



PINGO Abstimmung

<http://pingo.upb.de/5572>



Welche der folgenden Aussagen sind FALSCH?

- Die Spezifikation beschreibt die technische Architektur des zu erstellenden Systems.
- Stakeholder sind auch Benutzer des Systems.
- Klare Ziele bilden den Ausgangspunkt für die Anforderungsanalyse.
- Der Kunde erstellt alleine das Lastenheft.



Analyse – Anforderungen

- Begriff: **Anforderung**

requirement – (1) A condition or capability needed by a user to solve a problem or achieve an objective

...

IEEE Std 610.12 (1990)

- Anforderungsquellen sind
 - Dokumente
 - Systeme in Betrieb
 - Stakeholder

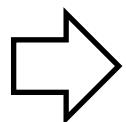
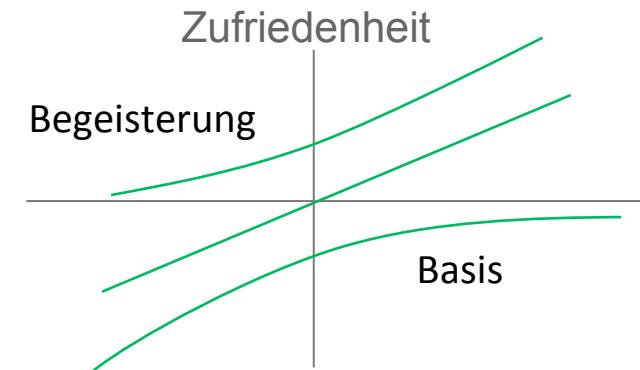


Aufgabe: Beurteilen Sie Ihr Smartphone/Handy nach entscheidenden Kaufmerkmalen. D. h. aus welchen Gründen haben Sie sich genau dafür entschieden?



Anforderungskategorisierung nach **KANO**

- **Basis-Merkmale:** grundlegend und selbstverständlich (implizite Erwartungen)
- **Leistungs-Merkmale:** bewusst wahrgenommen, schaffen Kundenzufriedenheit
- **Begeisterungs-Merkmale:** Nutzen stiftende Merkmale, rufen Begeisterung hervor. Differenzierung zur Konkurrenz
- **Unerhebliche Merkmale**
- **Rückweisungs-Merkmale:** Produktablehnung



Begeisterungs-Merkmale werden zu Leistungs-Merkmalen, und schließlich zu Basis-Merkmalen



Analyse

- Ergebnis der Analyse ist das **Lastenheft**
- Gemäß DIN69905 (Begriffe der Projektabwicklung) beschreibt das Lastenheft die

„vom Auftraggeber festgelegte Gesamtheit der Forderungen an die Lieferungen und Leistungen eines Auftragnehmers innerhalb eines Auftrages“.
- Inhalt variiert in der Praxis stark; besser: „Anforderungssammlung“
 - idealerweise sind auch **Ausgrenzungen** und **Annahmen** dokumentiert
 - zusätzlich sollte dokumentiert werden, von wem die Anforderung stammt
- Fachliche Anforderungen aus **Kundensicht**
- Lücken, Unklarheiten und Widersprüche sind „normal“
 - erschweren allerdings die weitere Planung und Risikoabschätzung



Ist-Analyse

- Das Wort „Analyse“ suggeriert Festlegung des Soll-Zustands
- Ist-Analyse ist aber ebenfalls essenziell
 - Wie haben die Kunden bislang gearbeitet?
 - Stärken des alten Systems werden erst wahrgenommen, wenn sie fehlen
 - Kunde denkt in „Verbesserungen“ zum Ist-Zustand; implizite Erwartung, dass alles „Gute“ erhalten bleibt
- Den Ist-Zustand zu erfassen ist nicht einfach!
 - Kommunikationsfähigkeiten sind gefordert
 - Fragetechniken helfen



Ist-Analyse

- Typischer Dialog für die Ist-Analyse

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.

Was bedeutet morgens?

...

[Beispiel aus LudewigLichter2007]



Soll-Analyse

- Aufgaben des **Analytikers**
 - Sammlung der Kundenanforderungen
 - Danach Klärung mit dem Klienten:
 - Welche Anforderungen widersprechen sich? Klärung unter den Kunden.
 - Welche Anforderungen sind technisch nicht umsetzbar?
 - Rechnet sich bei einigen Anforderungen der hohe Aufwand?



Analysetechniken

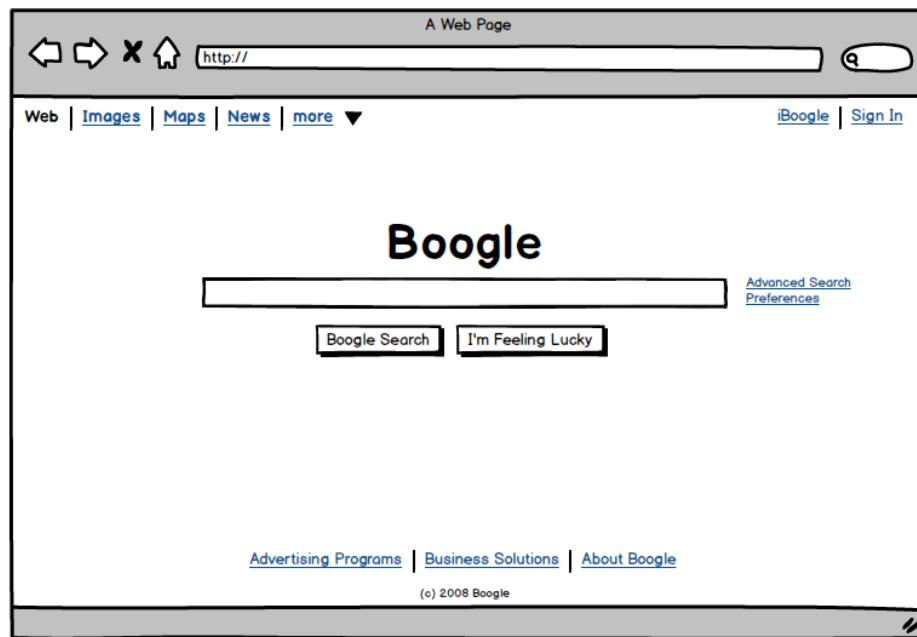
Analysetechnik	Schwerpunkt	Ist-Zustand	Soll-Zustand	Innovationsfolgen
Auswertung vorhandener Daten und Dokumente				
Beobachtung				
Befragung mit	{ geschlossenen strukturierten offenen } Fragen			
Interview				
Modell-Entwicklung				
Experimente				
Prototyping				
partizipative Entwicklung (nur hinsichtlich Analyseteil)				

© Ludewig, Licher, 2006



Analysetechniken

- Beobachtung des Kunden bei der Arbeit
- Durchführung von Workshops zur Aufnahme von Anforderungen
 - falls nötig, z. B. mit GUI-Mockups oder Prototypen



Beispiel:
GUI-Mockup mit Balsamiq

(Quelle: <http://balsamiq.com/>).



Aufgabe

Präzisierung von Anforderungen

Ihr Kunde nennt Ihnen die folgende Anforderung:

„Die Liste aller Kontendaten soll angezeigt werden.“

1. Welche Fragen ergeben sich für Sie?
2. Formulieren Sie diese Anforderung besser!



5 Minuten



Anforderungen – Eigenschaften

Vollständige, eindeutige, testbare Beschreibung einer Anforderung an das System.

Vollständig bedeutet hier, dass alle Details zu der Anforderung zu definieren sind. Es sollten so wenig wie möglich Aspekte als selbstverständlich eingeschätzt werden. Es ist in der Praxis davon auszugehen, dass diese Aspekte dann gerade so realisiert werden, wie es der Projekt-Auftraggeber nicht wollte.

Eindeutig meint, dass die Anforderung mit möglichst einfachen Worten so zu definieren ist, dass keine Missverständnisse zu erwarten sind.

Testbar müssen alle Anforderungen sein, da eine nicht testbare Anforderung nicht korrekt geprüft und daher niemals vom Auftraggeber abgenommen werden kann.



Formulierung von Anforderungen

- **Formulierungsregeln** für Anforderungen (nach C. Rupp)

Regel	Erläuterung
Formulieren Sie jede Regel im Aktiv.	Es wird deutlich, wer etwas tut – der Benutzer (und welcher Benutzer), oder das System
Drücken Sie Prozesse durch Vollverben aus	„erstellt“ im Ggs. zu „hat“ (, „ist“, ...)
Ermitteln Sie unvollständig spezifizierte Bedingungen	„Wenn-dann-sonst“ sollten vollständig angegeben sein
Bestimmen/Überprüfen Sie die Universalquantoren	„nie“, „immer“, „jedes“, ...
Überprüfen Sie Nominalisierungen	„Generierung“, „Datenverlust“ weisen auf komplexe, näher zu beschreibende Prozesse hin

Weitere in: C. Rupp: *Requirements-Engineering und -Management: Professionelle, iterative Anforderungsanalyse für die Praxis*



Formulierung von Anforderungen – **Hinweise**

- Jede Anforderung erhält eine eindeutige Nummer.
 - Nötig für die Verfolgung („Traceability“).
 - Hilft, Querbezüge herzustellen.
- Eine Anforderung muss testbar/prüfbar sein.
- Mischen Sie nicht mehrere Anforderungen in einer.
- Priorisieren Sie Anforderungen.



Anforderungsarten

- Es gibt unterschiedliche Arten/Perspektiven von Anforderungen
- Mögliche Klassifikationen:
 - Domäne (Anwendungsbereich)/Maschine
 - offen/latent
 - objektivierbar/vage
 - funktional/nichtfunktional



Anforderungsarten

Domäne↔Maschine

- **Domäne**

- beschreiben Eigenschaften eines Models eines Teilbereichs der Realität – die „Domäne“ in der das System arbeiten wird
- Beispiel:
Banksoftware: Regeln bzgl. Konten, Abbuchungen, Überziehungen, etc.; „Alle Transaktionen über 10.000 € benötigen die Zustimmung eines Vorgesetzten.“

- **Maschine**

- andere beschreiben Eigenschaften des Systems (der „Maschine“), das im Projekt gebaut wird
- Beispiel:
Banksoftware: Beschreibung, wie Zahlungen abgewickelt werden
→ Das Projekt definiert die „Maschine“, hat aber keinen Einfluss auf die Domänen-Eigenschaften! (diese definieren Einschränkungen/Constraints auf die „Maschine“)

„The speed of light is not an implementation decision.“



Anforderungsarten

offen↔latent

- **offen**
 - bereits vom Kunden formuliert
 - Analytiker braucht nur „einzusammeln“
- **latent**
 - Kunde ist sich der Anforderung nicht bewusst
 - Bedeutung muss ihm vom Analytiker durch Fragen oder Situation vor Augen geführt werden
→ *Hilfe*, dass sich der Kunde der Anforderung bewusst wird
- falls zu einer Frage/einer Anforderung keine Präferenz:
→ Projekt-Option

Anforderungsarten

- objektivierbar

objektivierbar↔vage

- Beispiel: Begrenzung des verfügbaren Speicherplatzes
 - Wichtig für die spätere Prüfung der Anforderung

• **vage**

- Beispiel: „wenig Speicher“, „benutzerfreundlich“, „performant“, ...
 - Nicht prüfbar!

- stets versuchen, Anforderungen in objektivierbarer Form zu dokumentieren; schwer, falls keine sinnvolle Quantifizierung möglich



Anforderungsarten

funktional \leftrightarrow nichtfunktional

- **funktional**
 - unmittelbarer Nutzen der Software
 - z. B. Überweisung tätigen, Nullstellen berechnen, Karten reservieren, ...
 - Kurzbeschreibung eines Systems ist stets eine grobe Charakterisierung der Funktion
- **nichtfunktional (neuer: „Qualitätsanforderung“)**
 - wartbar, performant, ressourcenschonend, intuitiv benutzbar, ...
 - oftmals *weich*
 - oft weggelassen oder durch unverbindliche Schlagwörter ausgedrückt
→ Schlecht! Vermeiden!
- **Grenzbereiche**
 - z. B. Bedienoberfläche kann je nach Anwendungsbereich nichtfunktional oder funktional sein



Bessere Nichtfunktionale Anforderungen

- **Bessere Formulierung nichtfunktionaler Anforderungen**
 - Ausformulierung anstelle von Schlagworten
 - Quantifizierung
 - Formulierung in Bezug auf Normen

Schlechtes Schlagwort	Bessere Formulierung
„schnell“	Das Programm muss eine Datei der Größe 10 MB innerhalb von 10 Sekunden erfolgreich verarbeitet haben.
„portabel“	Das System muss unter den Betriebssystemen Windows 8.1 und Suse Linux Mint 15 KDE arbeiten.
„benutzbar“	Gestaltung der Oberfläche nach <i>DIN EN ISO 9241, Teile 10-17: Grundsätze der Dialoggestaltung</i>
...	...



Prioritäten für Anforderungen

- Nötig, um Entwicklungsarbeit zu ordnen, z.B. nach dem Schema
 - (siehe auch http://en.wikipedia.org/wiki/MoSCoW_method)
 - **M** - MUST (unbedingt erforderlich)
 - **S** - SHOULD (sollte umgesetzt werden, wenn alle MUST-Anforderungen trotzdem erfüllt werden können)
 - **C** - COULD (kann umgesetzt werden, wenn die Erfüllung von höherwertigen Anforderungen nicht beeinträchtigt wird)
 - **W** - WON'T (wird diesmal nicht umgesetzt, aber für die Zukunft vorgemerkt)
 - Priorisierung fällt oftmals nicht leicht!
 - Möglichkeit: „Geld“ an Stakeholder verteilen (fester Geldbetrag; Scheine à 100, 50, 20, etc.) und auf Features „setzen“ lassen.



Feature/Anforderungs-Interaktion

- Features/Anforderungen sind populär, da sie einfach hinzuzufügen und zu ändern sind.
- Die „dunkle Seite“ sind **Interaktionen** zwischen Features.
- Management ist nötig!
 - potenzielle Interaktionen erkennen und dokumentieren
 - „gute“ von „schlechten“ Interaktionen unterscheiden
- Literatur:
<http://www2.research.att.com/~pamela/faq.html>



Feature-Interaktion Beispiel

- Beispiel: Behandlung von Besetzt-Zustand in der Telekommunikation
- Möglichkeiten
 - Weiterleitung des Anrufs an jemand anderen
 - den Angerufenen unterbrechen
 - den Anruf später wiederholen
 - die Mailbox aktivieren
- Falls zwei Aktionen B1 und B2 gleichzeitig aktiv sind, wird diejenige Aktion mit der höhere Priorität ausgeführt, die andere Aktion wird ignoriert
 - das ist gut so
 - die Implementierung sollte so sein, dass neue Aktionen ohne Modifikation der alten Aktionen später hinzugefügt werden können (ansonsten müssten die Aktivierungsbedingungen angepasst werden!)



Feature-Interaktion, schlechte Beispiele

- Bob hat Weiterleitung zu Carol aktiviert; Alice ruft Bob an, der Anruf wird an Carol weitergeleitet und die Adresse auf Bob geändert. Falls Carol Anrufe von Alice blockieren möchte, kommt dieser Anruf trotzdem durch.
- Alice ruft eine Hotline an; ein Feature wählt dort Bob als den aktuellen Bearbeiter aus und leitet den Anruf an ihn weiter. Bobs Telefon ist ausgeschaltet und die Mailbox wird aktiviert. Es wäre besser, wenn dann ein anderer Sachbearbeiter ausgewählt würde.
- Eine Familie möchte in einer Telefonkonferenz eine automatisierte Hotline anrufen. Der Konferenzruf filtert die nötigen DTMF-Töne zur Steuerung der Hotline, da diese Töne für die Steuerung der Konferenz selbst verwendet werden.
- Bob schickt eine E-Mail an eine Mailingliste. Ein Mitglied dieser Liste ist in Urlaub und hat eine Abwesenheitsnachricht aktiviert. Bob erfährt nun die Identität dieses Mitglieds.

(weitere interessante Interaktionen unter
<http://www2.research.att.com/~pamela/faq.html>)



Feature Interaktion – Wunsch und Realität in Anforderungserfassung



Wunsch: additive Komplexität



Realität: multiplikative Komplexität

Anforderungen lassen sich nicht „einfach“ additiv später hinzufügen!

→ auf eine gründliche „Requirements Analyse“ kann/darf im Projekt nicht verzichtet werden!
(die Folge wären u.a. teure Auswirkungen auf Architektur/Wartbarkeit/etc.)



Demos

- Lastenheft aus früheren Projekten
- Lastenheft aus dem WP „Das Softwareprojekt“



Spezifikation / Fachkonzept



Spezifikation

- Begriff: **(Software-Requirements-)Spezifikation**

software requirements specification (SRS) – Documentation of the essential requirements (functions, performance, design constraints, and attributes) of the software and its external interfaces.

...

IEEE Std 610.12 (1990)

- Auch: **Fachkonzept**
 - <http://de.wikipedia.org/wiki/Fachkonzept>



Spezifikation

Eine **Spezifikation** (vom lateinischen *specificatio* für die „Auflistung“ oder das „Verzeichnis“) ist eine **formalisierte Beschreibung** eines Produktes, **eines Systems** oder einer Dienstleistung. Ziel der Spezifikation ist es, Merkmale zu definieren und zu quantifizieren (Toleranzwerte), mit denen das Werk oder die Dienstleistung des Auftragnehmers bei der Übergabe an den Auftraggeber bzw. Käufer geprüft und durch den Auftraggeber abgenommen werden kann, bzw. nach der der Auftragnehmer bzw. Verkäufer die Bezahlung fordern kann, wenn die Merkmale der Spezifikation erreicht wurden. Die Spezifikation enthält in der Regel für jede spezifizierte Eigenschaft eine präzise Referenz zu der anzuwendenden Prüfmethode für das jeweilige Merkmal.

aus: Wikipedia



Spezifikation

- Eine Spezifikation beschreibt die „**Außensicht**“ des Systems.
- Wenn „Programmierer“ eine Spezifikation schreiben:
 - „schlecht getarnter Entwurf“ (→ Programmierer denken in technischen Lösungen!)
 - Anforderungen fehlen
 - Technische Entscheidungen werden vorzeitig und von den falschen Leuten getroffen



Spezifikation

- Was ist in einer Spezifikation enthalten?

