

PMRE

Zusammenfassung PMRE FS18

25.06.2019

Inhaltsverzeichnis

| | | |
|----------|--|----------|
| 1 | Einführung | 3 |
| 1.1 | Business Analyst | 4 |
| 1.2 | Setting | 4 |
| 1.3 | Ziele | 4 |
| 1.4 | Zieldiagramm | 6 |
| 2 | Stakeholder | 6 |
| 2.1 | Stakeholderanalyse | 7 |
| 2.2 | Stakeholderklassen | 7 |
| 2.2.1 | Management | 7 |
| 2.2.2 | Anwender | 7 |
| 2.2.3 | Wartungs- und Servicepersonal | 7 |
| 2.2.4 | Schulungs- und Trainingspersonal | 8 |
| 2.2.5 | Käufer | 8 |
| 2.2.6 | Marketing und Vertrieb | 8 |
| 2.2.7 | Entwickler | 8 |
| 2.2.8 | Projekt- und Produktgegner | 8 |
| 2.2.9 | Produktbeseitiger | 8 |
| 2.2.10 | Sicherheitsbeauftragte | 8 |
| 2.2.11 | Betriebsrat | 9 |
| 2.2.12 | Personen aus anderen Kulturkreisen | 9 |
| 2.2.13 | Gesetzgeber | 9 |
| 2.2.14 | Standardisierungsgremien | 9 |
| 2.2.15 | Öffentliche Meinung | 9 |
| 2.2.16 | Prüfer und Auditoren | 9 |
| 2.2.17 | Technische Experten | 9 |
| 2.2.18 | Produzenten des Produkts | 10 |
| 2.2.19 | Produktdesigner | 10 |

| | |
|--|-----------|
| 2.2.20 Prozessoptimierung und Arbeitsergonomie | 10 |
| 2.2.21 Experten für das Systemumfeld | 10 |
| 2.2.22 Produktlinienverantwortliche | 10 |
| 2.2.23 Forschung und Entwicklung | 10 |
| 2.2.24 Controllingabteilung | 10 |
| 2.3 Notation von Stakeholdern in einem Projekt | 11 |
| 2.4 Stakeholder-Map | 11 |
| 3 Kontextdiagramm | 12 |
| 3.1 Aspekte des Systemkontexts | 12 |
| 3.2 Typische Fehler | 12 |
| 4 Anforderungen ermitteln | 12 |
| 4.1 Anforderungsquellen | 14 |
| 4.1.1 Stakeholder | 14 |
| 4.1.2 Die 7 Produktdimensionen | 14 |
| 4.1.3 Kano-Modell | 15 |
| 4.1.4 Interview | 17 |
| 5 Anforderungen modellieren | 18 |
| 6 Diagramme in RE | 18 |
| 6.1 Use Case Diagramm | 18 |
| 6.2 Aktivitätsdiagramm | 18 |
| 6.3 Klassendiagramm | 19 |
| 6.4 CRUD-Matrix | 19 |
| 6.5 Zustandsdiagramm | 19 |
| 7 Agiles Requirement Engineering | 25 |
| 7.1 Überblick | 25 |
| 7.2 Backlog Items | 26 |
| 7.3 Repräsentation von Anforderungen in agilen Projekten | 26 |
| 7.3.1 Features | 26 |
| 7.3.2 Scrum User Stories | 27 |
| 7.3.3 Scrum Backlog Items | 29 |
| 7.4 Story Splitting | 31 |
| 7.5 Anforderungsmanagement in agilen Projekten | 31 |
| 7.5.1 Anforderungsmanagement in Scrum | 32 |
| 8 User Story Mapping | 33 |
| 8.1 Beispiel TriHow | 33 |
| 9 Prüfen von Anforderungen | 38 |
| 9.1 Motivation, Kriterien & Prinzipien | 38 |

| | |
|---|-----------|
| 9.2 Überprüfung von Anforderungen | 38 |
| 9.2.1 Prüfkriterien "Inhalt" | 38 |
| 9.2.2 Prüfkriterien "Dokumentation" | 39 |
| 9.2.3 Prüfkriterien "Abgestimmtheit" | 39 |
| 9.3 Prinzipien der Prüfung von Anforderungen | 39 |
| 10 Reviews | 39 |
| 10.1 Reviews nach Objekten | 40 |
| 10.1.1 Management Review | 40 |
| 10.1.2 Technical Review | 40 |
| 10.2 Review Typen | 40 |
| 10.3 Reviews im Projektzyklus | 40 |
| 10.4 Dauer und Zeitpunkt | 40 |
| 10.5 Vorgehen | 41 |
| 10.6 Meilensteine | 41 |
| 11 Anforderungsüberprüfung in hybriden Projekten | 41 |
| 11.1 Requirements Board / Backlog Grooming | 41 |
| 12 Anforderungen in der Umgangssprache formulieren | 41 |
| 12.1 Grundlegende Regeln | 42 |
| 12.2 Entscheidungstabelle | 43 |
| 12.3 Satzschemata | 43 |
| 12.4 Arten von Aktivitäten | 43 |
| 13 Qualitätskriterien im RE | 44 |
| 14 Anforderungsarten | 44 |
| 15 Methoden des Projektmanagements | 44 |
| 15.1 Metamodell nach Jenny | 46 |
| 15.2 Ansätze der Produktherstellung | 46 |
| 15.3 Projektziele | 46 |
| 16 Projektcontrolling | 46 |
| 16.1 Controlling im PDCA-Modell | 47 |
| 16.2 Earned Value | 48 |

1 Einführung

Requirements Engineering beinhaltet **Requirement Management** und **Requirement Development**, das wiederum *Elicitation, Analysis, Specification* und *Valididation* beinhaltet.

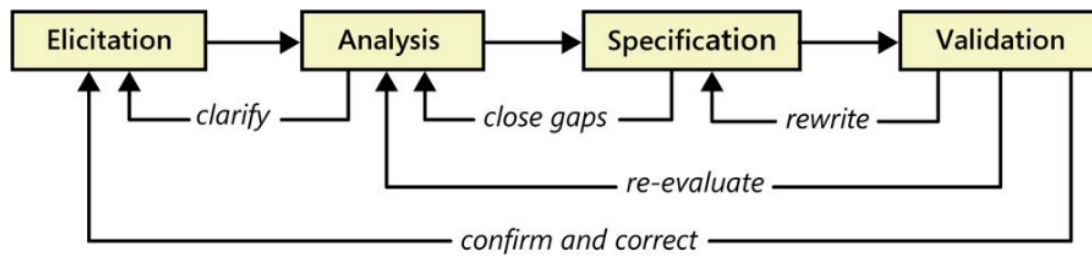


Abbildung 1: Requirements Development

1.1 Business Analyst

Der Business Analyst muss die Anforderungen verschiedenster Zielgruppen (→ *Stakeholder*) wie **Projekt-Sponsoren, Management, Entwicklung, Benutzer** und weiteren berücksichtigen und gegeneinander abwägen.

1.2 Setting

1.3 Ziele

Ein Ziel ist ein **angestrebter Zustand** oder eine **erwünschte Wirkung**. Ziele sollten bekannt sein, bevor man die Anforderungen ermittelt. Jede Anforderung muss auf ein Ziel zurückführbar sein.

Beispiele von Zielen:

Ziele der Geschäftsführung:

- Kosten senken
- Schnelle Abwicklung eines Projekts
- Kein zusätzliches Personal

Ziele Geschäftsstellenleiter:

- Mehr Servicepersonal
- Qualifizierte Mitarbeiter
- Gute Kundenkenntnis beim Berater

Ziele Mitarbeiter:

- Gute Ausbildung bei den Produktentwicklung
- Gute Bezahlung
- Gleichmässiger Arbeitsanfall

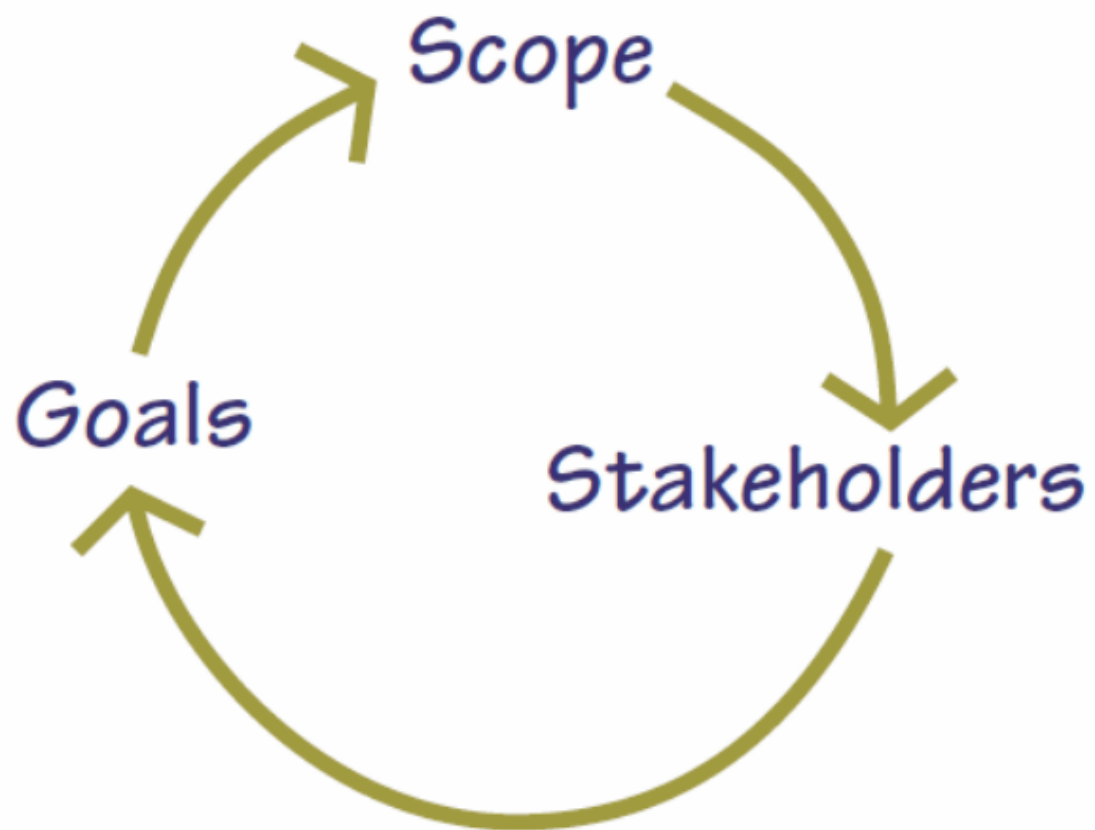


Abbildung 2: Setting

Ziele der Kunden:

- Schnelle Abwicklung
- Hochwertige Beratung
- Niedrige Preise

1.4 Zieldiagramm

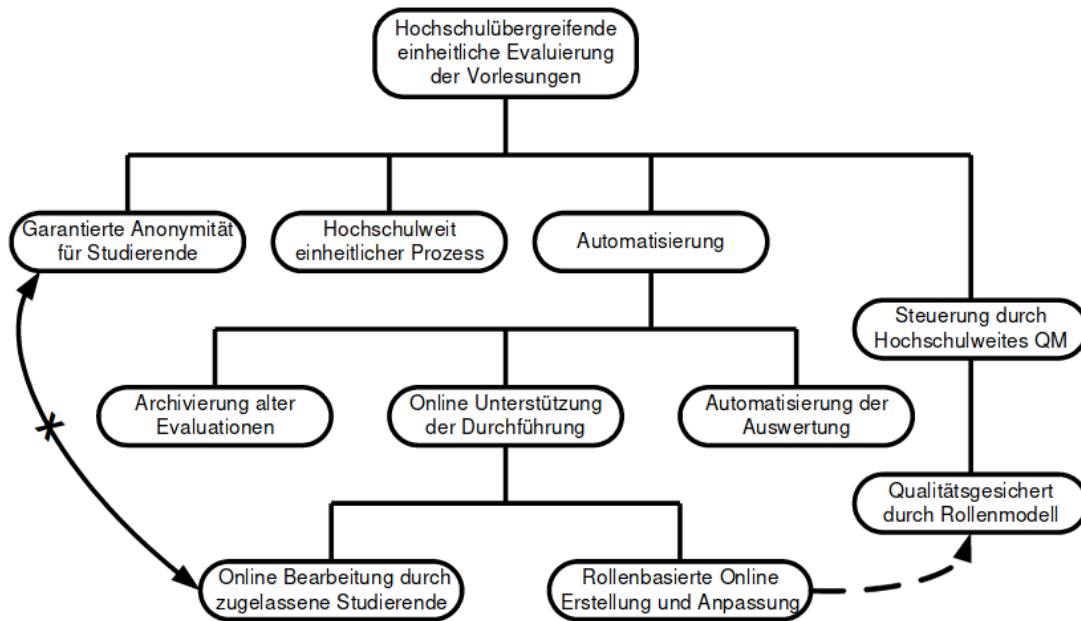


Abbildung 3: Zieldiagramm

2 Stakeholder

Stakeholder sind alle Personen, die direkt oder indirekt Einfluss auf die Anforderungen haben. Dies betrifft die Systementwicklung, den Einsatz und den Betrieb eines Produktes. Und umfasst auch Personen, die das System zwar nicht entwickelt haben aber es benutzen, betreiben oder schulen.

Stakeholder können in *interne* (Besitzer, Angestellte, Manager) und *externe* Stakeholder (Kunden, Kreditoren, Zulieferer, Gesellschaft, ...) eingeteilt werden.

2.1 Stakeholderanalyse

- Identifikation potentieller Stakeholder
- Sammeln von Informationen
- Strategische Einordnung
- Vorhersagen des Stakeholder-Verhaltens
- Massnahmen planen und umsetzen

Ziel ist es, die verschiedenen Interessen, Einflüsse und Wertvorstellungen zu ermitteln und festzulegen von welchen Gruppen und Individuen diese kommen.

2.2 Stakeholderklassen

2.2.1 Management

Sponsoren / Auftraggeber und Entscheider. Sorgen dafür, dass ein System die **Unternehmensziele** unterstützt und mit der **Unternehmensphilosophie** konform ist. Kommen i.d.R. erst im Genehmigungsverfahren mit dem Projektantrag im Kontakt.

Damit ein Projekt genehmigt wird müssen die Systemziele die Ziele des Managements unterstützen. Auch deshalb ist es wichtig, das Management als potentielle Stakeholder in ein Projekt einzubeziehen.

2.2.2 Anwender

Anwender liefern den Grossteil der **fachlichen Ziele**. Bei einer grossen Anzahl Benutzer können besonders erfahrene Benutzer als Benutzerrepräsentanten ausgewählt werden.

Benutzer können noch in direkte (= Benutzer, die aktiv mit dem System arbeiten) und indirekte Benutzer (= Personen, die vom System profitieren, z.B. Personen, die Belege aus dem System bekommen) unterteilt werden.

2.2.3 Wartungs- und Servicepersonal

Formulieren die wesentlichen Ziele für die Wartung und den Service des Systems. Dies beinhaltet beispielsweise **Anzeige von Fehlerzuständen**, Ansprüche an **Ausfallsicherheit**, **Wiederherstellbarkeit** und **Transport**.

2.2.4 Schulungs- und Trainingspersonal

Aspekte wie **Bedientbarkeit, Vermittelbarkeit, Dokumentation** und **Hilfesystem**.

2.2.5 Käufer

Benutzer und Käufer sind nicht immer identisch. **Lizenzkonzept, Service- und Vertragskonditionen, Preis**.

2.2.6 Marketing und Vertrieb

Interne Repräsentanten der externen Kunden, Ziel- und Anforderungslieferanten in der Produktentwicklung.

2.2.7 Entwickler

Technologiespezifische Ziele, eingesetzte Technologien. Dienen der Zukunftssicherung und der Motivation des Entwicklerteams.

2.2.8 Projekt- und Produktgegner

Potentielle Gegner, Konkurrenten berücksichtigen. Eventuell Überzeugungsarbeit oder Anpassung der Projektziele, um Widerstände zu vermeiden.

2.2.9 Produktbeseitiger

Personen, die mit der Beseitigung nach der Benutzung eines Produktes beauftragt sind. Bestimmungen betreffend **Umweltschutz** bei Hardware, **Deinstallationsprogramme** bei Software.

2.2.10 Sicherheitsbeauftragte

Anforderungen betreffend absichtlichen oder unabsichtlichen **Fehlverhalten** anderer Stakeholder und zweckwidrige Verwendung des Systems.

2.2.11 Betriebsrat

Spielt vor allem in grösseren Unternehmen eine wichtige Rolle bei der Einführung neuer Systeme und sollte deshalb früh in das Projekt integriert werden.

2.2.12 Personen aus anderen Kulturkreisen

Rahmenbedingungen wie **Darstellung** der Informationen, Verwendung von Symbolen und Begriffen.

2.2.13 Gesetzgeber

Rechtliche Rahmenbedingungen wie **Gesetze, Vorschriften und Verordnungen**. Beispielsweise das Datenschutzgesetz.

2.2.14 Standardisierungsgremien

Externe oder firmeninterne Standards, Richtlinien, Sicherheitsstandards, Corporate Identity.

2.2.15 Öffentliche Meinung

Meinungsführer wie marktdominierende Konkurrenz (Microsoft) oder öffentliche Meinung. Ziel: optimales Projektmarketing angepasst an unterschiedliche Gegebenheiten (USA, Asien, Europa).

2.2.16 Prüfer und Auditoren

Prüfen das System auf Konformität mit Richtlinien / Zielen.

2.2.17 Technische Experten

Personen mit technischem Fachwissen (Chemie, Werkstofftechnologie...). Überprüfen, ob ein Produkt überhaupt umgesetzt werden kann. **Ziele** und **Grenzen** aufgrund **technologischer Restriktionen** (z.B. elektromagnetische oder thermodynamische Anforderungen).

2.2.18 Produzenten des Produkts

Fertigungsanforderungen, End-of-line-Programmierung, Lieferbedingungen, Ausfallraten.

2.2.19 Produktdesigner

Anforderungen an das **Aussehen, Form** und **Ausbau** aus ästhetischen und technischen Gründen.

2.2.20 Prozessoptimierung und Arbeitsergonomie

Optimierung der **Benutzerschnittstelle, ergonomische Erfordernisse** (Lesbarkeit, Darstellung), **ökonomische Arbeitsabläufe** (z.B. Menüführung).

2.2.21 Experten für das Systemumfeld

Anforderungen und Rahmenbedingungen für die Verwendung und Einbettung des Systems in dem dafür vorgesehenen Systemumfeld.

2.2.22 Produktlinienverantwortliche

Einbettung in Produktfamilie, einheitliches Benutzerinterface, Farben, Logos, Wiederverwendung von Systemteilen.

2.2.23 Forschung und Entwicklung

Innovationen für neue Produkte.

2.2.24 Controllingabteilung

Entscheidungsträger in internen Finanzabteilungen, **finanzielle Rahmenbedingungen**, Anforderungen an die **Preisgestaltung**.

2.3 Notation von Stakeholdern in einem Projekt

Zur Dokumentation von Stakeholdern sind folgende Informationen notwendig:

- Name
- Funktion / Rolle
- Personen- und Kontaktdaten
- zeitliche und räumliche Verfügbarkeit
- Relevanz
- Wissensgebiet und -umfang
- Ziele und Interessen
- ...

Stakeholder können beispielsweise tabellarisch erfasst werden:

| Rolle | Beschreibung | Konkrete Vertreter | Verfügbarkeit | Wissensgebiet | Begründung |
|-------|--------------|--------------------|---------------|---------------|------------|
| ... | ... | ... | ... | ... | ... |

2.4 Stakeholder-Map

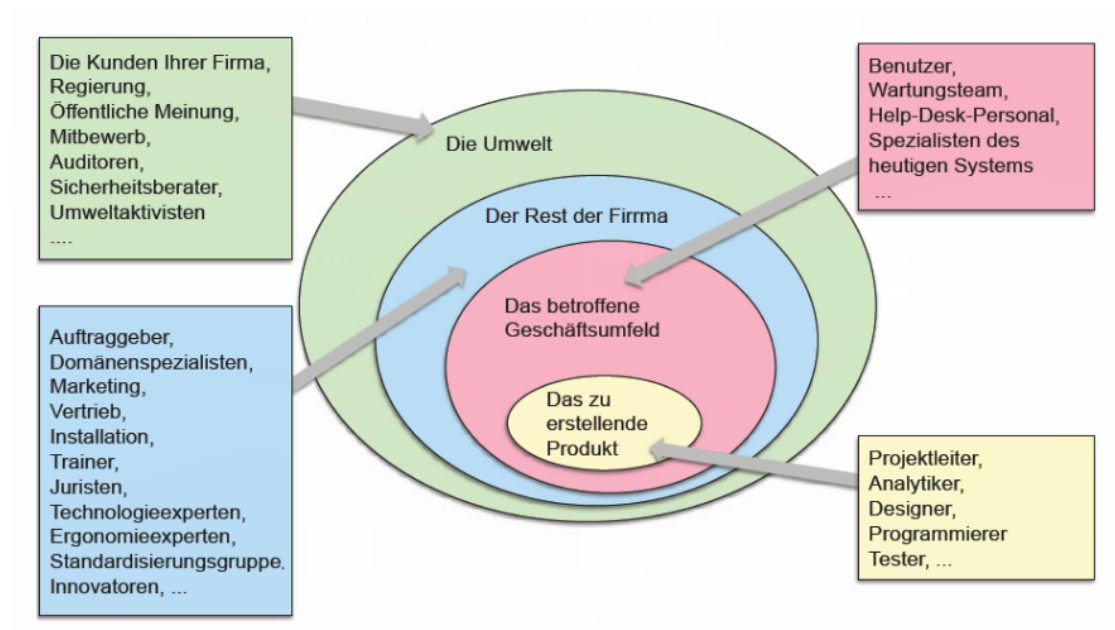


Abbildung 4: Stakeholder-Map

3 Kontextdiagramm

Im Kontextdiagramm wird das System und der Systemrelevante Kontext abgebildet. Ziel ist es klar zu definieren, was zum System gehört und was nicht, sowie Schnittstellen darzustellen.

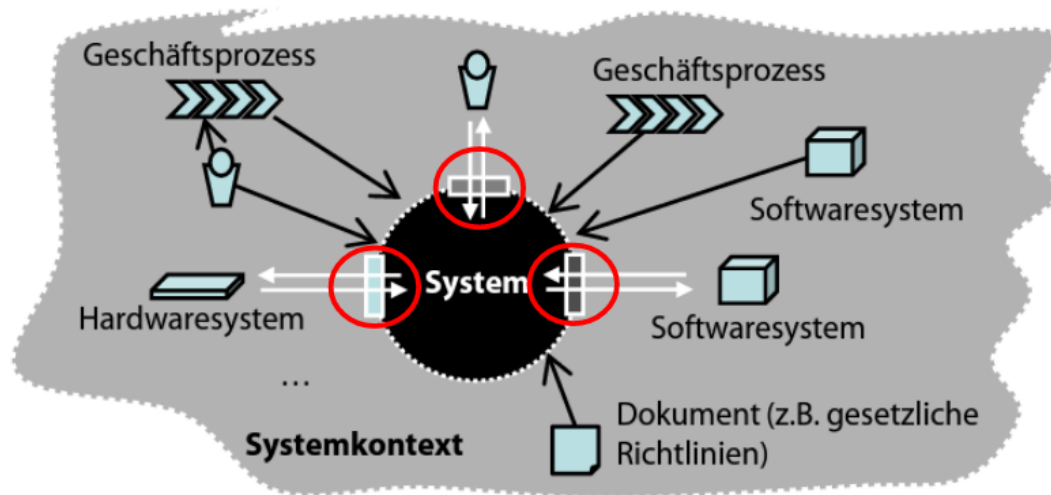


Abbildung 5: Kontext-Diagramm

3.1 Aspekte des Systemkontexts

3.2 Typische Fehler

4 Anforderungen ermitteln

- Bekanntes Wissen
- Unbekanntes Wissen
- Wissensbeschaffung & Erhebungstechniken
- Wissensdokumentation
- Validierung / Überprüfung von Wissen

Anforderungen können von verschiedenen Quellen kommen und sollten umfassend ermittelt, dokumentiert und bewertet werden.

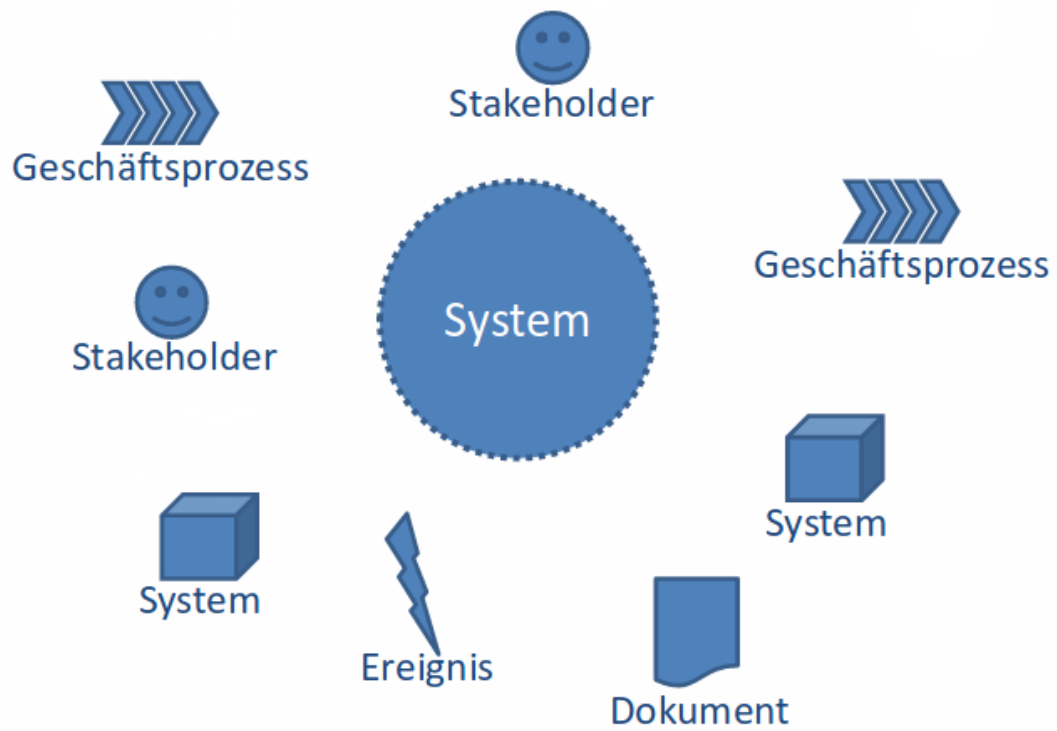


Abbildung 6: Aspekte des Systemkontexts

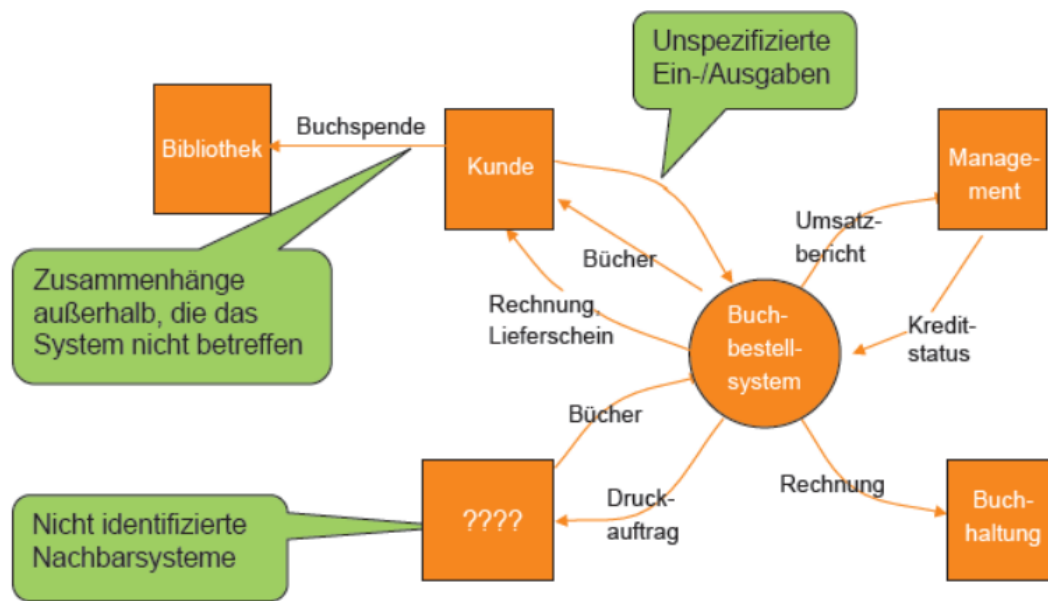


Abbildung 7: Typische Fehler im Kontextdiagramm

4.1 Anforderungsquellen

4.1.1 Stakeholder

Stakeholderstimmen einordnen:

- *Business Requirement*
- *Functional Requirement*
- *Quality Attribute*
- *Data Definition*
- *Business Rule*
- *Use Case*
- etc.

4.1.2 Die 7 Produktdimensionen

- **User:** Wer benutzt das System, wer erhält Output vom Produkt?
- **Interface:** Welche Interfaces senden Daten, Nachrichten zum oder vom Produkt?
- **Action:** Welche Aktionen erreichen die Ziele der Benutzer?








| | | | | | | |
|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |
| User | Interface | Action | Data | Control | Environment | Quality Attribute |
| Users interact with the product | The product connects to users, systems, and devices | The product provides capabilities for users | The product includes a repository of data and useful information | The product enforces constraints | The product conforms to physical properties and technology platforms | The product has certain properties that qualify its operation and development |

Abbildung 8: The 7 Product Dimensions

- **Data:** Was für Daten brauchen Benutzer vom Produkt?
- **Control:** Welche Policies, Regulationen und Regeln muss das Produkt einhalten?
- **Environment:** Wo wird das Produkt eingesetzt? Welche Software und Hardware wird verwendet für die Entwicklung und Benutzung des Produkts?
- **Quality Attribute:** Service Levels für Verfügbarkeit, Benutzung etc.

4.1.3 Kano-Modell

Das *Kano-Modell der Kundenzufriedenheit* teilt Kundenanforderungen in drei Kategorien ein:

- **Basisfaktoren:** Grundlegend und selbstverständlich (implizite Erwartungen), fallen erst bei Nichterfüllung auf. Nichterfüllung führt zu Unzufriedenheit:
 - *Bücher ausleihen in einer Bibliothek*
- **Leistungsfaktoren:** Explizite Erwartungen, beseitigen Unzufriedenheit oder schaffen Zufriedenheit, je nach Ausmass der Erfüllung
 - *Online-Reservation von Büchern, Benachrichtigung bei Ablauf der Ausleihfrist*
- **Begeisterungsfaktoren:** Nutzenstiftende Faktoren, die vom Kunden nicht erwartet werden, Abgrenzung von Konkurrenz
 - *Empfehlungen für ähnliche Bücher, Liefersdienst gekoppelt mit Pizza-Service*

Das Kano-Modell sagt, dass Leistungsfaktoren mit der Zeit zu Basisfaktoren werden und Begeisterungsfaktoren zu Leistungsfaktoren.

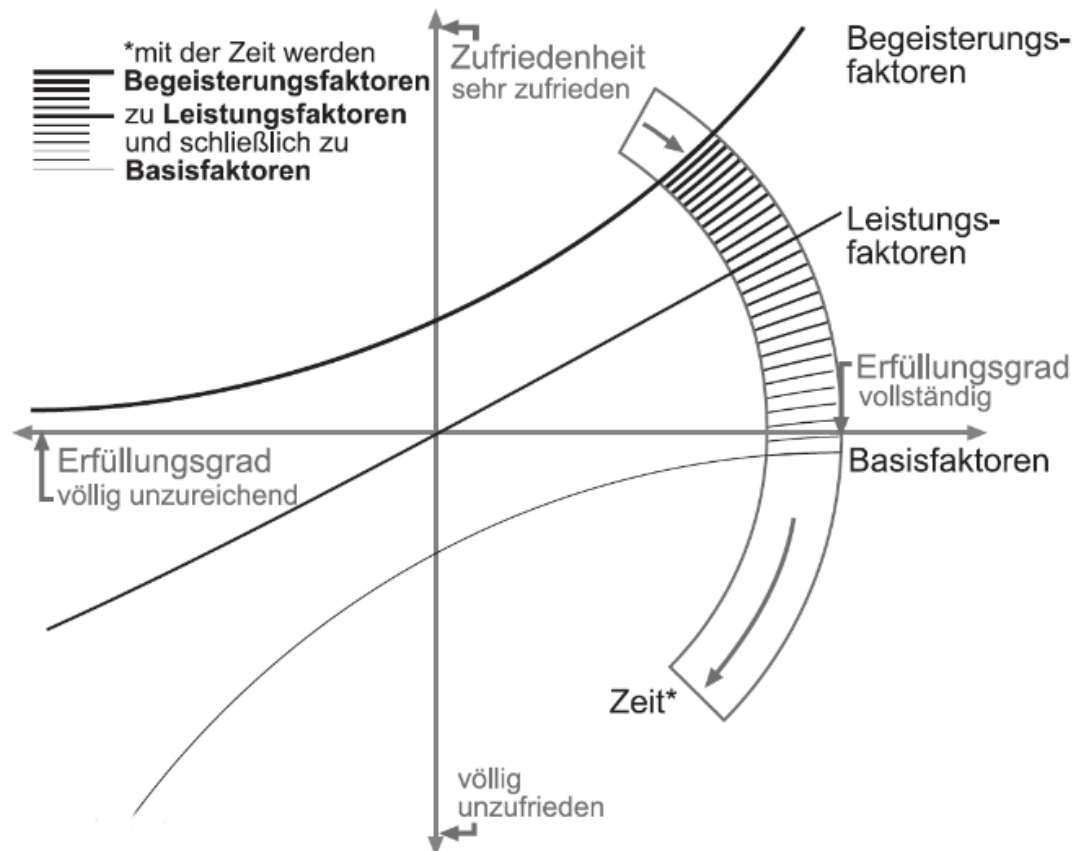


Abbildung 9: Kano Model

4.1.4 Interview

- **Geschlossene Fragen:** Bewertung auf Skala, Multiple-Choice.
- **Offene W-Fragen:**
 - Aufgabenmerkmale Was / Wie: *Aufgabenverrichtung*
 - * Woran: *Aufgaben-Objekt*
 - Merkmale der Aufgabenerfüllung
 - * Wer: *Aufgabenträger*
 - * Womit: *Sachmittel*
 - * Wann, wie lange: *Zeit*
 - * Wo, woher, wohin: *Ort*
 - * Wie oft, wieviel: *Menge*

Krokus-Regel:

- **Kurze Fragen:**
 - Interviewer kann Steuerungsfunktion besser gerechtfertigen
 - Höhere Chance auf kurze Antworten
 - Befragter wird nicht überfordert
- **Redundante Fragen vermeiden:**
 - Geringere zeitliche Belastung für den Befragten
 - Erleichtert Dokumentation
 - Bessere Transparenz, roter Faden ersichtlich
- **Offene Fragen stellen:**
 - Wecken Auskunftsbereitschaft
 - Geben Befragtem Zeit zum Nachdenken
 - Vermeidung von Manipulation und Spekulation
 - Fördern neue Gesichtspunkte
 - Engen Befragten nicht ein
- **Konkrete Fragen stellen:**
 - Straffen
 - Bremsen Vielredner
 - Fördern Verständnis
- **Unterfragen und Kettenfragen vermeiden:**
 - Jede Frage wird beantwortet
 - Befragter wird weniger verunsichert
- **Suggestive Fragen vermeiden:**
 - Befragter sagt, was er denkt und nicht, was erwartet wird
 - Weniger Widersprüche

5 Anforderungen modellieren

Modelle können *konkrete* (Modelleisenbahn) oder *abstrakte* (Atom-Modell, Finanzierungsmodell) Abbildungen realer Objekte sein. Der Verwendungszweck bestimmt, welches Modell geeignet ist.

Modelle sind entweder *Struktur*-, *Funktions*- oder *Verhaltens*- Analogie für ein Original.

- **Struktur:** Relevante Objekte und ihre Beziehungen werden modelliert. (→ Datenmodell)
- **Funktion:** Abläufe der Manipulation von Informationen werden modelliert. (→ Aktivitätsdiagramm, Datenflussdiagramm)
- **Verhalten:** Erlaubte Systemzustände und Zustandsänderungen durch Ereignisse werden modelliert (→ Zustandsdiagramm)

6 Diagramme in RE

Verschiedene Diagramme zur Darstellung verschiedener Perspektiven:

- **Struktur:** Klassendiagramm
- **Systemfunktionen:** USe Case Diagramm
- **Funktionsdetails:** Aktivitätsdiagramm
- **Verhalten:** Zustandsdiagramm, Sequenzdiagramm
- **System / Kontext:** Kontextdiagramm

6.1 Use Case Diagramm

Use Case Diagramme müssen immer auch eine Use Case Beschreibung enthalten, beispielsweise als Liste oder Tabelle.

6.2 Aktivitätsdiagramm

Modellelemente:

Beispiel:

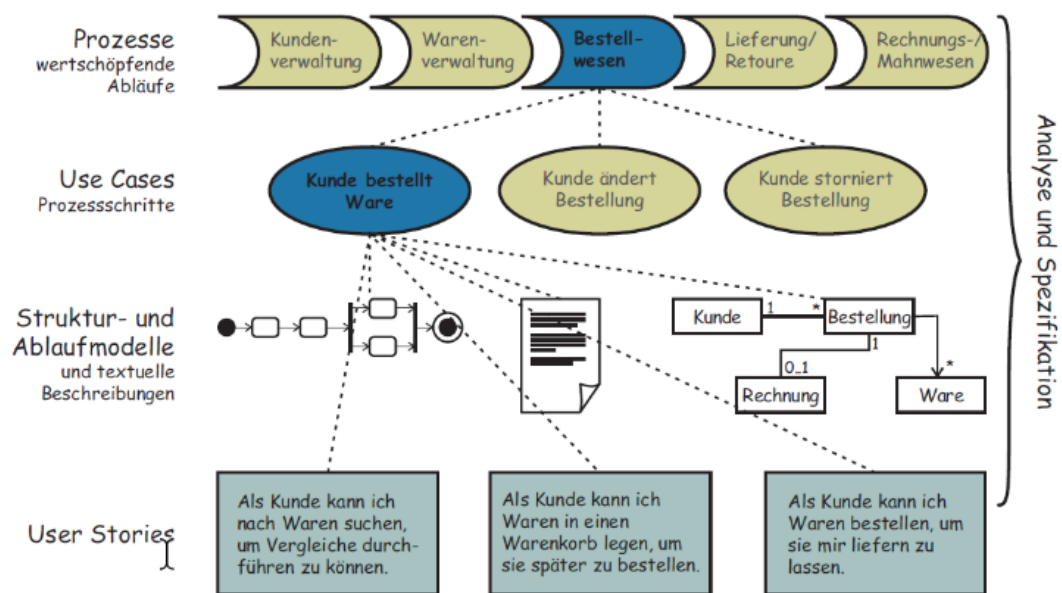


Abbildung 10: Überblick Modelle

6.3 Klassendiagramm

Modellelemente:

Beispiel:

6.4 CRUD-Matrix

Gegenüberstellung von UCs (Funktionen) und Klassen (Datenstrukturen).

6.5 Zustandsdiagramm

Modellelemente:

Beispiel:

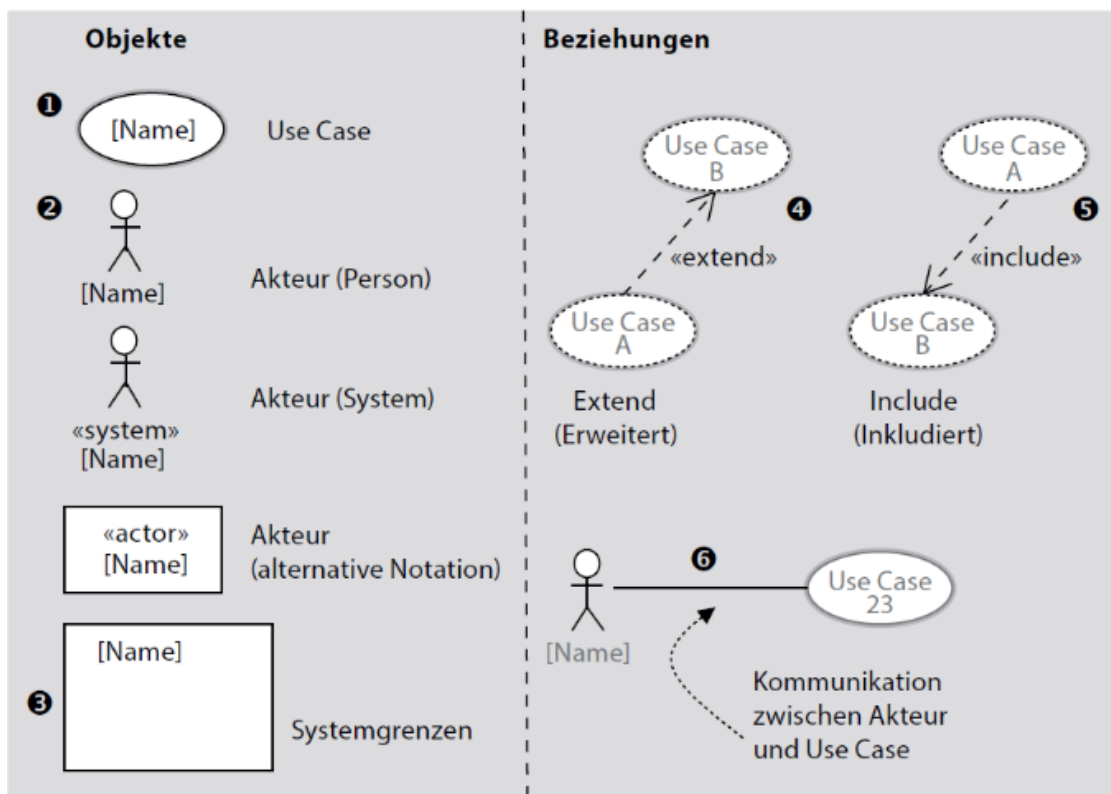


Abbildung 11: Use Case Diagramm

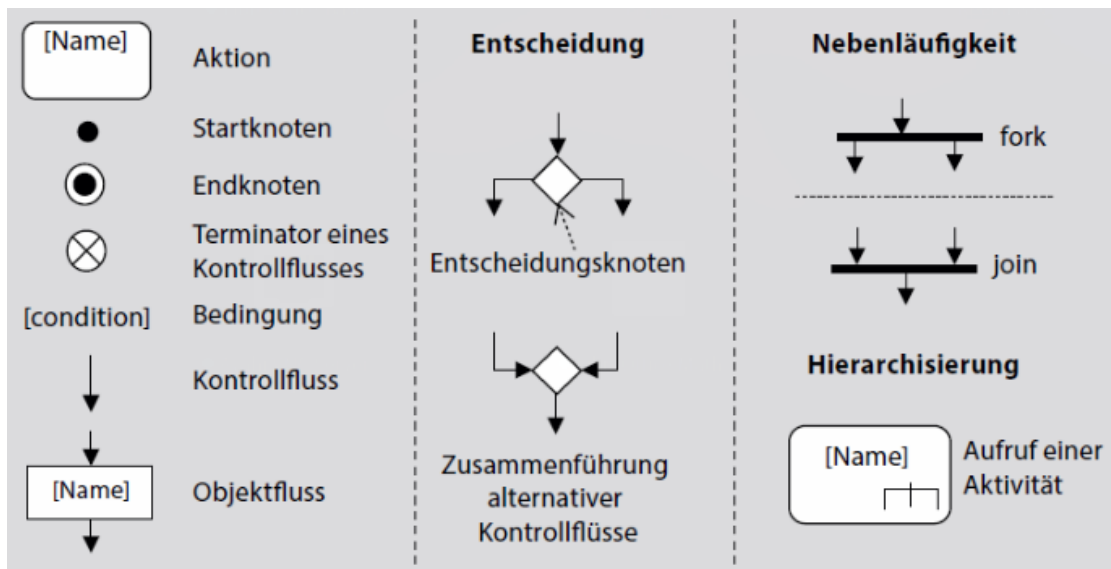


Abbildung 12: Aktivitätsdiagramm

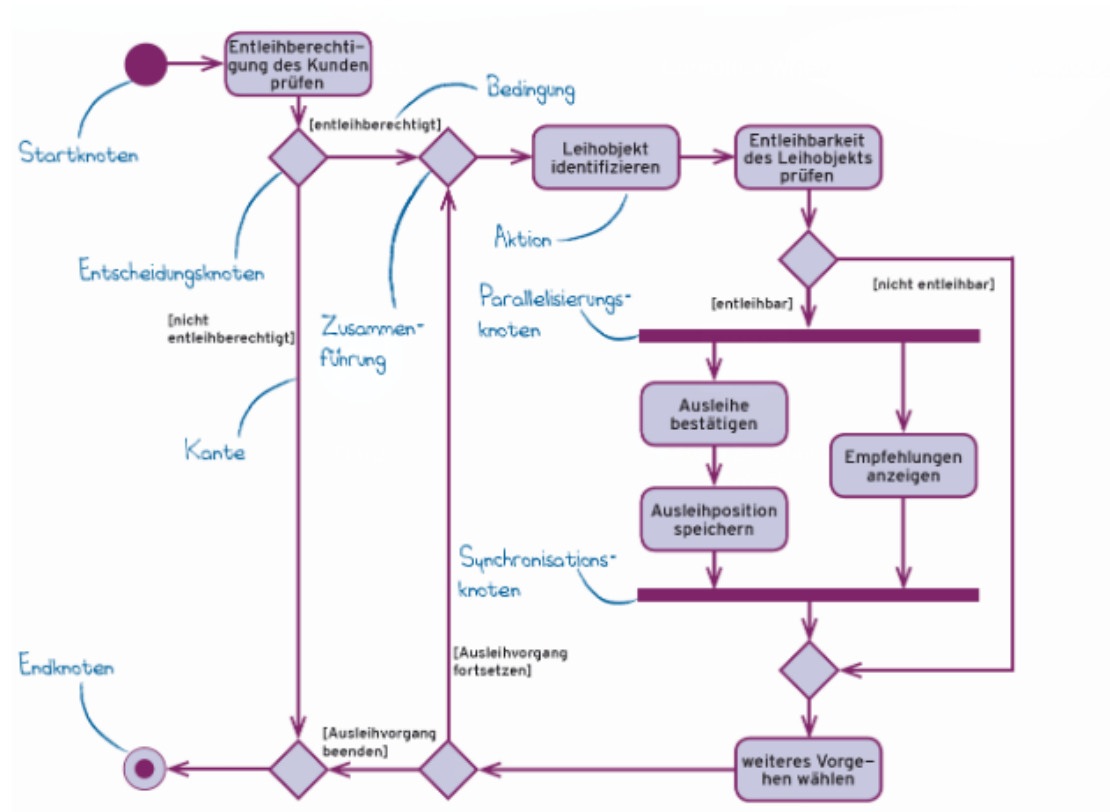


Abbildung 13: Aktivitätsdiagramm 2

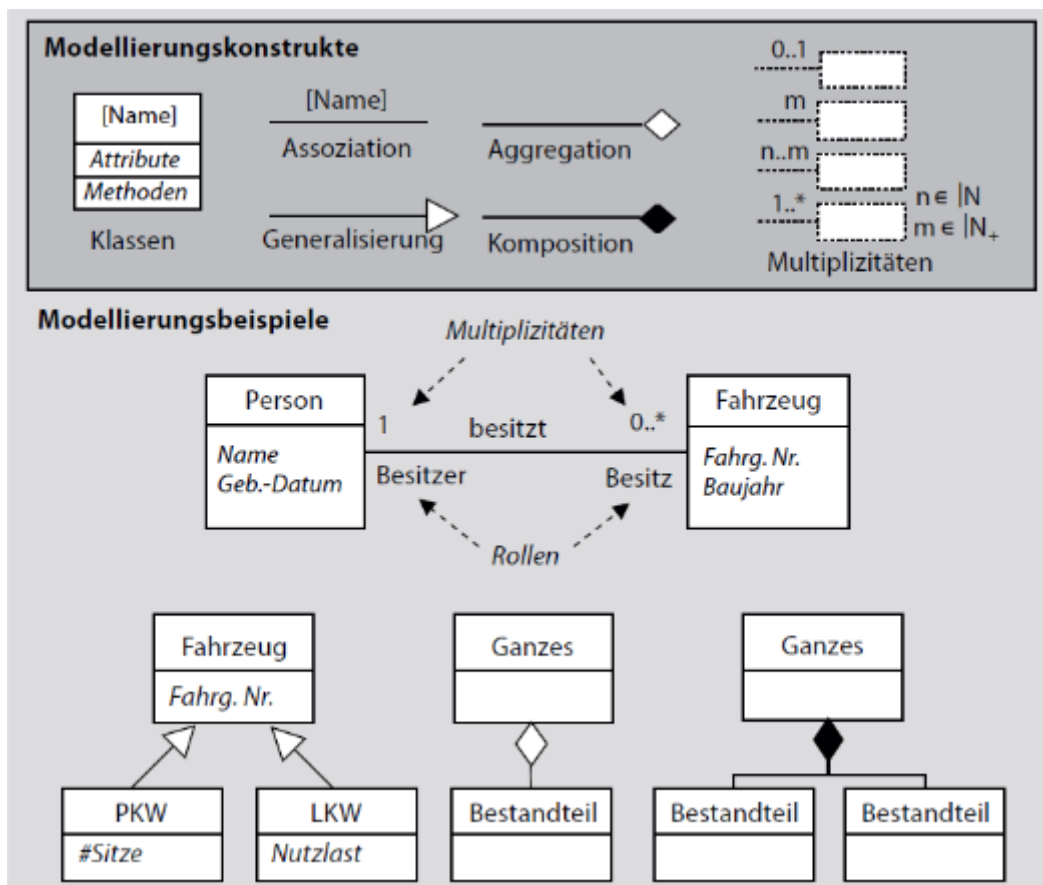


Abbildung 14: Klassendiagramm

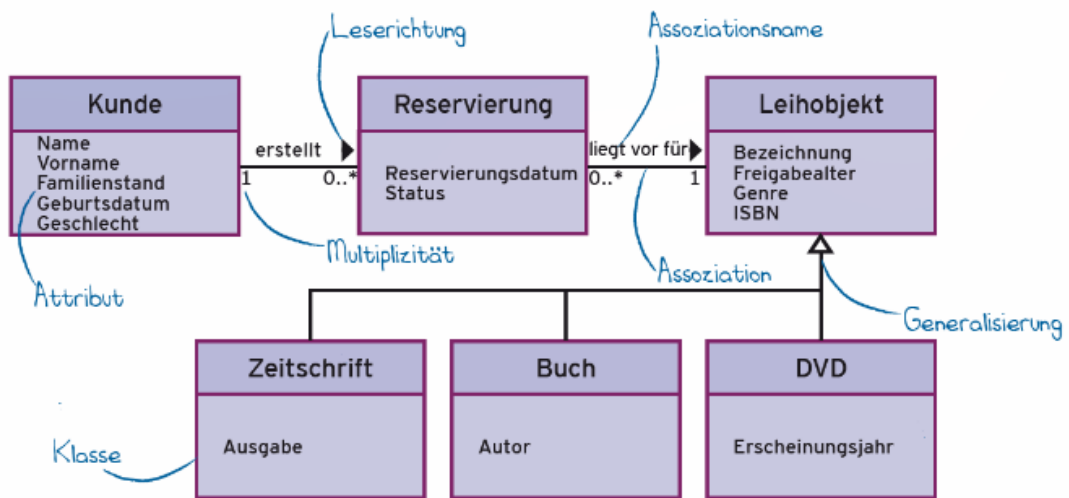


Abbildung 15: Klassendiagramm 2

| Entity-Klassen | | | | | | | |
|----------------|-----|-----|-------|-----|-------|---|-----|
| Use-Cases | | | | | | | |
| | C,R | | R | R,U | R,U | | |
| | | R | | | C,U,D | U | |
| | U | | C,U,D | R,U | | | |
| | | C,U | | | R,U | U | C,U |
| | R,U | D | R | | R | | R |

Abbildung 16: CRUD-Diagramm

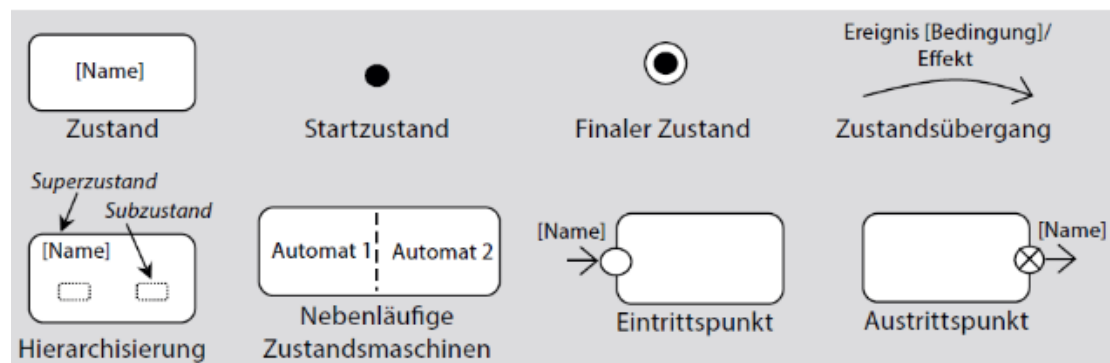


Abbildung 17: Zustandsdiagramm

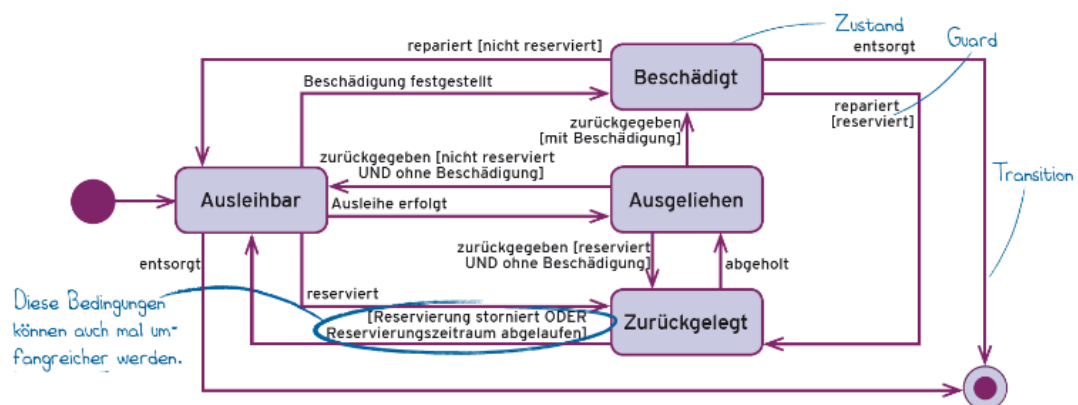


Abbildung 18: Zustandsdiagramm2

7 Agiles Requirement Engineering

7.1 Überblick

- **Requirements:** Anforderungen eines Stakeholders; jede Eigenschaft, die ein System besitzen soll.
- **Requirement-Spezifikation:** Jede Repräsentation eines oder mehrerer Requirements, unabhängig von Form und Granularität.
- Ebenen zur Strukturierung und Einordnung von Requirements:
 - High-Level Sichtweise: *Überblick über das Vorhaben*
 - Strukturierungsebene: *Artefakte in Zusammenhang bringen*
 - Detailebene: *Feingranulate Inhalte*

Überblick über Requirement-Artefakte:

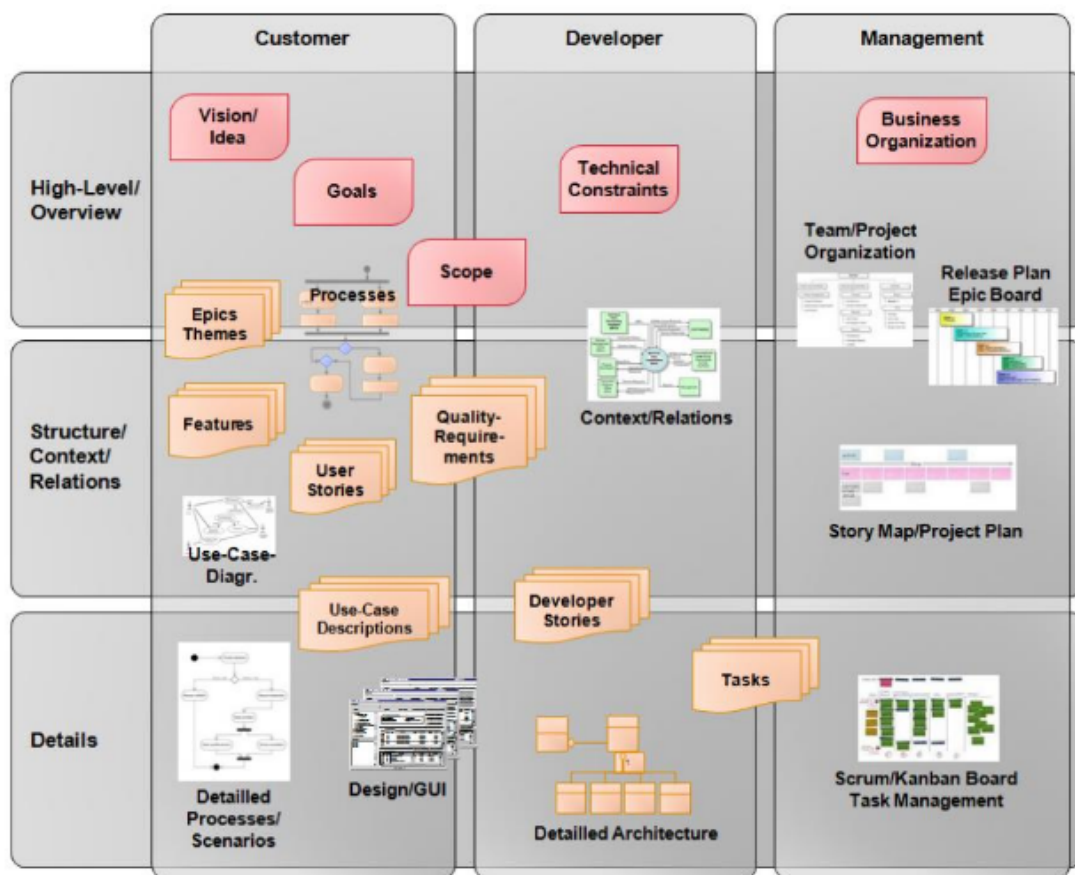


Abbildung 19: Überblick über Requirement-Artefakte

7.2 Backlog Items

Backlog-Items erfüllen folgende Funktionen:

- Strukturierung und Einordnung von Requirements
- Release-Planung
- Speicher für detaillierte Anforderungen
- Aufwandschätzung (bei Zuteilung zu einem Sprint)

Bevor Backlog-Items verwendet werden können, ist eine Verfeinerung (*Refinement*) notwendig.

- Details hinzufügen
- Schätzungen erstellen
- Reihenfolge festlegen

Das *Refinement* ist ein kontinuierlicher Prozess und wird vom *Product Owner* zusammen mit dem Entwicklungsteam durchgeführt.

7.3 Repräsentation von Anforderungen in agilen Projekten

7.3.1 Features

Features sind eine zusammenfassende Menge von Anforderungen. Features beschreiben *funktionale* und *nichtfunktionale* Anforderungen.

- Unterstützen Dialog unter den Stakeholders
- Verbessern die Anwendungskennntnisse der Entwickler
- Fördern Kreativität und Innovation
- Regen zu weiterem Nachforschen an

Features werden oft auch in der Produktwerbung eingesetzt.

ZU beachten bei Features:

- Entwicklungsteam versteht Features wirklich
- Gewichtung nach Bedeutung für das System
- Priorisierung für die Realisierung
- Hierarchisch strukturiert aufgeführt

7.3.2 Scrum User Stories

Kurzer, narrativer Text, der eine Interaktion zwischen einem Benutzer und dem System beschreibt. Diese Beschreibung dient als Grundlage für die Kommunikation und Ausgangspunkt für die Umsetzung von Produktfunktionalitäten.

User Stories werden im *Scrum Produkt Backlog* verwaltet und dienen als **Planungsinstrument** für die Sprints.

Das Diagramm zeigt ein Formular für ein User Story. Es besteht aus einem blauen Rahmen, der ein weißes Feld umschließt. Oben links befindet sich ein kleiner Kreis. Rechts daneben sind die Felder 'Story ID:' und 'Story Title:' angeordnet. Darunter befindet sich ein großer Textfeld 'User Story:' mit der Platzhalterstruktur: 'As a: <role>', 'I want: <some goal>' und 'So that: <some reason>'. Rechts neben diesem Feld befinden sich zwei weitere Felder: 'Importance:' und 'Estimate:', jeweils mit einem leeren Kasten. Unter dem 'User Story:' Feld befindet sich ein Feld 'Acceptance Criteria' mit der Platzhalterstruktur 'And I know I am done when:'. Rechts neben diesem Feld befindet sich ein Feld 'Type:' mit einer Liste von vier Optionen, jeweils mit einem leeren Kasten: 'Search', 'Workflow', 'Manage Data', 'Payment' und 'Report/ View'.

Abbildung 20: User Story

- **Name:**
 - prägnante Benennung der User Story
- **Beschreibung:**
 - Wer möchte was?
 - Durch welche Funktionalität wird der Nutzen realisiert?
- **Akzeptanzkriterien:**
 - Wie kann die Umsetzung den Anforderungen entsprechend festgestellt werden?
 - Was soll getestet werden?
- **Ergänzende Beschreibungen:**
 - Typischerweise als Link auf Use Case oder andere Anforderungsform

7.3.2.1 Formaler Aufbau

Name: *[Aussagekräftiger Titel]*

Beschreibung:

Als *[Rolle / Akteur]* **möchte ich** *[Funktionsbeschreibung]*, **damit** *[Begründung]*.

Akzeptanzkriterien:

[Messbare Kriterien für korrekte Implementation]

7.3.2.2 Beispiel:

Name: CSV-Export meiner Daten

Beschreibung:

Als Benutzer möchte ich _die Daten meiner Konten per CSV-File herunterladen können, um damit ausserhalb des Programms weitere Auswertungen und Berichte zu erstellen.

Akzeptanzkriterien:

- Ich kann die Daten für jedes meiner Konten herunterladen
- Es sind alle Werte in der CSV-Datei enthalten
- Ich kann die Daten für alle Berichtszeiträume ausgeben lassen

7.3.2.3 INVEST-Prinzip Anforderungen an User Stories:

- **Independent:** Eine Story ist unabhängig von anderen Stories.
- **Negotiable:** Kunder und Entwickler erarbeiten und präzisieren die Stories gemeinsam. Kunde beschreibt die Funktionalität grob, die Entwickler arbeiten die Details aus.
- **Valuable:** Die Stories sollten einen erkennbaren Mehrwert liefern.
- **Estimable:** Eine Story muss so überschaubar sein, dass die Entwickler die Umsetzung schätzen können.
- **Small:** Mindestens einen halben, maximal zehn Personentage Aufwand.
- **Testable:** Test sind der Massstab für den erfolgreichen Abschluss. Deshalb müssen Stories testbar sein.

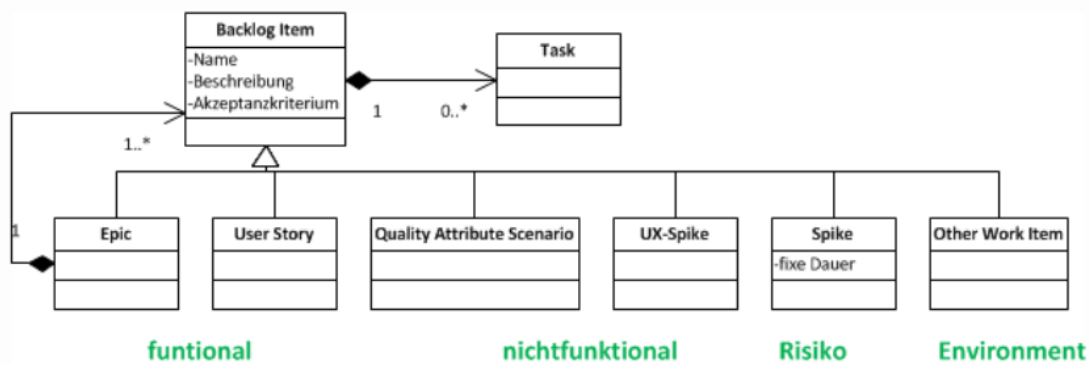


Abbildung 21: Backlog Items

7.3.3 Scrum Backlog Items

Backlog-Items in Scrum sind Strukturelemente für die Planung und haben unterschiedliche Ausprägungen:

Das Team verpflichtet sich, ausgewählte Backlog Items in einem Sprint vollständig umzusetzen.

7.3.3.1 Task Tasks sind Teilaufgaben innerhalb eines Backlog Items und dienen zur Aufgabenteilung und Aufwandschätzung.

7.3.3.2 Epic Epics sind grobgranulare Anforderungen, die zu komplex oder umfangreich sind, um sie in einem Sprint umzusetzen. Epics werden in andere Backlog Items zerlegt (*Story Splitting*), eine Aktivität innerhalb des RE in Scrum, Backlog Grooming)

Epics werden unterteilt in:

- Business Epics
 - Fachliche Anforderungen aus der Geschäftswelt
 - Organisations- oder prozessübergreifende Eigenschaften
- Architectural Epis
 - Weitreichende technische Vorgaben
 - Betreffen hauptsächlich Architekturfragen

Epics sind Anforderungen auf hoher Ebene und sollten deshalb kontinuierlich strukturiert und beschrieben werden.

Beispiel (attributiertes Epic):

| Attribut | Beschreibung |
|---------------------------|---|
| <i>Name</i> | Arbeitszeiterfassung auf mobilen Endgeräten |
| <i>Ziel</i> | Vorsprung gegenüber dem Wettbewerb aufbauen, mehr Flexibilität beim Einsatz der Zeiterfassung |
| <i>Beschreibung</i> | Die neue Softwarelösung für die Arbeitszeiterfassung wird auf allen Typen von Endgeräten zur Verfügung stehen und wir werden damit technologisch führend sein |
| <i>Art</i> | Architectural Epic |
| <i>Provider</i> | Herr Huber, Leiter Marketing |
| <i>Product Owner</i> | Herr Schmidt |
| <i>Priorität</i> | Mittel |
| <i>Termin</i> | Herbst-Fachmesse |
| <i>Aufwand</i> | Noch nicht abschätzbar |
| <i>Anmerkungen / Todo</i> | Ermittlung Machbarkeit, Zielgruppenanalyse |

7.3.3.3 Quality Attribute Scenario Messbar und testbar formulierte, nichtfunktionale Anforderung. Besteht aus sechs Schritten:

- **Source:** z.B. Benutzer, Computer, System
- **Stimulus:** zu betrachtende Bedingung
- **Environment:** In welchem Zustand befindet sich das System
- **Artefact:** Welcher Teil des Systems wird beeinflusst
- **Response:** Systemantwort
- **Response Measure:** Messbares Attribut der Antwort (z.B. Zeit)

Beispiele:

- Response Time:
 - When customers click check-out button, order information appears in less than 2 seconds
 - validity check takes no more than 5 seconds
- Scalability:

- Access to product backlog must scale up to 10'000 concurrent users and still fulfill the Response Time

7.3.3.4 Spikes, UX-Spikes & Other Work Items

- **Spike:** Time-boxed Arbeitspaket zur Minimierung von Risiken, z.B. das Erstellen eines Prototyps.
- **UX-Spike:** Arbeitspaket zur Erarbeitung und Überprüfung der User Experience und Erarbeitung der Benutzeroberfläche anhand von:
 - * Interviews mit Nutzern
 - * Kontextanalysen
 - * Online-Befragungen
 - * Use Cases
 - * Paper-Prototyping
 - * Usability-Tests
 - * etc.
- **Other Work Item:** Arbeiten zum Aufbau und Pflege der Arbeitsumgebung

7.4 Story Splitting

Epics sind zu gross, um eine realistische Schätzung abzugeben. Erst nach dem Aufteilen in funktionale (*User Stories*) und nichtfunktionale (*Quality Attribute Scenarios*) Anforderungen ist eine realistische Sprintplanung möglich.

Beim Aufteilen von Epics in User Stories muss sichergestellt werden, dass jede neu erzeugte User Story weiterhin einen Kundennutzen darstellt.

7.5 Anforderungsmanagement in agilen Projekten

- Anforderungen erarbeiten:
 - Aufwand (Zeit und Kosten)
- Anforderungen zeitgerecht für die Entwicklung zur Verfügung stellen:
 - Produkt Backlog mit detaillierten, verifizierten Anforderungen
- Anforderungen ändern sich im Projektverlauf
 - ~ 1% / Monat

7.5.1 Anforderungsmanagement in Scrum

- Produkt Backlog
 - Speicher für alle Anforderungen (neu und geändert)
 - müssen für *Sprint Backlog* aufbereitet werden (*splitting, grooming*)
- Sprint Backlog
 - Aktuell umzusetzende, verifizierte, detaillierte Anforderungen
- Scrum Taskboard / Kanban-Tafel
 - Planungstool für SW-Entwicklung (Sprintplanung und Controlling)
 - Planung der Anforderungserarbeitung

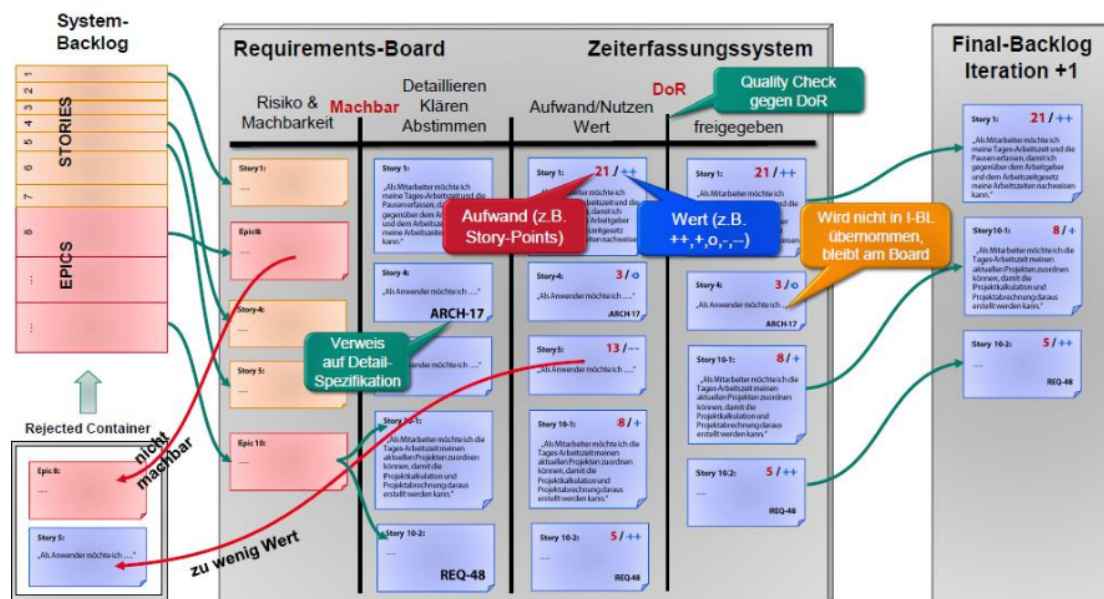


Abbildung 22: Requirement Board / Backlog Grooming

Tools für Requirements-Management

Aufgaben eines RM-Tools:

- Anforderungen versionieren
- Status der Anforderung dokumentieren
- Verlinkung von Anforderungen, um Rückverfolgbarkeit zu gewährleisten

Beispiel **ScrumDo**: Web-basiertes Tool zum einfachen Verwalten von User Stories: Produkt Backlog, Sprint- & Releaseplanung, Epics, Stories, Tasks etc.

8 User Story Mapping

Eine Story Map ist eine grafische, zweidimensionale Visualisierung des Produkt Backlogs. Jede Spalte ist ein Epic mit Überschrift und Stories, nach Priorität geordnet.

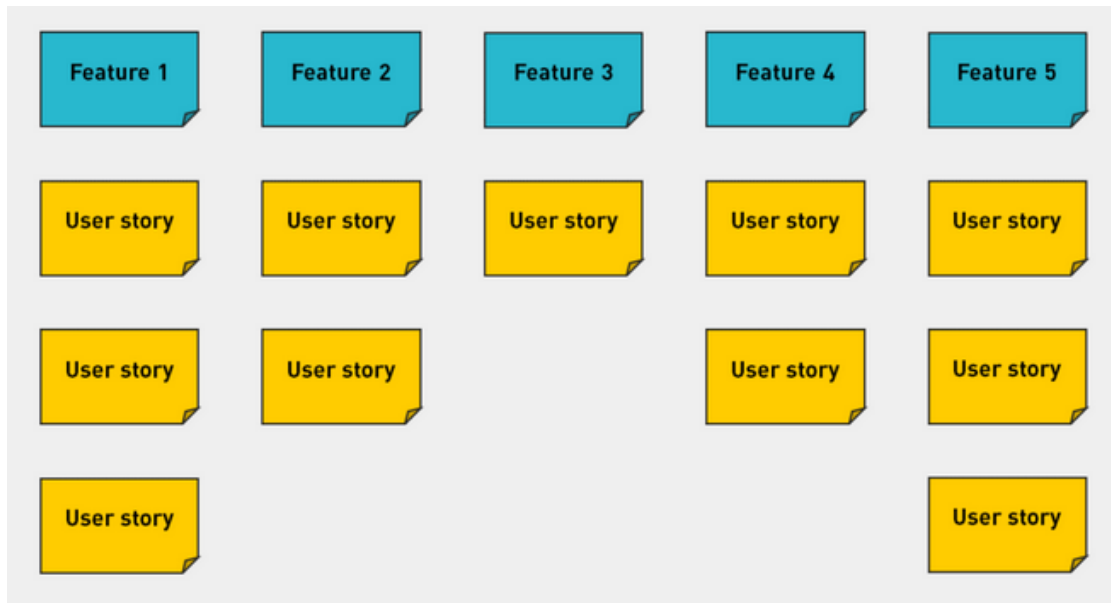


Abbildung 23: Story Map

Entwickeln der Story Map:

8.1 Beispiel TriHow

Entwicklung eines Workshop-Tools.

Kontextdiagramm:

Stakeholdermap:

UC-Diagramm:

Story Map:

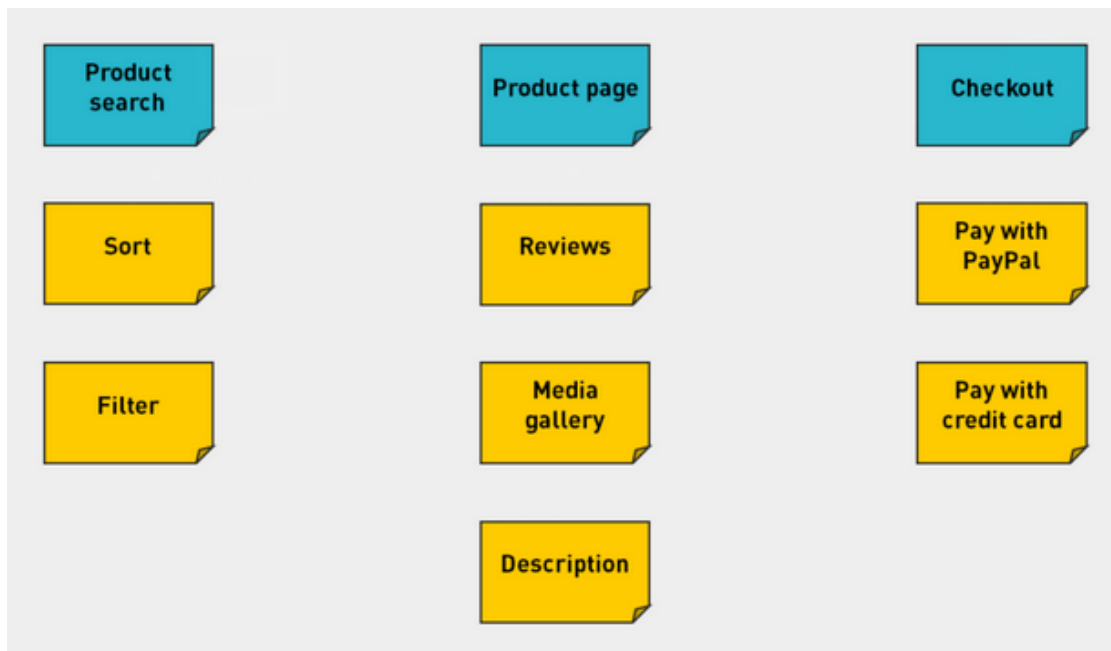


Abbildung 24: Story Map Step 1

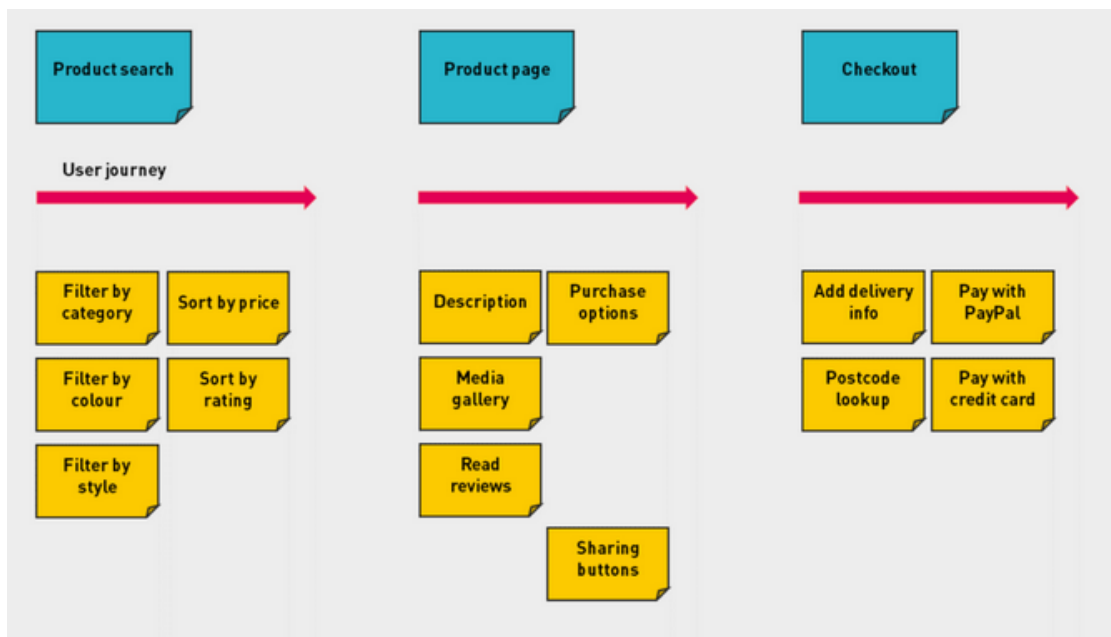


Abbildung 25: Story Map Step 2

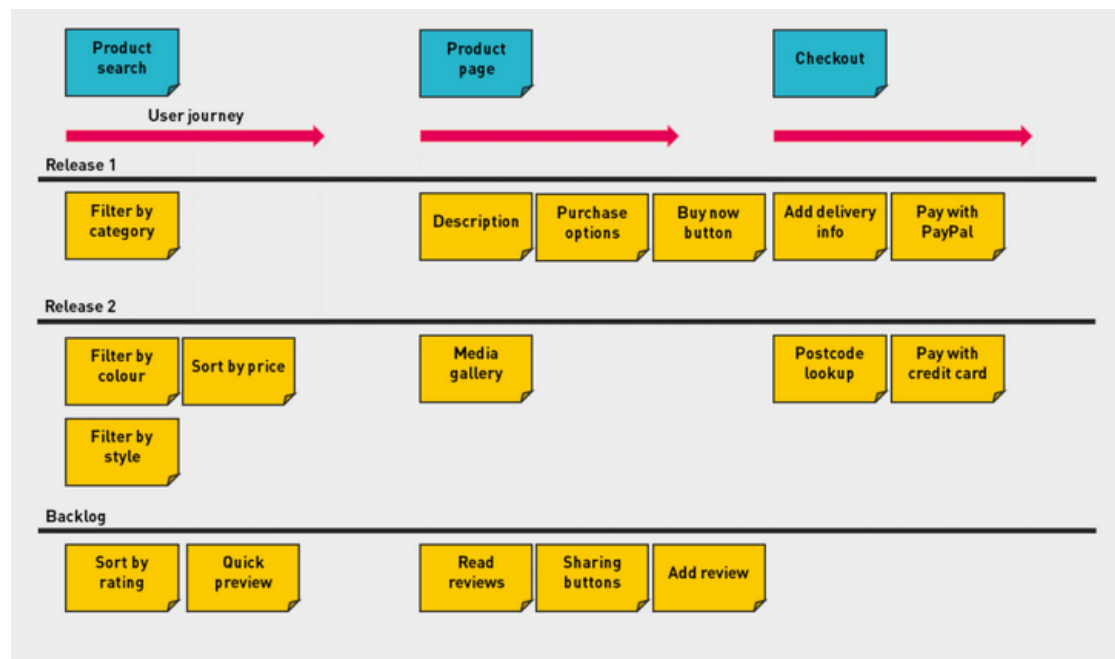


Abbildung 26: Story Map Step 3



Abbildung 27: Kontextdiagramm



Abbildung 28: Stakeholder Map

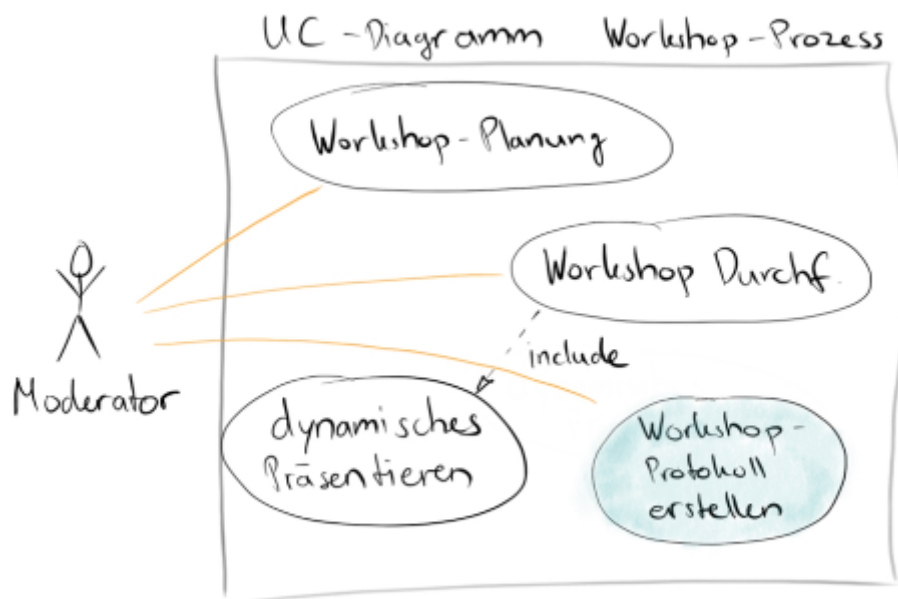


Abbildung 29: UC-Diagramm

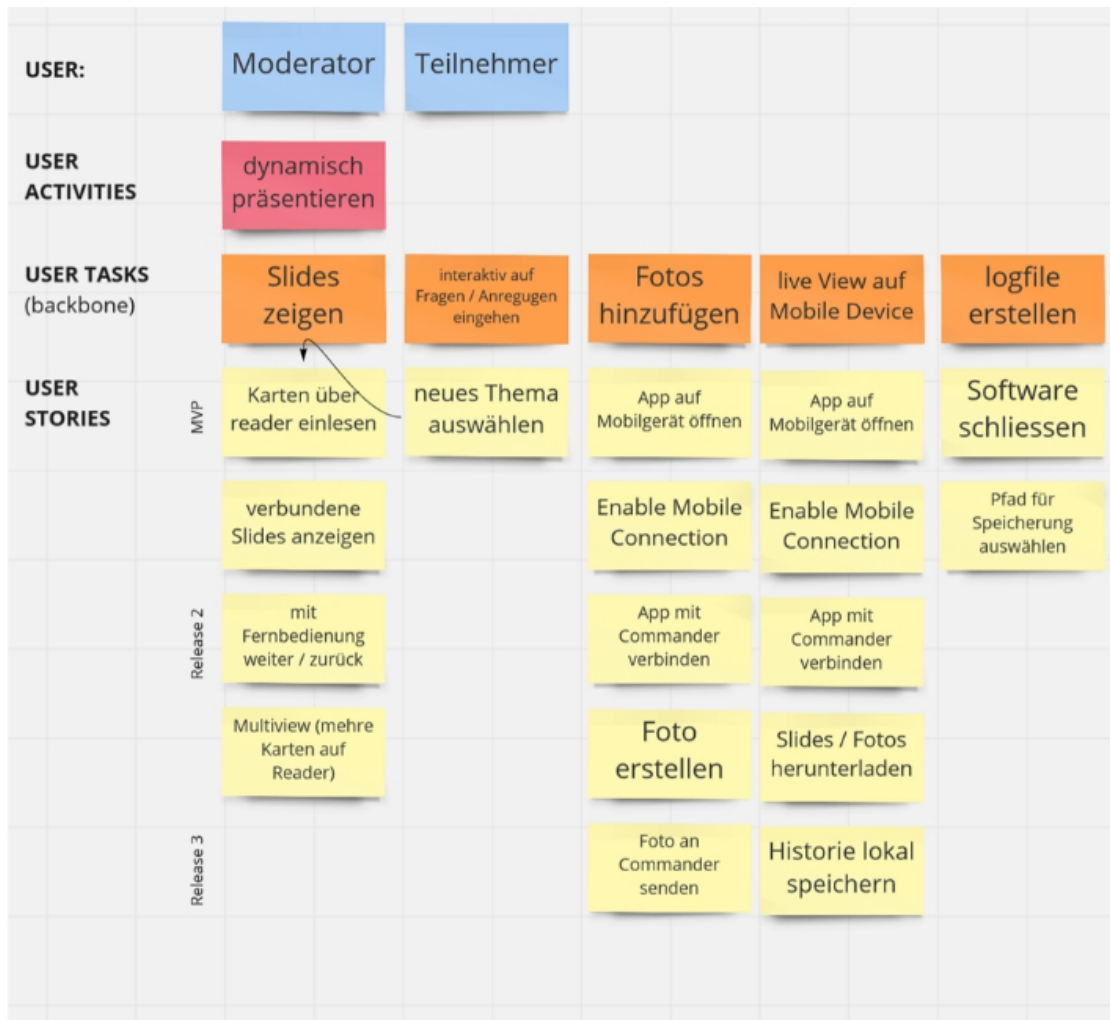


Abbildung 30: Story Map

9 Prüfen von Anforderungen

9.1 Motivation, Kriterien & Prinzipien

Warum Anforderungen überprüfen?

1. **Juristisch:** Entwicklungsauftrag auf Basis eines Anforderungsdokuments. Wenn dieses falsch oder unvollständig ist, kann das zu Streitigkeiten führen.
2. **Fehlerfortpflanzung:** Fehler in den Anforderungen pflanzen sich in der Umsetzung (Code, Architektur, Tests) fort und verursachen dort hohe Kosten.

Das Ziel ist es, Fehler früh im Entwicklungsprozess zu entdecken. Fehler in diesem Zusammenhang sind *mehrdeutige*, *widersprüchliche* oder *unvollständige* Anforderungen.

Anforderungsfreigabe: Anhand vorher festgelegter Prüf- und Abnahmekriterien wird entschieden, ob auf dieser Basis die Entwicklung angegangen wird.

9.2 Überprüfung von Anforderungen

Die Überprüfung von Anforderungen sollte analog zu den Zielen des Requirement Engineering die folgenden Hauptziele beachten:

- **Inhalt:** Wurden alle relevanten Anforderungen ermittelt und im erforderlichen Detaillierungsgrad erfasst?
- **Dokumentation:** Wurden die Anforderungen gemäss der festgesetzten Dokumentations- und Spezifikationsvorschriften dokumentiert?
- **Abgestimmtheit:** Stimmen alle Stakeholder mit den dokumentierten Anforderungen überein und sind alle bekannten Konflikte gelöst?

9.2.1 Prüfkriterien "Inhalt"

- Vollständigkeit: Alle notwendigen Informationen für jede Anforderung
- Verfolgbarkeit: Alle relevanten Anforderungsquellen
- Korrektheit / Adäquatheit: Bedürfnisse der Stakeholder
- Keine Widersprüche bei den Anforderungen
- Keine vorzeitigen Entwurfsentscheidungen, Lösungen nicht vorwegnehmen
- Überprüfbarkeit: Abnahme und Prüfkriterien
- Notwendigkeit: Jede Anforderung trägt zur Zielerfüllung bei

9.2.2 Prüfkriterien “Dokumentation”

- Konformität zum Dokumentationsformat und zur Dokumentenstruktur: Schablonen, Modelliersprache
- Verständlichkeit: Begriffe erklärt, Glossar
- Eindeutigkeit: Sprachlich eindeutig
- Konformität mit Dokumentationsregel: Syntax der Modelliersprache usw.

9.2.3 Prüfkriterien “Abgestimmtheit”

Weil Stakeholder im Verlauf des RE-Prozesses neues Wissen über das geplante System erwerben, können sich auch bestimmte Anforderungen ändern-

- Abstimmung: Jede Anforderung mit allen relevanten Stakeholdern abstimmen
- Abstimmung nach Änderung: Erneute Zustimmung nach Änderung
- Konflikt: Alle bekannten Konflikte auflösen

9.3 Prinzipien der Prüfung von Anforderungen

1. Beteiligung der richtigen Stakeholder
2. Trennung von Fehlersuche und Fehlerkorrektur
3. Prüfung aus unterschiedlichen Sichten
4. Geeigneter Wechsel der Dokumentationsform
5. Konstruktion von Artefakten
6. Wiederholte Prüfung

10 Reviews

Ein **Review** ist eine öffentliche, verbale, zwischenmenschliche Betrachtung eines Artefakts. (=> *Product Quality*)

Ein **Audit** ist eine unpersönliche Verifikation von Nachweisen des Prozesses wie beispielsweise eine Dokumentation (=> *Process Quality*)

Ein **Akzeptanztest** ist ein Black-Box-Test bei der Produktabnahme, in Scrum als fälschlicherweise “*Sprint Review*” bezeichnet. (=> *Validation*)

10.1 Reviews nach Objekten

10.1.1 Management Review

Betrifft **Prozesse**, z.B. Review des Projektmanagement-Plans.

10.1.2 Technical Review

Betrifft das Produkt, z.B. Review der Systemspezifikation.

10.2 Review Typen

- Inspektion
- Team Review
- Walkthrough
- Pair Programming
- Peer Deskcheck
- Passaround
- Ad Hoc Review

10.3 Reviews im Projektzyklus

- Customer Requirements => Feasability Review
- Detailed Requirement => Requirements Review
- System Design => Prelim. Design Review, Critical Design Review
- Detailed Design => Source Code Review, Product Release Review

10.4 Dauer und Zeitpunkt

Eine Review-Stizung sollte nicht nicht länger als ein bis maximal zwei Stunden dauern. Beim klassischen Projektverlauf sind Reviews eng an den Projektzyklus gekoppelt, und müssen dann angesetzt werden, wenn die zu überprüfenden Artefakte vorliegen.

Reviews bringen mehr, je früher man den Fehler entdeckt (besser bereits die Anforderungen und Konzepte reviewen als den Code und Detail-Design).

10.5 Vorgehen

- Erfolg hängt von der persönlichen Dynamik ab, flexibel sein wer wessen Arbeit reviewed.
- Nicht wertend sein
- Fehler nicht an die grosse Glocke hängen, sondern lokal lösen§
- Artefakte einige Tage vorher zur Verfügung stellen
- Entdeckte Fehler und Probleme protokollieren

10.6 Meilensteine

Ein geplanter Zeitpunkt im Projektverlauf an dem vorher festgelegte Zwischenergebnisse vorliegen, die es erlauben den Projektfortschritt zu messen.

Ein Meilenstein ist erreicht, wenn die geforderten Artefakte vorliegen und die Überprüfung (Reviews, Tests) erfolgreich war.

11 Anforderungsüberprüfung in hybriden Projekten

Im Gegensatz zum klassischen Modell werden die Anforderungen im Produkt Backlog nur grob erfasst und erst im Rahmen der Sprintplanung präzisiert. Es gibt deshalb keinen einzelnen Zeitpunkt, an dem eine generelle Anforderungsüberprüfung möglich ist.

In Scrum unterstützen diese drei Institutionen die Anforderungsüberprüfung:

- **Backlog-Grooming:** Priorisierung der Anforderungen, gegenseitige Abhängigkeit
- **Retrospektive:** Qualität der Anforderungen
- **Sprint-Abnahme:** Laufende Ergänzung und neue Abstimmung

11.1 Requirements Board / Backlog Grooming

12 Anforderungen in der Umgangssprache formulieren

- **Transformationsprozesse:** persönliches Wissen und Wahrnehmung sowie das Formulieren dieses Wissens in Sprache führen zur Wahrnehmungs- und Darstellungstransformation

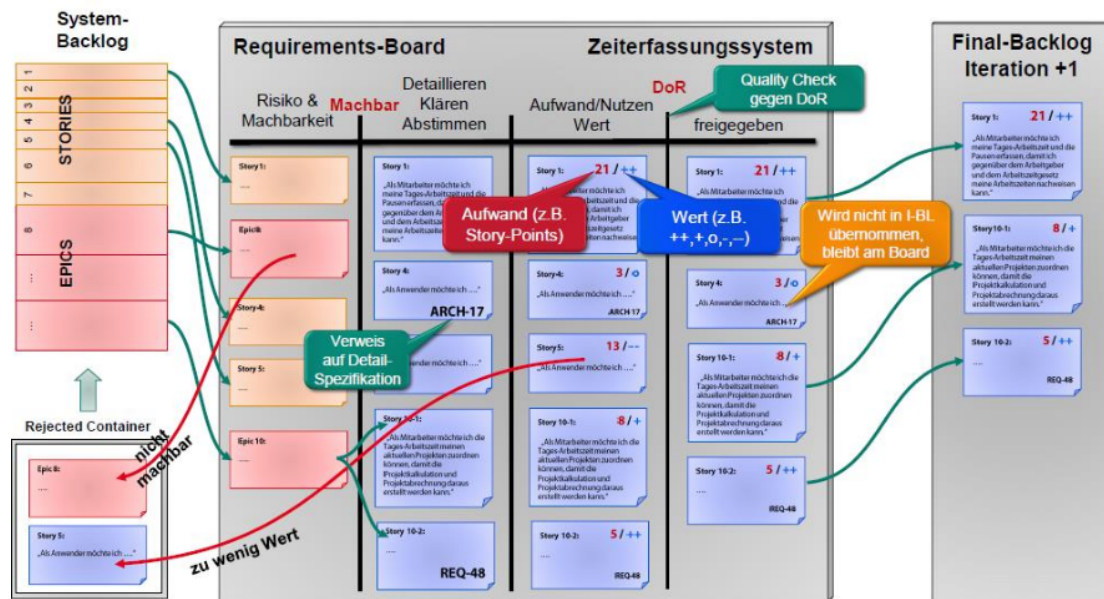


Abbildung 31: Requirements Board

- **Sprachlich Effekte:** Sprache ist mehrdeutig, am Entwicklungsprozess sind viele unterschiedliche Menschen mit beteiligt (Wissen, Kultur, soziale Prä-gung)
- **unterschiedliche Interpretation**

12.1 Grundlegende Regeln

- Anforderungen in ganzen Sätzen
- Formulierungen im Aktiv
- Begriffe konsistent verwenden, keine Synonyme & Homonyme
- Glossar erstellen für verwendete Begriffe
- Prozesse durch Vollverben formulieren
- Keine Nominalisierungen (*Daten archivieren* statt *Archivierung*)
- Spezifische Substantive (*Bibliothekar* statt *Benutzer*)
- Keine Universalquantoren (*die gespeicherten Registrierungsdaten* statt *alle Benutzerdaten*)
- Vollständig spezifizierte Bedingungen (Wenn ein *falls etwas*, dann auch ein *falls nicht*).
- Vollständig spezifizierte Prozesswörter (*Nicht zugelassene Kunden werden nicht angezeigt* was, wannn wie und wem wird in diesem Fall etwas angezeigt?)

12.2 Entscheidungstabelle

| Tabellenbezeichnung | R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 |
|----------------------------|----|----|----|----|----|----|----|----|
| Bedingungen | | | | | | | | |
| Lieferfähig | j | j | j | j | n | n | n | n |
| Angaben vollständig | j | j | n | n | j | j | n | n |
| Bonität in Ordnung | j | n | j | n | j | n | j | n |
| Aktionen | | | | | | | | |
| Lieferung mit Rechnung | x | | x | | | | | |
| Lieferung als Nachnahme | | x | | x | | | | |
| Angaben vervollständigen | | | x | x | | | | |
| Mitteilen: nicht lieferbar | | | | | x | x | x | x |

Abbildung 32: Entscheidungstabelle

12.3 Satzschablonen

Satzschablonen verwenden zur Fehlerminimierung:

12.4 Arten von Aktivitäten

1. **Selbständige Systemaktivität:** Das Bibliotheksystem (BS) muss dem Bibliothekar (B) spezifische Kundendaten anzeigen
2. **Benutzerinteraktion:** Das BS muss dem B die Möglichkeit bieten, Kundendaten einzugeben
2. **Schnittstellenanforderung:** Das BS muss fähig sein, Ausleihdaten aus einer anderen Bibliothek zu empfangen

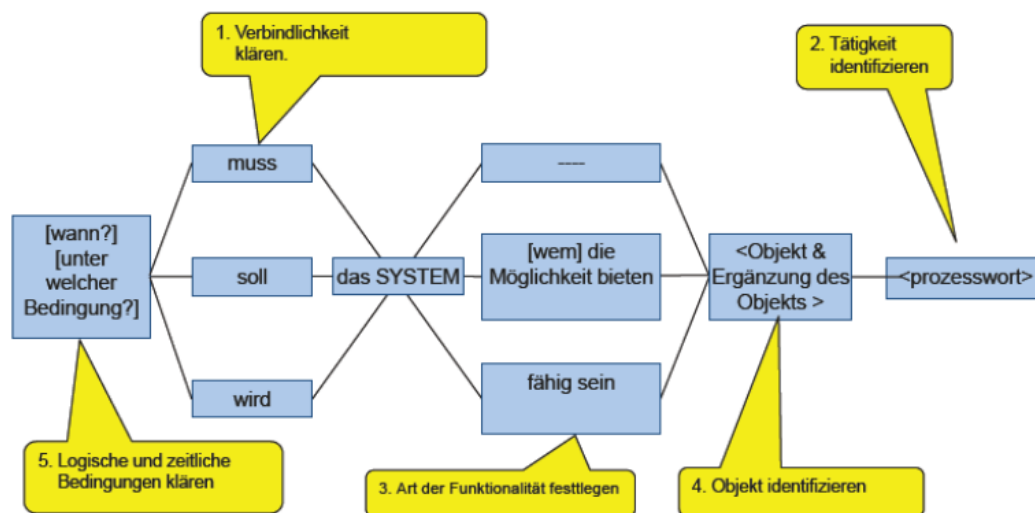


Abbildung 33: Satzschablone

13 Qualitätskriterien im RE

Ziel: Qualität im RE sicherstellen, qualitative / nichtfunktionale Anforderungen spezifizieren.

Validierung: Nachweis der Zweckmässigkeit von Anforderungen.

Entwickle ich das richtige System?

Verifikation: Nachweis von Korrektheit

Entwickle ich das System richtig?

Mangel: Fehlende Information, Anforderungslücke

Fehler: Falsche oder inkonsistente Information

14 Anforderungsarten

15 Methoden des Projektmanagements

Vorteile einer Projektmethode:

- Hilft Termine, Kosten und Qualität einzuhalten
- Abwicklungseffizienz

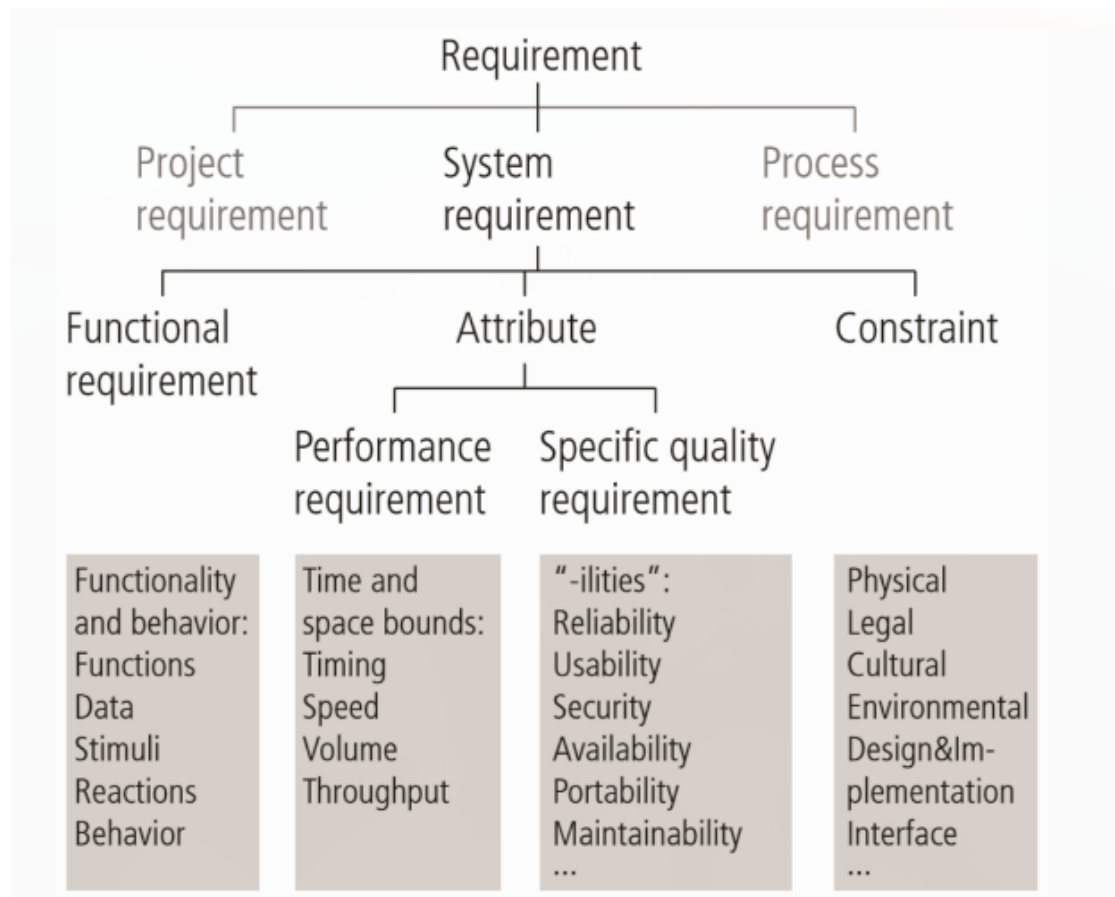


Abbildung 34: Anforderungen

- Unterstützt die Führung und Durchführung
- Standardisierte Werkzeuge
- Kommunikation innerhalb des Teams und mit Stakeholdern
- Wiederverwendbarkeit von Know-How, Best Practices
- Koordination, Priorisierung und Abstimmung von Projekten
- Abhängigkeiten zu anderen Firmen / Personen reduzieren

15.1 Metamodell nach Jenny

// TODO: * Jenny-Modell aus PMB einfügen * Zusammenfassen Projektmethoden:
https://elearning.hslu.ch/ilias/goto.php?target=file_3707827_download&client_id=hslu

15.2 Ansätze der Produktherstellung

- **Konstruktivistisch / sequentiell:** das vollständige Produkt wird zu 100% freigeben am Ende des Entwicklungsprozesses (*alles 100%*)
- **Inkrementell:** Einzelne, selbständige Teile des Produkts werden vollständig fertiggestellt und freigegeben und können unterschiedlich weit fortgeschritten sein (*Teil 1: 100%, Teil 3: 60%*)
- **Evolutionär:** Ganzes Produkt wird auf tiefem Funktions- oder Qualitätslevel fertiggestellt und freigegeben. (*Alles 30%*) Anschliessend Weiterentwicklung (*alles 60%, 80%, 100%, 120%...*)

15.3 Projektziele

SMART:

- Specific
- Measurable
- Achievable
- Relevant
- Timely

16 Projektcontrolling

Controlling: **Kontrollieren & Steuern**, Regeln anhand von Abweichungsanalyse, SOLL/IST-Vergeichen, Datenerhebung etc.

- Zeit

- Budget
- Zielerreichung

Das Projektcontrolling dient zur Überwachung des Ist-Zustandes eines Projekts.

16.1 Controlling im PDCA-Modell

- Plan
- Do
- Act
- Check

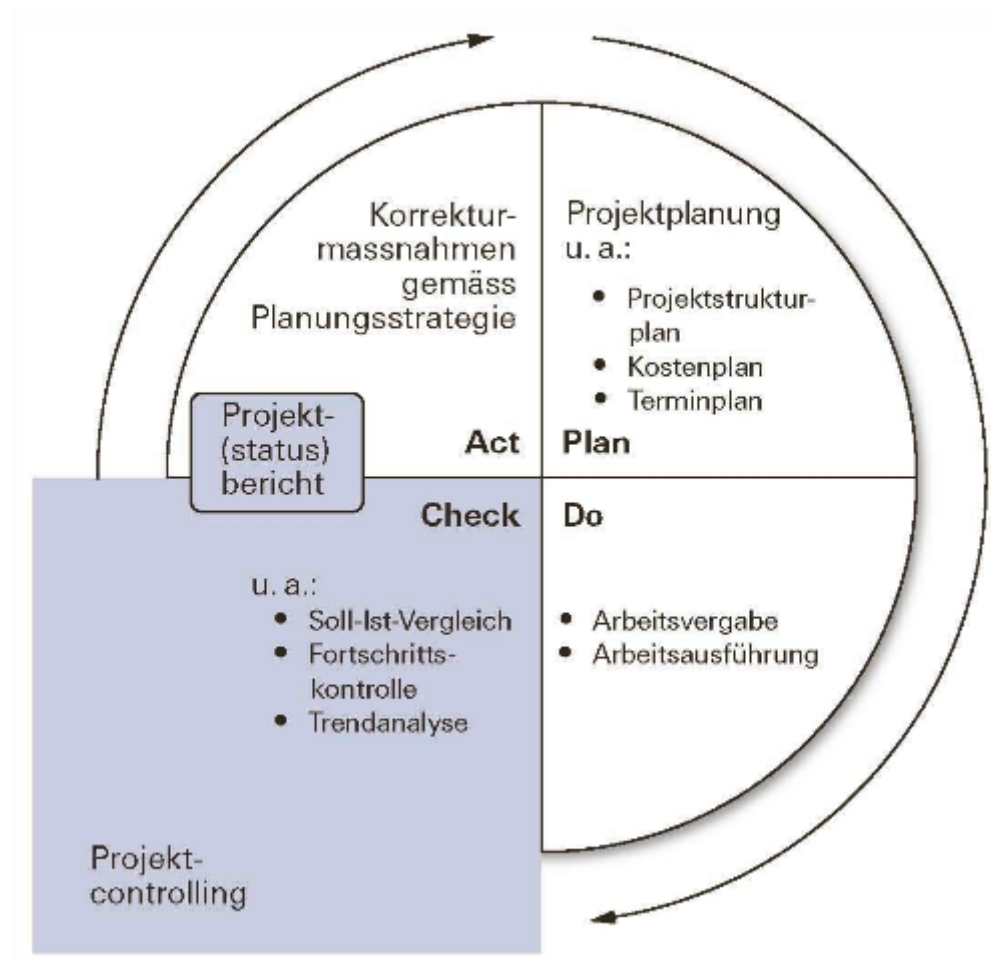


Abbildung 35: PDCA-Modell

16.2 Earned Value

TODO