



Media Engineering Requirements Engineering

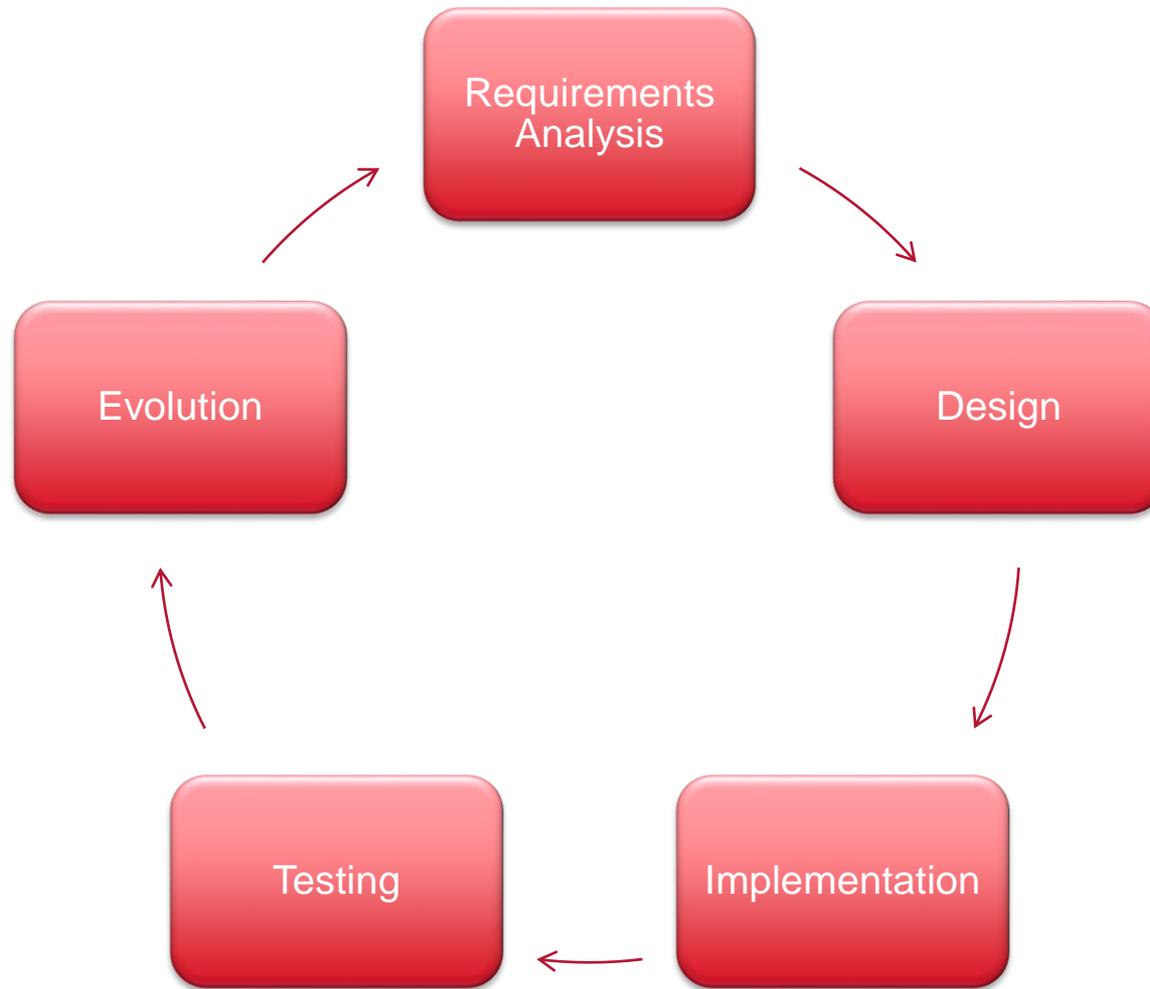


R. Weller

University of Bremen, Germany

cgvr.cs.uni-bremen.de

Der Software Development-Lifecycle



*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 as difficult to rectify later.*

Fred Brooks,
1987

- Die Anforderungen des Kunden an die Software sind **die wichtigsten Informationen in einem Software-Projekt.**

| | | 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 |

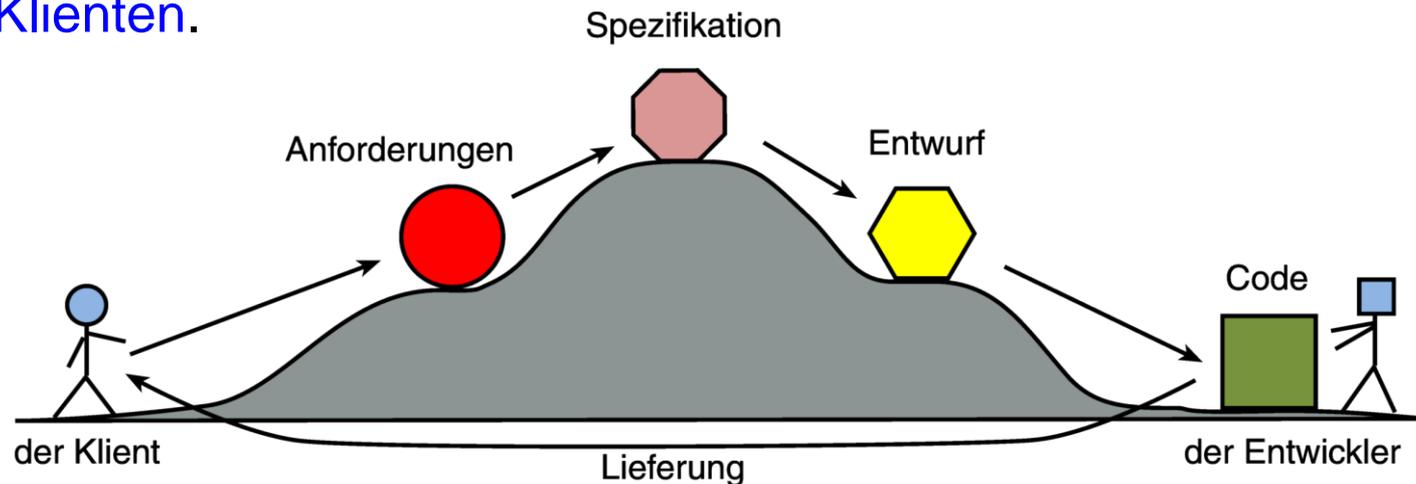
McConnell, Steve (2004). *Code Complete* (2nd ed.)

- Kosten für die Behebung von Fehlern abhängig von ihrer Verweildauer in der Software
- Fehler möglichst frühzeitig erkennen
- In allen Phasen aktiv gegen Fehler vorgehen
- Richtig betriebenes Requirements Engineering ist **wirtschaftlich**
- Analysten berichten, dass 71% aller IT-Projekte scheitern wegen einer schlechten Anforderungsanalyse

- Werden die Erwartungen, die Anforderungen des Kunden, nicht vollständig und präzise erfasst, ist damit zu rechnen, dass das entwickelte Produkt die Anforderungen nicht (vollständig) erfüllt.
- Die vollständige und präzise Erfassung der Anforderungen ist die **allerwichtigste technische Voraussetzung** für eine erfolgreiche Software-Entwicklung.



- Die Anforderungen werden **erhoben**, in der Spezifikation **formuliert**, **geprüft** und anschließend in den Entwurf **umgesetzt**. Schließlich wird **implementiert**, auf verschiedenen Ebenen **geprüft** und **korrigiert**. Das Resultat geht **zurück an den Klienten**.



- Der Klient bekommt nur dann, was er haben will, wenn seine Anforderungen **sorgfältig und vollständig** erhoben und unterwegs **nicht verfälscht** wurden.



Was sind überhaupt Anforderungen? Requirements



Definitionen (Nach IEEE Std 610.12-1990)

■ 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).

■ **Produktanforderungen**

- Beschreiben, **was** verschiedene Benutzer mit dem Produkt machen können
- Betrachten Softwareprodukt oder Dienst innerhalb der Umgebung, in der das System einmal arbeiten muss

■ **Marktanforderungen**

- Definieren, warum ein Projekt durchgeführt werden soll
- Enthalten oft Priorisierung aus betriebswirtschaftlicher Sicht

■ Qualitätsanforderungen

- Beschreiben eine qualitative Eigenschaft, die das System oder eine Funktion aufweisen müssen
- Sie ergänzen die Produktanforderungen: Beispiele sind Zuverlässigkeit, Verfügbarkeit, Wartbarkeit
- Sind schwer zu spezifizieren und zu testen
- Schwierige und oft unzureichende Diagnose und Fehlermanagement

■ Randbedingungen

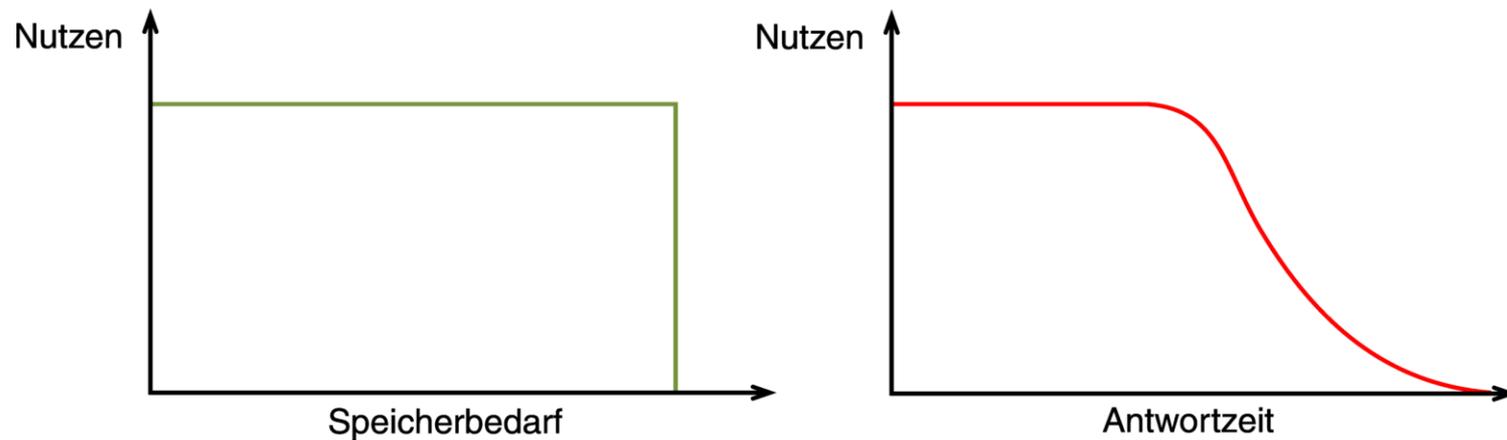
- Eine organisatorische oder technische Anforderung, die die Art und Weise einschränkt, wie ein System realisiert werden kann
- Sie ergänzen Produktanforderungen und Qualitätsanforderungen; Geschäftsprozesse, Gesetze sind typische Beispiele
- Organisatorische Randbedingungen spielen auch eine Rolle, obwohl sie nicht als Anforderungen zugegeben werden

Kategorien von Anforderungen: Offen vs latent

- **Offene und latente Anforderungen, Entwickler-Optionen**
 - **Offene Anforderungen:** vom Kunden selbst brauchbar formuliert
 - **Latente Anforderungen:** dem Kunden nicht bewusst.
 - **Entwickler-Optionen:** Der Kunde hat den Punkt offengelassen, es ist ihm gleich.
Auch das dokumentieren!

Harte und weiche Anforderungen

- Lehrbuchbeispiele enthalten typischerweise harte Anforderungen.
 - Beispiele: **Resultat einer Kontostandsberechnung, Obergrenze für den Umfang der installierbaren Software.**
- In der Praxis sind fast alle Anforderungen weich.



- Weiche Anforderungen sind **schwer zu erheben** und ebenso schwer zu formulieren.
- Prinzipiell möglich: **Kosten- oder Nutzenfunktionen.**

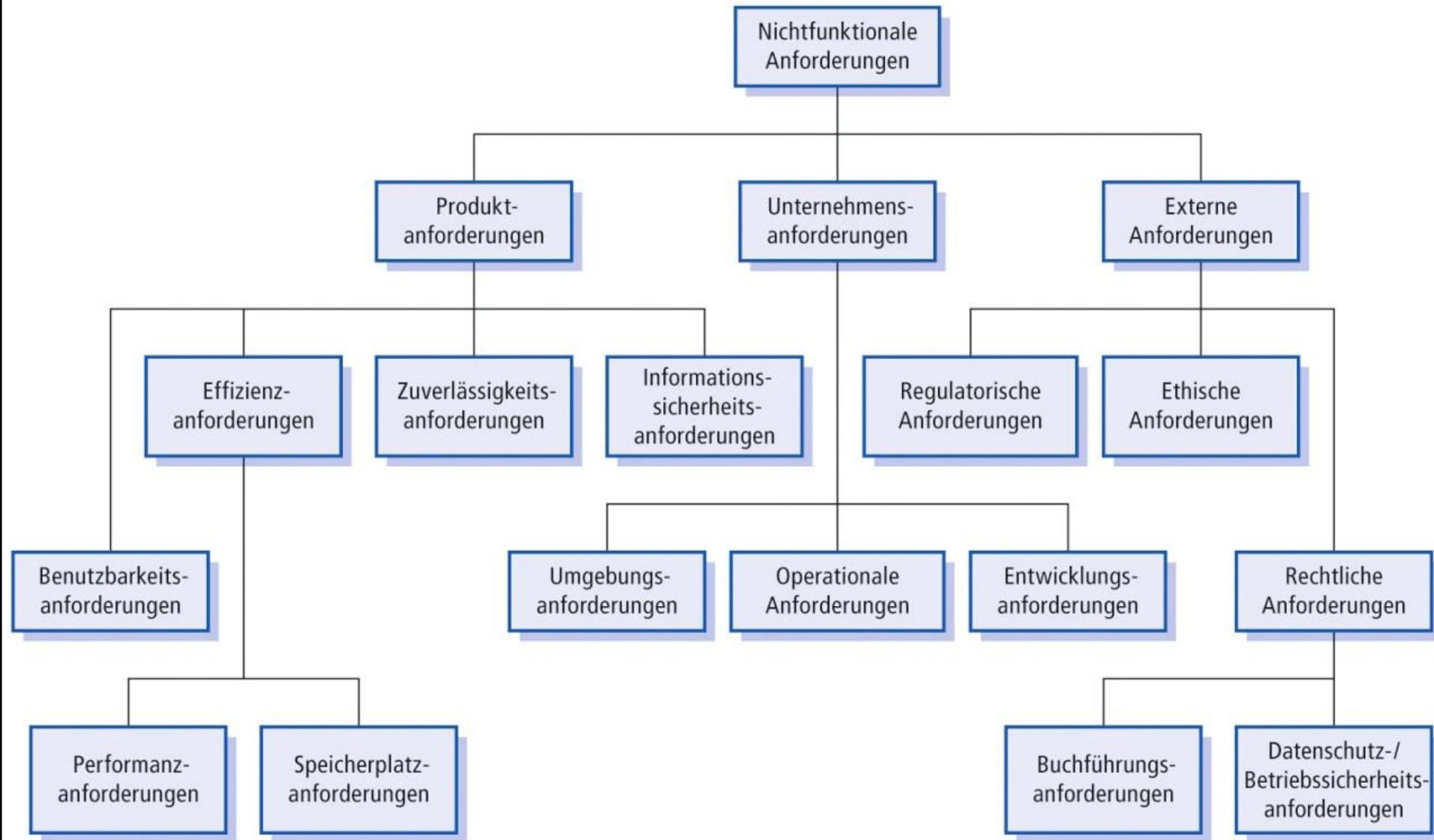
■ **Objektivierbare und vage Anforderungen**

- Anforderungen dienen nicht zuletzt als **Referenz bei der Prüfung der Software**.
- Dazu müssen die Anforderungen **objektivierbar** sein.
- Harte Anforderungen sind typisch objektivierbar, weiche nicht.

■ **Funktionale und nichtfunktionale Anforderungen**

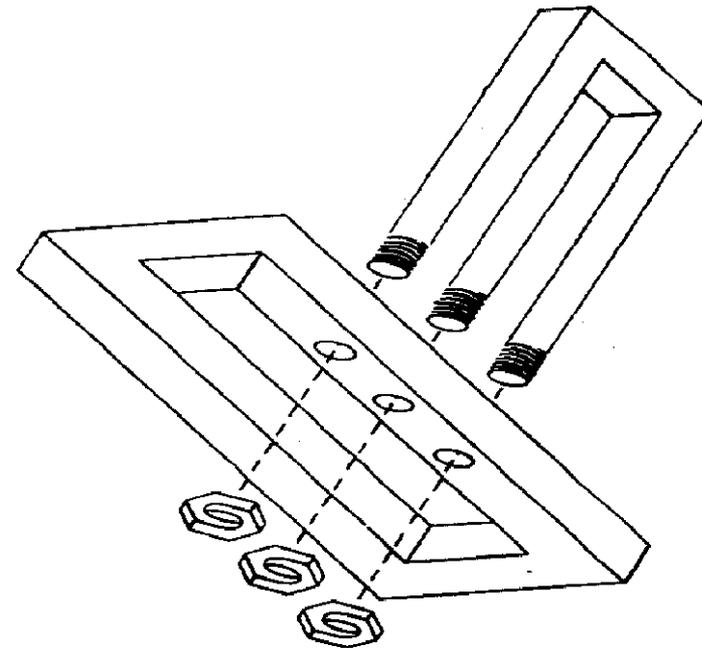
- Funktion der Software = Beziehung zwischen Ein- und Ausgaben, im weiteren Sinne auch das Zeitverhalten.
- Alle Anforderungen, die sich auf diese Funktion beziehen, sind **funktionale Anforderungen**, alle anderen sind **nichtfunktionale Anforderungen** (non-functional requirements, NFRs).

- Die Kurzbeschreibung eines Systems, also eine auf wenige Worte reduzierte Anforderung, ist stets eine grobe Charakterisierung der Funktion.
 - Beispiel: „Das System sichert nachts die Inhalte aller Festplatten.“
- Die **Abgrenzung** funktional – nichtfunktional ist **unscharf**; Anforderungen, die zwar die Funktion betreffen, aber nicht präzise formuliert werden können, werden vielfach als nichtfunktional eingeordnet.
 - Beispiele: **Robustheit, Bedienbarkeit**
 - Auch das Zeitverhalten wird meist als NFR behandelt.
- Funktionale **und** nichtfunktionale Anforderungen **werden gebraucht**.
- Praktisch alle Aussagen zur Wartbarkeit sind nichtfunktional.
 - „Das System muss leicht portierbar sein.“



- Die Formulierung nichtfunktionaler Anforderungen bereitet uns große Probleme. Die NFRs werden entsprechend stiefmütterlich behandelt, meist ganz weggelassen oder nur durch Schlagwörter ausgedrückt. **Das ist aber ihrer großen Bedeutung nicht angemessen.**
- NFRs lassen sich am besten mit Hilfe von Normen spezifizieren.
- **Vgl. DIN EN ISO 9241**
 - Teile 1 bis 9: Ergonomie, Technik der Dialoggestaltung
 - Teil 10: Grundsätze der Dialoggestaltung
 - Teile 11 bis 17: Details der Dialoggestaltung
- Allerdings bietet keiner der Standards harte Vorschriften, deren Einhaltung einfach und objektiv zu prüfen wäre.
- In keinem Falle sollte man auf einen Versuch zur Präzision verzichten!

- Anforderungen sollen u.a. folgende Kriterien erfüllen:
 - Vollständig
 - Verständlich
 - Korrekt
 - Widerspruchsfrei
 - Eindeutig
 - Prüfbar



- Anforderungen können mehrdeutig, überspezifisch, unvollständig, kontextspezifisch, unmöglich oder falsch sein, auf jeden Fall gibt es immer zu viele.
- Bei Anforderungen stellen sich viele Fragen:
 - Ist die Anforderung wichtig?
 - Sind alle Inhalte gleichermaßen wichtig und wie korrelieren sie?
 - Wer ist für die Anforderung verantwortlich?
 - Ist die Anforderung machbar? Welche Einflüsse hat sie?
 - Wurde diese Anforderung geändert und durch wen?
 - Wo sind die Analyseergebnisse dokumentiert?
 - Wie wird die Anforderung validiert?
 - Wie konkretisiert sich die Anforderung?



Wie findet man Anforderungen? Requirements Analysis



Definitionen (Nach IEEE Std 610.12-1990)

■ Requirements Analysis

- (1) The process of studying user needs to arrive at a definition of system, hardware, or software requirements.
- (2) The process of studying and refining system, hardware, or software requirements.

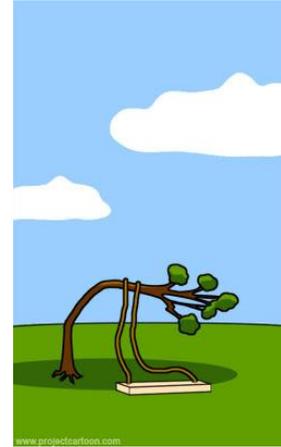
Methoden um Anforderungen zu finden

- Man analysiert den **IST-Zustand** und definiert einen **Soll-Zustand**
 - Man fragt den Kunden
 - Man nutzt sein Know-How
 - Man recherchiert
 - Lesen
 - Marktbeobachtung
 - Man phantasiert Beispiel-Szenarien
 - ...

Wer ist überhaupt der Kunde?

- Oft nicht so eindeutig wie man denkt:
 - Der, der es bezahlt (Auftraggeber, Chef, Endkunde)
 - Aber auch jeder, der nachher das Produkt verwendet (Benutzer)
 - Jeder, der unter dem Output zu leiden hat
 - Diejenigen, die es auf den Markt bringen
 - Z.B. Verkäufer, Support-Dienstleister
 - Eventuell auch diejenigen, die bislang Konkurrenzprodukte verwenden
- Drei Kategorien von Benutzern (Eason, 1987):
 - Primäre: Verwenden das Produkt regelmäßig selbst
 - Sekundäre: Gelegenheits- oder indirekte Benutzer
 - Tertiäre: Von der Einführung des Produkts Betroffene oder Menschen, die mit dem Verkauf des Produktes zu tun haben

- Ziel der Analyse: **Soll-Zustand** feststellen
- Es sollte also genügen, die Klienten nach ihren Wünschen zu fragen.
- **Aber**: Die Klienten konzentrieren sich auf das, was ihnen derzeit **nicht** gefällt
- Wir sind also bei jeder Umstellung auf die **Schwachpunkte des bestehenden Systems** fixiert
 - Seine Stärken nehmen wir erst wahr, wenn sie uns fehlen.
- **Implizit** erwarten die Klienten, dass alles, was bisher akzeptabel oder gut war, unverändert bleibt oder besser wird. Die Anforderungen an das neue System bestehen also nur zum kleinsten Teil aus Änderungswünschen, die allermeisten Anforderungen gehen in Richtung **Kontinuität**.



- Darum hat die Ist-Analyse, also die Feststellung des bestehenden Zustands, große Bedeutung für die erfolgreiche Projektdurchführung. Bei **Widersprüchen** ist der Ist-Zustand zudem ein **sicherer Halt!**
- Der Analytiker muss sich also **gut einfühlen** und **genau beobachten**, um vom Kunden zu erfahren, welche Aspekte des Ist-Zustands für ihn wichtig sind und beibehalten werden sollten.
- Die Ist-Analyse ist eine **undankbare Tätigkeit!**

Beispiel

Sie öffnen also morgens das Schloss am Haupteingang?

Ja, habe ich Ihnen doch gesagt.

Jeden Morgen?

Natürlich.

Auch am Wochenende?

Nein, am Wochenende bleibt der Eingang zu.

Und während der Betriebsferien?

Da bleibt er natürlich auch zu.

Und wenn Sie krank sind oder Urlaub haben?

Dann macht das Herr X.

Und wenn auch Herr X ausfällt?

Dann klopft irgendwann ein Kunde ans Fenster, weil er nicht reinkommt.

Was bedeutet „morgens“? ...

- Gefahr: Der Analytiker wird zum Prellbock zwischen
 - den **Kunden** (mit großen Wünschen) und
 - den **Entwicklern** (mit der Sorge, die Wünsche nicht erfüllen zu können, und dem Wunsch, eigene Vorstellungen zu realisieren).
- Aber der Analytiker sollte sich nicht als deus ex machina fühlen, der die unlösbaren Probleme löst.
- Sinnvoll:
 - Alle Wünsche sammeln, nicht kritisieren („Wunschzettel“)
 - Widersprüche offenlegen
- Wünsche sind technisch **nicht realisierbar oder zu teuer**: →
 - Anforderungen müssen reduziert oder Mittel aufgestockt werden.
- **Konflikte** zwischen Wünschen der Klienten: →
 - Probleme vor Klienten ansprechen und auf Einigung drängen.



- In aller Regel haben die Kunden kein absolut festes Bild des Produkts. Das verschafft dem Analytiker Spielräume.
 - Ein System, das **nicht völlig autonom** arbeitet, kann seine Benutzer mehr oder minder stark in Anspruch nehmen.
 Wenn eine bestimmte Funktion schwierig zu implementieren ist, aber nur selten gebraucht wird, kann man sie auch von Hand ausführen.
 - Die **zentrale Komponente** des Systems hat Vorrang; andere Teile werden nicht sofort benötigt. → Entwicklung in Ausbaustufen. Vorteile: Zentrale Funktion steht schnell zur Verfügung, und die weitere Entwicklung kann auch erste Erfahrungen und neue Fakten berücksichtigen.
 - Wenn **käufliche (oder vorhandene) Komponenten** integriert werden, sinken die Kosten und die Entwicklungszeit. Der Kunde muss den Preis „handgeschmiedeter“ Lösungen kennen.

- Die Ausgangslage der Analyse ist **extrem unterschiedlich**:
 - Anwendungsgebiet der Software
 - Umgebung des Zielsystems
 - Aufgabenstellung
 - Vorkenntnisse des Analytikers
 - Verständnis der Gesprächspartner usw.
- Folge: Es gibt **kaum allgemeine Regeln für die Analyse**.
- Immer wichtig und richtig:
 - Die Analyse als Tätigkeit **wahr- und ernst nehmen!**
 - Aufwand und Personal dafür im **notwendigen Maße einplanen!**
 - Resultate in eine (auch dem Kunden) **verständliche Form bringen!**

| Analysetechnik | Schwerpunkt | | |
|--|-------------|--------------|-------------------|
| | Ist-Zustand | Soll-Zustand | Innovationsfolgen |
| Auswertung vorhandener Daten und Dokumente | ■ | | |
| Beobachtung | ■ | | |
| Befragung mit $\left\{ \begin{array}{l} \text{geschlossenen} \\ \text{strukturierten} \\ \text{offenen} \end{array} \right\}$ Fragen | ■ | | |
| | ■ | | |
| | ■ | | |
| Interview | | ■ | |
| Modell-Entwicklung | | ■ | |
| Experimente | | ■ | |
| Prototyping | | ■ | |
| partizipative Entwicklung (nur hinsichtlich Analyseteil) | | | ■ |

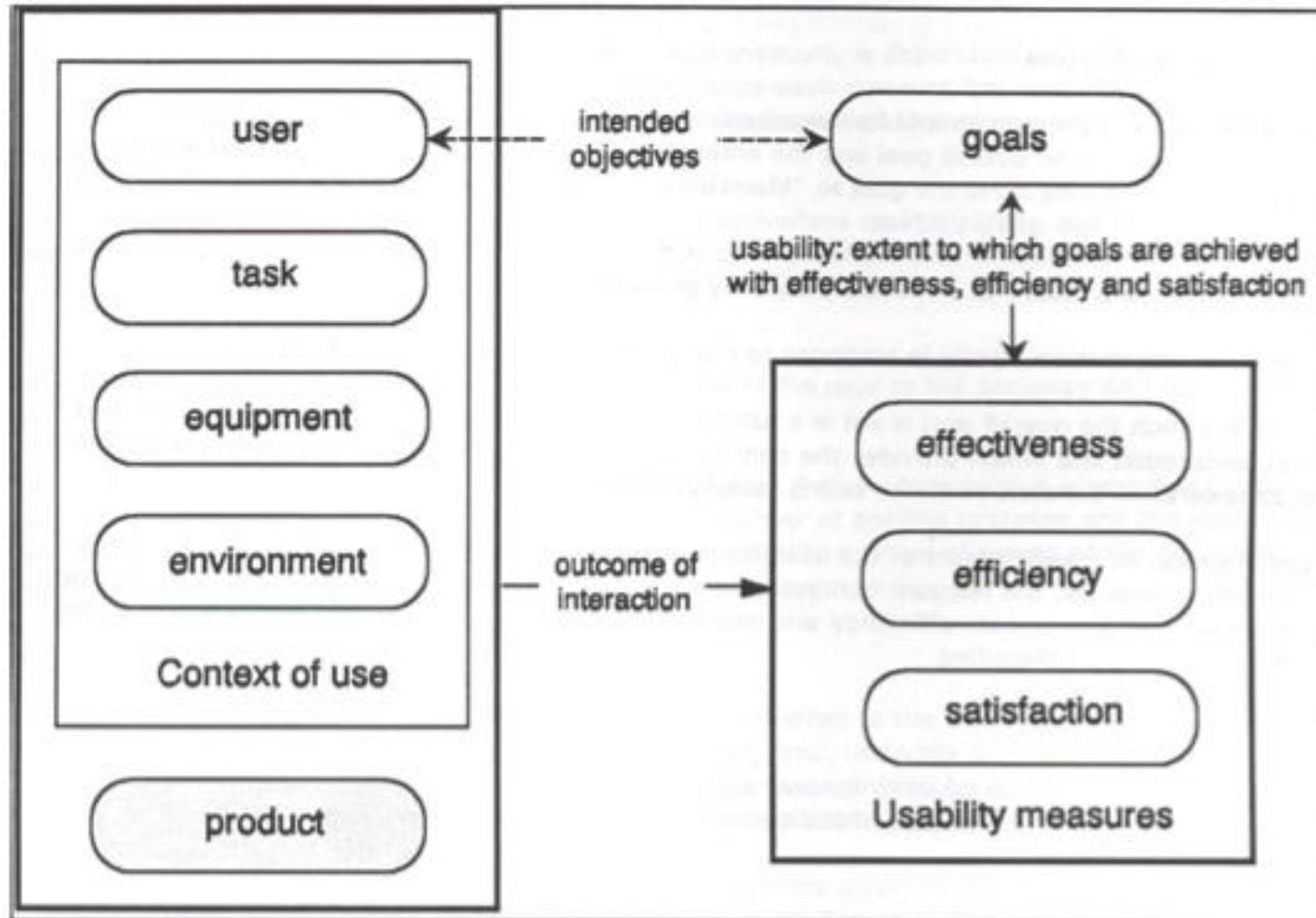
- Natürlich sollte der Analytiker sollte die einschlägigen **Vorschriften und Regeln** kennen und beachten.
 - Auswertung der Dokumente
- Wenn möglich, direkt **am Alltag** derer **teilnehmen**, die später mit der Software umgehen werden oder deren Funktion später von der Software übernommen wird (Ethnographische Analyse).
 - **Über die Schulter schauen** oder, besser noch, selbst mitarbeiten.
- Dann kann der Analytiker die Klienten **systematisch befragen**
 - geschlossene und strukturierte, auch offene Fragen
 - **Gespräch** bringt mehr als Fragebogen-Versand!

- Frageformulierung
 - einfache und eindeutige Formulierung
 - kurze prägnante Fragen (möglichst keine Unterfragen) verwenden
 - nicht zu viel bzw. zu wenig Fragen verwenden
 - einfache Sachverhalte ansprechen
 - Vermeidung einer Überbeanspruchung des Befragten
 - Vermeidung von komplizierten Sätzen und unbekanntem Begriffen
 - Vermeidung von doppelten Negationen
 - Vermeidung von suggestiven Fragen
 - Verwendung neutraler Fragen
- vollstandardisiert
 - explizite Formulierung der Fragen
 - festgelegte Reihenfolge der Fragen
 - kein Spielraum des Interviewers vorhanden
- teilstandardisiert
 - Fragebogengerüst
 - hauptsächlich "offene" Fragen
 - Möglichkeit der Mitstrukturierung des Interviewers
- nicht standardisiert
 - völliger Verzicht auf einen Fragebogen
 - nur Stichwort- oder Themenvorgabe

- „offene“ Fragen:
 - keine Antwortvorgabe
 - vollkommene Freiheit des Befragten
 - Geeignet um Wissen zu prüfen
 - Offenheit beschränkt Vergleichbarkeit

- „geschlossene“ Fragen
 - Fragen und Antworten vorgegeben
 - Bessere Quantifizierung der Antworten
 - Spektrum der Antwortmöglichkeiten kann unvollständig sein
 - Oft verwendet: Likert-Skala
 - Quantitative Auswertung mit Software wie SPSS

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



Usability Framework (ISO 9241)

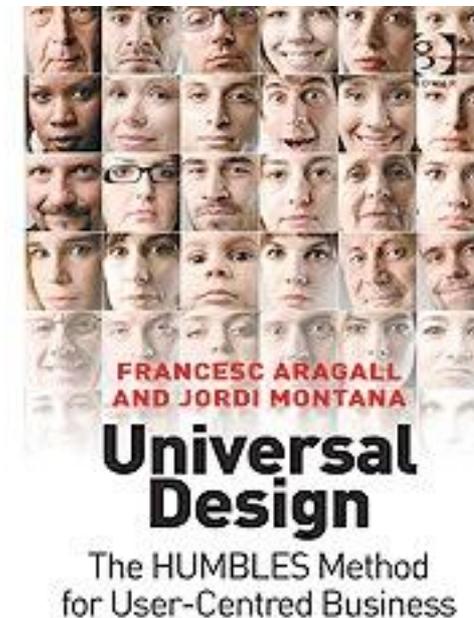
- 3: Visual display requirements
- 4: Keyboard requirements
- 5: Workstation layout and postural requirements
- 6: Guidance on the work environment
- 7: Requirements for display with reflections
- 8: Requirements for displayed colours
- 9: Requirements for non-keyboard input devices
- 10: Dialogue principles (formerly 10)
- 11: Guidance on usability
- 12: Presentation of information
- 13: User guidance
- 14: Menu dialogues
- 15: Command dialogues
- 16: Direct manipulation dialogues
- 17: Form filling dialogues

Nicht gesetzlich vorgeschrieben, aber sehr hilfreich (und manchmal vom Kunden gewünscht)

* Bis 2006: Ergonomic requirements for office work with visual display terminals

- Web Content Accessibility Guidelines (WCAG 2.0)
 - Perceivable
 - Operable
 - Understandable
 - Robust

**Gesetzlich vorgeschrieben für
öffentliche Einrichtungen (BIT-V)**



§4 Barrierefreiheit:

- Barrierefrei sind bauliche und sonstige Anlagen, Verkehrsmittel, technische Gebrauchsgegenstände, Systeme der Informationsverarbeitung, akustische und visuelle Informationsquellen und Kommunikationseinrichtungen sowie andere gestaltete Lebensbereiche, wenn sie für behinderte Menschen in der allgemein üblichen Weise, ohne besondere Erschwernis und grundsätzlich ohne fremde Hilfe zugänglich und nutzbar sind.

Behindertengleichstellungsgesetz (BGG)

§ 11 Barrierefreie Informationstechnik

- (1) Träger öffentlicher Gewalt im Sinne des § 7 Abs. 1 Satz 1 gestalten ihre Internetauftritte und -angebote **sowie die von ihnen zur Verfügung gestellten grafischen Programmoberflächen**, die mit Mitteln der Informationstechnik dargestellt werden, nach Maßgabe der nach Satz 2 zu erlassenden Verordnung schrittweise technisch so, dass sie von behinderten Menschen grundsätzlich uneingeschränkt genutzt werden können. [...]
- (2) Die Bundesregierung wirkt darauf hin, dass auch **gewerbsmäßige Anbieter von Internetseiten sowie von grafischen Programmoberflächen**, die mit Mitteln der Informationstechnik dargestellt werden, durch Zielvereinbarungen nach § 5 ihre Produkte entsprechend den technischen Standards nach Absatz 1 gestalten.

§ 3 Anzuwendende Standards

- (1) Die in § 1 genannten Angebote der Informationstechnik sind nach der Anlage 1 so zu gestalten, dass alle Angebote die unter Priorität I aufgeführten Anforderungen und Bedingungen erfüllen. Weiterhin sollen zentrale Navigations- und Einstiegsangebote zusätzlich die unter Priorität II aufgeführten Anforderungen und Bedingungen berücksichtigen.

§ 3 Anzuwendende Standards

■ (2) Auf der Startseite des Internet- oder Intranetangebotes (§ 1 Nummer 1 und 2) einer Behörde im Sinne des § 7 Absatz 1 Satz 1 des Behindertengleichstellungsgesetzes sind gemäß Anlage 2 folgende Erläuterungen in Deutscher Gebärdensprache und in Leichter Sprache bereitzustellen:

1. Informationen zum Inhalt,
2. Hinweise zur Navigation sowie
3. Hinweise auf weitere in diesem Auftritt vorhandene Informationen in Deutscher Gebärdensprache oder in Leichter Sprache.

Die vier Prinzipien der BITV 2.0

- **Prinzip 1: Wahrnehmbarkeit** – Die Informationen und Komponenten der Benutzerschnittstelle sind so darzustellen, dass sie von den Nutzerinnen und Nutzer wahrgenommen werden können.
- **Prinzip 2: Bedienbarkeit** – Die Komponenten der Benutzerschnittstelle und die Navigation müssen bedient werden können.
- **Prinzip 3: Verständlichkeit** – Die Informationen und die Bedienung der Benutzerschnittstelle müssen verständlich sein.
- **Prinzip 4: Robustheit** – Inhalte müssen so robust sein, dass sie von möglichst allen Benutzeragenten, **einschließlich assistiver Technologien**, zuverlässig interpretiert werden können.



Braillezeile



Fußmaus



Kopfmaus



Mund-Maus

Datenschutz als Grundrecht: Das Volkszählungsurteil des BVerfG (1983)

„Das Persönlichkeitsrecht umfaßt [...] auch die aus dem Gedanken der Selbstbestimmung folgende Befugnis des Einzelnen, grundsätzlich selbst zu entscheiden, wann und innerhalb welcher Grenzen persönliche Lebenssachverhalte offenbart werden. [...] Diese Befugnis bedarf unter den heutigen und künftigen Bedingungen der automatischen Datenverarbeitung in besonderem Maße des Schutzes.“

„Mit dem Recht auf **informationelle Selbstbestimmung** wären eine Gesellschaftsordnung und eine diese ermöglichende Rechtsordnung nicht vereinbar, in der Bürger nicht mehr wissen, wer was wann und bei welcher Gelegenheit über sie weiß.[...] Das Grundrecht gewährleistet insoweit die Befugnis des Einzelnen, grundsätzlich selbst über Preisgabe und Verwendung seiner persönlichen Daten zu bestimmen“

BDSG §1 (1)

Zweck dieses Gesetzes ist es, den **Einzelnen** davor zu schützen, dass er durch den Umgang mit seinen **personenbezogenen** Daten in seinem **Persönlichkeitsrecht** beeinträchtigt wird.

BDSG §3 (1)

Personenbezogene Daten sind Einzelangaben über persönliche oder sachliche Verhältnisse einer bestimmten oder bestimmbaren natürlichen Person (Betroffener).

Grundprinzipien des Datenschutzrechts

- Verbot mit Erlaubnisvorbehalt
- Zweckbindung
- Erforderlichkeit
- Betroffenenrechte
- Transparenz

Grundsatz

Die Verarbeitung und Nutzung personenbezogener Daten ist prinzipiell verboten.

(§ 4 Abs. 1 BDSG)

- Verarbeitung ist nur zulässig, wenn
 - a) der/die Betroffene oder
 - b) der Gesetzgeber (auch im BDSG) sie hinsichtlich Umfang und Zweck gebilligt hat („bereichsspezifische Regelungen“).

Gesetz zum Datenschutz im Schulwesen (Schuldatenschutzgesetz) in der FHB

Fassung vom 27.2.2007

- §12 Schülerverzeichnis
- (1) Zur Überwachung der Schulpflicht und zur Vorbereitung, Durchführung und Auswertung schulorganisatorischer Maßnahmen sowie für schulstatistische und berufsvorbereitende Zwecke können beim Senator für Bildung und Wissenschaft und beim Magistrat Bremerhaven nachstehende Daten in automatisierten Dateien verarbeitet werden:
 - 1. Bei allgemeinbildenden Schulen Name, Geburtsdatum, Adressdatum, Geschlecht, Staatsangehörigkeit, Muttersprache, Aussiedlereigenschaft und Einschulungsdatum der Schülerin oder des Schülers und die von ihm oder ihr besuchte Klasse sowie von den Erziehungsberechtigten Name und Adressdatum;
 - 2. bei beruflichen Schulen darüber hinaus die Daten des Ausbildungsberufes, des betrieblichen Ausbildungsbeginns und des Ausbildungsendes der Schülerin oder des Schülers.

Gesetz zum Datenschutz im Schulwesen (Schuldatenschutzgesetz) in der FHB

Fassung vom 27.2.2007

§4 Einwilligung und Unterrichtung der Betroffenen

- (1) Die Daten gemäß der Verordnung nach §2 Abs. 2 dürfen grundsätzlich nur mit Einwilligung der Betroffenen verarbeitet werden, es sei denn, dass
 - 1. die Verarbeitung im Interesse der oder des Betroffenen oder für die pädagogische Arbeit an der Schule notwendig ist,
 - 2. die fehlende Kenntnis der Daten über Erkrankungen und Behinderungen eine gesundheitliche Gefährdung der oder des Betroffenen bedeuten könnte
 - 3. die Verarbeitung zur Wahrnehmung der Dienst- und Fachaufsicht oder innerhalb eines Verwaltungs-, Rechtsbehelfs- oder Gerichtsverfahrens erforderlich ist oder
 - 4. der Schutz einer oder eines Betroffenen die Einholung der Einwilligung verbietet.

- Wenn die abstrakte Vorstellung vom Zielsystem nicht ausreicht, muss etwas **ausprobiert** oder **gezeigt** werden.
 - **Modell-Entwicklung**
 - **Experiment**
 - **ausführbarer Prototyp**

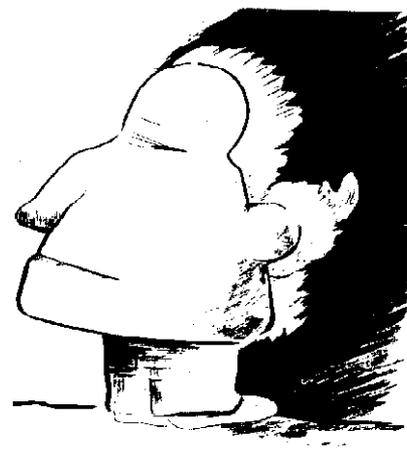
- Bei der **partizipativen Entwicklung** wird die Trennung zwischen Analyse und Entwicklung bewusst aufgehoben. Die neue Software wächst in den sozialen Kontext hinein.
 - Vergleiche auch die Ansätze der „**agilen Software-Entwicklung**“!

- Der Kunde kann sehr viele Anforderungen nicht nennen, **weil er sie nicht hat**. Er hat nur **Wünsche und Ziele**, die sich im Gespräch mit dem Analytiker auf Anforderungen abbilden lassen.
- Der Analytiker hat (bewusst oder unbewusst) eigene Interessen, die er durchsetzen möchte (eine „**hidden agenda**“).
- Der Kunde hat **Anforderungen, die er nicht sagen will** (also auch eine hidden agenda)
- Der Kunde hat Anforderungen, die ihm so **selbstverständlich** scheinen, dass er sie nicht erwähnt.
- Am Ende der Analyse und Spezifikation sollte eine Vision entstanden sein, in der
 - der Kunde die **Erfüllung seiner** (reduzierten) **Wünsche**,
 - der Entwickler ein **realisierbares Produkt** sieht.
- Beschrieben durch ein **Dokument** evtl. auch durch einen **Prototypen**.

- Warum wird die Analyse (und besonders die Ist-Analyse) in der Praxis oft vernachlässigt oder falsch fokussiert?
 - Wo die **Spezifikation keine Rolle** spielt, wird auch die Analyse keine Bedeutung haben.
 - Der Kunde will **keine Veränderung, sondern eine Verbesserung**. Entwickler, die das nicht begreifen, vernachlässigen die Ist-Analyse.
 - Entwickler sind oft der (grundfalschen) Überzeugung, **bereits zu wissen, was gewünscht oder benötigt wird**.
 - **Kunden** legen der Analyse **Steine in den Weg**:
 - Den Analytikern stehen die Kunden (insbesondere die Fachleute auf Kundenseite) nicht zur Verfügung.
 - Die Analyse-Aktivität wird als unproduktiv denunziert.
 - Kunden sabotieren den Prozess durch späte Änderungen.

Aus einem Praxisbericht

- Analyse wird vom **Kernteam** durchgeführt (ca. 3-4 Leute). Die Mitglieder kommen teilweise aus der **Anwendung**, teilweise handelt es sich um **Entwickler**. Die Anforderungen an diese Leute sind **sehr hoch**.
- An der Analyse müssen **Entscheidungsträger**, **Fachleute** und **Anwender** beteiligt sein. Die Anwender werden typisch durch **Interviews** beteiligt. (Zwei Leute befragen eine Person ca. 90 min lang).
- Abstimmung der Analyse in „**Workshops**“ mit 6 bis 10 Teilnehmern, ein halber bis ganzer Tag.
- Es ist aussichtslos, den Kunden mit **formalen Darstellungen** zu kommen.
- Resultat der Analyse ist ein **Begriffslexikon**.



- Hilfreich ist es, während der Analyse ein **Begriffslexikon** anzulegen; dieses wird während der gesamten Software-Entwicklung verwendet und ergänzt.
- In den frühen Phasen wird dieses Dokument **aufgebaut und weiterentwickelt**.
- Das Begriffslexikon enthält solche Begriffe, die
 - **wichtig** sind und
 - von verschiedenen Leuten, v.a. von Klienten, Analytikern und Entwicklern, **unterschiedlich ausgelegt werden** könnten.
- Dazu gehören häufig **Begriffe, die auf den ersten Blick völlig klar zu sein scheinen**.
- Beispiel: Informationssystem für die Prüfungsdaten von Studenten
 - „Student“, „Prüfung“, „Note“, „Erstprüfung“

- 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)
- Die Angaben werden **aus den Gesprächen und Interviews abgeleitet** oder, wenn sie von den Analytikern kommen, **sorgfältig mit den Klienten überprüft**.
 - Typische Quellen sind im genannten Beispiel die Angestellten in der Verwaltung, die Juristen der Uni, die Gesetze und Ordnungen der Universität.

| | |
|---------------------|---|
| Begriff | Student , synonym Studentin, Studierender, Studierende |
| Bedeutung | Eine Person, die an der Universität Stuttgart immatrikuliert ist und noch nicht exmatrikuliert wurde, die folglich legal einen Studentenausweis der Universität Stuttgart hat oder haben könnte. |
| Abgrenzung | Gasthörer und Studierende anderer Hochschulen sind im Sinne dieses Systems keine Studenten. |
| Gültigkeit | Mit der Immatrikulation an der Universität Stuttgart entsteht ein neuer Student; er existiert bis zur Exmatrikulation, gleichgültig, wie sie zustande kommt. Ein Fachwechsel oder eine Namensänderung implizieren keine Exmatrikulation. Hat sich eine Person im Laufe ihres Lebens mehrfach an der Universität Stuttgart immatrikuliert, so handelt es sich um mehrere, nicht identische Studenten. |
| Bezeichnung | Ein Student ist durch die Matrikelnummer und einen Zeitpunkt (zu dem die Matrikelnummer gültig war oder ist) eindeutig bestimmt, alle anderen Attribute, insbesondere der Name, können mehrfach vorkommen. |
| Unklarheiten | Es ist noch ungeklärt, wie Namen aus anderen Schriftsystemen (z. B. Russisch, Arabisch, Chinesisch) dargestellt werden. |
| Querverweise | Gasthörer, Matrikelnummer, Studentenausweis |



Wie dokumentiert man Anforderungen? Requirements Specification



- 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 Realisierungsvorhaben aufgrund der Umsetzung des vom Auftraggeber vorgegebenen Lastenhefts

- Anforderungsspezifikation (Pohl, Rupp, 2011)
 - Eine systematisch dargestellte Sammlung von Anforderungen (typischerweise für ein System oder eine Komponente), die vorgegebenen Kriterien genügt.

- Lastenheft wird vom Auftraggeber (Kunden) geschrieben
 - welche Funktionalität ist gewünscht
 - welche Randbedingungen (SW/ HW) gibt es

- Pflichtenheft wird vom Auftragnehmer (Software-Entwicklung) geschrieben
 - welche Funktionalität wird realisiert
 - auf welcher Hardware läuft das System
 - welche SW-Schnittstellen (Versionen) berücksichtigt

- Variante: Kunde beauftragt Auftragnehmer direkt in Zusammenarbeit Pflichtenheft zu erstellen
 - ein gemeinsames Heft ist sinnvoll
 - Pflichtenheft ist meist (branchenabhängig) zu bezahlen

- Beschreibt die **fachlichen Anforderungen aus Klientensicht**.
- Obwohl es Definitionen für diesen Begriff gibt (z.B. in DIN 69905), bleibt er in der Praxis unscharf, weil der Inhalt des Lastenhefts von Firma zu Firma **stark variiert**.
- Lücken, Unklarheiten und Widersprüche sind darin normal.
- Erst das Pflichtenheft sollte **vollständig, klar und konsistent** sein.



- Das Pflichtenheft ist im Idealfall inhaltlich
 1. **zutreffend**: Es gibt die Vorstellungen des Kunden richtig wieder.
 2. **vollständig**: Jede (in einem Kopf oder in einem Dokument) vorhandene Anforderung ist in der Spezifikation enthalten.
 3. **widerspruchsfrei** (oder **konsistent**): Jede Anforderung ist mit allen anderen Anforderungen vereinbar.
Ein inkonsistentes Pflichtenheft ist nicht realisierbar, denn kein System kann widersprüchliche Anforderungen erfüllen.
 4. **neutral** (oder **abstrakt**): Das Pflichtenheft schränkt die Realisierung nicht über die wirklichen Anforderungen hinaus ein.
 5. **nachvollziehbar**: Die Quellen der Anforderungen sind dokumentiert, die Anforderungen sind eindeutig identifizierbar.
 6. **objektivierbar**: Das realisierte System kann gegen die Anforderungen geprüft werden. auch (missverständlich) „testbar“.

- Eine **ideales Pflichtenheft** ist
 1. **leicht verständlich**: Alle Interessenten sind in der Lage, das Pflichtenheft zu verstehen.
 2. **präzise**: Das Pflichtenheft schafft keine Unklarheiten und Interpretationsspielräume.
 3. **leicht erstellbar**: Die Anfertigung und Nachführung des Pflichtenhefts sind einfach und erfordern keinen nennenswerten Aufwand.
 4. **leicht verwaltbar**: Die Speicherung des Pflichtenhefts und der Zugriff darauf sind einfach und erfordern keinen nennenswerten Aufwand.
- Diese Merkmale **konkurrieren!**
- Auch hier suchen wir also einen **Kompromiss**.

- Die Forderung nach Neutralität war in der Frühzeit des Software Engineerings radikal:
- Idee: Das Pflichtenheft soll aussagen, **was** das System leistet, aber nicht, **wie** diese Leistung erbracht wird („**What, not how!**“).
- **Beispiel: Programm zur Ermittlung der Nullstellen einer Funktion kann ohne Aussagen zum Algorithmus spezifiziert werden.**
- Inzwischen wurde die Forderung nach Neutralität aufgeweicht.
- Als Ideal bleibt sie aber gültig. Denn in der Praxis kann man immer wieder beobachten, dass ein Pflichtenheft gefordert, aber ein Entwurf geliefert wurde. Auf diese Weise wird (evtl.) die optimale Lösung ausgeschlossen.

- Die geforderte Neutralität der Spezifikation ist als Ideal sinnvoll, aber praktisch nicht durchzuhalten. Dafür gibt folgende Gründe:
 1. Vorhandene Komponenten sind einzubeziehen.
 2. Vorgaben von außen betreffen die Struktur und Details
 3. Die Beschreibung des Lösung ist einfacher.
 4. Erst das gegliederte System ist überschaubar.
 5. Die Konsistenz der Spezifikation wird erst durch die Realisierung geklärt.
 6. Die Fragen an die Spezifikation entstehen erst im Entwurf.

- Darum kann eine abstrakte Spezifikation nur unter Idealbedingungen entstehen: **Das Problem ist gut verstanden, sicher lösbar, stabil und relativ „flach“.**

0. Administrative Daten: von wem, wann genehmigt, ...
1. Zielbestimmung und Zielgruppen
 - In welcher Umgebung soll System eingesetzt werden?
 - Ziele des Systems, welche Kunden betroffen?
2. Funktionale Anforderungen
 - Produktfunktionen (Use Cases, Anforderungen)
 - Produktschnittstellen (GUI-Konzept , andere SW)
3. Nichtfunktionale Anforderungen
 - Qualitätsanforderungen
 - weitere technische Anforderungen
4. Lieferumfang
5. Abnahmekriterien
6. Anhänge (insbesondere Begriffslexikon)

Darstellung: Formal, graphisch, natürlichsprachlich

- Eine Pflichtenheft soll einerseits **präzise** und einer Bearbeitung mit Werkzeugen zugänglich, andererseits auch für **Laien handhabbar**, mindestens aber verständlich sein.
- Da die Informatik starke Wurzeln in der Mathematik hat, war es naheliegend, für die Spezifikation **formale Notationen** zu entwickeln.
- Mit diesen Notationen verwandt sind **grafische Darstellungen** der Spezifikation.
 - Dabei steht aber weniger die formale Präzision als die Anschaulichkeit im Vordergrund.
- Wer weder formal noch grafisch spezifiziert, bedient sich der **natürlichen Sprache**.
 - Diese hat den Vorteil, für jeden Menschen verständlich zu sein und ganz ohne spezielle Regeln und Werkzeuge auszukommen..

Die sieben Regeln zur Formulierung von Anforderungen

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

- **Schlecht:** Das geplante System soll sowohl von Experten als auch von unerfahrenen Personen (Nutzerinnen und Nutzern) benutzbar sein. Unerfahrene User sollen auch ohne große Vorkenntnisse der Bedienerführung oder des Vorgängersystems die vorgesehene Systemfunktionalität nutzen können. Zur Benutzung des Systems sollen die Anwender also keinerlei Schulungen oder spezielle Hilfestellungen benötigen. Der Umgang mit dem System muss daher leicht verständlich sein und ohne große Erfahrung mit dem Vorgängersystem oder vergleichbaren Systemen erfolgen können.

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

- **Besser:** Ein unerfahrener Benutzer soll das System ohne spezielle Schulung verwenden können

- **Schlecht:** Die Dauer für die Erfassung und Verarbeitung der Messdaten soll halbiert werden. Dadurch soll die Wartezeit bis zum Vorliegen von Ergebnissen verkürzt werden.

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

- **Besser:** Das System erfasst und verarbeitet die Messdaten im Vergleich zum System xy doppelt so schnell. Dadurch muss der Nutzer kürzer auf das Vorliegen von Ergebnissen warten

- **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:

- ...

- ...

- **Schlecht:** Die Benutzung des Systems soll selbsterklärend sein

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

- **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

- **Schlecht:** Das System soll leicht benutzbar sein.

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

- **Besser:** Das System soll ... so dass sich die Nutzer auf andere Aufgaben konzentrieren können

- **Schlecht:** Das System soll auch von ungeschulten Benutzern intuitiv benutzbar sein.

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

- **Besser:** ... weil es auch in Mietfahrzeugen einsetzbar sein soll

- **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.

| | Regel | Erläuterung, Beispiel |
|--------|---|--|
| R 1 | Formulieren Sie jede Anforderung im Aktiv. | Der Akteur wird angegeben, und es wird sichtbar, ob das System oder der Benutzer etwas tut. Fordert man, dass etwas »gelöscht wird«, so bleibt das unklar. |
| R 2 | Drücken Sie Prozesse durch Vollverben aus. | Vollverben (wie »liest«, »erzeugt«, nicht »ist«, »hat«) verlangen weitere Informationen (Objekte, Ergänzungen), die den Prozess genauer beschreiben. Nicht: »Wenn die Daten konsistent sind«, sondern »Wenn Programm ABC die Konsistenz der Daten geprüft hat«. Und natürlich muss auch spezifiziert sein, was geschehen soll, wenn sie <i>nicht</i> konsistent sind (R4). |
| R 3 | Entdecken Sie unvollständig spezifizierte Prozesswörter (Verben). | Fehlen Angaben (Objekte), dann wird nach diesen Angaben gesucht, um vollständige Aussagen zu erhalten. Wenn eine Komponente <i>einen Fehler meldet</i> , fragt sich, wem. |
| R 4 | Ermitteln Sie unvollständig spezifizierte Bedingungen. | Für Bedingungen der Form »Wenn-dann-sonst« müssen sowohl der Dann- als auch der Sonst-Fall beschrieben sein (vgl. das Beispiel für R2). |
| R 5 | Bestimmen Sie die Universalquantoren. | Sind Sätze mit »nie«, »immer«, »jedes«, »kein«, »alle« wirklich universell gültig, oder gibt es Einschränkungen? Sind »alle Personen« wirklich alle Personen, oder nur die Anwesenden, die Mitarbeiter, die Besitzer einer Eintrittskarte? |

| | Regel | Erläuterung, Beispiel |
|----|---|--|
| R6 | Überprüfen Sie Nominalisierungen. | Nomen (z. B. »Generierung«, »Datenverlust«) weisen oft auf einen komplexen Prozess hin, der beschrieben werden muss. Verwendet man das Substantiv »Anmeldung«, dann fehlen meist die Ergänzungen, die beim Verb »anmelden« erwartet werden: Wer meldet sich wo und wofür an? |
| R7 | Erkennen und präzisieren Sie unbestimmte Substantive. | Oft bleibt unklar, ob es sich um einen generischen Begriff oder um ein bestimmtes Objekt handelt. Ist vom »Bediener« die Rede, so fragt es sich, ob es nur einen gibt oder welcher gemeint ist. Ähnliches gilt für viele Begriffe (Gerät, Meldung). |
| R8 | Klären Sie die Zuständigkeiten bei Möglichkeiten und Notwendigkeiten. | Ein Zwang muss realisiert werden. Steht in einer Anforderung, dass etwas <i>möglich</i> oder <i>unmöglich</i> ist, <i>passieren kann</i> , <i>darf</i> oder <i>muss</i> , so ist zu klären, wer dies erzwingt oder verhindert. |
| R9 | Erkennen Sie implizite Annahmen. | Begriffe in den Anforderungen (»die Firewall«), die nicht erläutert sind, deuten oft auf implizite Annahmen (hier vermutlich auf die, dass es <i>eine Firewall gibt</i>). |

| Schlagwort | bessere Anforderung |
|------------------------------|---|
| einfach bedienbar | Das Programm soll auch von Laien ohne weitere Einweisung benutzt werden können. |
| robust | Eine Bedienung über die Tastatur darf unter keinen Umständen zu einem irregulären Abbruch des Programms führen. |
| portabel | Das Programm muss von einem Entwickler in höchstens 6h von Windows XP auf Linux portiert werden können. |
| übersichtlich kommentiert | Die Module des Programms müssen einen Kopfkomentar nach Std. xyz enthalten. |
| plattformunabhängig | Alle prozessorspezifischen Teile des Programms müssen in einem speziellen Modul liegen. |
| nicht zu große Module | Module dürfen max. 300 Zeilen ausführbaren Code enthalten. |
| angemessenes Handbuch | Das Benutzungshandbuch wird gemäß Richtlinie ABC aufgebaut. Es wird vom Gutachter N.N. geprüft. |

- **Def.:** (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.

- Szenarientypen
 - Positive, negative, und Missbrauchsszenarien
 - Deskriptive, explorative, oder erklärende Szenarien
 - Systeminterne, Interaktive, Kontextszenarien
 - Haupt-, Alternativ- und Ausnahmeszenarien

- Anwendungsfälle (Use Cases) gruppieren Szenarien

Beispiel „automatisches Bremsmanöver“

- (nach Pohl) Der Fahrer fährt mit konstanter Geschwindigkeit auf der Autobahn. Das vorausfahrende Fahrzeug bremst scharf. Der Fahrer tritt auf das Bremspedal. Das System erkennt eine Unterschreitung des Sicherheitsabstands und gibt eine Warnung aus. Der Abstand zwischen den Fahrzeugen verringert sich weiter. Das System leitet eine automatische Vollbremsung ein und informiert den Fahrer über den Bremsvorgang. Als der Abstand sich nicht mehr verringert, beendet das System die automatische Vollbremsung. Danach stellt es den Mindestabstand zum vorausfahrenden Fahrzeug wieder her und informiert den Fahrer über die Rückgabe der Kontrolle.
- Was wird klar, was bleibt unklar?

Attribute von Szenarien

- **Positiv:** Wie wird das Ziel erfüllt?
- **Negativ:** Wie wird das Ziel nicht erfüllt?
- **Missbrauchsszenarien:** unerwünschte Verwendung

- **Deskriptiv:** verdeutlicht typische Abläufe
- **Explorativ:** verdeutlicht Lösungsraum
- **Erklärend:** verdeutlicht Gründe für Systemverhalten

- **Systemintern:** Handlungsstränge innerhalb des Systems
- **Interaktiv:** Handlungsstränge zwischen Akteuren
- **Kontextszenarien:** weitere Kontextinformationen

- **Hauptszenario:** wesentlicher Funktionsablauf
- **Alternativszenario:** partielle Abwandlung eines Hauptszenarios
- **Ausnahmeszenario:** Reaktion auf Ereignisse, die Hauptszenario verhindern

1. Der Fahrer schaltet das Navigationssystem ein.
2. Das System ermittelt den aktuellen Standort des Fahrzeugs.
3. Das System erfragt den gewünschten Zielort.
4. Der Fahrer gibt den Zielort in das System ein.
5. Das System ermittelt den benötigten Fartenausschnitt.
6. Das System zeigt die Karte des Zielgebietes am Bildschirm an.
7. Das System erfragt die Optionen für die Routenberechnung.
8. Der Fahrer wählt die Optionen für die Routenberechnung.
9. Das System bestimmt die Wegführung.
10. Das System zeigt eine Erfolgsmeldung am Bildschirm.
11. Das System stellt eine Liste von Wegpunkten zusammen.
12. Das System zeigt den nächsten Wegpunkt der Navigation.

| Szenario „Navigation zum Zielort“ | |
|--|--|
| Fahrer | Navigationssystem |
| 1. Einschalten des Geräts | |
| | 2. Ermittelt den aktuellen Standort |
| | 3. Erfragt den Zielort |
| 4. Eingabe des Zielorts | |
| | 5. Ermittelt den benötigten Kartenausschnitt |
| | 6. Zeigt am Display Karte vom Zielgebiet |
| | 7. Erfragt die Optionen für die Routenberechnung |
| 8. Eingabe der Optionen für die Routenberechnung | |
| | 9. Ermittelt die Wegführung |
| | 10. Anzeige „Weg wurde berechnet“ |
| | 11. Ermittelt die Liste der Wegpunkte |
| | 12. Anzeige des jeweils nächsten Wegpunkts |

- Stark vereinfachtes Bild von den Anforderungen in der Praxis:
 - Wenn der Klient eine Anforderung nicht **von selbst genannt** hat, dann **hat** er diese Anforderung **nicht**.
 - Wenn eine Anforderung **funktional** oder **quantifizierbar** ist, wird sie **dokumentiert**.
 - Alle anderen Anforderungen werden **pauschal als nichtfunktionale** Anforderungen bezeichnet und **kaum erwähnt**. Das gilt auch für
 - die Gebrauchsqualität (z.B. der **Bedienbarkeit**)
 - die Wartungsqualität (z.B. der **Erweiterbarkeit**).
- Empfehlung:
 - **Anforderungen nicht „vergessen“, nur weil sie Mühe machen.**



Warum machen wir das Ganze?



- In der Praxis gibt es viele schlechte Anforderungsanalysen; oft gibt es gar keine.
- Die Anforderungsanalyse ist aber notwendig für
 1. die **Abstimmung** mit dem **Kunden** bzw. mit dem Marketing,
 2. den **Entwurf** und die **Implementierung**,
 3. das **Benutzungshandbuch**,
 4. die **Testvorbereitung**,
 5. die **Abnahme**,
 6. die **Wiederverwendung**,
 7. die **Klärung** späterer Einwände, Regressansprüche usw.,
 8. eine spätere **Re-Implementierung**.

Nachteile bei fehlender Anforderungsanalyse

1. Die Anforderungen bleiben **ungeklärt**, sie werden darum auch nicht erfüllt.
2. Den Entwicklern **fehlt die Vorgabe**, darum fragen sie „auf dem kurzen Dienstweg“ Bekannte, die beim Kunden arbeiten, oder sie legen mangels Kontakten die eigenen Erfahrungen und Erwartungen zu Grunde.
3. Die **Basis für das Handbuch fehlt**, es wird darum phänomenologisch, d.h. experimentell, verfasst.
4. Ein gutes Handbuch ist ein umformulierter Auszug aus der Anforderungsanalyse!
5. Ein **systematischer Test** ist ohne Anforderungsanalyse unmöglich, denn es ist nicht definiert, welche Daten das System akzeptieren muss und welche Resultate es liefern soll.

6. Wenn bei der **Abnahme** nicht entschieden werden kann, ob das System richtig arbeitet, wird die Korrektheit zur **Glaubensfrage**.
 7. Oft zeigen sich **echte oder vermeintliche Mängel** der Software erst nach längerem Gebrauch. Ohne Anforderungen kann diese Unterscheidung aber nicht getroffen werden.
 8. Wer eine Software(-Komponente) **wiederverwenden** will, muss wissen, was sie leistet. Das ist in der Spezifikation dokumentiert.
 9. Wenn ein System ausgemustert und ersetzt wird, ist **Aufwärtskompatibilität** gefordert (vgl. Heninger et al., 1978).
- „**Spezifikation im Kopf**“ gibt es nicht!



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