

Mitschrift von Andreas Groll

Vorlesung

Informationssysteme

2002/2003 Prof. Jasper

Hinweise, Verbesserungen, Vorschläge, Kritik und natürlich Dank an

TheChaos@networkchallenge.de

<http://chaos.networkchallenge.de>

Diese Mitschrift ist leider in keinster Weise vollständig, vor allem da auch in der Vorlesung oft auf das Buch „Carl Steinweg – Projektkompass Softwareentwicklung“ verwiesen wurde. Falls jemand ergänzendes Material in digitaler Form hat, kann er es mir bitte zukommen lassen.

Weiteres BNC – relevantes Material unter:

<http://www.fabianpetzold.de/bnc/>

2.4. IT-Projekt(e) definieren (High Level Requirements)

2.4.1. Ergebnisse (für jedes Projekt)

- Projektziele und Rahmenbedingungen
- Projektscope → Prozesse / grobes Datenmodell
→ gut kommentiertes Kontextdiagramm
- Prozesse und Leistungsmerkmale
- Lösungsansatz: Systemkonfiguration
- Projektorganisation (grob)
- Vorgehen und Meilensteine (grob)

2.4.2. Machbarkeit prüfen / Wirtschaftlichkeit darlegen

Literatur!

3. Konzeption

3.1. Ziele:

- Der Leistungsumfang des zu erstellenden Systems ist festgelegt
- Lösungsalternativen sind erarbeitet und bewertet
- Die Wirtschaftlichkeit der gewählten ist dargestellt
- Ein Umsetzungsangebot ist erstellt

→ Sign-off des Arbeitgebers

Voraussetzungen:

- Ziele und Rahmenbedingungen existieren
- Scope des Projekts ist aus Geschäftssicht festgelegt
- existierende Vorgaben für die Systemkonfiguration sind dargestellt

Ergebnisse:

- fachliche Konzept
 - überarbeitete PHDs (Prozesshierarchiediagramme)
 - Prozesskettendiagramme (PTDs)
 - Beschreibung der elementaren Geschäftsprozesse
 - Use-Case-Diagramme
 - Beschreibung der Use-Cases
 - Kontextdiagramm aus Systemsicht
 - Business-Objektmodell

- IT-Lösung
 - Lösungsszenarien
 - Lösungsvorschläge
 - Machbarkeit
 - Wirtschaftlichkeit

- Leistungsumfang und Abnahmekriterien
 - Anforderungskatalog
 - Abnahmekriterien

3.2. Fachliches Konzept erarbeiten

3.2.1. Geschäftsprozesse identifizieren und beschreiben

Ziele: - Die wesentlichen Prozesse innerhalb des Scopes des Projekts sind identifiziert
- Die Prozesse sind aus Geschäftssicht hierarchisch strukturiert und beschrieben

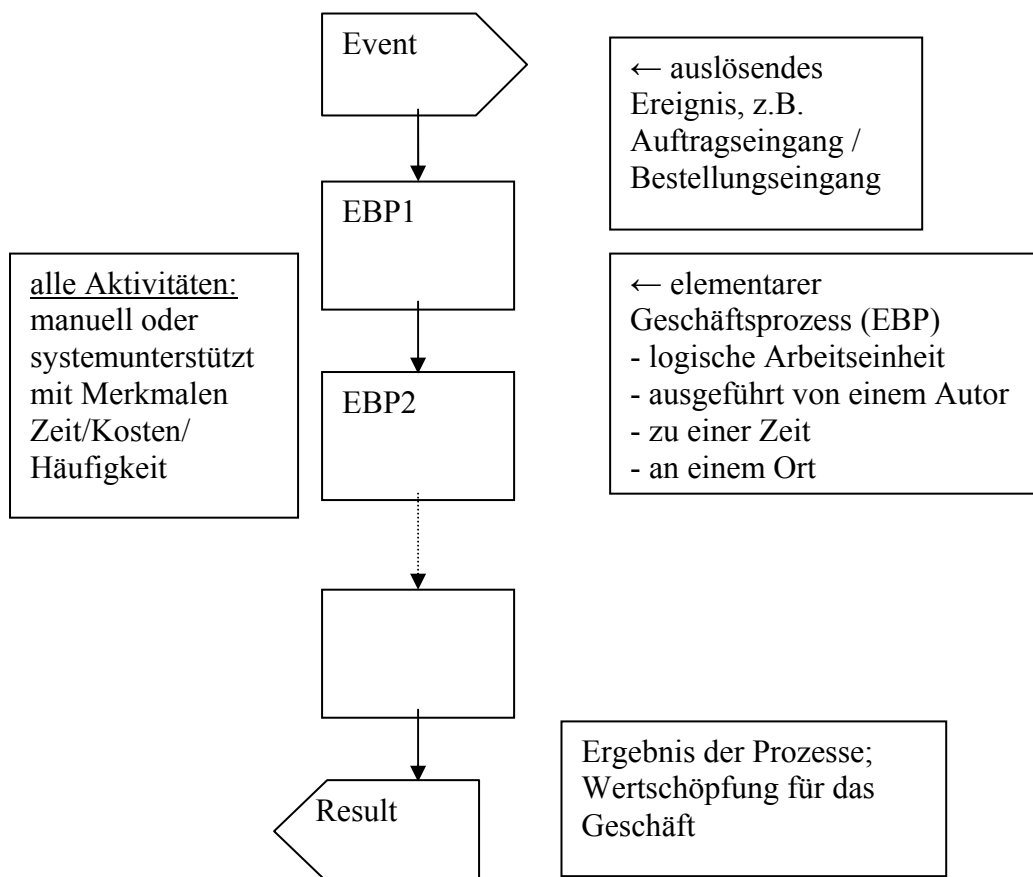
Aufgaben/Vorgehen

1. Benennen der Prozesse
2. Strukturieren
3. Einordnen in eine Hierarchie
4. Beschreibung der Prozesse (in SE)
 - Name
 - Ziel / Zweck / Beschreibung
 - Leistungsmerkmale

3.2.2. Prozessketten und elementare Geschäftsprozesse

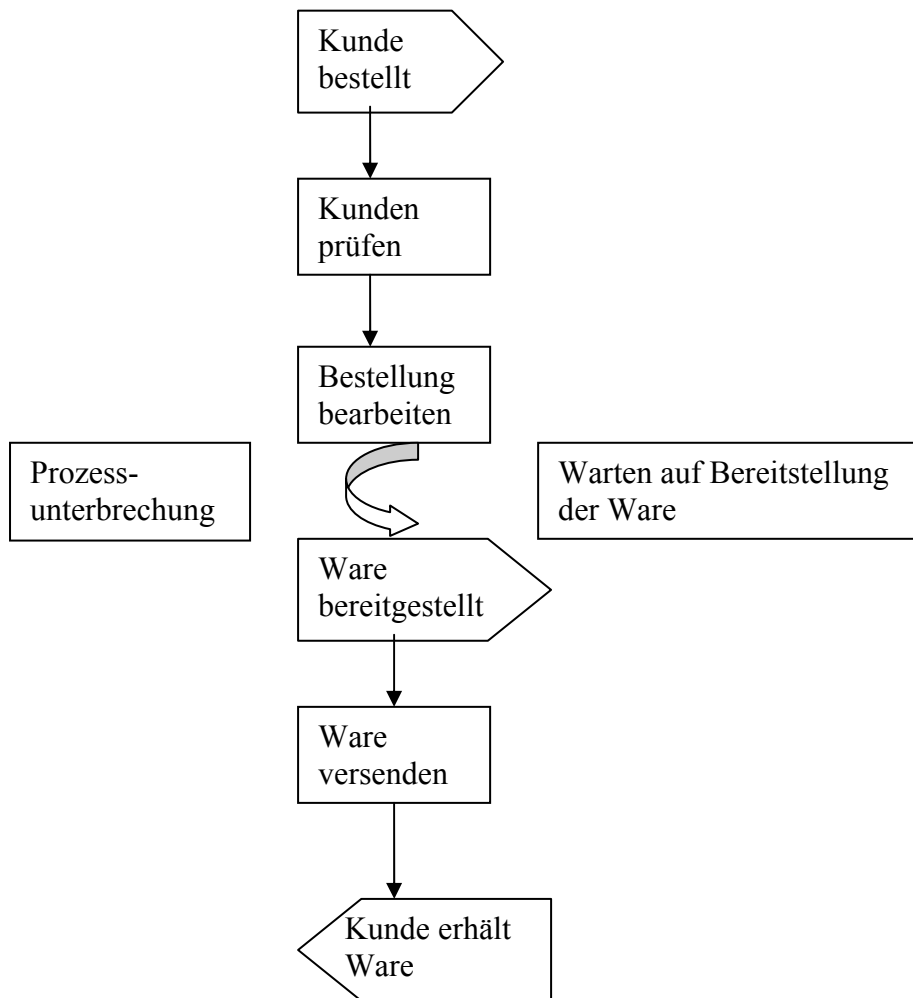
Ziele:

Exkurs Prozesskette



- für die Prozesse der Blätter des PHD sind die Prozessketten identifiziert und bewertet
- alle elementaren Geschäftsprozesse, die auslösenden Ereignisse und die Ergebnisse sind beschrieben
- jeder EBP ist beschrieben durch
 - Name
 - Ziel/Zweck (Objective)
 - Beschreibung des Ablaufs (Description)
 - Actor (Name / Funktion)
 - Leistungsmerkmale (Volumetrics)

Beispiel: Auftrag bearbeiten



Beschreibung der EBP:

Name: Kunden prüfen
Objective: Kunde ist bekannt, Daten aktualisiert
Description: 1. Falls Kunde neu: Daten erfassen
2. relevante Daten ändern
3. Adresse prüfen
Actor: Kundenbetreuer (CCA)
Volumenries: 10 Kunden/h

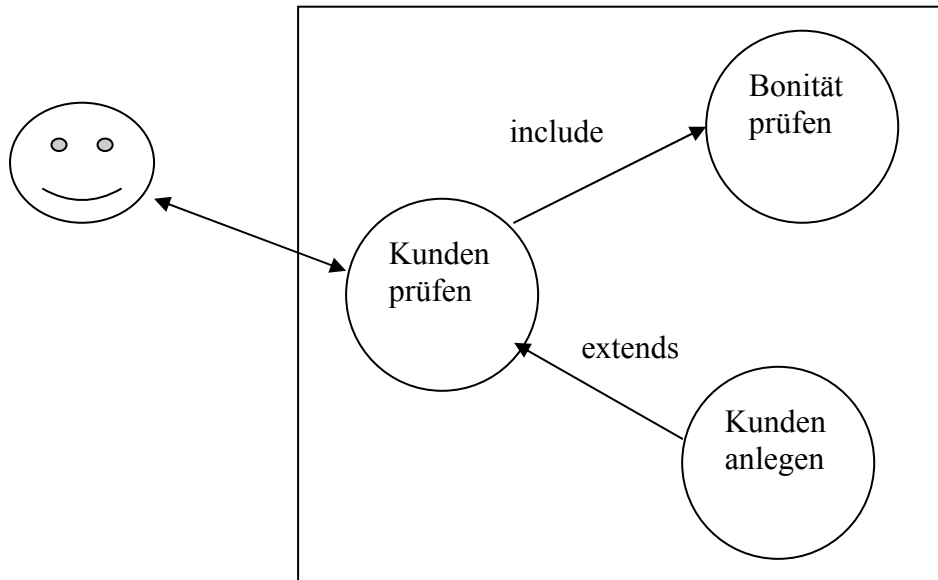
möglichst vollständige Beschreibung des fachlichen Ablaufs!

1. main course (Normalfall)
2. unbekannte Alternativen

3.2.3. Use Case entwerfen

Ein Use-Case ist die Zusammenfassung einer Reihe von zusammengehörigen Funktionen, die ein Actor mit dem System durchführt.

Dargestellt als Oval in einem Use-Case-Diagramm (UCD)



Ziele:

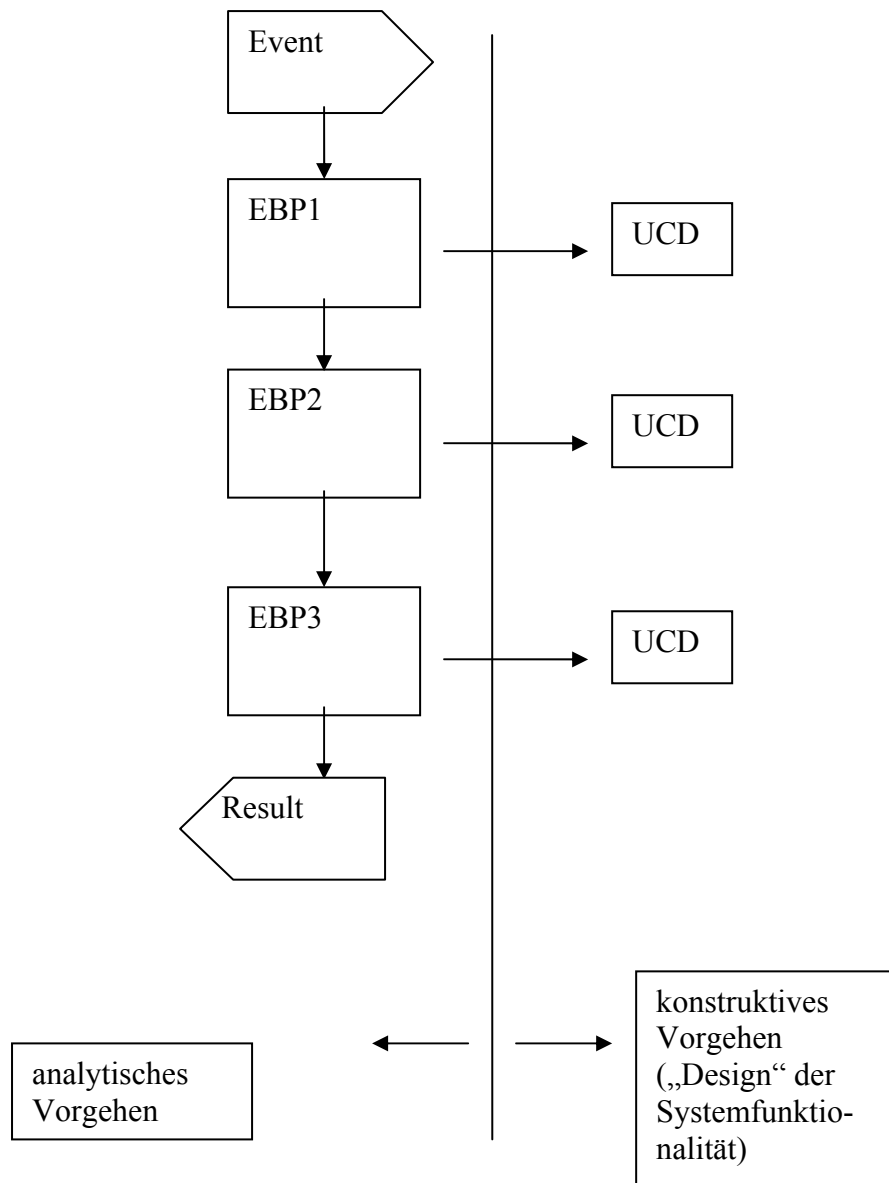
- Die Use-Cases als die Systemunterstützung der elementaren Geschäftsprozesse sind identifiziert
- Die Actors sind den Use-Cases zugeordnet
- alle Use-Cases sind beschrieben

Ergebnisse:

- Liste der Actors
- Liste aller Use-Cases
- Beziehungen zwischen Use-Cases und Actors definiert
- Use-Cases und Actor beschrieben

Aufgaben/Vorgehen:

1. Use-Case Diagramme entwerfen
Prozessbezogenes Vorgehen



Anmerkungen:

Falls Systemunterstützung für ein EBP, dann meist 3-5 Use-Cases für ein EBP.

Kein Use-Case für EBP → keine Unterstützung durch das System.

Zur Prüfung der entworfenen Funktionalität: Use-Cases nach Actor geordnet in Diagrammen anordnen.

2. Use-Cases beschreiben

Name: Kunden prüfen
Intent: Ein Kunde ist geprüft
Description: 1. Kunde anhand KundenID oder Namen identifizieren
2. Falls der Kunde Neukunde ist „Kunde anlegen“ (eigener Use-Case)
3. Informationen zum Kunden aktualisieren
4. Adresse prüfen (Aufruf von „Post adress“)
5. „Bonität prüfen“ (eigener Use-Case)
Precondition: Bestandskunden bekannt
Postcondition: Kundendaten sind geprüft
Alternate Courses:
- Kundendaten unvollständig
- Adressprüfung nicht verfügbar
- Bonitätsprüfung nicht verfügbar

3. gegebenenfalls User-Interface entwerfen
→ nutze Werkzeuge!
→ ein Entwurf pro wichtigem Use-Case

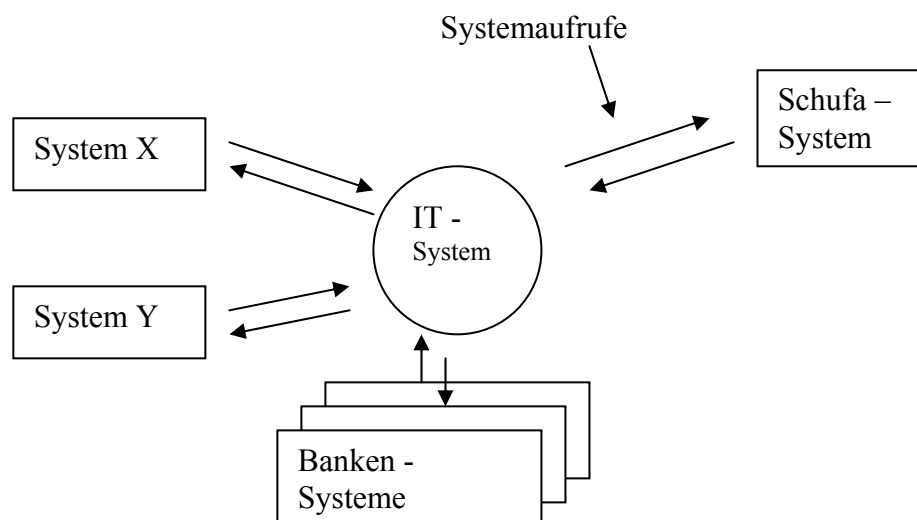
3.2.4. Business Objektmodell entwerfen

Business-Objekt ↔ Entität

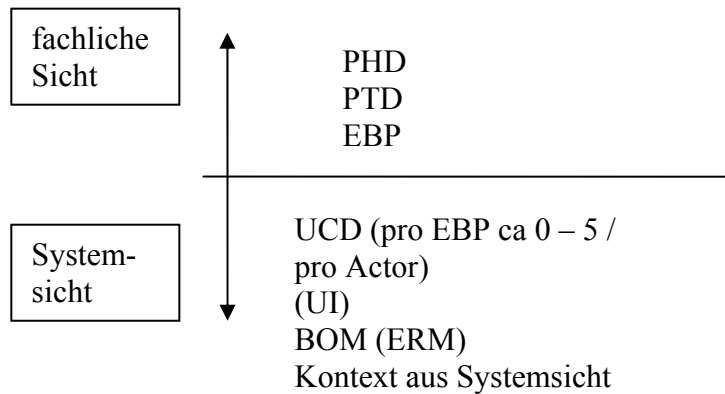
Ziele: - alle wesentlichen Objekte, auf deren Datenhaushalt die zu unterstützenden Prozesse zugreifen sind identifiziert
- zu jedem Objekt sind die wichtigsten Attribute (und Methoden) identifiziert und beschrieben
- die Objekte sind zueinander in Beziehung gesetzt

Ergebnis: (grobes) UML – Klassendiagramm
(alternativ: ER – Diagramm)

3.2.5. Kontextdiagramm aus Systemsicht



Konzeption:



3.2.6. Anforderungen zusammenstellen

- Ziele: - Anforderungskatalog
 - Priorisierung
 - funktionale & nichtfunktionale Anforderungen

Prozess	Bedeutung	Use-Case	Bedeutung	Erfüllung
Auftragsbearbeitung	A	Kunden prüfen	A	
		Bonität prüfen	A	
Kundenbewertung	B	Bewertungsfunktion anlegen	C	

Funktionale Anforderung

Aspekt	Bedeutung	Anforderung	Bedeutung	Erfüllung
Verfügbarkeit	A	24 x 7 x 365	A	
		parallele Verarbeitung von Massendaten	B	

3.3. IT – Lösung konzipieren

3.3.1. Lösungsalternativen ermitteln und bewerten

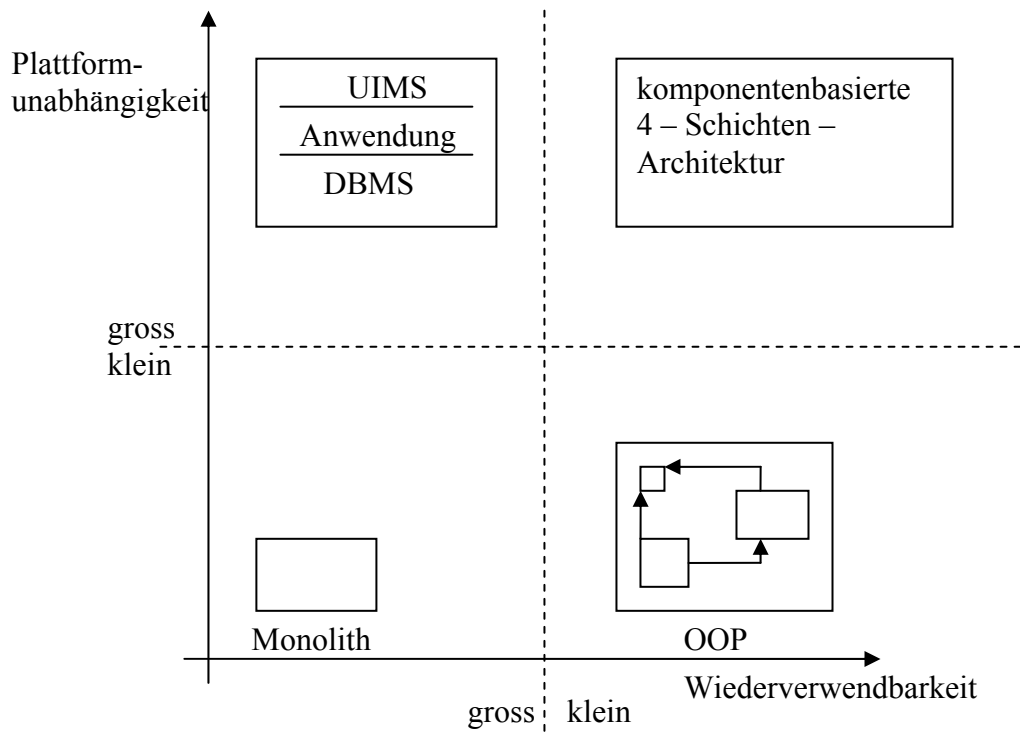
- selber machen, klonen
- funktionale Anforderungen (Use-Cases unterstützen?)
- gibt es schon Erfahrungen mit so einer Lösung?

3.4. Leistungsumfang und Abnahmekriterien festlegen

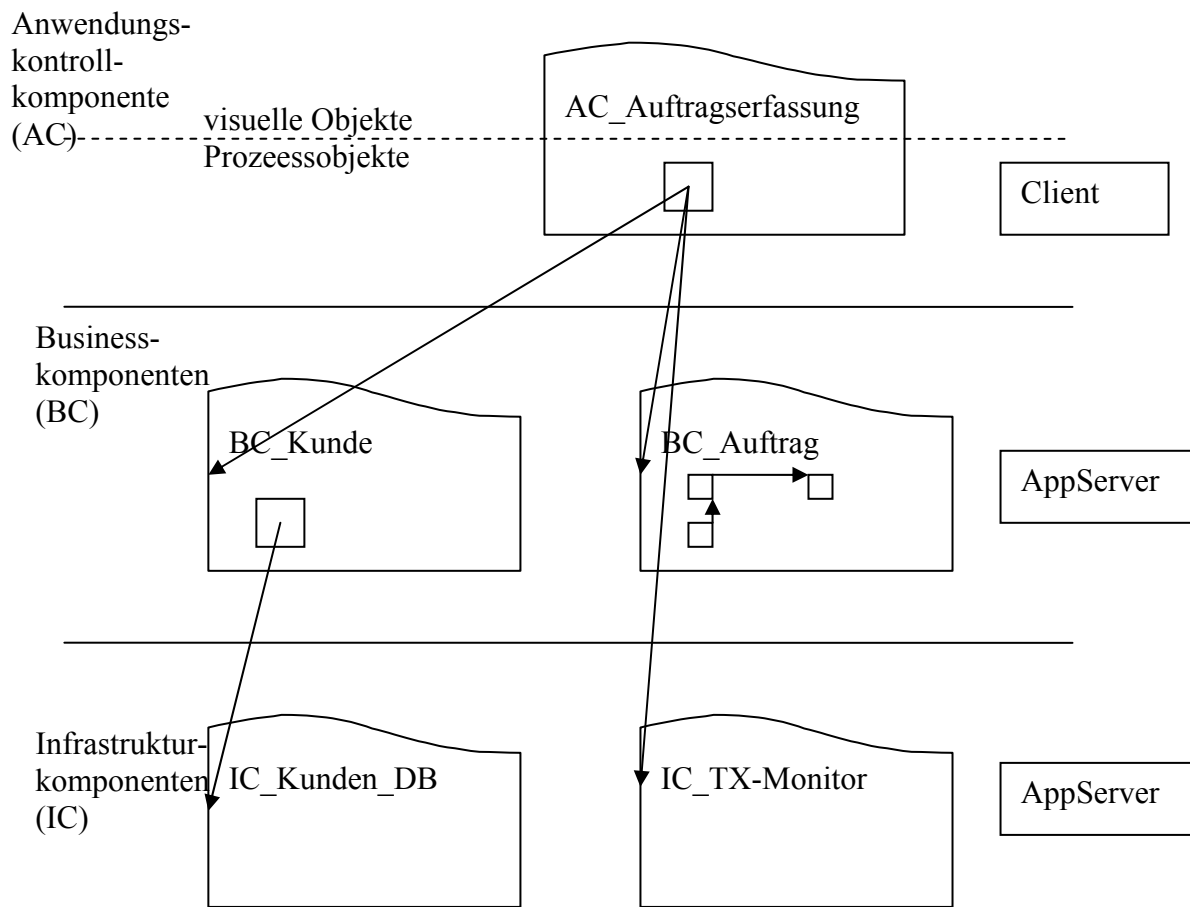
Abnahmekriterien sind Testfälle!

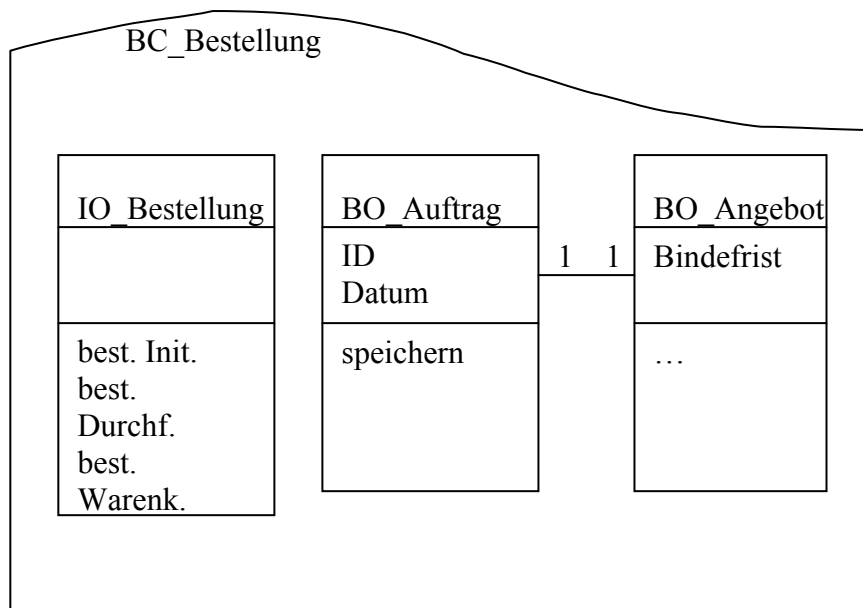
→ Unterschriften

4. Design



4 – Schichten – Architektur





Beispiel:
Jedem EBP (und damit UCD) ist ein Actor zugeordnet.

Kontextdiagramm: - stehen allein
- externe Partner des Systems können Actors sein

BOM: - steht allein
- Objekte (Klassen) sind die relevanten

4.1. Orientierung

- Komponente
- komponentenbasierte, objektorientierte 4 – Schichten – Architektur (3 Schichten)

4.1.1. Ziele

- Die Komponentenarchitektur ist entworfen
- Die zu realisierenden Komponenten sind beschrieben
- Die Komponentenumsetzung ist definiert

4.1.2. Ergebnisse

zur Komponentenarchitektur

- Komponentenarchitektur
- Beschreibung der Schnittstellen der Komponenten
- Sequenzdiagramme für die Use-Cases mit benötigten visuellen Objekten und Prozessobjekt(e) und Schnittstellenobjekte der Komponenten
- vollständige Beschreibung der Use-Cases

Zu jeder einzelnen zu realisierenden Komponente

- Klassenmodell der Komponente mit Attributen, Methoden und Assoziationen
- Sequenzdiagramme zu den Methoden der Schnittstellen der Komponenten
- ggf. Lebenszyklen komplexer Objekte → STD
- ggf. User Interface Design

Zur Komponentenumsetzung

- logisches Datenmodell (Create Table)
- physisches Datenmodell (Create Index)
- Realisierungskonzept (Sprache/Systeme)

4.1.3. Voraussetzungen

- UCD
- BOM
- Kontext (Systemsicht)

4.1.4.

Komponentenarchitektur → Komponentenentwurf → Komponente umsetzen → gutes Design

4.2.

5. Realisierung

5.1. Ziele

- die Infrastruktur für die Realisierung des IT – Systems steht allen zuständigen Mitarbeitern des Projekts produktiv zur Verfügung
- die Schnittstellen zu existierenden Systemen sind realisiert, getestet und dokumentiert
- die umzusetzenden Komponenten sind dokumentiert, implementiert und getestet
- die Komponenten und die Schnittstellen zu den externen Systemen sind zum IT – System integriert und getestet
- alle notwendigen Dokumente und Prozeduren für die Wartung und Weiterentwicklung des IT – Systems existieren und sind erprobt

6. Einführung

→ Einpassung des IT – Systems in seine Umwelt

Skizze Seite 172 3. Auflage

6.1.1. Ziele

- Das von der Realisierung gelieferte Werk ist in die Betriebsinfrastruktur übernommen (Produktionsumgebung)
- alle notwendigen Softwarekomponenten stehen zur Verfügung
- Anwender, Betriebsteam etc. sind ausreichend geschult
- das System ist vom Kunden offiziell abgenommen

6.1.2. Voraussetzungen

Alle Programme und Dokumente aus den früheren Phasen liegen vor

6.1.3. Ablauf

1. Werkabnahme
2. Pilotsbetrieb
3. Offizielle Abnahme des System
4. Roll Out
5. Training → typischerweise auch früher
6. Going Life

6.2. Integration des Testkonzepts

Skizze S. 179 3. Auflage

7. Betrieb

- dauert lange (hoffentlich)
- ist teuer! (=80% der Gesamtkosten), abhängig von Dauer
- im Wesentlichen Personalkosten

7.1. Orientierung

Gewährleistung der Produktionssicherheit

7.1.1. Ziele

- das Anwendungssystem läuft störungsfrei und performant
- die Nutzer sind mit dem System zufrieden
- Die Änderungswünsche des Kunden gehen berechenbar in die Optimierung und Weiterentwicklung des Systems ein
- das Anwendungssystem wird ständig verbessert und weiterentwickelt (state-of-the-art)

7.2. Ziele

- das Anwendungssystem läuft störungsfrei und performant
- die User sind mit dem System zufrieden
- die IT-bezogenen Services werden entsprechend der vereinbarten Servicelevel bereitgestellt
- Änderungswünsche der Kunden gehen berechenbar in die Optimierung und Weiterentwicklung des Systems ein
- das Anwendungssystem wird ständig verbessert und optimiert

7.3. Ergebnisse

- Service – Level – Management
- Service – Desk (help desk bzw Service)
- Availability Management
- Incident Management
- Continity Management
- Capacity Management
- Problem Management
- Change Management (kleine Projekte)

8. Projektmanagement (steuert Erfolg des Projekts)

8.1.1. Ziele

- Lenkung & Gestaltung des Projektes sind inhaltlich und organisatorisch abgesichert
- die Projektziele werden unter den vereinbarten Rahmenbedingungen erreicht

8.1.2. Ergebnisse

- Projektorganisation und –kommunikation
- Gestaltung der Projektprozesse
- Gestaltung des Projektleitungsinstrumentariums

8.2. Grundlegende Begriffsbestimmung

- DIN 69901

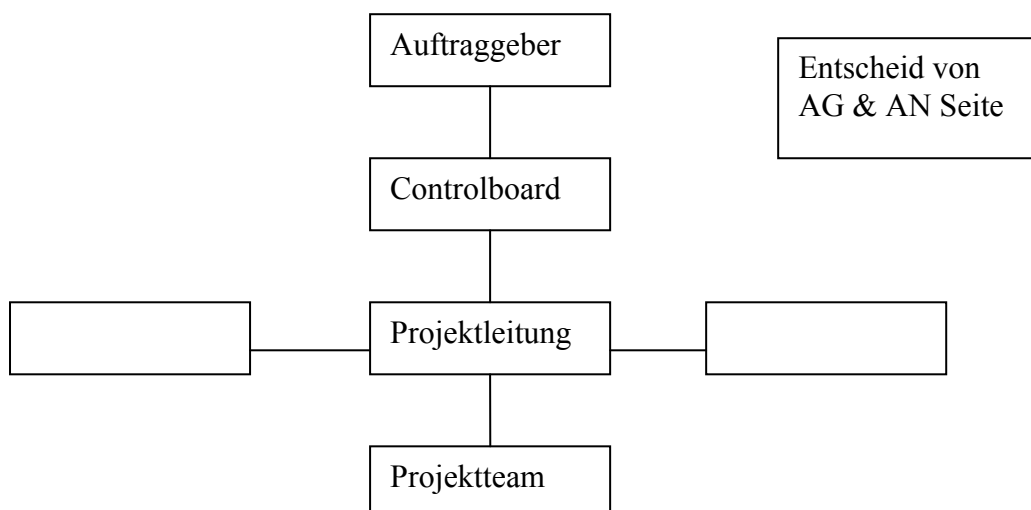
„Ein Vorhaben, das im Wesentlichen durch die Einmaligkeit der Bedingung ihrer Gesamtheit gekennzeichnet ist“

- klare Zielvorgabe und erzeugt etwas Neues
- Start und Ende Termin
- einmalige Bedingungen (Budget, Personal, Vertrag)
- eigene Organisation

Projektprozess

Initialisieren → Planung → Steuerung → Abschluss

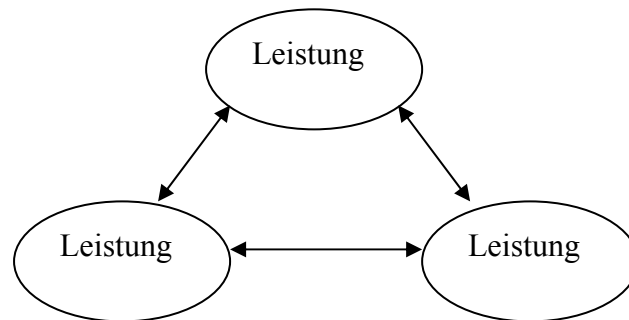
Projektorganisation



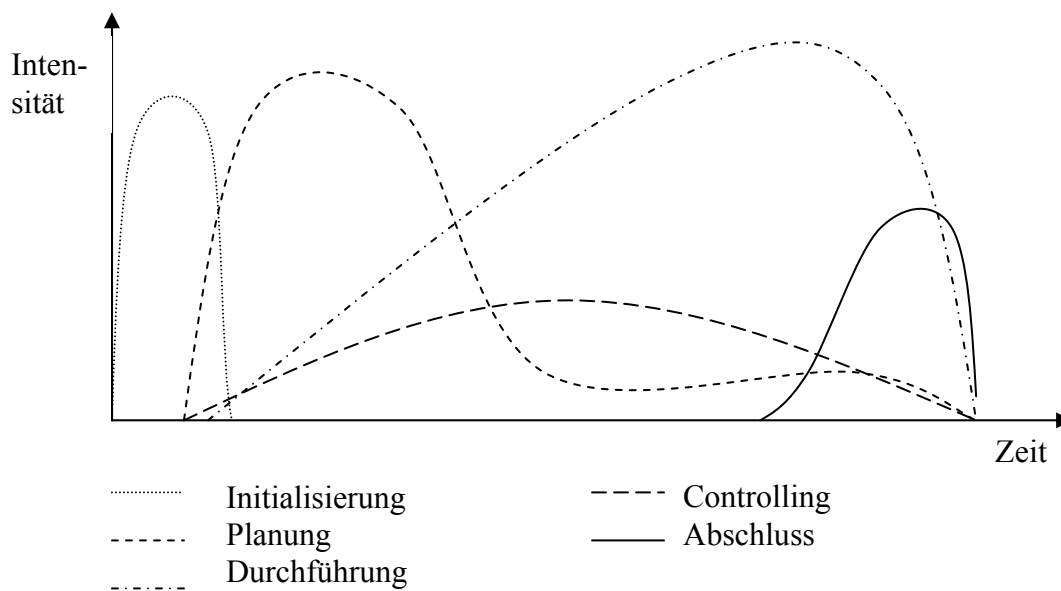
8.3. Projektmanagement

DIN 69901: „Projektmanagement bezeichnet die Gesamtheit von Führungsaufgaben, Führungssituationen und Führungstechniken und Mitteln für die Abwicklung eines Projekts“

8.3.1. Ziel



8.3.2. Projektmanagement – Prozess



8.4. Die Phasen des Projektmanagementprozesses

8.4.1. Initialisierung

- Ergebnisse:
- Projektrahmen
Ziele, Umfang, Ergebnisse, Budget, Rahmenbedingungen,
Abnahmekriterien
 - Projektablauf
Meilensteine, grober Zeitplan
 - Projektorganisation
 - Projektabsicherung
Risiko, Qualitätsanforderungen

Aufgaben / Vorgehen:

- Vorgespräche mit Personen führen
- Management – Kick – Off WS durchführen
- Projektauftrag formulieren und Abnahme durch Auftraggeber
- Team – Kick – Off WS durchführen

8.4.2. Projektplanung

- Ergebnisse:
- vom Controlboard abgenommener Projektplan
 - Projektrahmen (Details zu Deliverables und Arbeitspaketen)
 - Projektablauf (Meilensteine mit Arbeitspaketen, Zeit / Kostenplan,
Ressourcenplan, Aus- und Weiterbildungsplan)
 - Projektorganisation (Organisation, Team mit Kompetenzen und
Verantwortlichkeit)
 - Projektabsicherung (Risikomanagement – Planung, Qualitätsplan,
Änderungsmanagement)

Vorgehen:

- Deliverables; Arbeitspakete und Projektablauf planen
- pro Arbeitspaket einen Verantwortlichen bestimmen
- pro Arbeitspaket Ressourcen festlegen CMA, Räume techn. Infrastrukturen,
Spezialisten
- Reihenfolge der Arbeitspakete festlegen (möglichst konfliktfreier Plan)
- Termine und Ressourcen zuordnen
- vollständiger Projektplan liegt vor

- Grundlage: Erfahrungswerte
- 1. Erfahrung IS – Projekt
- Problem: Kommunikation- und Knowledge Management

- Beschaffung des Projekts: Infrastrukturen planen
- es kann zyklisch werden

8.4.3. Phase Projektdurchführung und –controlling

→ Ergebnisse (wesentlich)

- es existiert stets ein aktueller Projektplan
 - Projektstatus, Projektlogbuch (alle Planänderungen und Gründe)
 - Projektabstimmung
 - Liste offener Punkte
 - Liste von Änderungsanträgen
 - Analysen / Prognosen zu Risiken

→ Aufgaben / Vorgehen (wichtige)

- Projektaktivitäten werden veranlasst und dokumentiert (wöchentliche Meetings der Verantwortlichen der Arbeitspakete)
- Projektplan detaillieren und gegebenenfalls anpassen (wöchentliche Ausgaben pro Arbeitstag)
- Zielgruppen gerecht über den Stand des Projekts informieren
 - Projektteam
 - CB und AG
 - GF
 - „Öffentlichkeit“
- Controllboards durchführen
- Änderungsmanagement
- Team(weiter)entwicklung
- QS
- Projektmarketing

8.4.4. Projektabschluss

Ergebnisse

- unterschriebene Projektabnahme → Geld
- Feedback & Erfolgswertung → „NGK“ → Geld
- Wissenssicherung

Aufgaben / Vorgehen

- Projekt abnehmen lassen
- Risikobetrachtung abschließen
- Feedback einholen
- Projektwissen sichern
- Übergabe des Systems
- Projektabschluss Team → Freizeit

9. Qualitätsmanagement

Strukturqualität	Prozessqualität	Produktqualität
Image	Sicherheit	Kontrollen
Unternehmensstruktur	Flexibilität	Effizienz
Qualität	Kommunikation	Ergonomie
Leistungsstandards		Zuverlässigkeit

^- „Missionstatements“

Ablauf

Qualitätsplanung →	Qualitätssicherung →	Qualitätss
- Qualitätsziele des Projekts (siehe auch Konzeption, NFA)	- durchgeführte und protokollierte Qualitätsmassnahmen	- Qualitätsmassnahmen bewerten
- Qualitätsorganisation		- Arbeitsergebnisse ggf. freigeben, Korrekturmassnahmen veranlassen
- Qualitätsplan		- Qualitätsplan anpassen

9.2. Qualitätsmanagement auf Unternehmensebene

.
.
.

9.6. Anhang Kriterienkatalog zum Qualitätsmanagement bei Software

Qualitätsziel	Qualitätsmerkmal
Vollständigkeit	Abdeckungsgrad
Korrektheit	(Fehlerfreiheit)
Ergonomie	Bedienbarkeit, Erlernbarkeit, Verständlichkeit
Effizienz	Antwortzeit, Speichereffizienz, Laufzeiteffektivität, Zugriffseffektivität
Zuverlässigkeit	Antwortzeitzeit, Integrität, Reproduzierbarkeit, Stabilität, Genauigkeit
Sicherheit	Datenschutz
Portabilität	Kompatibilität, Geräteunabhängigkeit
Testbarkeit	Verständlichkeit, Offenheit
Änderbarkeit	Erweiterbarkeit, Strukturiertheit, Programmiermethodik
Wartbarkeit	Analysierbarkeit, Verständlichkeit der Software
Wiederverwendbarkeit	Schichten, Protokolle

10. Testen im IT – Entwicklungsprozess

10.1. Orientierung

10.1.1:

Der Leistungsumfang des IT – Systems gemäss der funktionalen und nichtfunktionalen Anforderungen ist geprüft

Die Qualität des IT – Systems ist geprüft

Der Testumfang ist stets transparent

Das System ist auch für weitere Releasezyklen gut und effizient testbar

Die Wiederverwendbarkeit von Komponenten ist gesichert (bei entsprechender Zielstellung des Auftragsnehmers)

10.1.2. Voraussetzungen

- Testverantwortlicher existiert
- enge Kommunikation zwischen Auftraggeber, Auftragnehmer, Testverantwortlichen über Testobjekte, Testfälle und Prioritäten
- für jeden Test sind die erwarteten Ergebnisse
- das System muss testbar sein → es gibt Deliverables, Konzept

10.1.3. Ergebnisse

- eine detaillierte Beschreibung der Testziele, Testszenarien, Testfälle
- eine Testperson
- Testumgebung mit Testsystem, Testdaten, Testprozeduren, ggfls. Automatisierungsmechanismen
- Testprotokolle und –anwendungen

10.2. Systematisches Testen

10.2.1. Begriffe

- Testfälle: Menge von Eingaben und geforderte Ergebnisse
- Testobjekte: Systemteile auf denen Testfälle ausgeführt werden sollen
- Testszenario: Menge von Testfällen

10.2.2. Testprozess

- | | |
|--------------------------------------|---------------------------------|
| 1. Testinitialisierung | → Testfälle |
| 2. Testplanung | → Testkonzept |
| 3. Testdurchführung / Testauswertung | → Testergebnisse und Soll / Ist |
| 4. Testabschluss | → Abdeckungsgrad |

Bsp: Abnahmetest

- Testinitialisierung erfolgt parallel zur Konzeption!
- Planung am Ende der Designphase
- Testdurchführung und Testauswertung und Testabschluss (Einführung)

Komponententest

- Testinitialisierung: am Ende des Komponentenentwurfs
- Testplanung: anschliessen
- Testdurchführung und Testauswertung und Testabschluss (Realisierung der Komponentenintegration)

Dokumententest (z.B. Analyse)

- alles in der Analysephase

Zur Spezifikation von Testfällen (hier Abnahmetest)
funktionale Anforderungen

(Sparzulage berechnen) ← exakte Beschreibung!

Berechnung der Sparzulage

- 0 Kinder → 0 %
- 1 & 2 Kinder → 10%
- 2 und mehr Kinder → 20%
- max. Einkommen 50000 Euro

Name: *Systemtest Sparzulage berechnen – Keine Bewilligung*

ID: T 4711

Ziel: Die Anforderung *Sparzulage berechnen* ist geprüft
Testfall keine Bewilligung

Zweck: Fachlicher Systemtest

Testobjekt: UseCase „Sparzulage berechnen“

Argumente: Einkommen 50001, 1 Kind

Sollergebnis: Sparzulage = 0 %

Umgebung: ...

Bedingung: ...

Beschreibung: ...

- 6 Testfälle

10.3. Testen im Entwicklungsprozess

Phase	Testobjekt	Testfall
Analyse	Prozeß „Einkommenssteuererklärung abwickeln“	- Angestellter mit Familie - Selbständige - Landwirte
Konzeption	UseCase „Sparzulage berechnen“	- kinderreiche Familien mit geringem EK - kinderlose Familien mit geringem EK
Design	Komponente „Prämienberechnung“ - Schnittstellentest	- berechne Sparzulage (Kunde: 4711 Jahr: 2000) - getSparzulage(...)
Realisierung	Klasse „Kunde“ - Methode	- getAnzahl Kinder - getEK
Einführung	Teilsystem	Testfälle analysieren

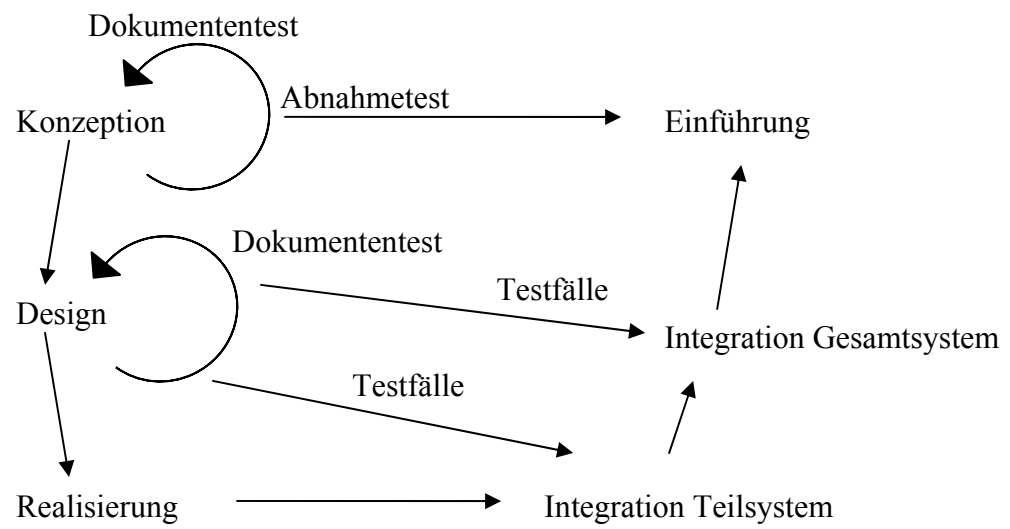


Bild: Systemtests

→ zusätzliche Tests: Benutzerhandbuch