschte Anforderungsänderung?

AM beantwortet Fragen verschiedener Rollen (II)

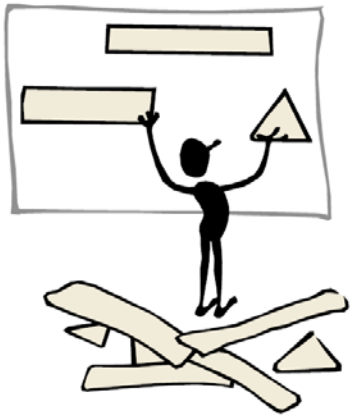
- Anforderungsingenieur
  - Sind die Anforderungen konsistent zu Vorgängerdokumenten?
  - Welche Anforderungen sind von einer Änderung betroffen?
- Entwerfer (Designer, SW Architekt)
  - Sind alle Anforderungen im Entwurf umgesetzt?
  - Auf welches Anforderungsdokument kann ich mich beziehen?
  - Warum wurde eine Anforderung wie umgesetzt?
- Tester
  - Erfüllt das System die Anforderungen?
  - Welche Testfälle müssen bei einer Änderung der Anforderungen erneut durchgeführt werden?

Anforderungsmanagement ist umso wichtiger, je ...

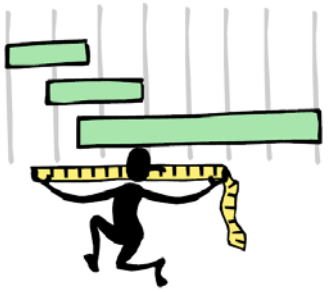
- ... größer die Zahl der Anforderungen
- ... länger die geschätzte Lebensdauer der Software
- ... wahrscheinlicher Anforderungen geändert werden müssen
- ... größer die Zahl der Beteiligten
- ... wichtiger die Qualität der Software

3.3.2 Aktivitäten des Anforderungsmanagements

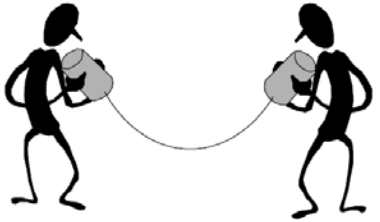
- 3.3.2 Aktivitäten
  - 3.3.2.1 Strukturieren



  - 3.3.2.2 Bewerten

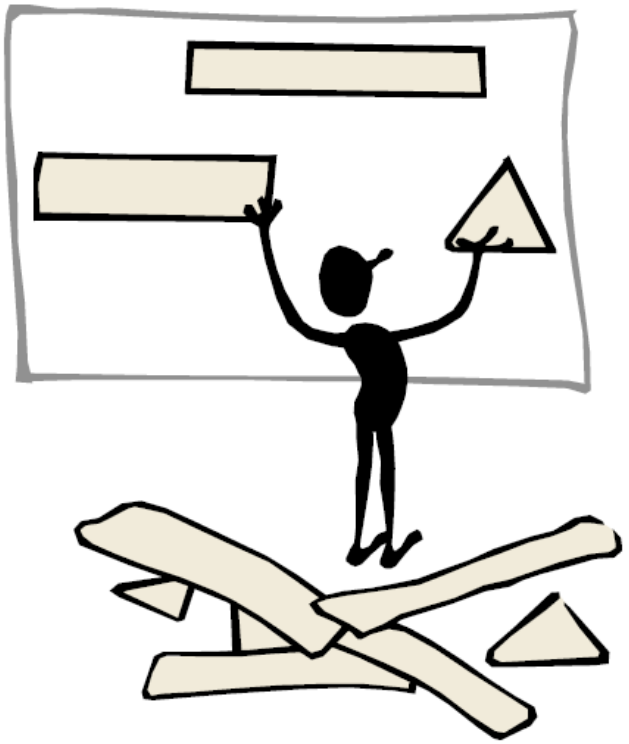


  - 3.3.2.3 Verfolgen



3.3.2.1 Aktivitäten - Strukturieren

- Klassifizieren
  - Gruppieren ähnlicher Anforderungen
- Identifizierbar machen
  - Anforderungen „griffig“ machen



Ziele der Klassifikation von Anforderungen

- **Zusammengehörigkeit und Überlappungen** von Anforderungen in einer großen Menge (geschrieben von unterschiedlichen Autoren) erkennen
- **Fehlende Anforderungen** erkennen
- Eine Organisation von Anforderungen aufbauen, die **orthogonal zur sequentiellen Dokumentenstruktur** ist
- **Ausschnitte** aus dem Anforderungsdokument für bestimmte Rollen generieren („Sichten”)

Beispiele für mögliche Klassifikationen

- Gegenstand der Anforderung
  - Systemfunktionalität, Benutzungsschnittstelle, Datenbank, Kommunikation, Beschränkungen (z.B. Performanz)
- Betroffene Benutzergruppe
  - System-Administrator, Verwaltungsangestellter, Lagerarbeiter
- Priorität der Anforderung
  - Essentiell, notwendig, wünschenswert
- Kosten/Aufwand einer Anforderung
  - 0-5 PM, 5-10 PM, >10 PM (PM=Personenmonat)

Vorgehen zur Definition von Klassen

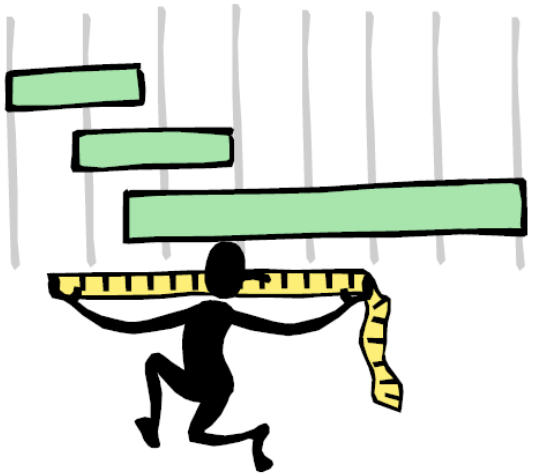
- Wähle für Projekt passende Struktur aus einem Standard aus
  - z.B. unternehmensinterne Vorgaben
- Identifiziere die (wesentlichen) Rollen im konkreten Projekt
- Befrage die Rollen, welche Fragen sie beantworten möchten
  - Welche Sichten auf die Gesamtheit der Anforderungen werden benötigt?
- Überlege mit welchen Informationen diese Fragen beantwortet werden können
- Lege Informationen fest, die zu jeder Anforderung erfasst werden
- Definiere, wer die Informationen wann und wie erfasst
- **Achtung:** je mehr Informationen, desto höher der Aufwand bei der Erfassung!!

Anforderungen identifizierbar machen

- Anforderungen sollten durch ein eindeutiges **und sich nicht mehr änderndes** Kürzel identifizierbar sein
- Dieses Kürzel dient der Referenzierung und sollte zur Vermeidung von Missverständnissen
  - den Typ der Anforderung widerspiegeln und
  - das Herkunftsdokument
- Beispiele
  - LH-FA11 = Funktionale Anforderung 11 im Lastenheft
  - PH-FA23 = Funktionale Anforderung 23 im Pflichtenheft
  - PA17 = Performanzanforderung 17
  - LH-FA-Temp19 = Funktionale Anforderung 19 des Teilsystems „Temperatur“ im Lastenheft

3.3.2.2 Aktivitäten - Bewerten

- Priorisieren
  - Gewichtung der Anforderungen
  - Beispiel für eine 3 stufige Priorisierung
    - *Essentiell (PrioA)*
    - *Notwendig (PrioB)*
    - *Wünschenswert (PrioC)*
- Risiken bewerten
  - Planung des mit einer Anforderung verbundenen Implementierungsrisikos
- Realisierungsaufwand abschätzen
  - Planung der Kosten einer Anforderung



Ziele der Priorisierung von Anforderungen

- Die Analyse von Anforderungen steuern
  - Akzeptable Kompromisse finden zwischen in Konflikt stehenden Zielen
  - Releases der Software planen (zuerst die wichtigen Anforderungen realisieren, dann die unwichtigen)
- Beispiel
  - ***Essentiell*** *(PrioA):* die Anforderung muss implementiert werden, sonst ist das System nutzlos
  - ***Notwendig*** *(PrioB):* das System ist ohne Umsetzung dieser Anforderung weniger effizient/effektiv
  - ***Wünschenswert*** *(PrioC):* das System wäre mit dieser Anforderung attraktiver
- Die Zuordnung von Prioritäten zu Anforderungen hängt von dem zu entwickelnden System ab.

Prüfen von Prioritäten

- Werden alle Anforderungen als essentiell eingestuft, dann ist die Priorisierung sinnlos.
- Die Prioritäten müssen überprüft werden:
  - Wäre die Anforderung noch immer essentiell, wenn es doppelt soviel kosten würde, diese Anforderung zu realisieren?
  - Dafür die Anforderung X des Beteiligten Y nicht realisiert werden kann?
- Anforderungen können auch relativ zueinander priorisiert werden:
  - Beispiel: {REQ7, REQ8, REQ9} > REQ1 > REQ2
- Wenn sich Beteiligte nicht über die Priorität einer Anforderung einigen können, dann liegt ein Konflikt vor.

Mögliches Vorgehen zur Priorisierung

- Schritt 1:
  Kunde bestimmt Zufriedenheit auf Skala von 1-5 wenn Anforderung im System umgesetzt ist
- Schritt 2:
  Kunde bestimmt Unzufriedenheit auf einer Skala von 1-5 wenn Anforderung nicht im System umgesetzt ist
- Schritt 3:
  Entwickler priorisiert die Anforderung basierend auf den zwei Zahlen

Ziele des Bewertens von Risiken

- Unvollständige Informationen identifizieren
- Notwendige Korrekturen an den Anforderungen frühzeitig erkennen
- Projektmanagement unterstützen (Kostenabschätzung und Risikomanagement)
- Risikoanalyse ist insbesondere wichtig bei neuen Softwaresystemen.

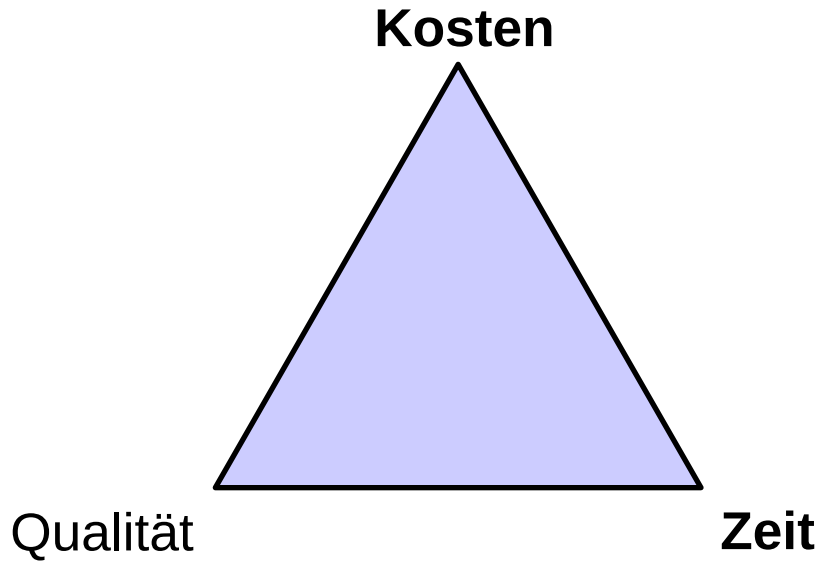
Beispiele für Arten von Risiken

- Performanz
  - Fraglich ob gewählte Hardware Performanz-Anforderungen genügt
- Sicherheit
  - Fraglich ob Sicherheitsanforderungen realisiert werden können
- Aufwand
  - Realisierung erscheint technisch schwierig und im Aufwand unvorhersagbar (z.B. verschiedene Implementierungsmechanismen müssen erst evaluiert werden)
- Stabilität
  - Anforderung ist nicht stabil und wird sich in absehbarer Zeit ändern
- Abhängigkeit
  - Die Änderung zieht weitere Änderungen nach sich und bedeutet extremen Änderungsaufwand

Ziele der Abschätzung des Aufwands

- Genaue Abschätzung der Kosten des gewünschten Systems
- Genaue Planung der Systementwicklung
- Verbesserung der Messung über verschiedene Projekte hinweg

- Beispiele für Maße:
  - Function Points
  - Boehm‘s COCOMO



Beispiel: Anforderungsspezifikation (eine Anforderung)

- clreichm
- makuehl
- dagebauer
- jomatheis
- dc
- Projektleiter
- User Requirements (USER)
  - Oberfläche (grafischen Darstellungen)
    - Perspektiven
    - Ansichten
      - Modellansicht (Modellbaum) (MA)
      - Diagrammübersicht (DÜ)
      - Objektkonfigurationsansicht (OK)
      - Labelsettingansicht (LA)
      - Suchergebnisansicht (SA)
      - Eigenschaftsansicht (EA)
      - Fehleransicht (FA)
      - Projektansicht (PA)
      - Outline (OL)
      - Clusteransicht (CA)
      - Mappingansicht (MAP)
      - Bedarfsansicht (BA)
      - Leitungssatzansicht (LSA)
      - Einstellungsansicht (ESA)
    - Zugeschnittene Oberfläche
    - Internationalisierung
    - Bildlaufleiste
    - Editoren
  - Bedienfunktionen
  - Metrik
  - Modellverwaltung
  - Start und Beenden der Applikation
  - Variantenmanagement
  - Wiederverwendung
  - Schnittstellen
  - Hilfe und Dokumentation

1.1.2.8.2 Diagramm Selektion (PA) ac158cdf1070cd71b97f

Details

Owner:	dagebauer	**Version:**	12	**Status:**	accepted
Type:	WHAT	**Priority:**	prioB		

Description: Das aktive Diagramm folgt der Selektion in der Projektansicht.

Dieses Feature ist an und abschaltbar in der Actionbar (Doppelpfeil).

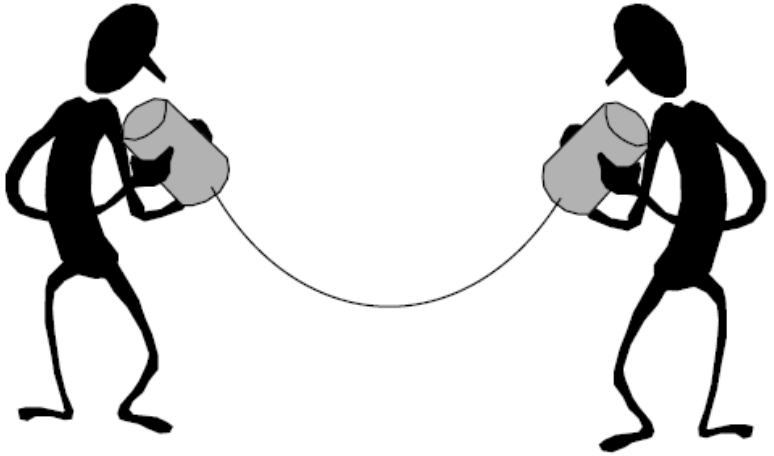
Comment:

Attributes

Attribute	Value
Preconditions	
Postconditions	
Reviewed By	clreichm; meberw
Revised Description	
Review Comments	
Estimate	1
Number of Ambiguities Found	0
Target Version	V1.0.1
Part of Test Suite @ Dev Team	false
System Test Needed	true
System Test Priority	A
System Test Frequency	always
TraceTo	1.1.2 Ansichten 1.1.6 Editoren
TraceFrom	
ReceiveFroms	
ReceiveTos	

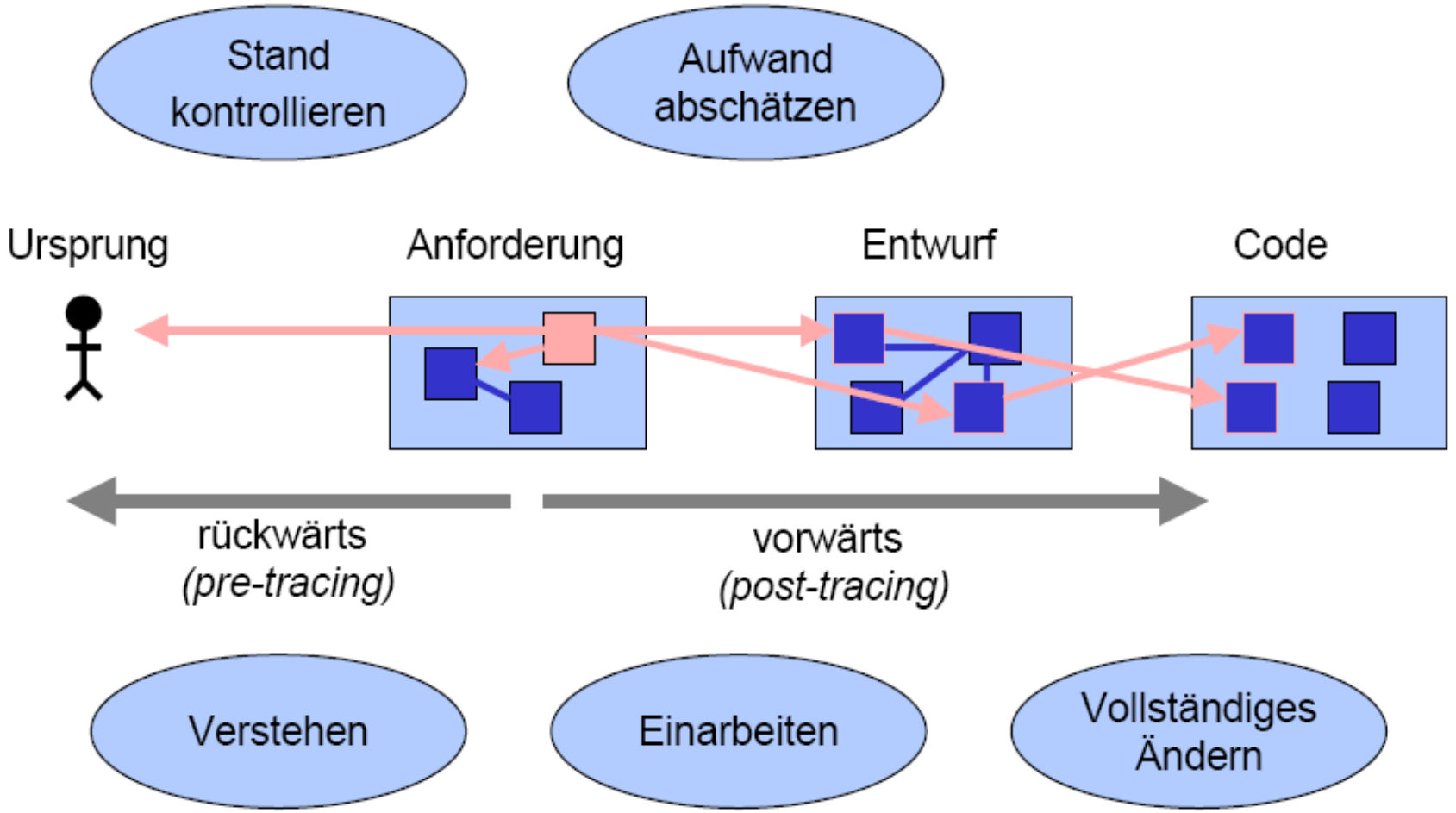
3.3.2.3 Aktivitäten - Verfolgbarkeit

- Dokumentation von Beziehungen einer Anforderung zu:
  - anderen Anforderungen (horizontal)
  - Dokumentationselementen in anderen Dokumenten (vertikal)
  - anderen Versionen der Anforderung (evolutionär)



Ziele von Verfolgbarkeit

- Anforderungsverfolgbarkeit „Fähigkeit, das Leben einer Anforderung zu beschreiben und verfolgen zu können und zwar sowohl vorwärts als auch rückwärts ...“ (Gotel et al.)



Typische Verknüpfungspunkte der Anforderungsverfolgung

- *Quellen:*
  Verweise auf Personen/Stakeholder, die Anforderungen einbringen
- *Begründungen:*
  Verweise auf Begründungen (z.B. Gesetze, Normen, Geschäftsziele)
- *Anforderungen:*
  Verweise auf andere Anforderungen (z.B. übergeordnete Anforderungen)
- *SW–Architektur:*
  Verweise auf Teilsysteme, die Anforderungen umsetzen
- *Testfälle:*
  Verweise auf Testfälle, die die Anforderungen abprüfen
- *Schnittstellen von Fremdsystemen*

Wer dokumentiert wann und wie Beziehungen?

• Wer?	
– Ersteller eines Dokuments	Wissen
– Verantwortlicher für Verfolgbarkeit	Kein Entwicklungsaufwand
	Zugriffsrechte einfacher
• Wann?	
– Während Entwicklung	Wissen
– Nach Fertigstellung	Aufwand
• Wie?	
– Explizite Links (z.B. Querverweise, Matrizen)	Analyse einfach
– Implizite Links (z.B. Beziehungen durch Namen oder Dokumentenstruktur)	Aufwand geringer
• Tool?	
– Mit einem RM-Werkzeug	Verwaltung einfach
– Ohne spezielles Werkzeug	Anschaffungskosten

Darstellung von Verfolgbarkeits-Information

- Implizit

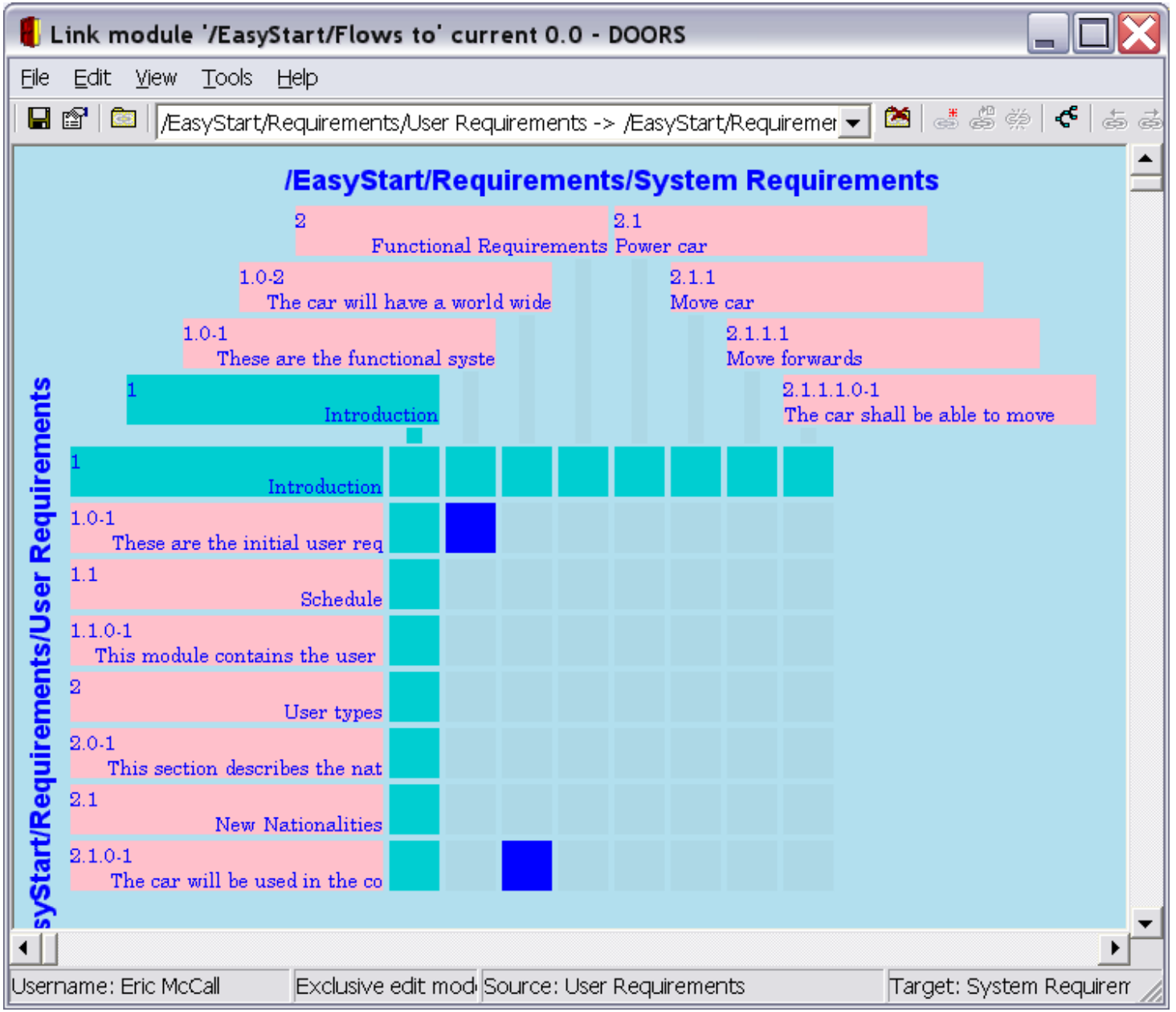
- Traceability-Listen

Anforderung	Testfall
UR100	T1, T2, T6
UR101	T1, T3

- Traceability-Matrix

	T1	T2	T3	T4
UR100	X	X		
UR101	X		X	

Beispiel Verfolgbarkeits-Information (DOORS)



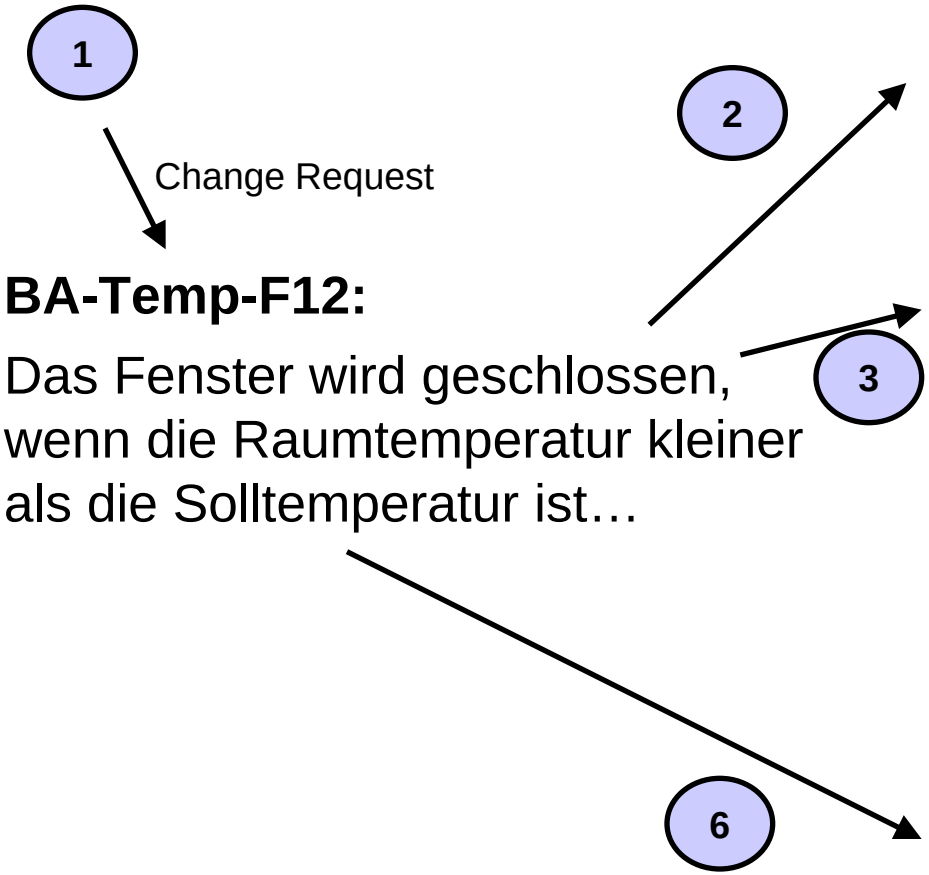
Richtlinien zur Nutzung von Beziehungen

- Definition des Zwecks z.B.
  - Planung von Änderung
  - Durchführung von Änderungen
  - Verstehen einer Entwurfskomponente
  - Wiederverwendung einer Anforderung
- Festlegung der zu analysierenden Beziehungen z.B.
  - Alle Beziehungen
  - Nur bestimmte Beziehungen

Richtlinien zur Nutzung – Beispiel Änderungen

1. Identifiziere Startmenge von Änderungskandidaten
2. Identifiziere weitere Änderungskandidaten im Dokument durch Analyse der „repräsentiert“ und „wird verfeinert durch“-Beziehungen
3. Bestimme sekundäre Änderungskandidaten im Dokument durch Analyse der „ist abhängig von“-Beziehungen
4. Prüfe, ob sekundäre Änderungskandidaten geändert werden müssen
5. Geg. Schritt 3
6. Identifiziere weitere Änderungskandidaten durch Analyse der „wird umgesetzt durch“-Beziehungen

Richtlinien zur Nutzung – Beispiel Änderungen



- **BA-Temp-F12:**
  Das Fenster wird geschlossen, wenn die Raumtemperatur kleiner als die Solltemperatur ist…

- **Übersicht:**
  Das System schließt die Fenster

Die Raumtemperatur wird mit Hilfe von 3 Raumtemperatur- Sensoren gemessen…

Das Fenster wird mit Hilfe von 2 Motoren geschlossen….

3.3.3 Aktivitäten – Änderungsmanagement

- Änderungsmanagement umfasst Aktivitäten zur
  - Identifikation von Änderungen
  - Planung von Änderungen
  - Durchführung von Änderungen

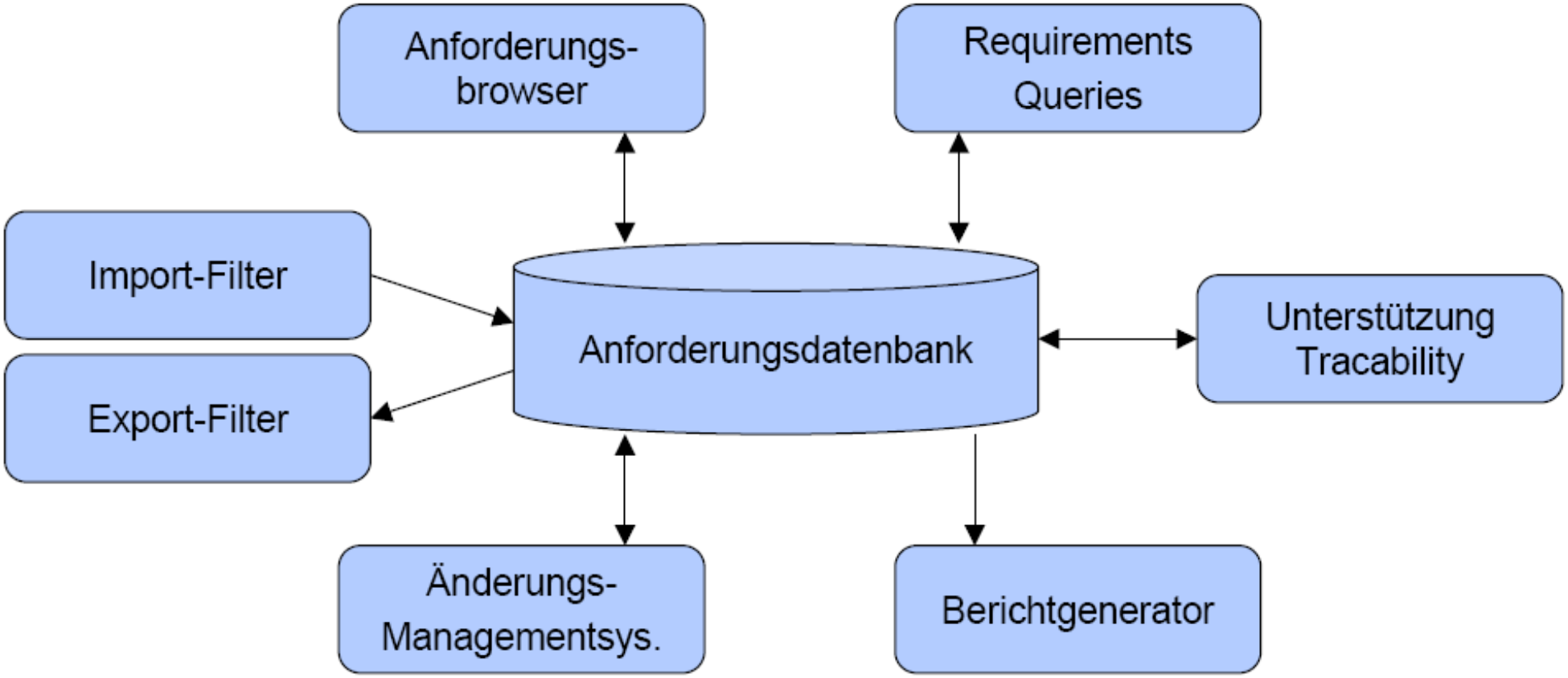
Zusammenfassung - Anforderungsmanagement

- Anforderungsmanagement ist wichtig um:
  - die Zusammenarbeit von verschiedenen Rollen zu unterstützen und
  - 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

Aufgaben von Requirements Management Werkzeugen

- Verwalten aller Dokumente
  - z.B. Anforderungen, Modelle, Testpläne, Änderungswünsche
- Verwaltung logischer Beziehungen zwischen den Dokumenten
  - Verfolgbarkeit
- Editieren von Dokumenten
  - Mehrbenutzerfähigkeit, Zugriffskontrolle, Konfigurations- und Versionsmanagement)
- Organisation der Information
  - Gruppierung, Hierarchisierung, Attributierung mit Zusatzinformation
- Reporte generieren, beliebig konfigurierbar
  - Z.B.: Wo sind die Anforderungen realisiert?
  - Z.B.: Warum steht dieses Stück Code hier?

3.3.4 Grundlegende Struktur von RM Systemen



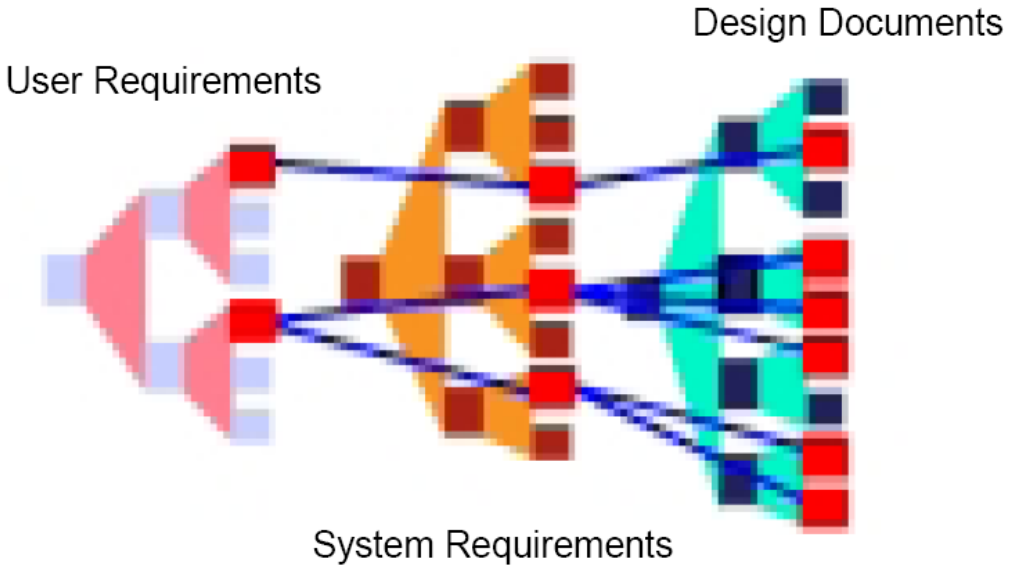
Übersicht Requirements Management Systeme

Tool Name	Vendor	Description
AnalystStudio	Rational Software	Tool Suite. Includes RequisitePro, Rose, SoDA and ClearCase
Caliber-RM	Technology Builders, Inc (TBI)	Requirements traceability tool
CORE	Vitech Corporation	Full life-cycle systems engineering CASE tool. It supports the systems engineering paradigm from the earliest days of concept development and proposal development, requirements management, behavior modeling, system design and verification process.
CRADLE/REQ	3SL (Structured Software Systems)	Requirements Management tool capable of storing within its database, graphs, spreadsheets, tables, diagrams and any other information as part of a requirement.
DOORS	Telelogic (was QSS)	Requirements traceability tool
DOORSrequireIT	Telelogic (was QSS)	Requirements trace tool that is integrated with Microsoft Word. Data can be merged with DOORS databases
GMARC	Computer Systems Architects (CSA)	Generic Modeling Approach to Requirements Capture (GMARC). Toolset will also generate quality metrics for a specification enabling formal proof that use of the GMARC has improved the requirement set.
icCONCEPT	Integrated Chipware	Requirements traceability tool. Replaces RTM
IRqA	TCP Sistemas e Ingenieria	Integral Requisite Analyzer. A requirements management tool, but also a requirements analysis environment, that includes facilities to support problem domain modeling and automatic domain analysis.
ITraceSE	ITrace Systems	Requirements traceability tool
Life*CYCLE	Computer Resources International	Requirements traceability tool. (No longer available)
RDT	IGATECH Systems Pty Limited	Requirements traceability tool
RequisitePro	Rational Software	Requirements traceability tool. Also part of AnalystStudio
RIMS	Sygenex Incorporated	Requirements and Information Mamagement System (RIMS).
RTM	Integrated Chipware	Requirements traceability software. See icCONCEPT product.
RTS	Electronic Warfare Associates, Inc.	Requirements Traceability Systems (RTS). Complete foundation for tracking the requirements of a software/hardware project through the accompanying documentation and source code. This includes tracking the development and testing status of requirements
SLATE	SDRC SSG	System Level Automation Tool for Engineers (SLATE) is used to capture, organize, build and document system-level designs from raw concepts through structural partitioning. Interfaces to Office 97, Project and CASE tools.
Systems Engineer	Blue Spruce	Requirements trace tool
Tofs	Tofs AB	Tool For Systems. Assists you in realizing and managing not only software, but also the manual (human) and hardware (electronic, hydraulic, mechanic, etc) parts of a system, which complete the system's missions together with the software.
Tracer	RBD, Inc.	Requirements traceability tool
XTie-RT	Teledyne Brown Engineering	Requirements traceability tool

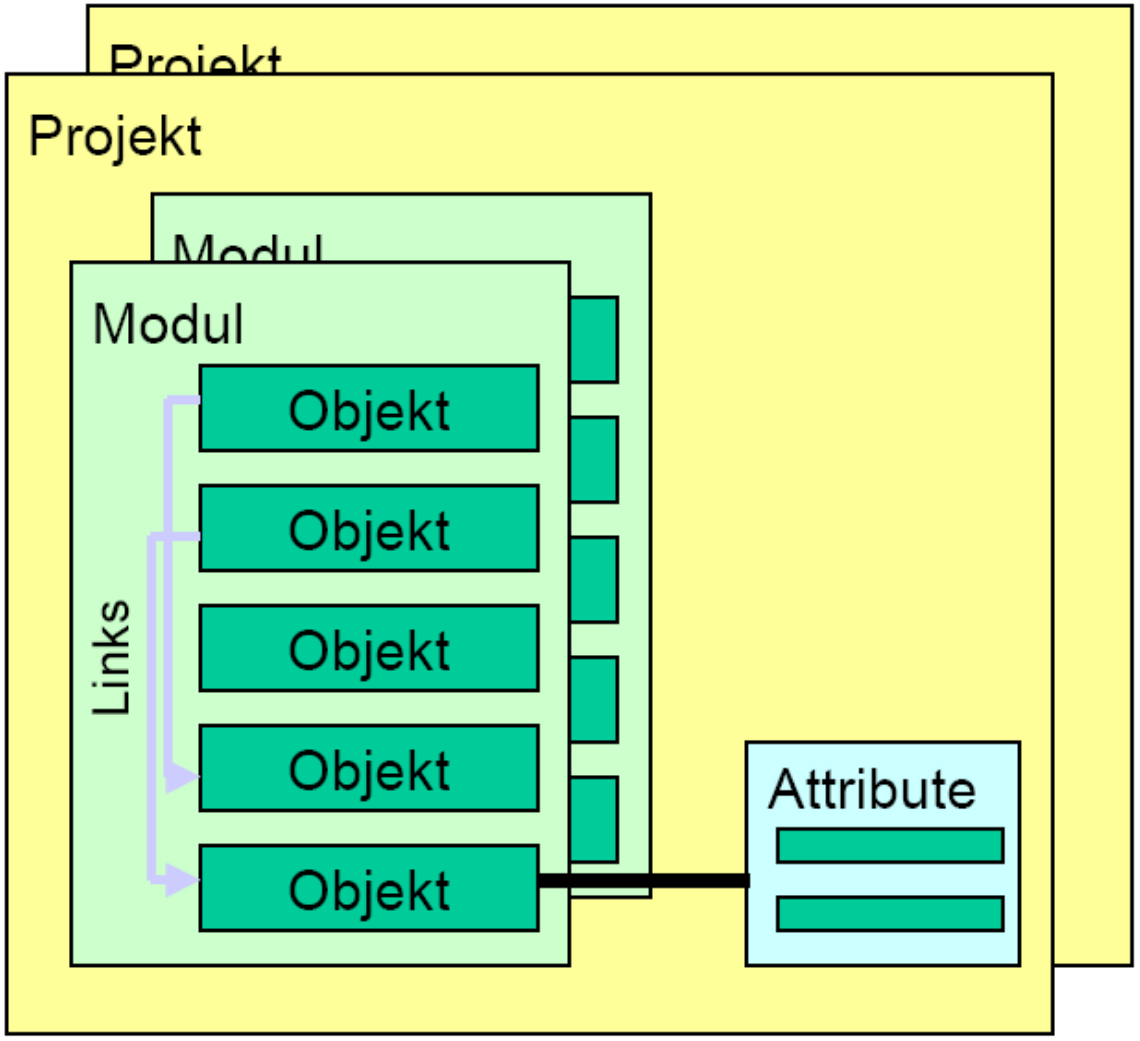
http://www.incose.org/productspubs/products/setools/tooltax/reqtrace_tools.html, 2013

RM Werkzeug DOORS - Philosophie

- Verwaltung verschiedener Klassen von Dokumenten, z.B.
  - User Requirements
  - System Requirements
  - Design Documents
  - Test Cases
- Nachvollziehen des Zusammenspiels der **Dokumente** über Links (→ Traceability)



RM Werkzeug DOORS - Grundstruktur



- Ein Objekt entspricht einer Anforderung
- Gleichartige Anforderungen (bzgl. Abstraktion) werden in einem Modul zusammengefasst
- Neben „Formalen“ Modulen (siehe links) gibt es auch Deskriptive Module (Plain Text) und Link Module (verwalten die Links)

Dokumentation von Requirements

- Jedes DOORS-Objekt sollte genau eine Anforderung enthalten

- Objekte haben Überschrift-Eigenschaft
- Objekte stehen in einer Objekt-Hierarchie (keine Vererbung o.ä.)

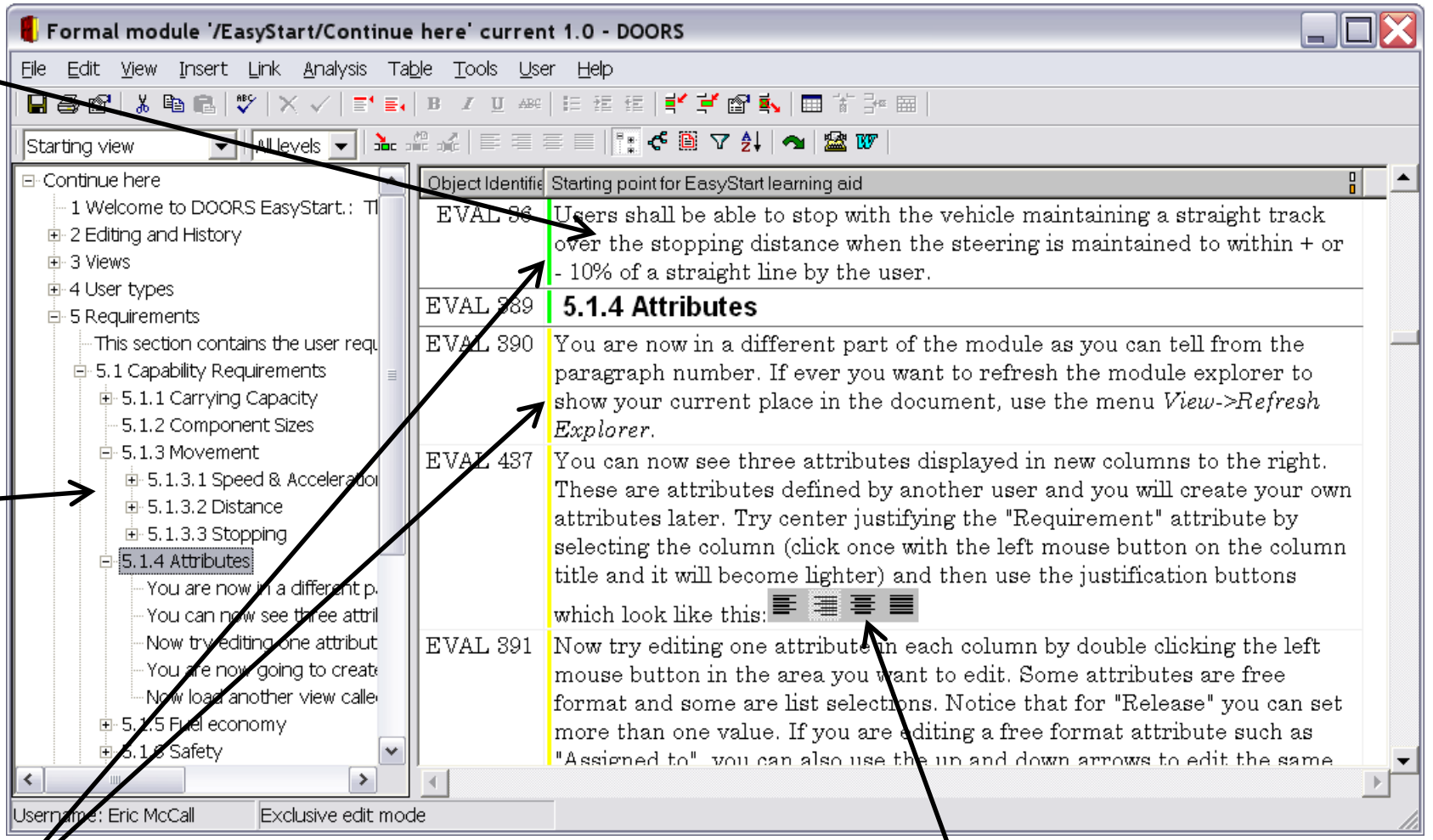
} Realisierung von Dokumenten-Strukturen mit Kapitelnummern (Ein- und Ausblenden von Ebenen ist möglich)

- Der Inhalt eines Objekts ist völlig frei, die inhaltliche Verantwortung liegt voll beim Autor
- Übliche Editierfunktionalität wird unterstützt

Dokumentation von Requirements

Textuelle Anforderung (keine Vorgabe bezüglich der Struktur o.ä.)

Dokumenten-struktur

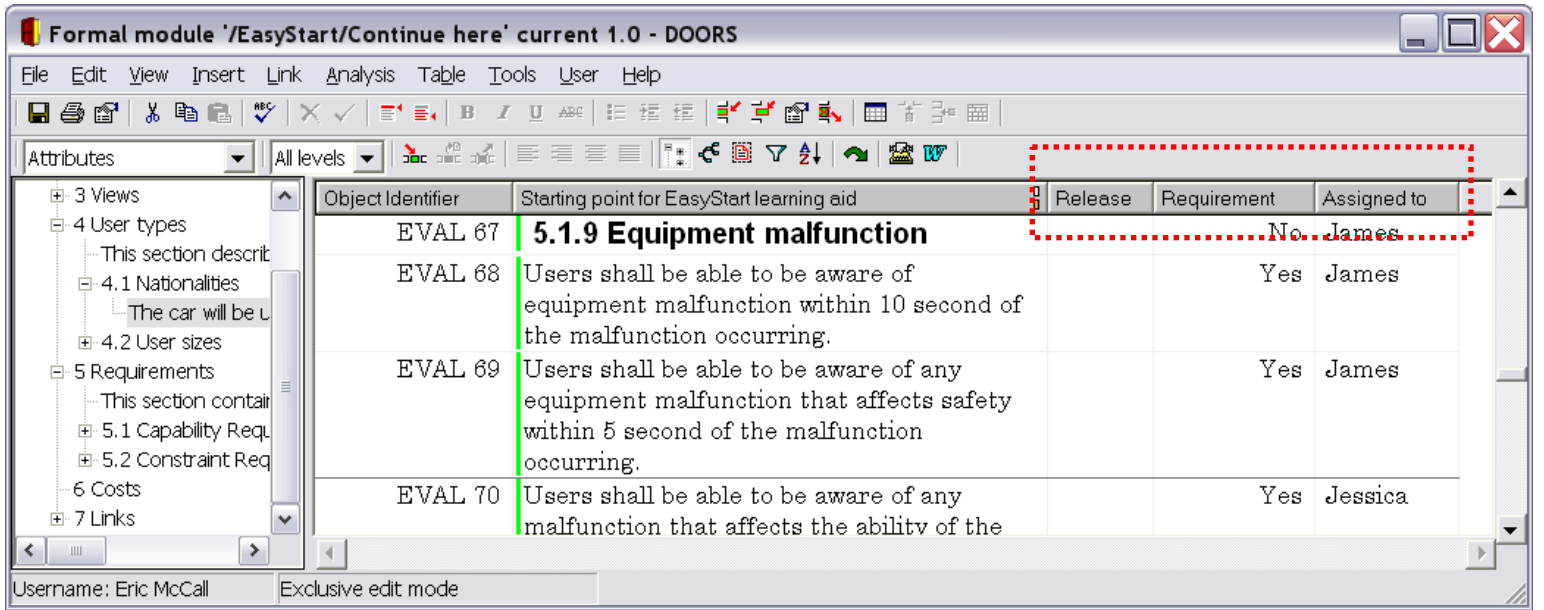


Bearbeitungsstandanzeige:
Grün (nicht verändert), Gelb (neu), Rot (geändert)

Abbildung

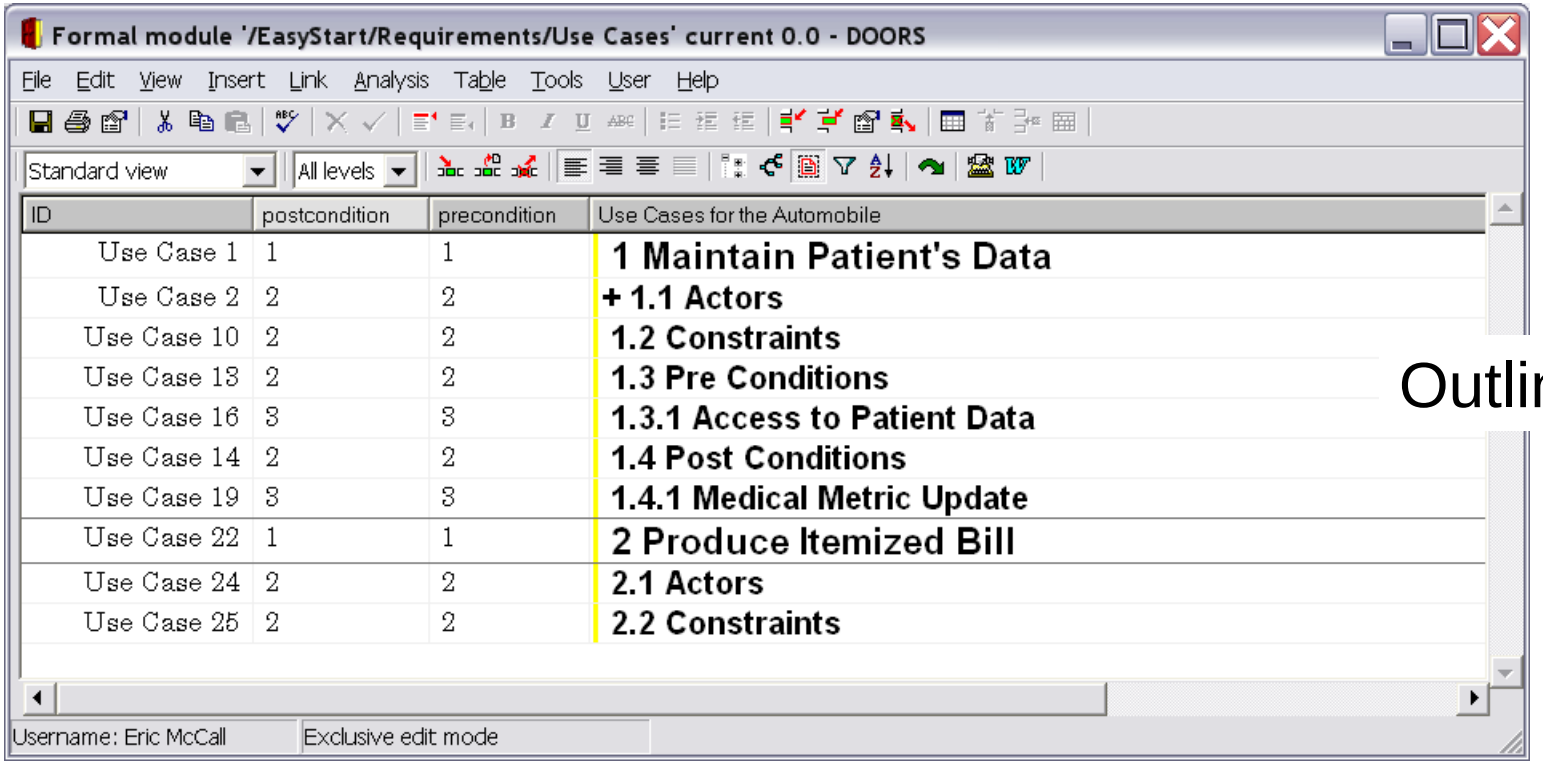
Attribute und Sichten (1)

- Jedem Objekt sind Attribute zugeordnet
  - Systemseitige Attribute (z.B. Object Identifier, Ersteller, Datum, ...)
  - Benutzerdefinierte Attribute (z.B. Bearbeitungsstand, Testbar, ...)
- Attributtypen
  - Zahlen
  - Zeichenketten
  - Aufzählungstypen
  - Mengenwertige Aufzählungstypen



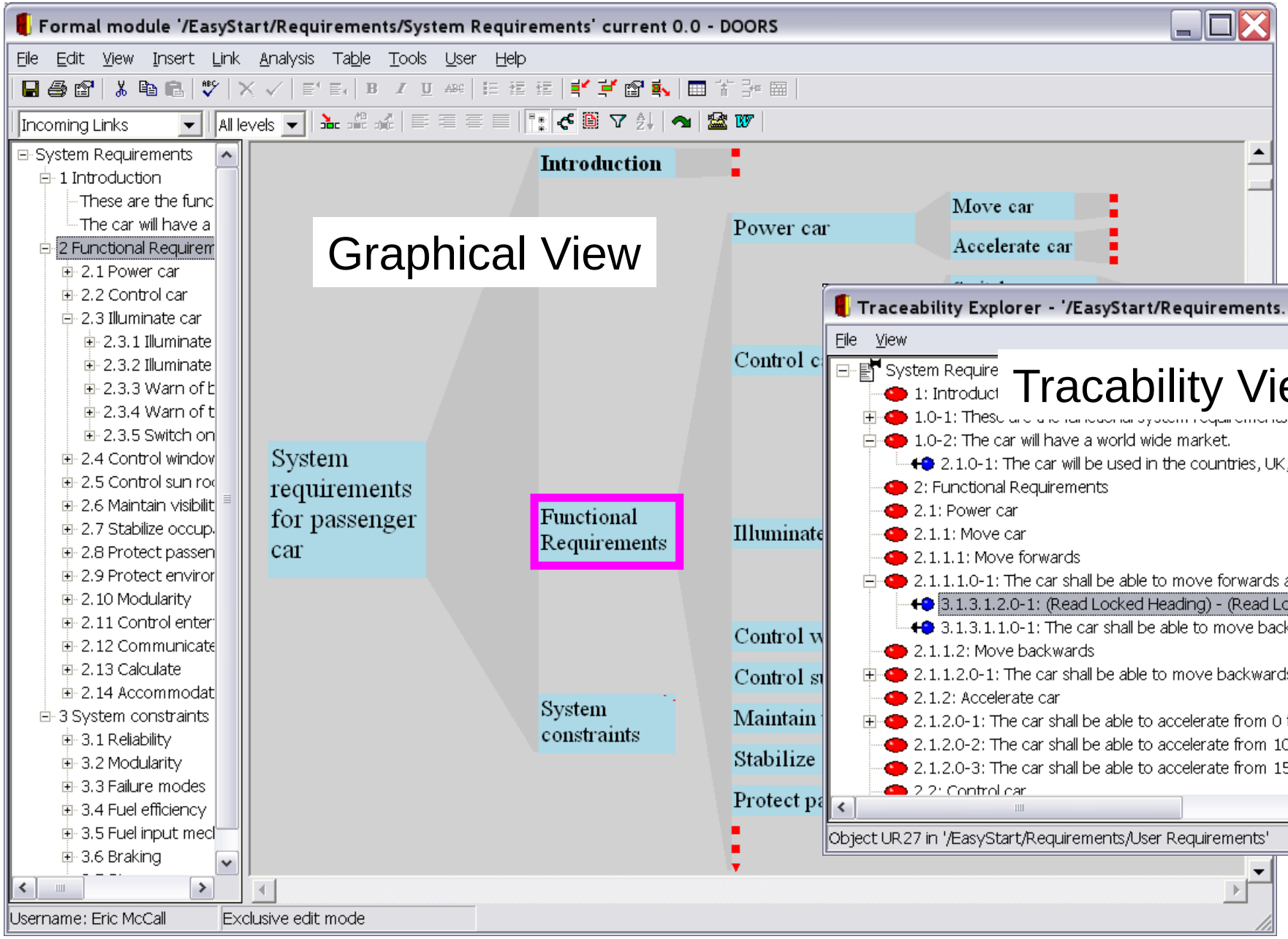
Attribute und Sichten (2)

- 3 Darstellungsformen
  - Outline View (textuelle Darstellung, mit und ohne Filterung)
  - Graphical View (graphische Darstellung der Struktur)
  - Traceability View (Zusammenhang der Anforderungen)



Outline View

Attribute und Sichten (3)

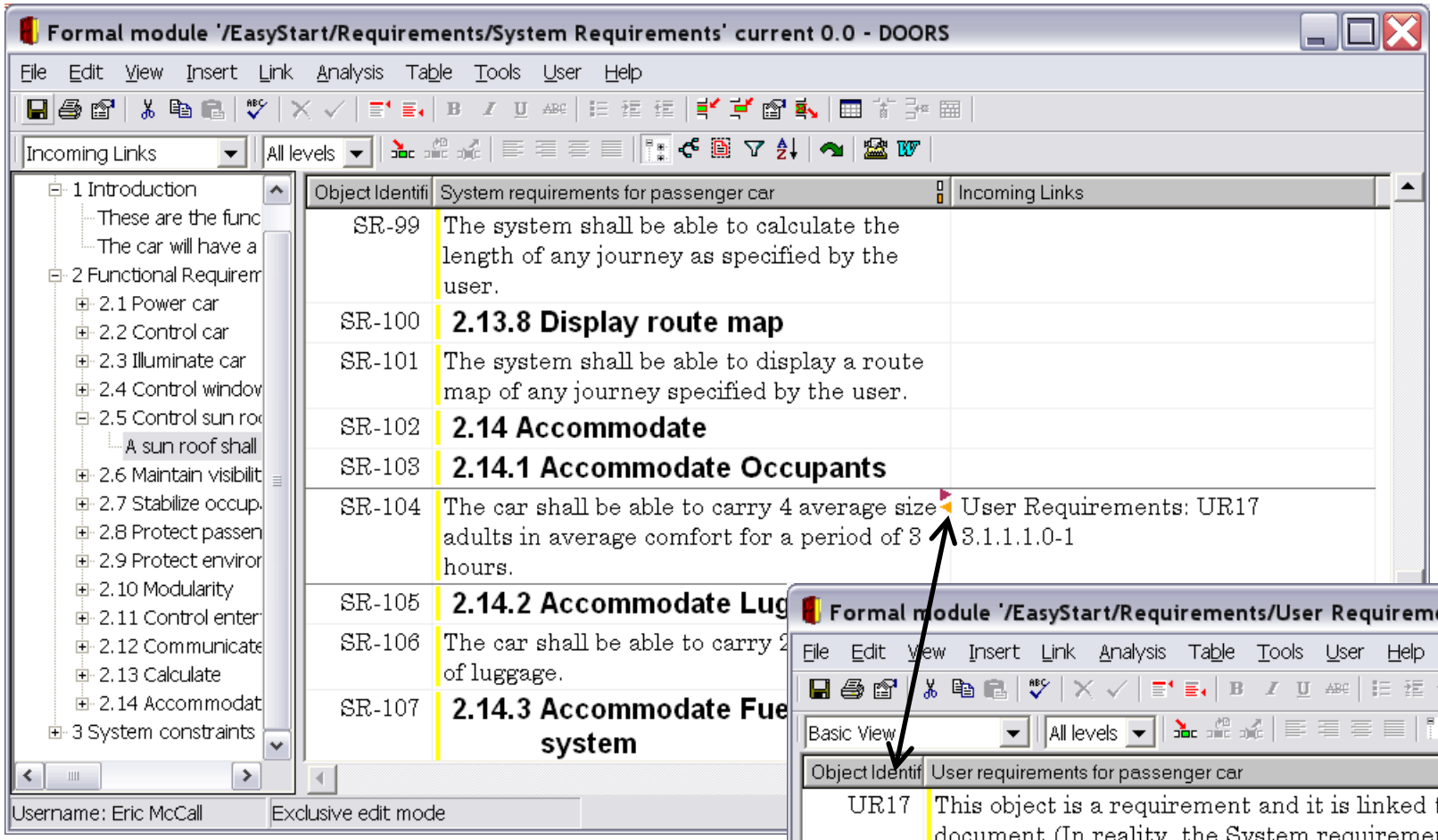


Links und Requirements Tracing

- Links verbinden zwei Objekten (bidirektional) miteinander
  - Innerhalb eines Moduls
  - Zwischen Modulen

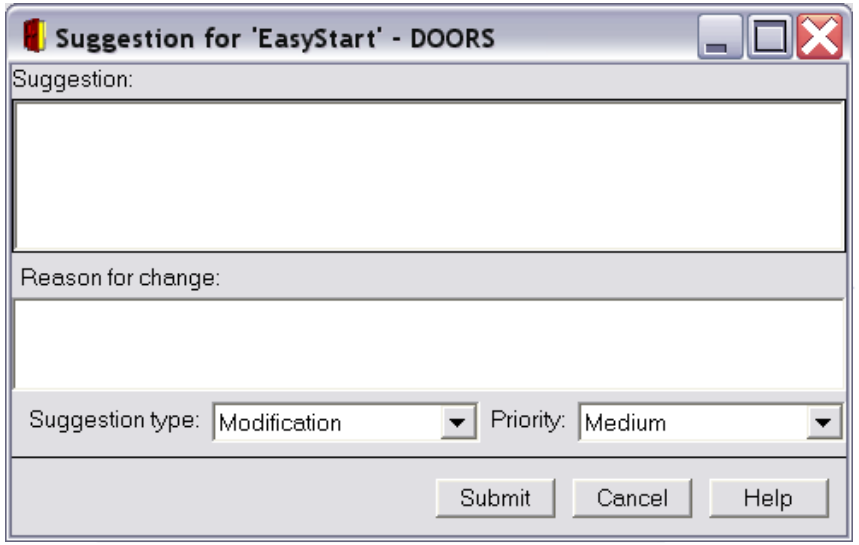
SR-103	**2.14.1 Accommodate Occupants**	
SR-104	The car shall be able to carry 4 average size adults in average comfort for a period of 3 hours.	User Requirements: UR17 3.1.1.1.0-1

Links und Requirements Tracing

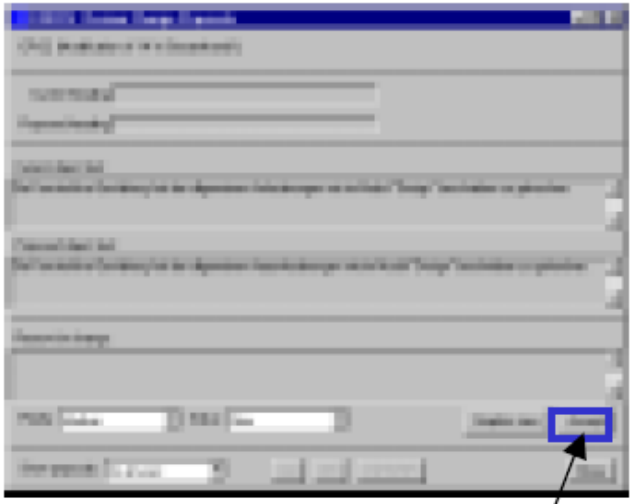


RM Werkzeug DOORS - Weitere Features

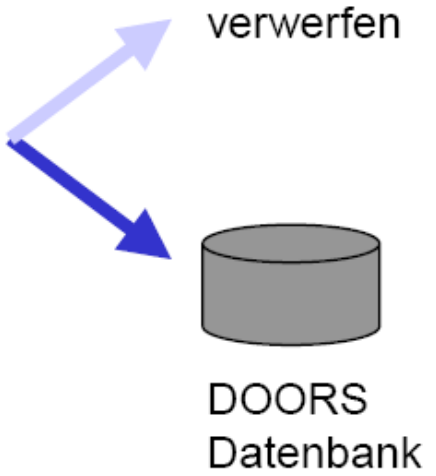
- Change Request System



Erstellen eines Änderungsantrags



Entscheidung über Antrag in einem Review → Accept oder Clear



- Einfaches Konfigurationsmanagement über sog. Baselines (d.h. Einfrieren eines Änderungsstandes unter einer neuen Versionsnummer)
- Für jedes Objekt wird die Historie mitgeführt (wer hat wann was verändert)

Fragen ?