

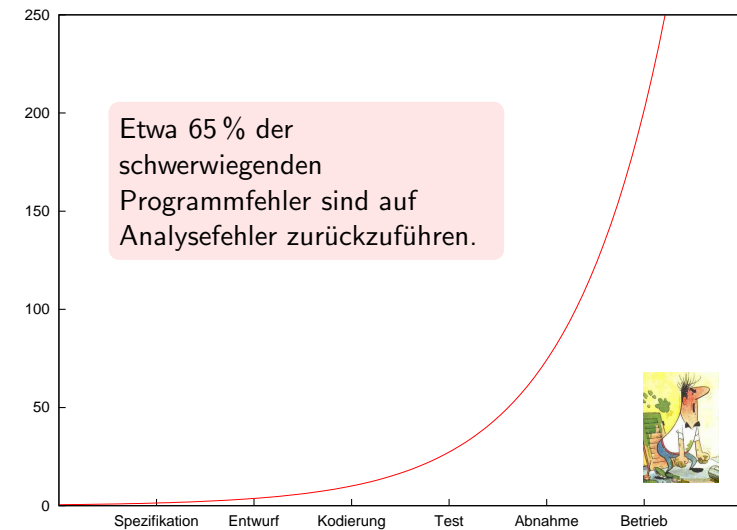
Überblick

Anforderungsanalyse

Einleitung
Aufgabenfelder
Darstellungsmethoden
Zusammenfassung

Wozu?

Relative Fehlerbehebungskosten



Analyse der Problemstellung

Methodisch gestütztes Aufstellen von Anforderungen

Anforderung (engl. *requirement*)

- ▶ eine Aussage über eine zu erbringende Leistung eines Produktes, Systems oder Prozesses
- ▶ eine Eigenschaft, die erfüllt sein muss, damit ein bestimmter Vorgang gelingen kann
- ▶ ein Leistungsmerkmal (nicht nur) von Software

Anforderungen werden im **Lasten-/Pflichten**heft zusammengefasst

- ▶ als Bestandteil eines zu erstellenden Anforderungsdokuments, das...
 - ▶ die Aufgaben beschreibt, die das zu entwickelnde System lösen soll
 - ▶ die von dem Entwicklungsprojekt zu erreichenden Ziele definiert
 - ▶ den Benutzerkreis des zu entwickelnden Systems festlegt
- ▶ in Zusammenarbeit mit dem „Kunden“ (Auftraggeber) entsteht

Anforderungen ↔ Lasten und Pflichten

Aggregiert in verschiedenen Dokumenten

Lastenheft (Anforderungsspezifikation)

- ▶ beschreibt unmittelbare Anforderungen, Erwartungen und Wünsche
- ▶ legt fest, **was** und **wofür** etwas gemacht werden soll

Pflichtenheft (Sollkonzept, Fachfeinkonzept, fachliche Spezifikation)

- ▶ detaillierte Beschreibung einer zu erfüllenden Leistung
 - ▶ liegt am Ende als „schwarzer Kasten“ (engl. *black box*) vor
 - ▶ enthält i. d. R. nicht die Lösung des Problems (der Implementierung)
 - ▶ Inhalte sind präzise, vollständig und nachvollziehbar dargestellt,
- ▶ gibt an, **wie** und **womit** etwas realisiert werden soll
 - ▶ verknüpft mit techn. Festlegungen der Betriebs-/Wartungsumgebung

Nach DIN 69905 [4] enthält das Pflichtenheft die vom Auftragnehmer erarbeiteten Realisierungsvorgaben, die sich aus der Umsetzung des vom Auftraggeber vorgegebenen Lastenheftes ergeben haben.

Gliederung für Pflichtenhefte

Beispiel: Prozesssteuerung

1. Allgemeines
 - 1.1 Einführung
 - 1.2 Referenzen
2. Systembeschreibung
 - 2.1 Funktionelles Zusammenwirken
 - 2.2 Funktionelle Arbeitsweise
 - 2.3 Aufteilung in Hard-/Software
3. Softwareanforderung
 - 3.1 Daten: Name, Typ, Struktur, Wertevorrat, Dimension, Genauigkeit, Zeitbedingungen, Bedeutung
 - 3.2 Funktionen: Ergebnis, Bedingungen, Initialisierung, Sonderfälle, Wiederholfrequenz/Durchlaufzeit, Bedeutung
4. Sonstiges: Programmiersprache, Verfahrensvorschriften

Abgrenzung RE und RM

Zuordnung der einzelnen Schritte im Entwicklungsprozess

Requirements Engineering

- ▶ Erfassung
- ▶ Analyse
- ▶ Dokumentation
- ▶ Prüfung
- ▶ Abstimmung

Generierung „guter“ Anforderungen

Requirements Management

- ▶ Strukturierung
- ▶ Bewertung
- ▶ Verfolgung
- ▶ Änderungsverfolgung
- ▶ Berichtswesen

Verwaltung dieser Anforderungen

Anforderungsanalyse \leadsto Anforderungsdefinition

Folgeschritt im Entwicklungsprozess

Anforderungstechnik (engl. *requirements engineering*, RE)

- ▶ wichtig(st)e Voraussetzung zur Ermittlung der Anforderungen:
 - ▶ Interessenvertreter (engl. *stakeholder*) identifizieren
 - ▶ d. h., die richtigen zu befragenden Personen/Institutionen...
- ▶ oft auch als Synonym verwendet für „Anforderungsanalyse“

Anforderungspflege (engl. *requirements management*, RM)

- ▶ umfasst die Anforderungsanalyse und geht darüber hinaus:
 - ▶ Maßnahmen zur Anforderungssteuerung, -kontrolle und -verwaltung
 - ▶ d. h., Risiko-, Änderungs- und Umsetzungsmanagement
- ▶ elementarer Prozess der „Software- und Systemreifegrad-Modelle“

CMMI *Capability Maturity Model*TM *Integration* [5]

SPICE *Software Process Improvement and Capability Determination* (ISO 15504, [6])

- ▶ auch bekannt als *Software Requirements Specification* (SRS, [7])

Qualitätsmerkmale

Merkmale einer „guten“ Anforderungsspezifikation

Adäquatheit das beschreiben, was der Auftraggeber will bzw. was benötigt wird

Vollständigkeit alles beschreiben, was der Auftraggeber will bzw. was benötigt wird

Widerspruchsfreiheit sonst ist die Spezifikation nicht realisierbar

Verständlichkeit für den Auftraggeber und Auftragnehmer

Eindeutigkeit um Fehler durch Fehlinterpretationen zu vermeiden

Prüfbarkeit feststellen können, ob das realisierte (implementierte) System die Anforderungen erfüllt

Einzelschritte der Anforderungsanalyse

Übersetzungsprozess zwischen Fachseite und Entwickler

1. Anforderungserhebung

- ▶ Kriterien zur **Aufnahme** der Anforderungen:
 - ▶ vollständig, eindeutig definiert/abgegrenzt, verständlich
 - ▶ atomar, identifizierbar, dokumentiert, notwendig
 - ▶ nachprüfbar, rück- und vorwärtsverfolgbar
- ▶ abschließende **Erfassung** der Anforderungen im Lastenheft

2. Anforderungsdefinition

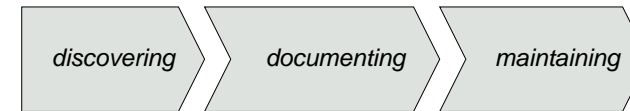
- ▶ Kriterien zur **Strukturierung** der Anforderungen:
 - ▶ abhängig, zusammengehörig, rollenbezogen
 - ▶ funktional/nichtfunktional, fachlich/technisch motiviert
- ▶ abschließende **Abstimmung** zwischen Kunde und Entwickler

3. Anforderungsbewertung → Prüfung und Bewertung

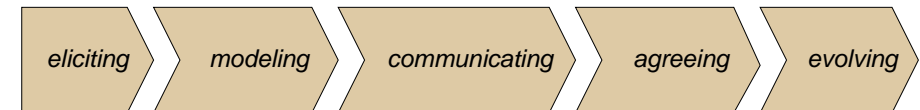
- ▶ Qualitätssicherung der Anforderungen nach folgenden Merkmalen:
 - ▶ korrekt (widerspruchsfrei), machbar, notwendig, priorisiert
 - ▶ nutzbar (nützlich), benutzerfreundlich
- ▶ Ergebnis dieses Schritts ist die Basis für das Pflichtenheft

Einzelschritte der Anforderungsanalyse (Forts.)

Vom Lasten- zum Pflichtenheft...



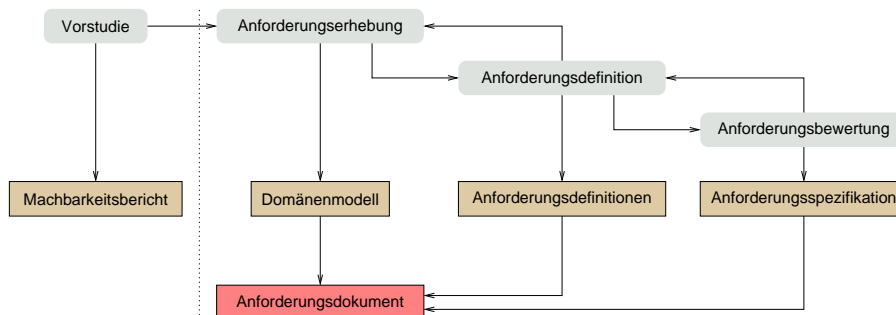
Sommerville & Sawyer [8]



Nuseiher & Easterbrook [9]

Prozess der Anforderungsanalyse

Aktivitäten und Dokumente [10]



Typen von Anforderungen

Nach SRS [7]

1. funktionale Anforderungen
 - ▶ Beschreibung des kompletten deterministischen Systemverhaltens
2. externe Schnittstellen
3. Performanz (statisch/dynamisch)
4. logische Datenbasis
 - ▶ z. B. Nutzungsfrequenz, Zugriffsfähigkeiten, Daten inkl. Beziehungen
5. Entwurfseinschränkungen
 - ▶ Einhaltung von Normen, Systemattribute (von Software)
6. Systemattribute der Software
 - ▶ Zuverlässigkeit, Verfügbarkeit, Sicherheit, Wartbarkeit, Übertragbarkeit

☞ Anforderungen 2.–6. gelten als **nicht funktional**

Typen von Anforderungen (Forts.)

Externe Schnittstellen...

- ▶ Name des Elements bzw. Postens
- ▶ Gegenstandsbeschreibung (Zweck)
- ▶ Quelle der Eingabe und Ziel der Ausgabe
- ▶ Gültigkeitsbereich, Genauigkeit und Abweichung
- ▶ Maßeinheit
- ▶ Zeitvorgabe
- ▶ Beziehung zu anderen Ein-/Ausgaben
- ▶ Bildschirmformate/-organisation
- ▶ Fensterformate/-organisation
- ▶ Datenformate
- ▶ Befehlsformate

Herausfinden

... was der Kunde will bzw. machbar ist

Erhebung (engl. *elicitation*)

- ▶ Identifizierung von Anforderungen, Auflagen und Einschränkungen
 - ▶ Fragebogeninterviews, offene Interviews, Besprechungen
- ▶ Wiederverwendung von Anforderungen vorangegangener Projekte

Abstimmung (engl. *negotiation*)

- ▶ Auflösung evtl. bestehender **Konflikte**...
 - ▶ zwischen Fähigkeiten und Einschränkungen
 - ▶ zwischen Anforderungen und Betriebsmittel (Ressourcen)
 - ▶ wegen inkompatibler Merkmale verschiedener Interessensvertreter
- ▶ „Verhandlung“ mit den Interessensvertretern (Auftraggeber)
 - ▶ Konsensfindung, Kompromisswege/-lösungen herausarbeiten

Formulieren

... des Problems und ggf. auch einer Lösungsidee

Analyse (engl. *analysis*)

- ▶ Grenze des Systems und Interaktion mit der Umgebung erläutern
 - ▶ dabei ggf. auch verschiedene Sichten (engl. *viewpoint*) einnehmen
 - ▶ z. B.: unterschiedliche Entwicklerrollen oder Beschreibungstechniken
- ▶ widersprüchliche Anforderungen identifizieren und ggf. auflösen

Spezifikation (engl. *specification*)

- ▶ vollständige Menge zusammenhängender Anforderungen gestalten
- ▶ Subsysteme/Komponenten definieren und Anforderungen zuordnen

Modellierung (engl. *modeling*)

- ▶ Systemeigenschaften durch **konzeptionelle Modelle** untersuchen
 - ▶ Daten-/Kontrollfluss-, Zustands-, Objekt-, Anwendungsfallmodelle
- ▶ die operative Umgebung samt Daten und Kommunikation verstehen

Organisieren

... um die Problemkomplexität zu beherrschen

Dokumentation (engl. *documentation*) \leadsto **Anforderungsdokument**

- ▶ die Menge aller beschriebenen Anforderungen zusammenstellen
- ▶ Lastenheft erzeugen, das später ins Pflichtenheft überführt wird

Strukturierung (engl. *structuring*)

- ▶ Anforderungen nach verschiedenen Kriterien klassifizieren
 - ▶ Gruppierung nach z. B. Priorität (bei der Erfüllung der Gesamtziele), Herkunft, Gültigkeitsbereich, Stabilität, usw. vornehmen
 - ▶ in funktionale und nicht-funktionale Anforderungen einstufen
- ▶ Attribute für jede Anforderung festlegen
 - ▶ Beschreibung, Grund, Urheber, Status, Akzeptanzkriterien, Implikationen, Abhängigkeiten, ...
 - ▶ dient u. a. auch der weiteren Gruppierung (s. o.)
- ▶ den Anforderungen eindeutige Bezeichner zuordnen

Hinterfragen

... ob das Problem richtig verstanden wurde

Validierung (engl. *validation*)

- ▶ sicherstellen, dass das beschriebene System die ursprünglichen Intentionen (des Auftraggebers) adäquat wiedergibt
 - ▶ ein sich zu verschiedenen Prozesszeitpunkten wiederholender Vorgang
- ▶ das Anforderungsdokument examinieren, in Form von Inspektionen oder formalen Besprechungen durch Gutachtergruppen
 - ▶ Fehler, irrtümliche Annahmen, unklar bestimmte Begriffe, Abweichungen von üblichen Vorgehensweisen identifizieren
 - ▶ Gutachter sind u. a. auch Beauftragte der Benutzer des Systems
- ▶ ggf. einen Prototypen zeigen, um die ursprüngliche Intention (s. o.) mit der eigenen Interpretation des Systems zu konfrontieren
 - ▶ manchmal genügen bereits einfache Papierskizzen

Spezifikationstechniken

Allgemeine Klassifikation bzw. Ansätze

formal (engl. *formal*)

- ▶ rigorose, mathematische Grundlage \leadsto **formale Notation**

informell (engl. *informal*)

- ▶ wenn die **Transkription** („Umkodierung“) in eine formale Notation mit zugeordneten Regeln nur eingeschränkt möglich ist
 - ▶ z. B. ein Ablaufdiagramm (engl. *flowchart*)
- ▶ bestenfalls werden Anforderungsverletzungen/-konflikte sichtbar

halbförmlich (engl. *semiformal*)

- ▶ Ansätze, die formale und informelle Züge zeigen, wie z. B. UML:
 - ▶ das Zustandsdiagramm (engl. *statechart*) ist formal
 - ▶ andere Konzepte sind jedoch eher pseudomathematischer Natur

Echtzeitsysteme (mit strikt einzuhaltenden Anforderungen) erfordern eine formale Begründung der Leistungscharakteristiken von Anforderungen.

Vorbereiten

... für die Phasen der Systementwicklung danach

$\neg (RE \wedge RM)$

Entwurf (engl. *design*)

- ▶ überlegen, wie die Anforderungen umgesetzt werden können

Implementierung (engl. *implementation*) und Integration

- ▶ es tun, d. h., die Anforderungen umsetzen

Verifikation (engl. *verification*) und Testen

- ▶ das Ergebnis mit dem ursprünglichen „Plan“ vergleichen

Einführung (engl. *rollout*)

- ▶ das „Produkt“ ausliefern

Natürliche Sprache

- ▶ weit verbreitete Technik
- ▶ Strukturierung durch Nummerierungs- und Gliederungsschemata
- ▶ Qualitätsverbesserung durch **linguistische Methoden**
 - ▶ Sätze mit Standardstruktur
 - ▶ kein Passiv
 - ▶ beschränkte Menge von Verben mit festen Bedeutungen
 - ▶

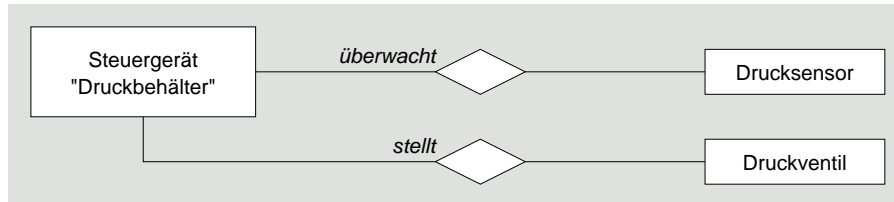
pro leicht zu schreiben/lesen, ausdrucksmächtig

cons unübersichtlich, fehleranfällig, schwierig zu prüfen

☞ ungeeignet als alleiniges Beschreibungsmittel

Datenmodellierung

- ▶ Grundlage ist der *Entity-Relationship*-Ansatz
- ▶ modelliert werden Ausschnitte der Realität durch...
 - ▶ Gegenstandstypen (engl. *entity types*)
 - ▶ Beziehungstypen (engl. *relationship types*)
 - ▶ Attribute (engl. *attributes*)

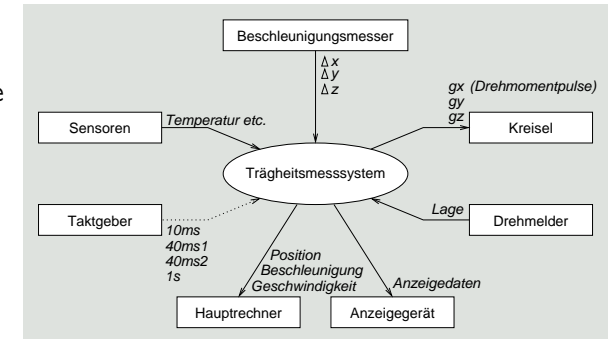


pro vergleichsweise einfach und klar, ideal für Datenbankrealisierungen

cons beschreibt weder Funktionalität noch Verhalten von Systemen, hat keine Mittel zur Systemdekomposition bzw. Datenkapselung

Strukturierte Analyse

- ▶ Grundlage bilden Datenflussdiagramme
- ▶ Modellierung von Systemfunktionalität
- ▶ Beschreibung des Systemkontextes
 - ▶ Interaktion
 - ▶ Ein-/Ausgabe



pro vergleichsweise anschaulich, unterstützt Systemdekomposition

cons keine Lokalität von Daten, begrenzte Kapselungsfähigkeiten, nicht-funktionale Anforderungen nicht adäquat beschreibbar, „Strukturbruch“ zwischen Spezifikation und Implementierung

Objektorientierte Spezifikation

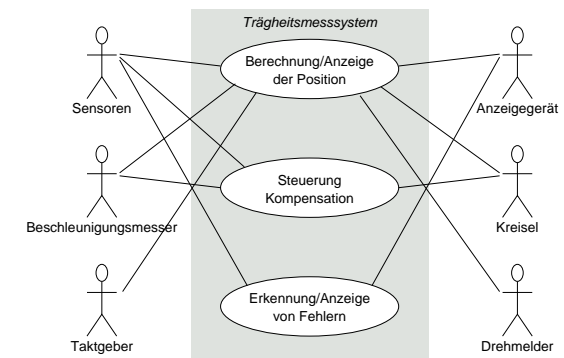
- ▶ Modellierung der statischen Struktur eines Systems unter Verwendung von Objekt- und Klassendiagrammen
- ▶ Objekte/Klassen beschreiben Daten, Funktionen und zeitliches Verhalten

pro gut geeignet zur Beschreibung der Systemstruktur, unterstützt Lokalität von Daten und Einkapselung, motiviert strukturähnliche Implementierung, ermöglicht Systemdekomposition

cons nicht-funktionale Anforderungen nicht adäquat beschreibbar

Szenarien und Anwendungsfälle

- ▶ Modellierung der Interaktion zwischen System und Umwelt
 - ▶ d. h., Akteure
- ▶ Interaktionssequenzen entsprechen Szenarien
 - ▶ Anwendungsfall
 - ▶ engl. *use case*



pro leicht verstehbar und überprüfbar, modelliert Funktionalität aus Benutzersicht, Abgrenzung des Systems vom Kontext, ermöglicht Systemdekomposition

cons keine Erfassung von Zusammenhängen/Abhängigkeiten von Szenarien, statische Struktur, keine Datenmodellierung

Formale Methoden

- ▶ Grundlage bilden mathematisch-logische Formalismen
 - ▶ formal definierte Syntax und Semantik
- ▶ große theoretische Vorteile, praktisch aber nur marginal aufzufinden
 - ▶ punktueller Einsatz: **sicherheitskritische Systeme**

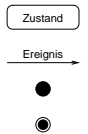
pro Eindeutigkeit durch formal definierte Semantik, Widerspruchsfreiheit formal abprüfbar, Nachweisbarkeit der Erfüllung von Anforderung, Lösungsneutral

cons aufwändige Erstellung, Prüfung auf Adäquatheit schwierig (Expertenwissen), umfangreiche Spezifikationen auch für Fachleute schwer verständlich

Zustandsdiagramm (engl. *statechart*)

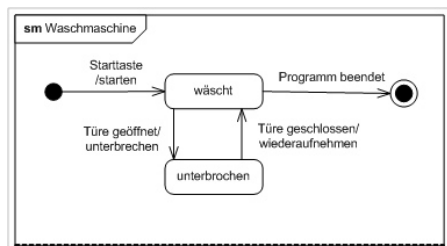
Formale Methoden (Forts.)

- ▶ ein **endlicher Zustandsautomat** (engl. *finite state machine*, FSM) kombiniert mit Datenflussdiagrammen
 - ▶ dokumentiert die verschiedenen Zustände einer „Klasse“
 - ▶ beschreibt die Ereignisse, die zur Zustandsänderung führen
- ▶ grundlegende Beschreibungselemente (Notation):
 - Zustand** der „Modus“ einer Entität
 - Übergang** die **ereignisbedingte Zustandsänderung**
 - Initialzustand** genau einer, vor allen Übergängen
 - Endzustand** einer von ggf. vielen
- ▶ verschiedene Ausprägungen, je nach Modellierungszweck:
 - Verhaltenszustandsautomat** (engl. *behavioral state machine*)
 - ▶ modelliert das **Verhalten** eines Modellelements (z. B. Klasse)
 - Protokollzustandsautomat** (engl. *protocol state machine*)
 - ▶ modelliert die **zulässige Nutzung** der Verhaltensmerkmale

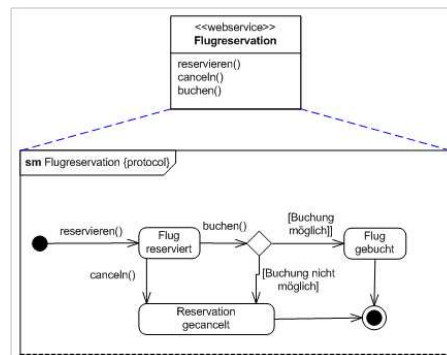


Zustandsdiagramm (Forts.)

Beispiele in UML



Verhaltenszustandsautomat



Protokollzustandsautomat

Resümee

Einleitung

- ▶ Anforderung, Qualitätsmerkmale, Typen von Anforderungen
- ▶ Anforderungsanalyse (-technik) vs. Anforderungspflege: RE \mapsto RM
- ▶ Einzelschritte bzw. Prozess der Anforderungsanalyse
- ▶ Anforderungsspezifikation; Lastenheft, Pflichtenheft

Aufgabenfelder \leadsto herausfinden, formulieren, organisieren, hinterfragen

- ▶ Erhebung, Abstimmung
- ▶ Analyse, Spezifikation, Modellierung
- ▶ Dokumentation, Strukturierung
- ▶ Validierung

Darstellungsmethoden

- ▶ formale, informelle, halbformale Spezifikationstechniken
- ▶ natürliche Sprache, Datenmodellierung, strukturierte Analyse, objektorientierte Spezifikation, Anwendungsfälle, formale Methoden
- ▶ Zustandsdiagramme