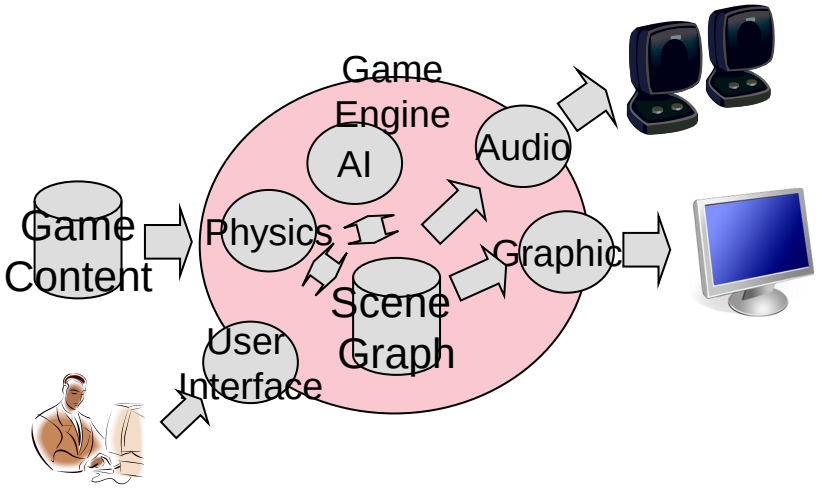
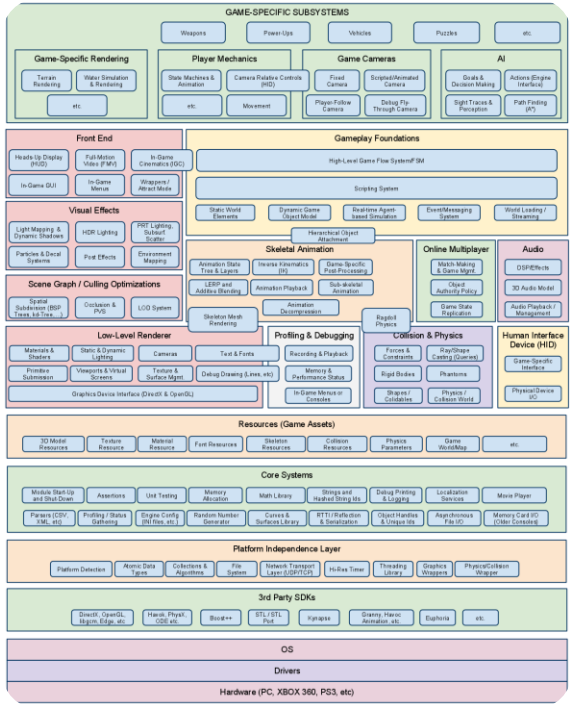
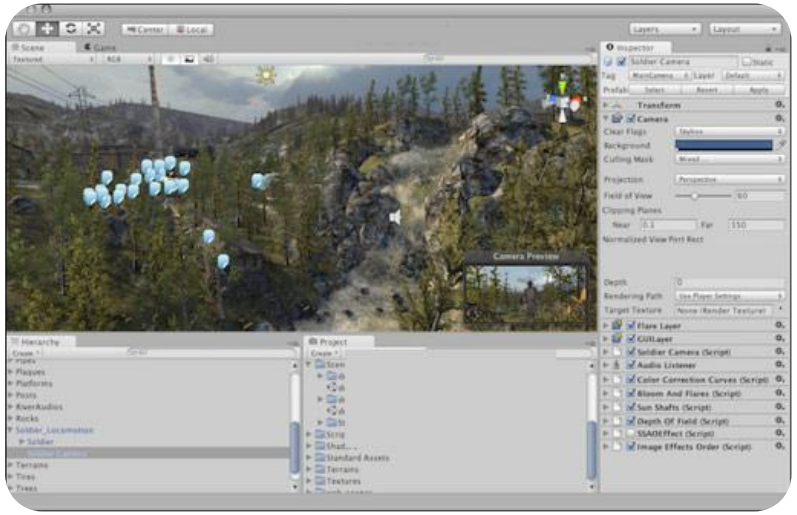
Kurzer Rückblick – Letzte Vorlesung







R. Weller Media Engineering WS 26/27 Requirements Engineering 1

Media Engineering
Requirements Engineering

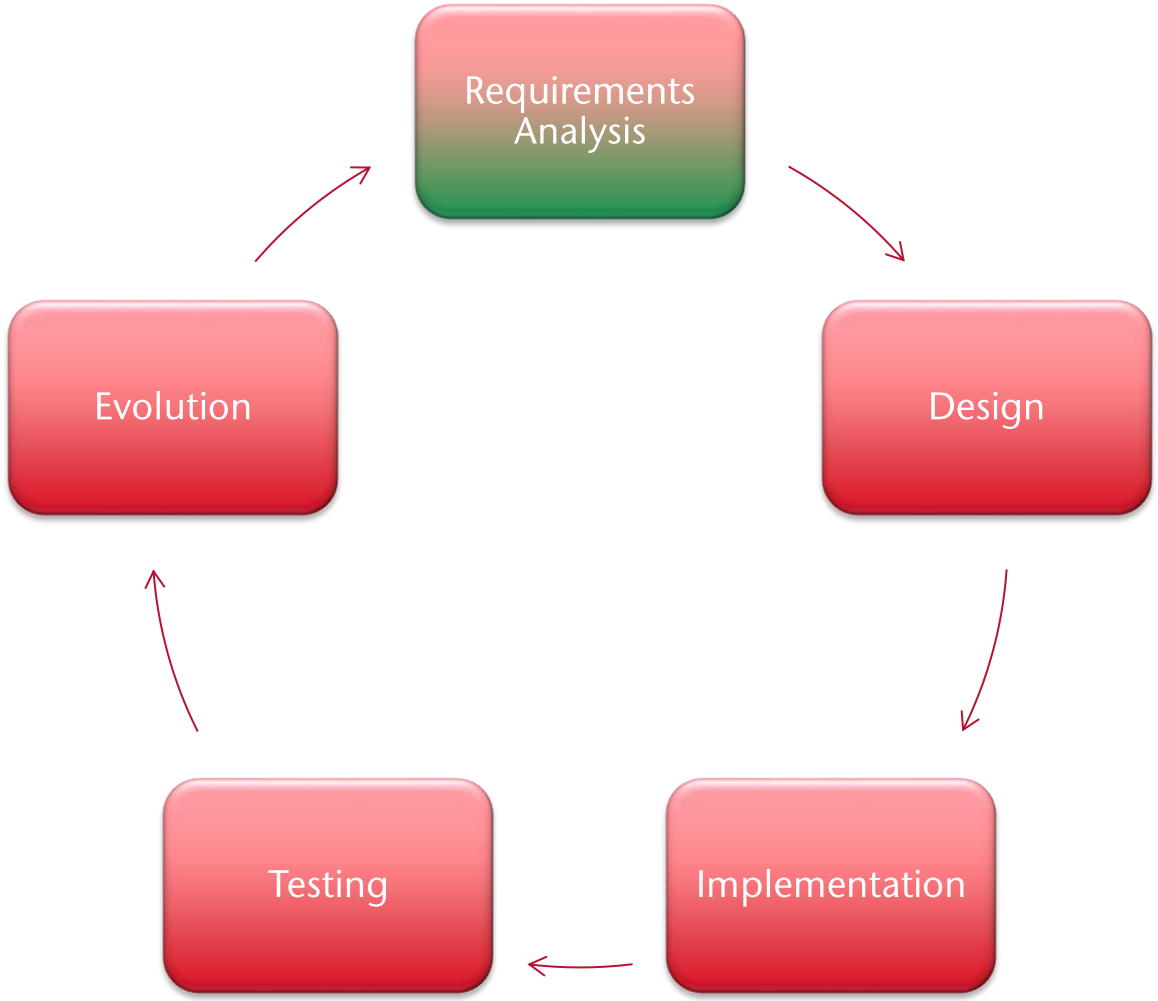


R. Weller

University of Bremen, Germany

cgvr.cs.uni-bremen.de

Der Software Development-Lifecycle



Übersicht

- Warum braucht man Anforderungen?
- Was sind überhaupt Anforderungen?
- Wie findet man Anforderungen?
- Wie hält man Anforderungen fest?

Zur Erinnerung: Concord (2024)



Bild: Sony Interactive Entertainment / Firewalk Studios

Beispiel: No Man’s Sky (2016)



Bild: Hello Games (Steam)

Stellenwert der Anforderungen

The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements … No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.

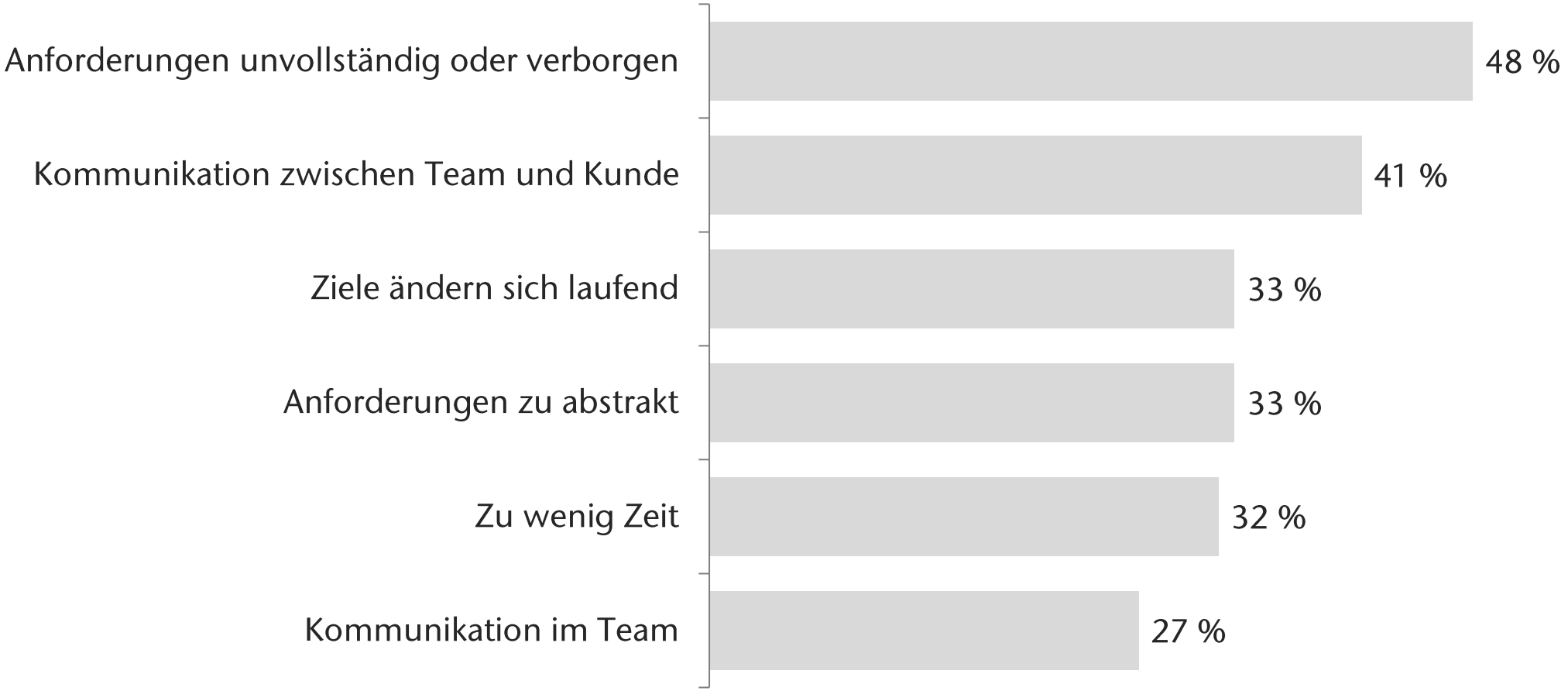
Fred Brooks,
1987

Kostenfortpflanzung eines Fehlers

		Zeitpunkt der Erkennung des Fehlers				
		Anforderungen	Architektur	Entwurf	Test	Release
Zeitpunkt der Einführung des Fehlers	Anforderungen	1x	3x	5-10x	10x	10-100x
	Architektur	-	1x	10x	15x	25-100x
	Implement.	-	-	1x	10x	10-25x

Woran hakt es in der Praxis?

Anteil der Unternehmen, die das Problem nennen



Méndez Fernández et al.: Naming the Pain in Requirements Engineering. Empirical Software Engineering 22, 2017 (228 Unternehmen, 10 Länder)



Was sind überhaupt Anforderungen?
Requirements

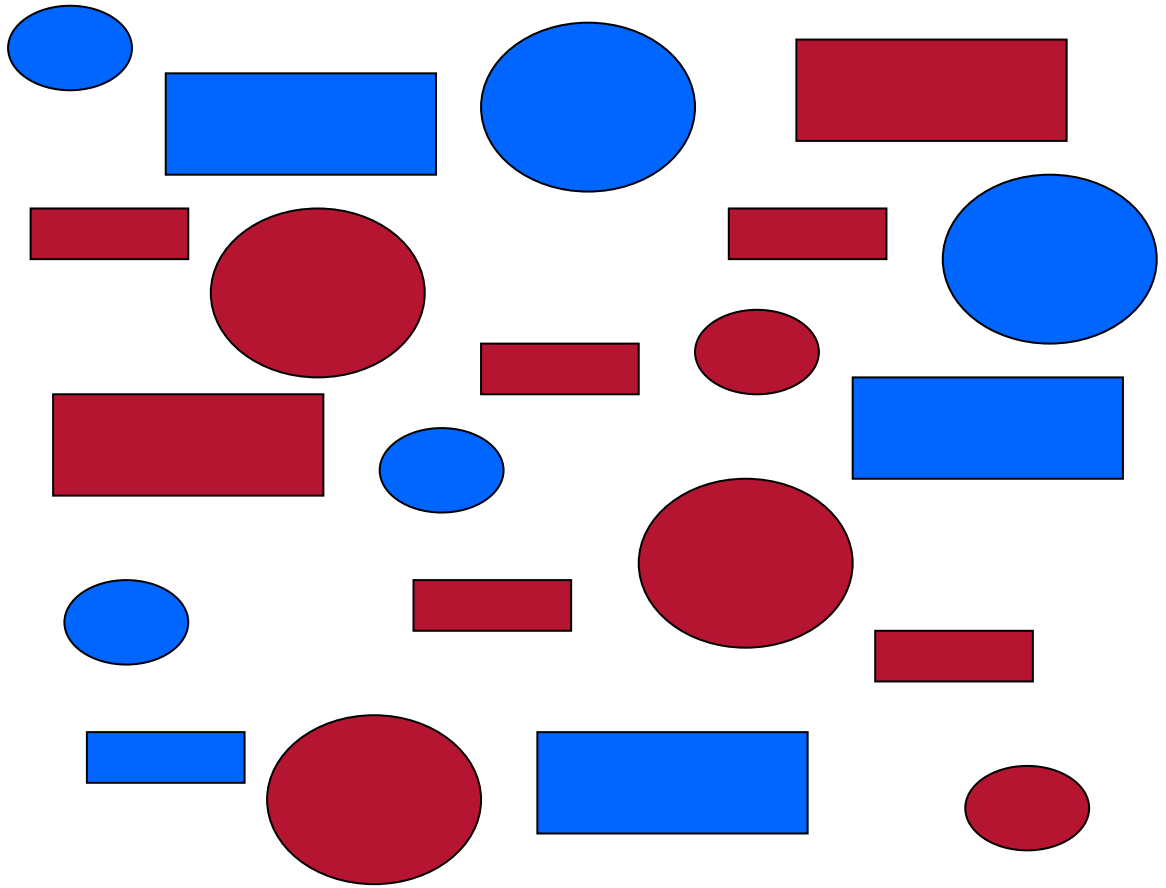


Definition (nach ISO/IEC/IEEE 24765)

Requirement

- (1) A condition or capability needed by a user to solve a problem or achieve an objective.
- (2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.
- (3) A documented representation of a condition or capability as in (1) or (2).

Welche Arten von Anforderungen gibt es?

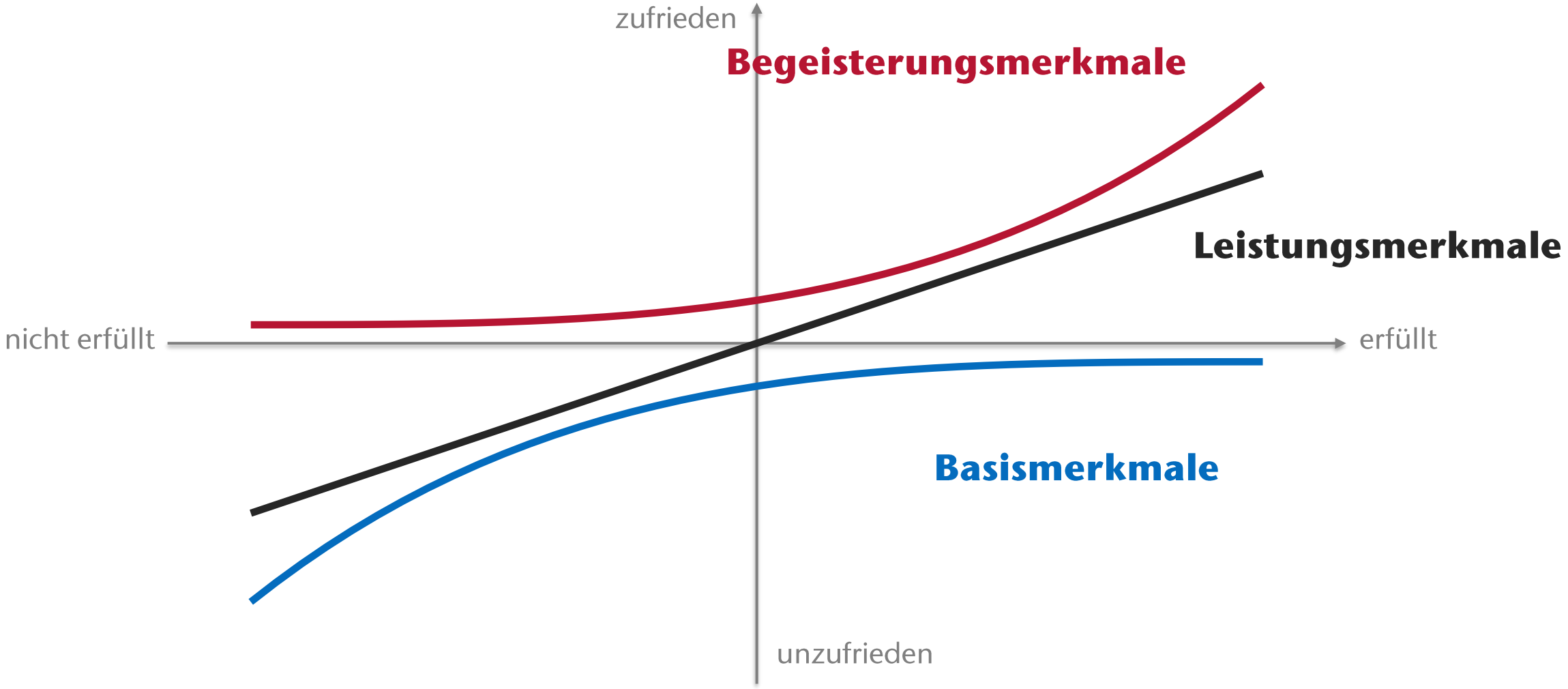


Einteilung nach Anforderungssteller

- Offene Anforderungen
- Latente Anforderungen
- Entwickler-Optionen



Das Kano-Modell



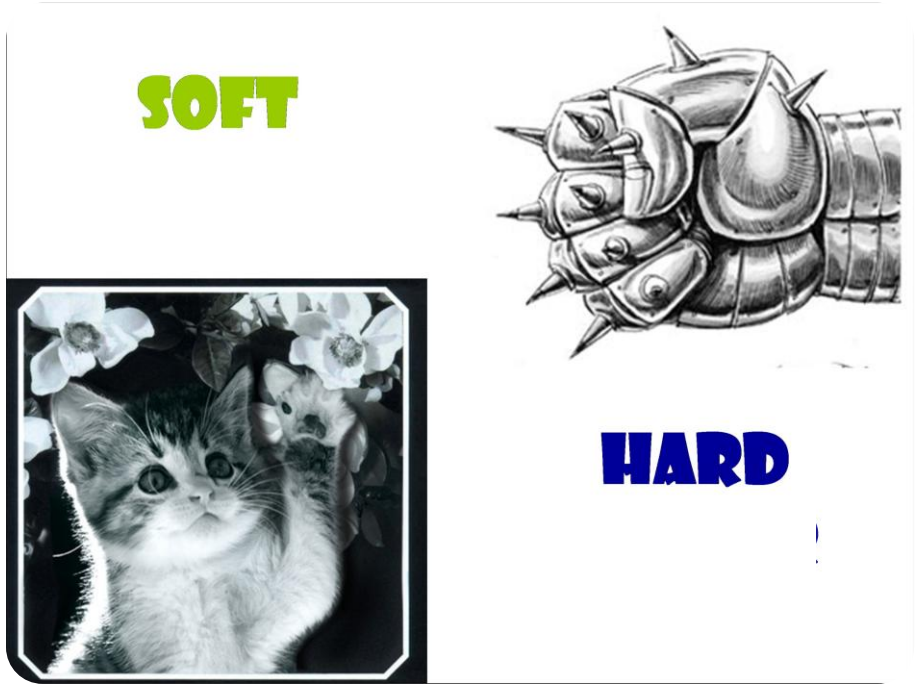
nach Kano et al.: Attractive Quality and Must-Be Quality, 1984

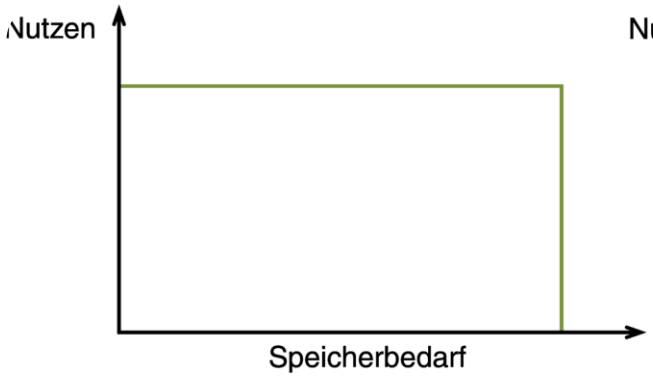
Wer ist überhaupt der Kunde? Stakeholder

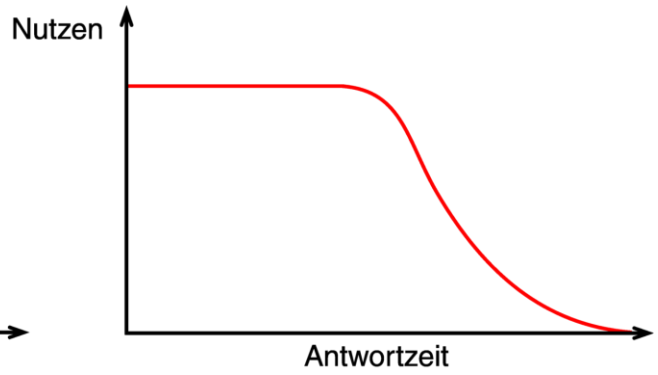
- Drei Kategorien von Benutzern (Eason, 1987):
  - Primäre: Verwenden das Produkt regelmäßig selbst
  - Sekundäre: Gelegenheits- oder indirekte Benutzer
  - Tertiäre: Von der Einführung des Produktes Betroffene
    - z.B. Verkäufer, Support



Einteilung nach Art der Anforderung: Hart vs. weich

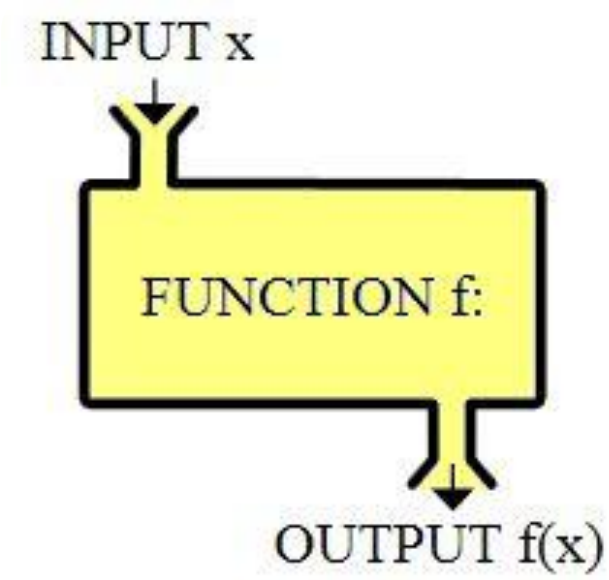




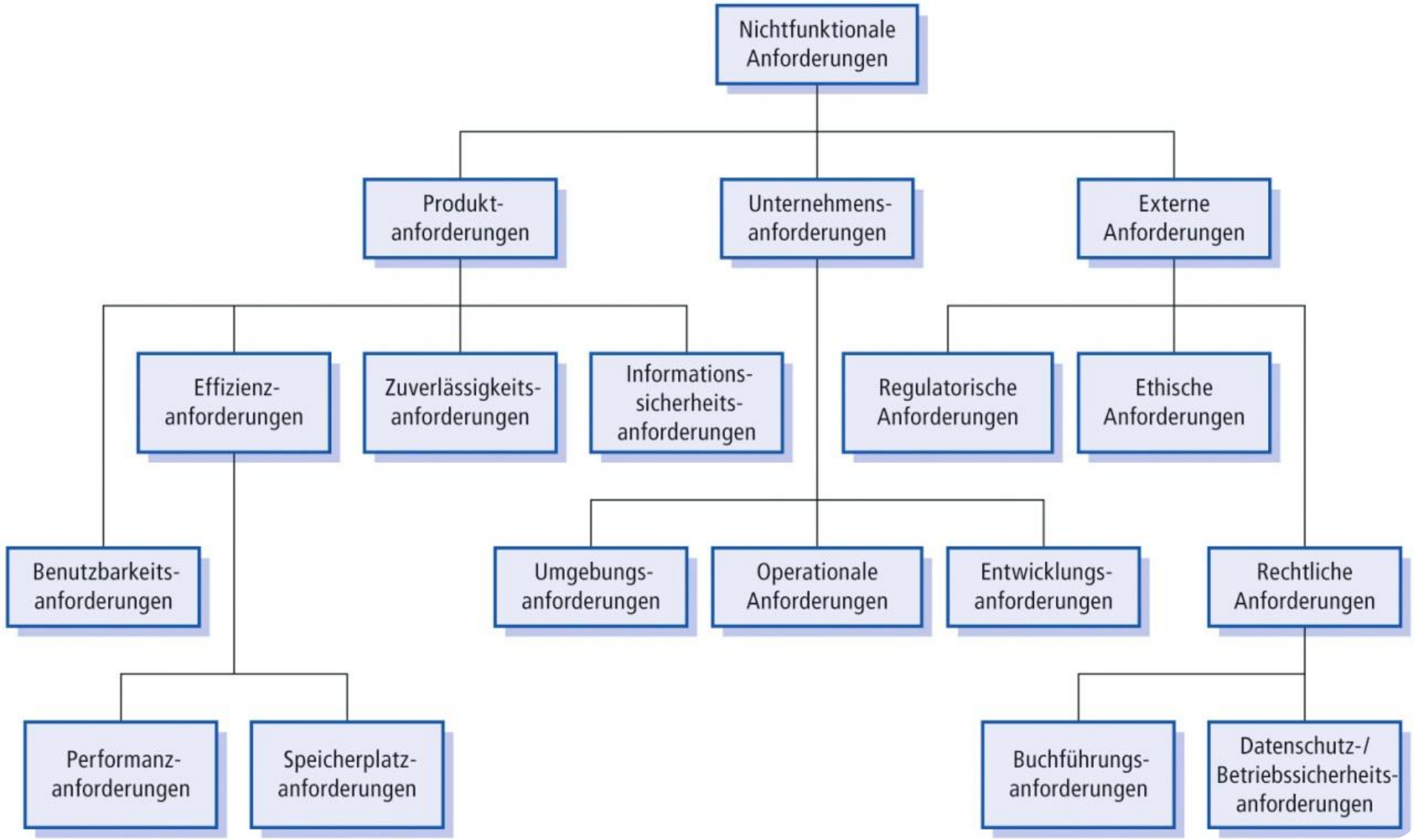


Weitere Kategorien von Anforderungen: Funktional vs. Nichtfunktional

- Funktion der Software = Beziehung zwischen Ein- und Ausgaben
- funktionale Anforderungen
- nichtfunktionale Anforderungen (non-functional requirements, NFRs)



Nichtfunktionale Anforderungen

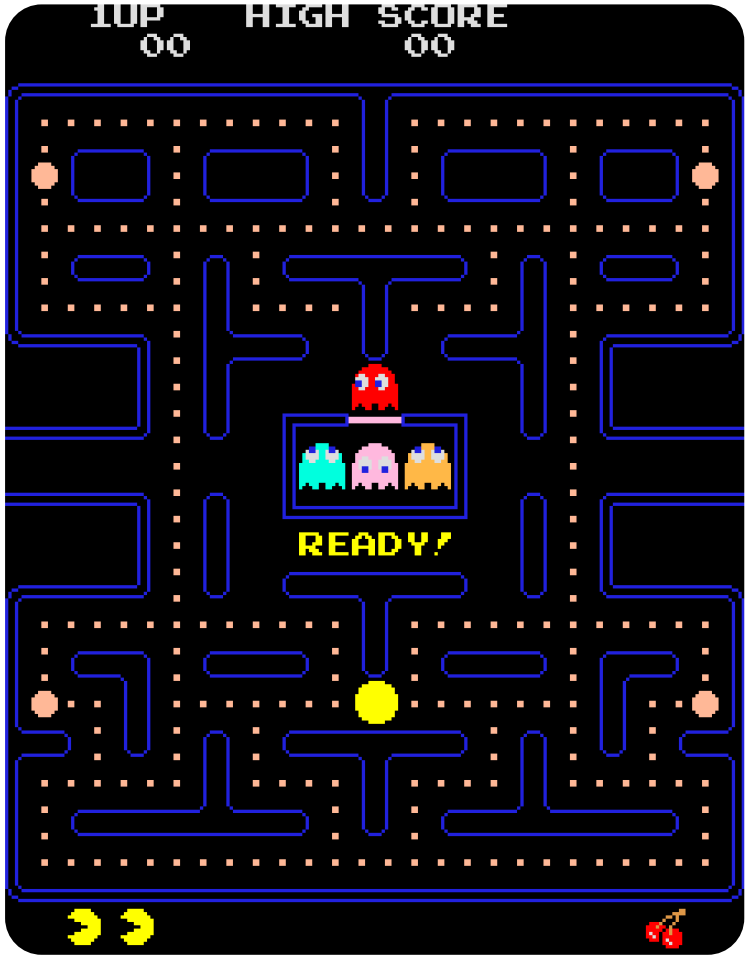


Richtlinien für Barrierefreiheit

- Web Content Accessibility Guidelines (WCAG 2.2)
  - Perceivable
  - Operable
  - Understandable
  - Robust



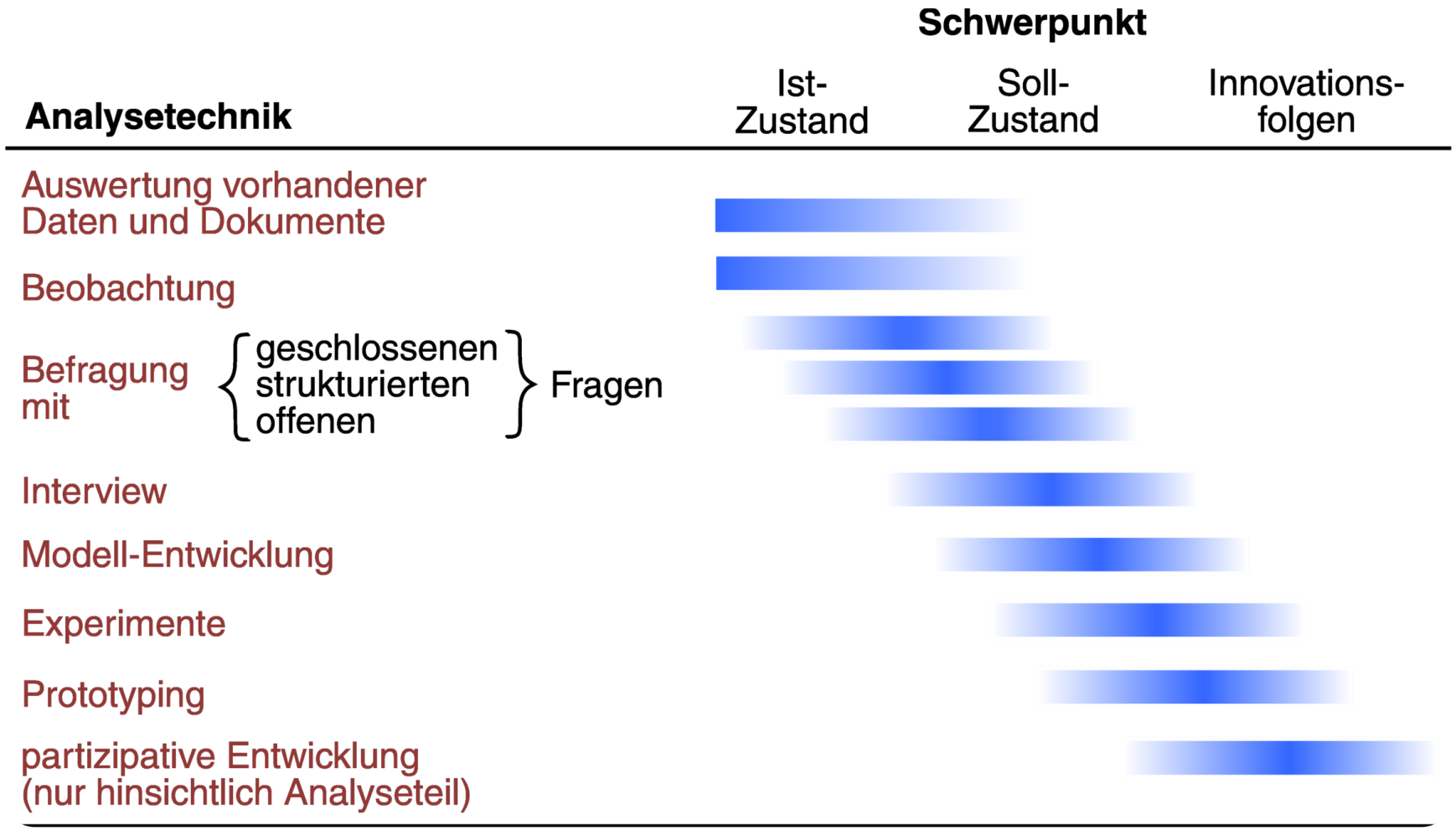
Was heißt das für Ihr Spiel?



Wie findet man Anforderungen?
Requirements Analysis



Techniken der Analyse



Probleme beim Fragen: Beispiel



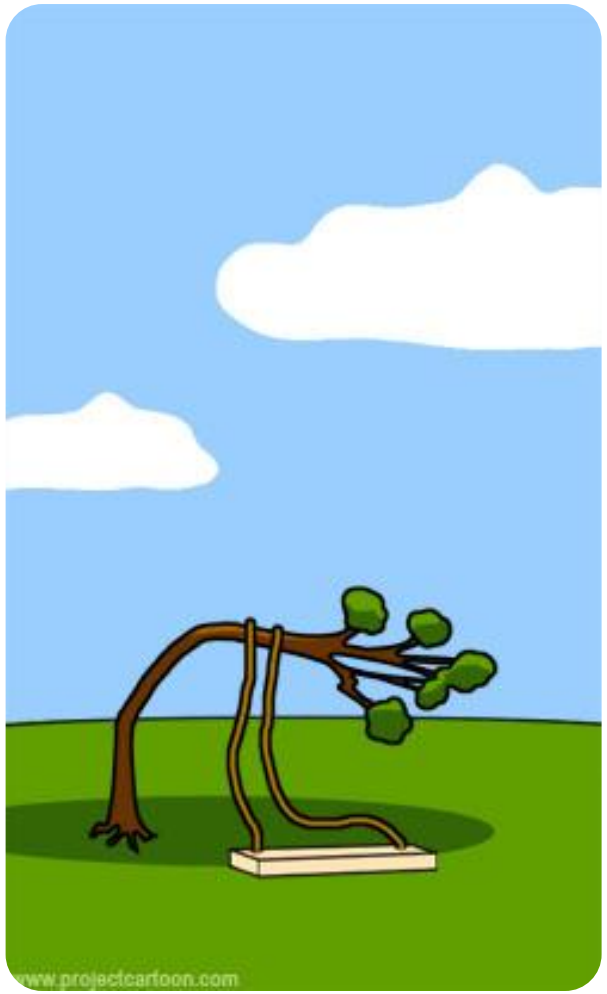
Systematisches Fragen

- „offene“ Fragen
- „geschlossene“ Fragen

Zustimmung			Ablehnung		
stark	mittel	schwach	stark	mittel	schwach
+3	+2	+1	-1	-2	-3

- Vollstandardisiert, teilstandardisiert, nicht standardisiert

Weitere Probleme beim Fragen



Wissen wir, was Nutzer wollen?

Neue Funktionen im Test mit echten Nutzern (Microsoft)



ein Drittel | ein Drittel | ein Drittel

Kohavi, Crook, Longbotham: Online Experimentation at Microsoft, 2009

Szenarien und Use Cases

Definition (Pohl):

Ein *Szenario* beschreibt ein konkretes Beispiel für die Erfüllung bzw. Nichterfüllung eines oder mehrerer Ziele. Es konkretisiert dadurch eines oder mehrere Ziele. Ein Szenario enthält typischerweise eine Folge von Interaktionsschritten und setzt diese in Bezug zum Systemkontext.

Beispiel „Kraftpille“

- Der Spieler steuert Pac-Man durch das Labyrinth. Pac-Man frisst eine Kraftpille. Das System färbt alle Geister blau und kehrt ihre Laufrichtung um. Der Spieler steuert Pac-Man auf einen blauen Geist. Das System erkennt die Berührung, entfernt den Geist und zeigt 200 Punkte an. Die Augen des Geistes kehren ins Geisterhaus zurück. Nach einigen Sekunden beginnen die übrigen Geister zu blinken. Das System färbt sie wieder ein, und sie jagen Pac-Man erneut.

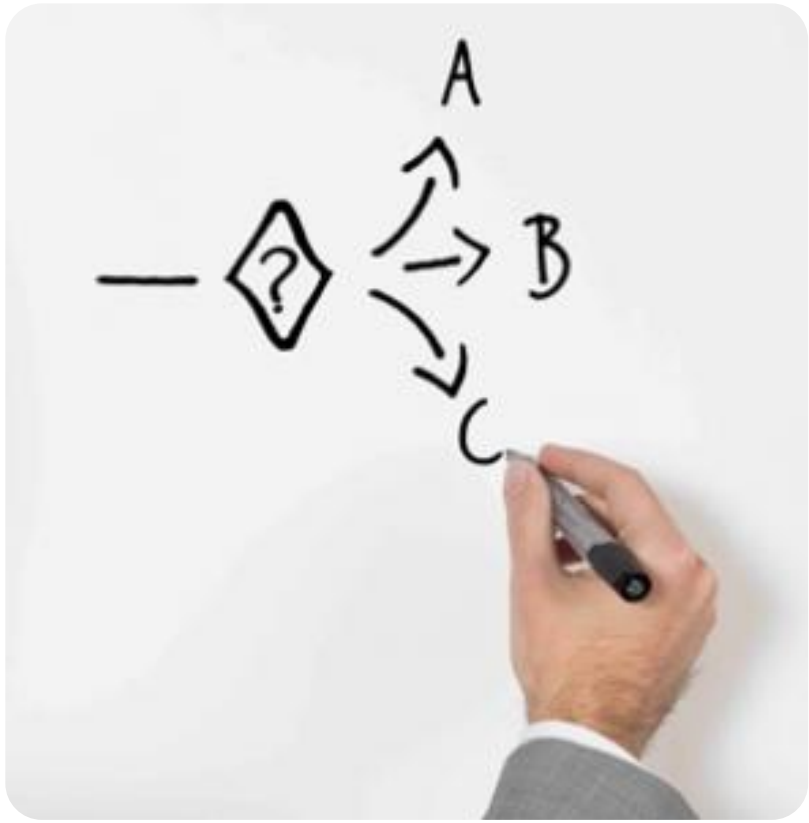
Szenarientypen

- **Positiv**
- **Negativ**
- **Missbrauchsszenarien**

- **Deskriptiv**
- **Explorativ**
- **Erklärend**

- **Systemintern**
- **Interaktiv**
- **Kontextszenarien**

- **Hauptszenario**
- **Alternativszenario**
- **Ausnahmeszenario**



Szenario als strukturierte Folge von Schritten

1. Der Spieler wählt im Hauptmenü „Spiel starten“.
2. Das System lädt das Labyrinth.
3. Das System setzt Pac-Man und die Geister auf ihre Startpositionen.
4. Das System zeigt „Ready!“ und spielt die Startmelodie.
5. Der Spieler drückt eine Richtungstaste.
6. Das System bewegt Pac-Man in die gewählte Richtung.
7. Der Spieler steuert Pac-Man auf eine Münze.
8. Das System entfernt die Münze und erhöht den Punktestand um 10.
9. Das System spielt das Fressgeräusch.
10. Der Spieler steuert Pac-Man auf einen Geist.
11. Das System zieht ein Leben ab.
12. Das System setzt alle Figuren auf ihre Startpositionen zurück.

Bremen



CG VR

Wie dokumentiert man Anforderungen?
Requirements Specification



Definitionen: Lasten- / Pflichtenheft

Lastenheft (DIN 69901 – 5):

- Vom Auftraggeber festgelegte Gesamtheit der Forderungen an die Lieferung und Leistungen eines Auftragnehmers innerhalb eines Auftrags.

Pflichtenheft (DIN 69901 – 5):

- Das vom Auftragnehmer erarbeitete Realisierungsvorgaben aufgrund der Umsetzung des vom Auftraggeber vorgegebenen Lastenhefts

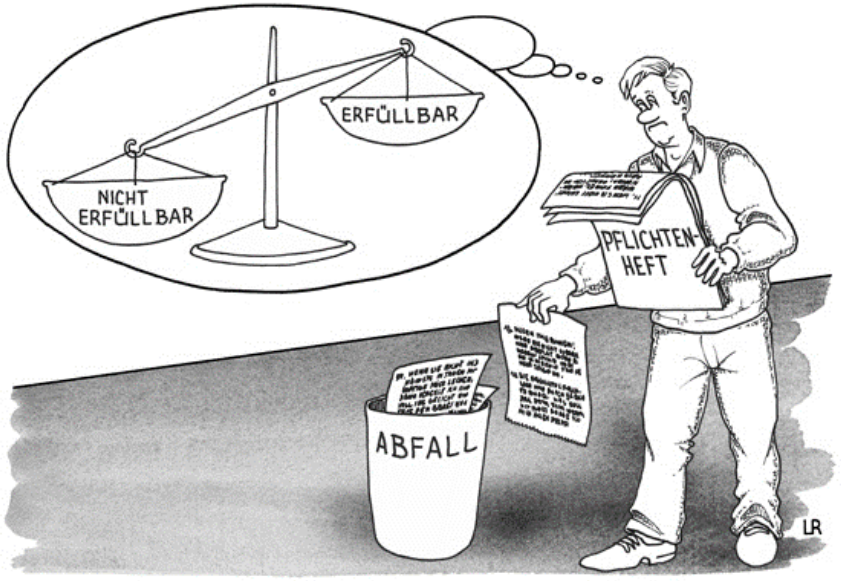
Anforderungsspezifikation (Pohl, Rupp, 2021)

- Eine systematisch dargestellte Sammlung von Anforderungen (typischerweise für ein System oder eine Komponente), die vorgegebenen Kriterien genügt.

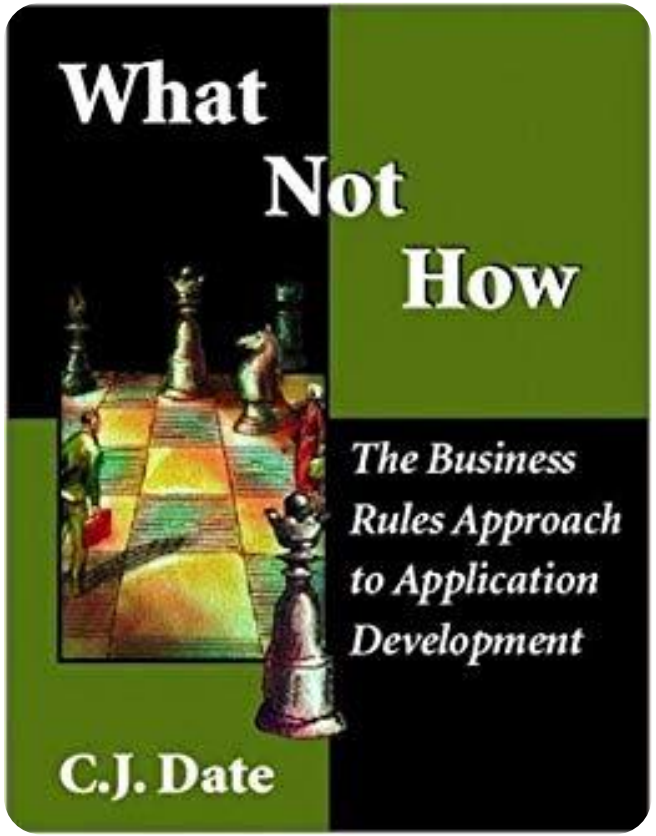
Angestrebte Eigenschaften des Pflichtenhefts

- Das Pflichtenheft ist im Idealfall inhaltlich

1. zutreffend
2. vollständig
3. widerspruchsfrei
4. neutral
5. nachvollziehbar
6. objektivierbar



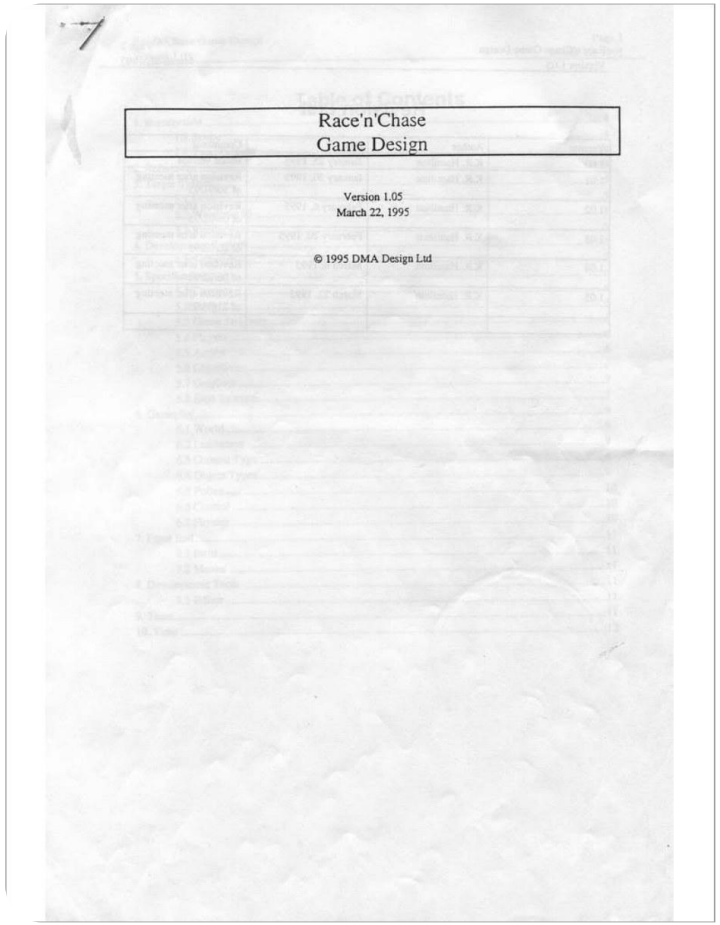
Pflichtenheft vs Entwurf



Lastenheft / Pflichtenheft: möglicher Aufbau

0. Administrative Daten: von wem, wann genehmigt, ...
1. Zielbestimmung und Zielgruppen
2. Funktionale Anforderungen
3. Nichtfunktionale Anforderungen
4. Lieferumfang
5. Abnahmekriterien
6. Anhänge (insbesondere Begriffslexikon)

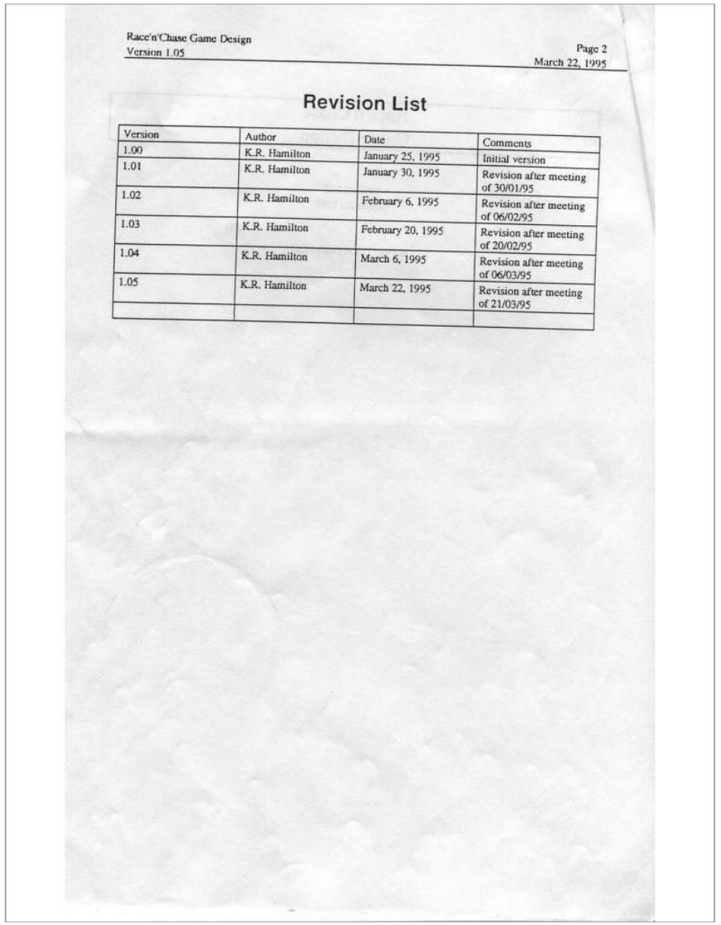
In der Spielebranche: das Game Design Document



Race'n'Chase
Game Design

Version 1.05
March 22, 1995

© 1995 DMA Design Ltd



Revision List

Version	Author	Date	Comments
1.00	K.R. Hamilton	January 25, 1995	Initial version
1.01	K.R. Hamilton	January 30, 1995	Revision after meeting of 30/01/95
1.02	K.R. Hamilton	February 6, 1995	Revision after meeting of 06/02/95
1.03	K.R. Hamilton	February 20, 1995	Revision after meeting of 20/02/95
1.04	K.R. Hamilton	March 6, 1995	Revision after meeting of 06/03/95
1.05	K.R. Hamilton	March 22, 1995	Revision after meeting of 21/03/95

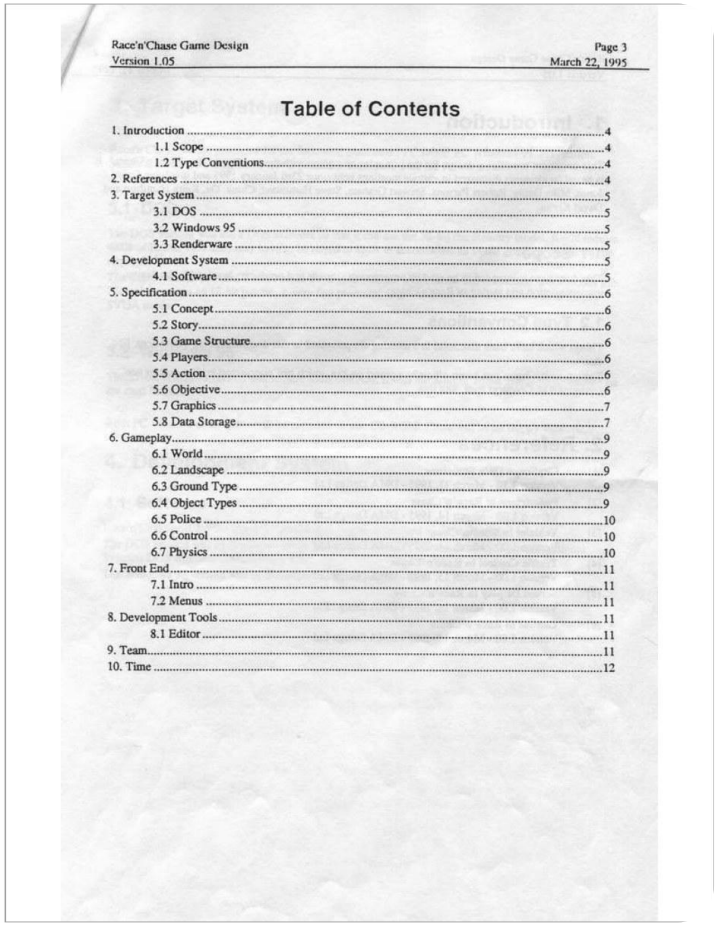


Table of Contents

1. Introduction 4
1.1 Scope 4
1.2 Type Conventions 4
2. References 4
3. Target System 5
3.1 DOS 5
3.2 Windows 95 5
3.3 Renderware 5
4. Development System 5
4.1 Software 5
5. Specification 6
5.1 Concept 6
5.2 Story 6
5.3 Game Structure 6
5.4 Players 6
5.5 Action 6
5.6 Objective 6
5.7 Graphics 7
5.8 Data Storage 7
6. Gameplay 9
6.1 World 9
6.2 Landscape 9
6.3 Ground Type 9
6.4 Object Types 9
6.5 Police 10
6.6 Control 10
6.7 Physics 10
7. Front End 11
7.1 Intro 11
7.2 Menus 11
8. Development Tools 11
8.1 Editor 11
9. Team 11
10. Time 12

DMA Design: Race'n'Chase Game Design, Version 1.05, 22. März 1995 (später: Grand Theft Auto)

User Stories

Als Spieler
möchte ich nach einer Kraftpille Geister fressen können,
um mich für kurze Zeit wehren zu können.

Akzeptanzkriterien

✓ Alle Geister werden blau und kehren um.

✓ Ein gefressener Geist bringt 200 Punkte.

✓ Nach sechs Sekunden jagen die Geister wieder.

Sieben Regeln für die Formulierung (nach Pohl)

1. Formulieren Sie die Ziele kurz und prägnant!
2. Verwenden Sie Aktivformulierungen!
3. Formulieren Sie überprüfbare Ziele!
4. Verfeinern Sie nicht überprüfbare Ziele!
5. Stellen Sie den Mehrwert eines Ziels dar!
6. Geben Sie eine Begründung für das Ziel an!
7. Formulieren Sie deklarativ, d.h. vermeiden Sie die Beschreibung von Lösungsansätzen!

1. Prägnant
2. Aktiv
3. Überprüfbar
4. Verfeinerbar
5. Wertschöpfend
6. Begründbar
7. Deklarativ

Überprüfbare, quantitative Anforderungen

- Schlecht: Das System soll besser sein als das Vorgängersystem.

1. Prägnant
2. Aktiv
3. Überprüfbar
4. Verfeinerbar
5. Wertschöpfend
6. Begründbar
7. Deklarativ

- Besser: Das System soll folgende Verbesserungen gegenüber dem System xy bieten:
  - ...
  - ...

Verfeinerung von Anforderungen

1. Prägnant
2. Aktiv
3. Überprüfbar
4. Verfeinerbar
5. Wertschöpfend
6. Begründbar
7. Deklarativ

- Schlecht: Die Benutzung des Systems soll selbsterklärend sein

- Besser: Das System soll selbsterklärend sein, d.h. ein durchschnittlicher Nutzer soll durchschnittlich nach 2 Min. folgende Funktionen aufrufen können: ...
- Das System soll selbsterklärend sein, d.h. den Vorgaben nach W3C... in Bezug auf ... folgen

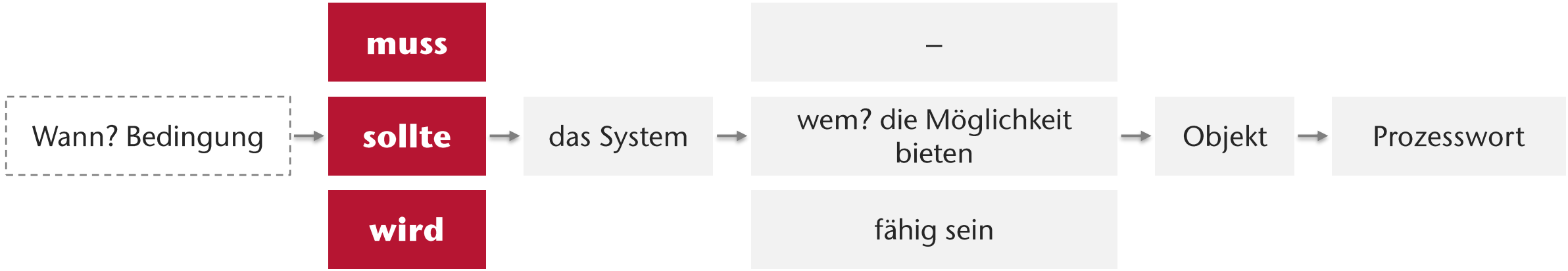
Keine Lösungsvorwegnahme

- Schlecht: Durch komprimierte Datenübertragung im Cache soll das geplante System um 10% kürzere Antwortzeiten aufweisen

1. Prägnant
2. Aktiv
3. Überprüfbar
4. Verfeinerbar
5. Wertschöpfend
6. Begründbar
7. Deklarativ

  - Besser: Das System soll um 10% kürzere Antwortzeiten aufweisen als System xy.

Eine Schablone für Anforderungssätze



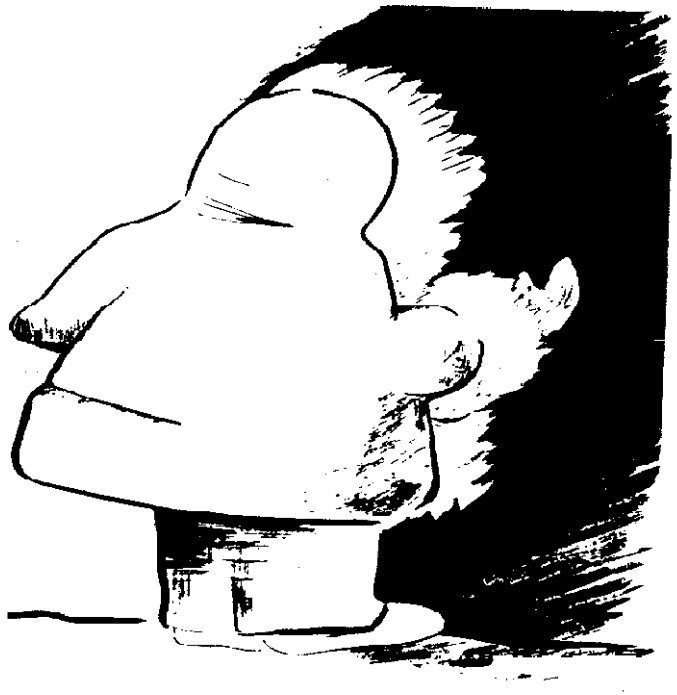
Nach dem Fressen einer Kraftpille **muss** das Spiel dem Spieler sechs Sekunden lang die Möglichkeit bieten, Geister zu fressen.

nach Rupp & die SOPHISTen: Requirements-Engineering und -Management

Was zuerst? Priorisieren

Must	Should	Could	Won't
muss sein	sollte sein	wenn Zeit bleibt	diesmal nicht

Das Begriffslexikon



- Beispiel: Pac-Man
  - „Geist“, „Kraftpille“, „Leben“, „Level“

Informationen im Begriffslexikon

a) Begriff und Synonyma (im Sinne der Spezifikation)

b) Bedeutung (Definition, Erklärung)

c) Abgrenzung (wo ist dieser Begriff nicht anzuwenden?)

d) Gültigkeit (zeitlich, räumlich, sonst)

e) Fragen der Bezeichnung, Eindeutigkeit u. A.

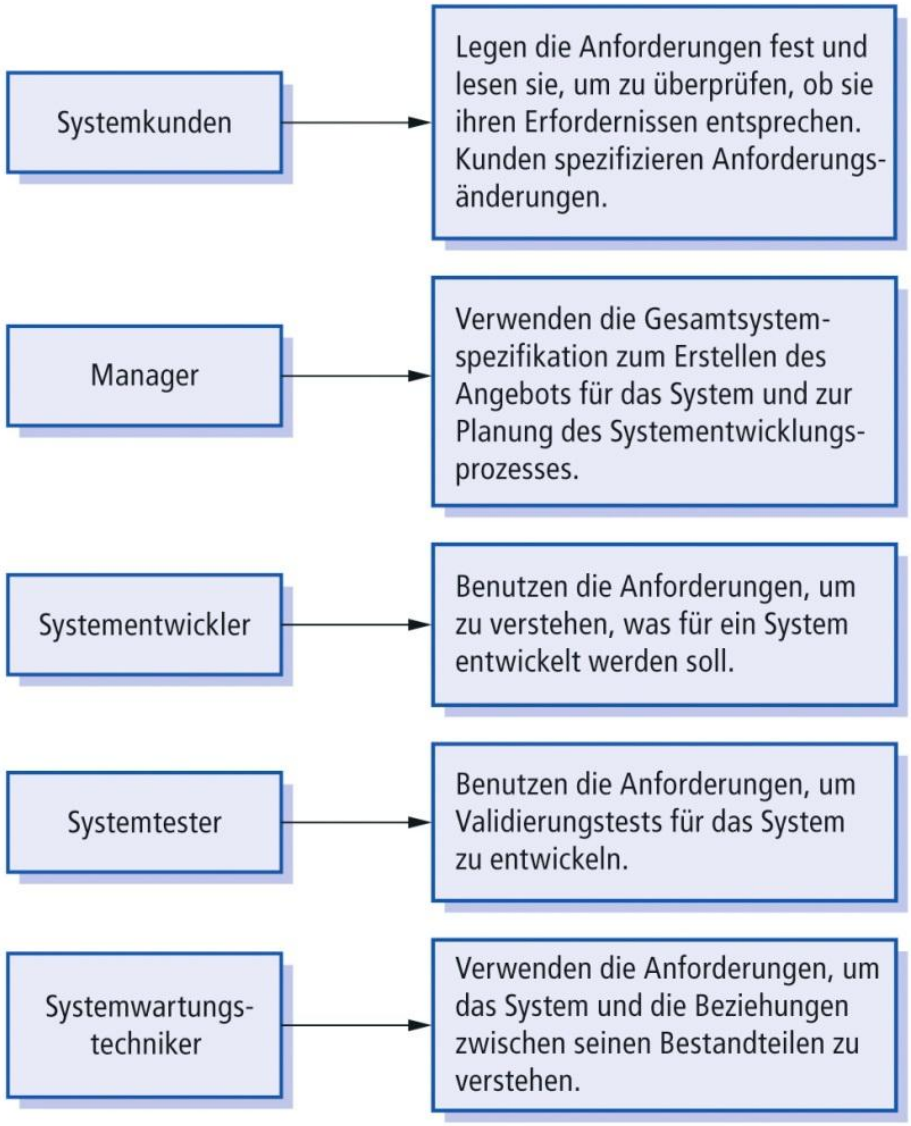
f) Unklarheiten, die noch nicht beseitigt werden konnten

g) verwandte Begriffe (Querverweise)

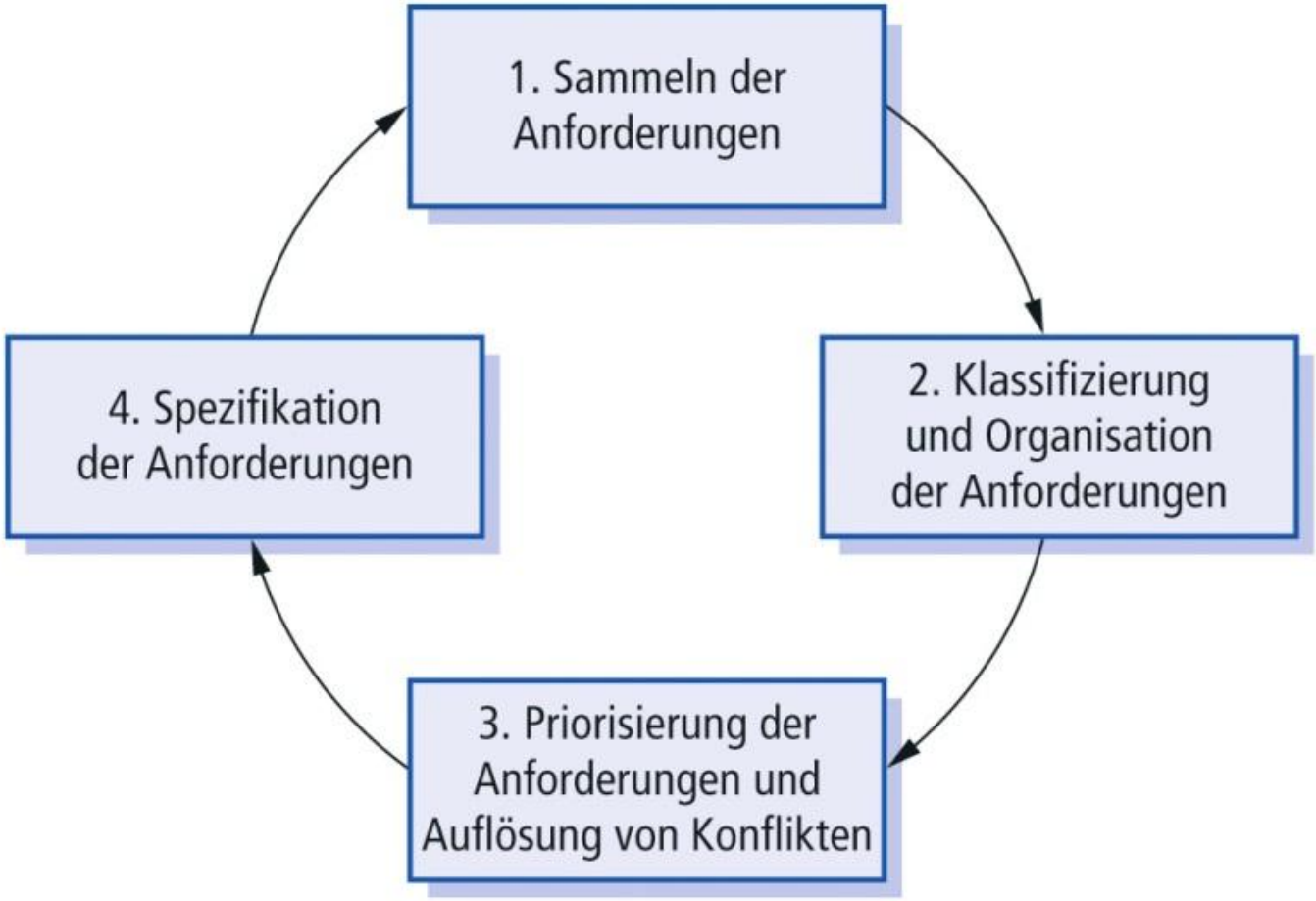
Beispiel

Begriff	**Geist, synonym Gegner, Monster**
Bedeutung	Eine vom Spiel gesteuerte Figur, die Pac-Man verfolgt und ihn bei Berührung ein Leben kostet.
Abgrenzung	Die Augen eines gefressenen Geistes auf dem Rückweg ins Geisterhaus sind im Sinne dieses Spiels kein Geist; sie können Pac-Man nicht fangen.
Gültigkeit	Ein Geist entsteht beim Start eines Levels im Geisterhaus und existiert bis zum Ende des Levels. Wird er gefressen, entsteht er nach der Rückkehr der Augen ins Geisterhaus neu.
Bezeichnung	Ein Geist ist durch seine Farbe eindeutig bestimmt (rot, rosa, türkis, orange).
Unklarheiten	Es ist noch ungeklärt, ob sich zwei Geister gleichzeitig auf demselben Feld aufhalten dürfen.
Querverweise	Kraftpille, Geisterhaus, Leben

Wer liest überhaupt das Pflichtenheft?



Requirements-Engineering



Übungsfrage 1/3

„Die Geister sollten eventuell schneller werden, wenn es schwieriger wird.“

?

Nennen Sie zwei Mängel dieser Anforderung und formulieren Sie sie besser.

1 Eine Minute allein nachdenken

2 Stichworte notieren

3 Antworten sammeln, dann Auflösung

Übungsfrage 2/3

Microsoft testet neue Funktionen an echten Nutzern, bevor sie eingeführt werden.

? **Was folgt aus dem Ergebnis für Ihr Spiel? Eine Antwort ist richtig.**

A Fachleute wissen in aller Regel, was Nutzer wollen; Tests sind meist überflüssig.

B Nur etwa ein Drittel der Ideen wirkt wie gedacht – also ausprobieren und beobachten statt nur fragen.

C Neue Funktionen verschlechtern fast immer das Produkt.

D Man sollte Nutzer gar nicht mehr fragen, sondern nur noch messen.

1 Eine Minute allein nachdenken **2** Stichworte notieren **3** Antworten sammeln, dann Auflösung

Übungsfrage 3/3

Aus Kapitel 2: Game Engines

Ein Ego-Shooter soll stabil mit 60 Bildern pro Sekunde laufen.

?

Funktionale oder nichtfunktionale Anforderung? Welches Merkmal im Kano-Modell – und was sagt die Studie aus Kapitel 2 dazu?

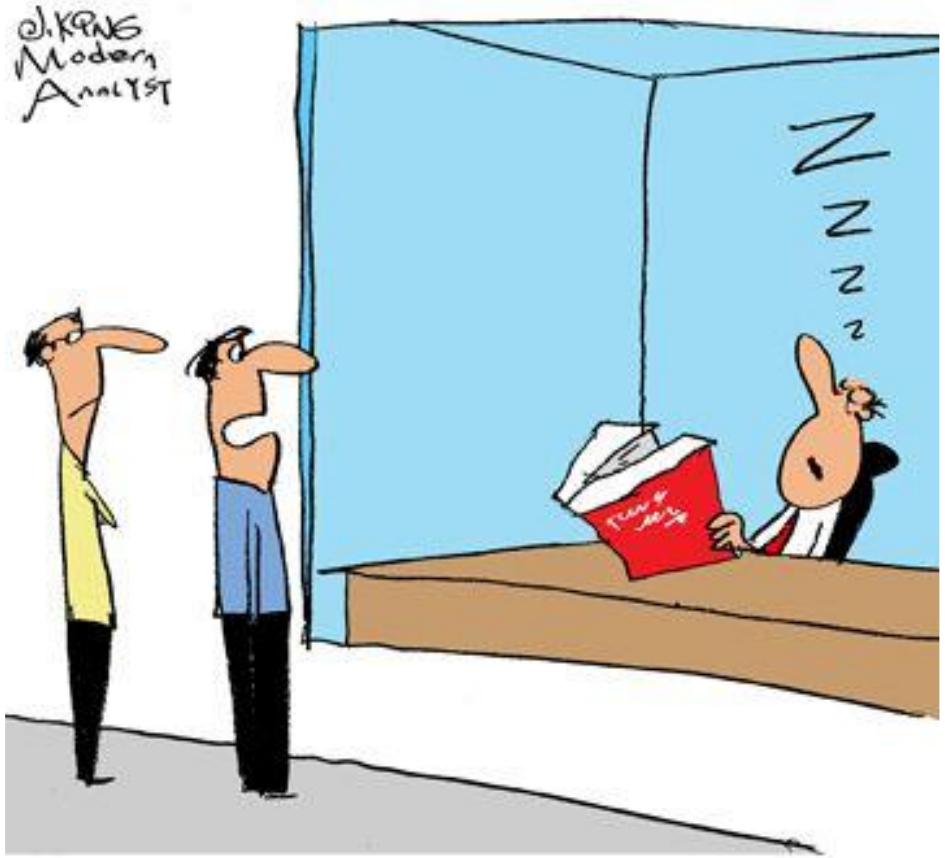
1 Eine Minute allein nachdenken

2 Stichworte notieren

3 Antworten sammeln, dann Auflösung

Lessons Learned

- ✓ Fehler in den Anforderungen sind die teuersten, weil alles Spätere auf ihnen aufbaut.
- ✓ Kunden sagen nicht alles: Selbstverständliches muss man gezielt erfragen, beobachten oder ausprobieren.
- ✓ Das Pflichtenheft sagt, WAS das System leisten soll, nicht WIE – in aktiven, prüfbaren, verbindlichen Sätzen. Und es hält fest, was nicht kommt.



"Based on our tests, the business stakeholders fall asleep around page 37 of the Functional Requirements Specification. Put the Issues Section on page 40."