Softwaretechnik C - Softwaremanagement

LE 05: Anforderungsmanagement

Fachhochschule Dortmund
University of Applied Sciences

Organisatorisches

- VL am 22.11.2022
  - Findet **ausschließlich online (Webex)** statt

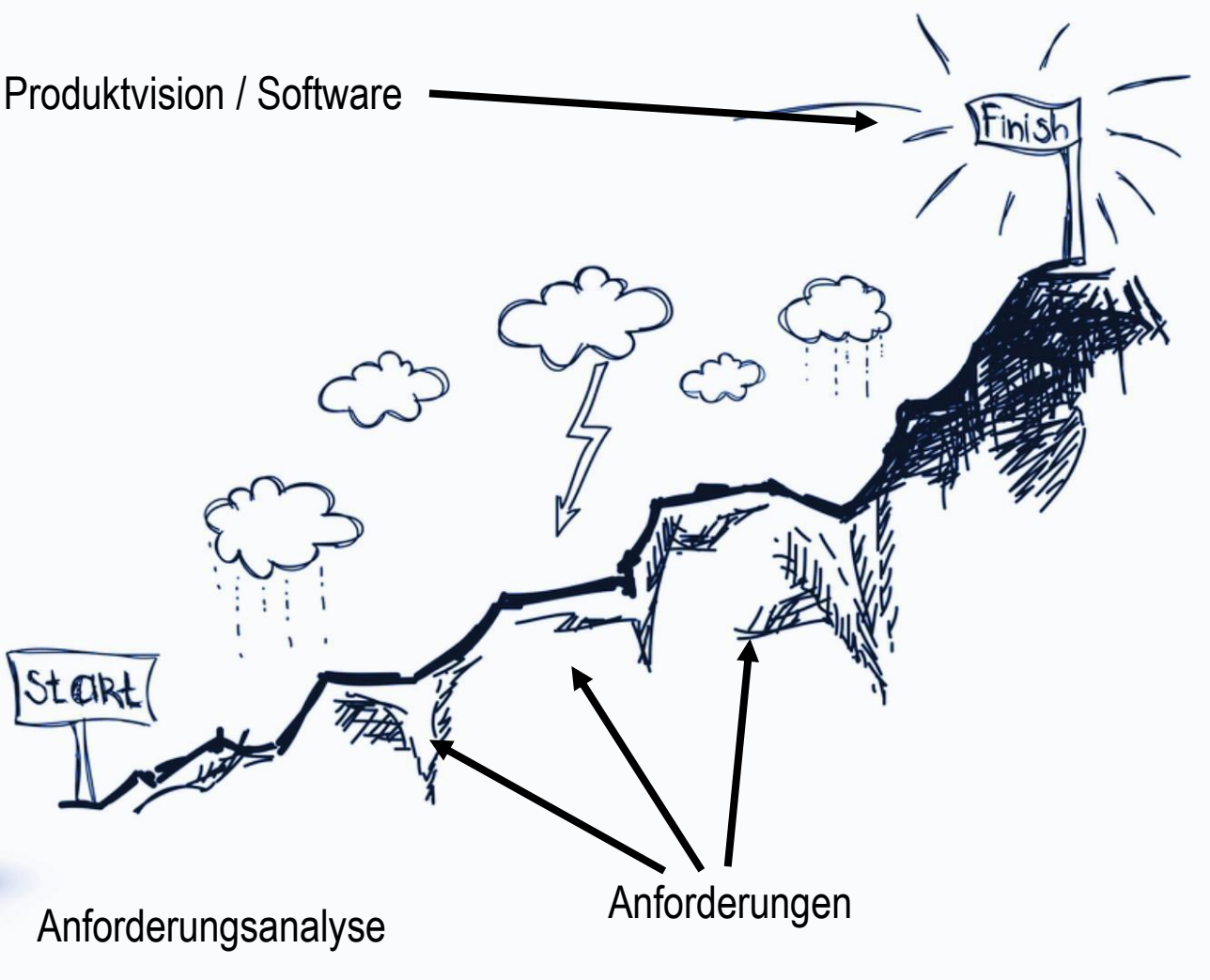
Agenda

- Anforderungsmanagement
  - Einleitung
  - Anforderungstypen
  - Anforderungserhebung
  - Anforderungsmanagement in der agilen Softwareentwicklung
  - Anforderungsänderungen

- Anforderungsmanagement

  - Einleitung
  - Anforderungstypen
  - Anforderungserhebung
  - Anforderungsmanagement in der agilen Softwareentwicklung
  - Anforderungsänderungen

Einleitung



Quelle: https://www.juptr.io/juptrblogs/_07cdc8da-4e43-4061-a64d-dc08e3f81410.html

Einleitung

- Die Anforderungen an ein Softwareprodukt
  - zu **ermitteln**,
  - zu **spezifizieren**,
  - zu **analysieren**,
  - zu **validieren** und
  - daraus eine **fachliche Lösung** abzuleiten bzw. ein Produktmodell zu entwickeln,
- gehört zu den anspruchsvollsten Aufgaben innerhalb der Softwaretechnik (vgl. Balzert, 2009)

Definitionen

Die Anforderungen an ein Softwaresystem

- legen fest, **was** das Softwaresystem leisten soll,
- **Welche Eigenschaften** es aufweisen muss und soll und
- definieren **Einschränkungen** seiner Funktion und Implementierung.

Die Anforderungsanalyse

- dient der Festlegung der **quantitativen und qualitativen Eigenschaften** eines Software-Produkts
- aus Sicht des Auftraggebers bzw. des Kunden.

Definitionen

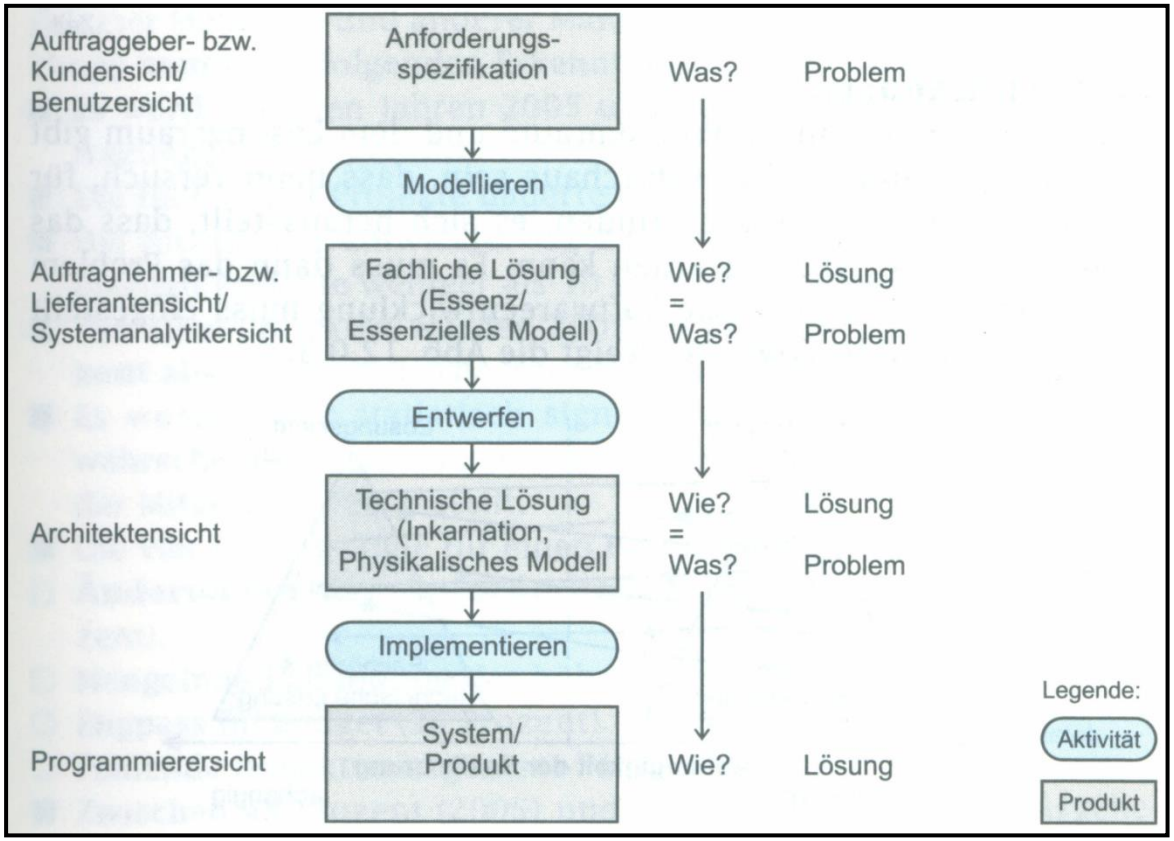
Das Anforderungsdokument

- Wird auch als „**Anforderungsspezifikation**“ bezeichnet
- umfasst sämtliche **relevanten und zu realisierenden Anforderungen** an ein Softwareprodukt
- ist integraler Bestandteil des **Pflichtenhefts**, das der Auftragnehmer (oftmals in enger Kooperation mit dem Auftraggeber) erstellt
- Die Anforderungsspezifikation sollte alle **relevanten funktionalen und nicht-funktionalen Anforderungen** beinhalten
- Umfassende und präzise **Beschreibung** sämtlicher **Funktionen, Dienste und Eigenschaften** des zu entwickelnden Softwaresystems
- Unvollständige Anforderungen sollten – sofern bekannt – auch als unvollständig im Dokument gekennzeichnet werden
- Zentrales **Ergebnisdokument** der **Anforderungsanalyse**

Problem und Lösung

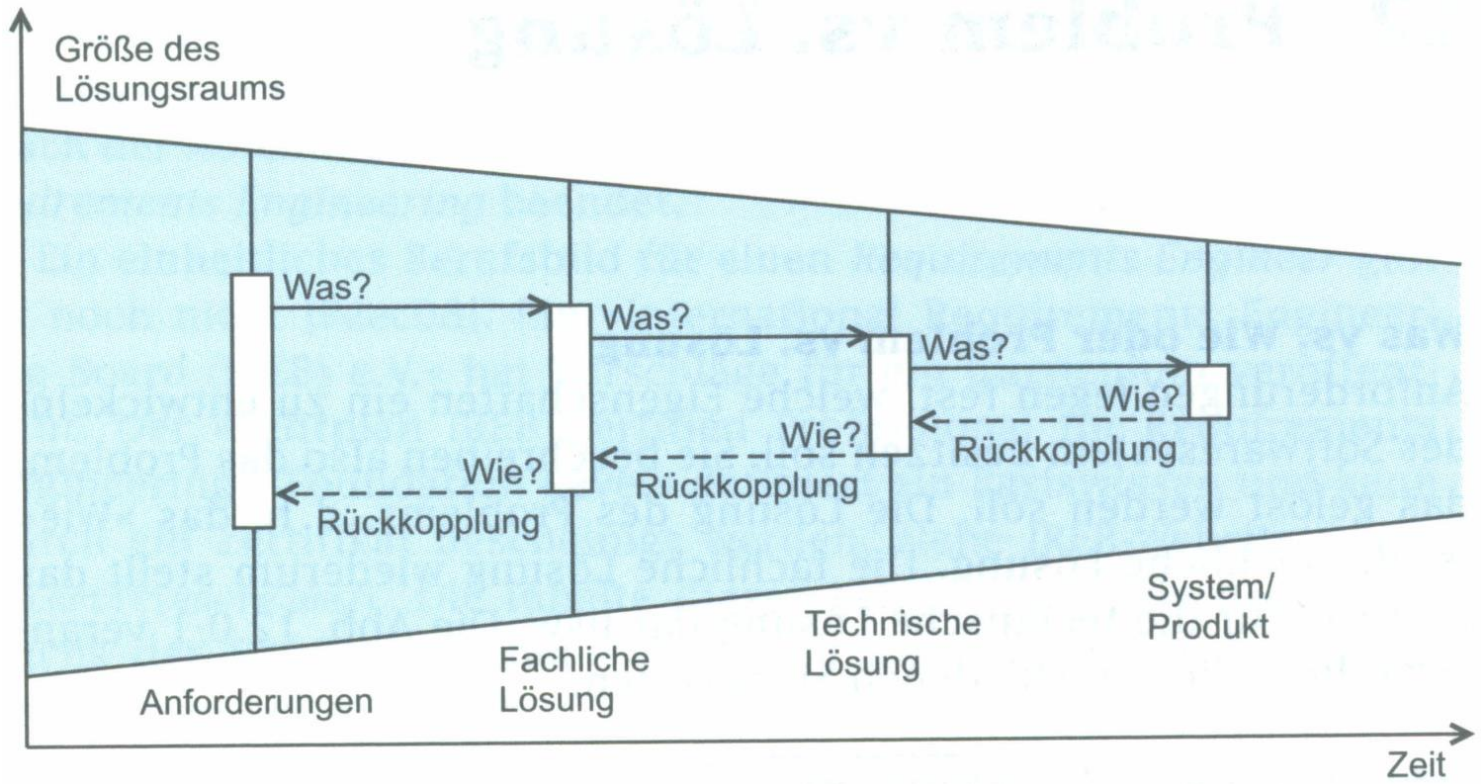
- Anforderungen legen fest, welche **Eigenschaften** ein zu entwickelndes Softwaresystem besitzen soll
- Sie beschreiben somit das **Problem**, das gelöst werden soll
- Die Lösung des Problems (Anforderungsspezifikation), das heißt das „wie“, ist die **fachliche Lösung**
- Die fachliche Lösung stellt wiederum das Problem für die technische Lösung dar usw.

Problem und Lösung



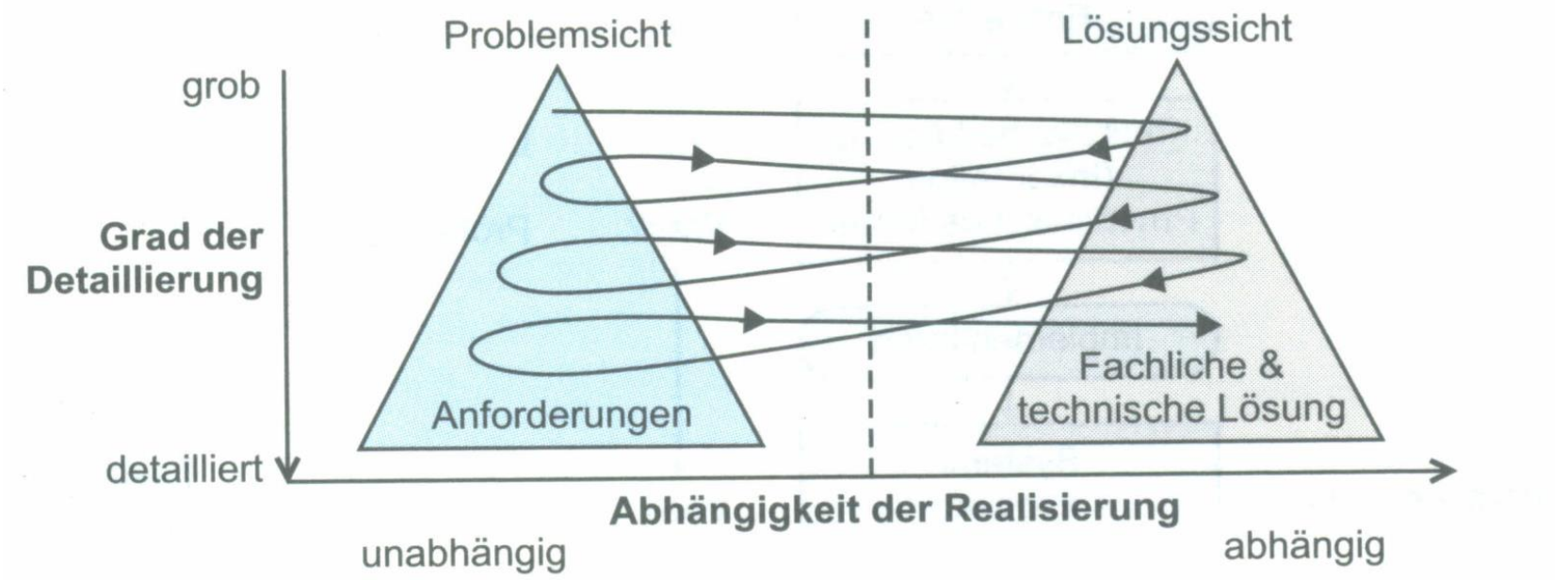
Lösungsraum

- Im Laufe der Softwareentwicklung wird **Lösungsraum** immer stärker eingeschränkt
- Daher ist es wichtig, dass bei der initialen Aufstellung der Anforderungen möglichst keine Einschränkungen für nachfolgende Aktivitäten erfolgen



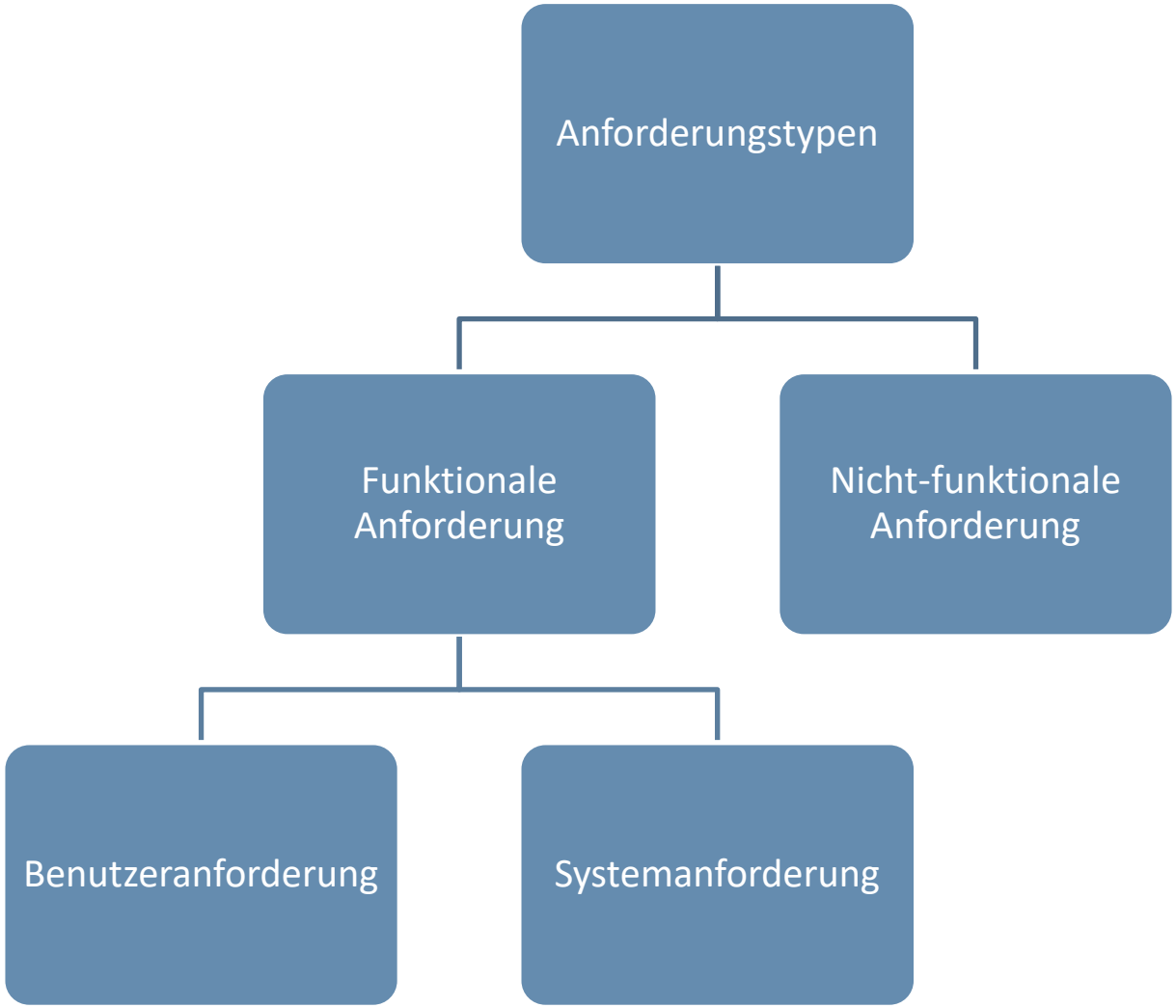
Wechselwirkung

- Zwischen **Problem- und Lösungsraum** gibt es **Abhängigkeiten**
- Es kann sich z.B. herausstellen, dass ein Problem nicht wie geplant gelöst werden kann:
  - => dann ist eine Modifikation des Problems notwendig oder
  - => ggf. auch die Einstellung der weiteren Softwareentwicklung



- Anforderungsmanagement
  - Einleitung
  - Anforderungstypen
  - Anforderungserhebung
  - Anforderungsmanagement in der agilen Softwareentwicklung
  - Anforderungsänderungen

Anforderungstypen



Funktionale Anforderungen

- Beschreiben die **Funktionen und Dienste**, die ein Softwaresystem bereitstellen soll
  - Reaktion des Softwaresystems auf bestimmte Eingaben
  - Auch: Beschreibung, was das System NICHT leisten soll

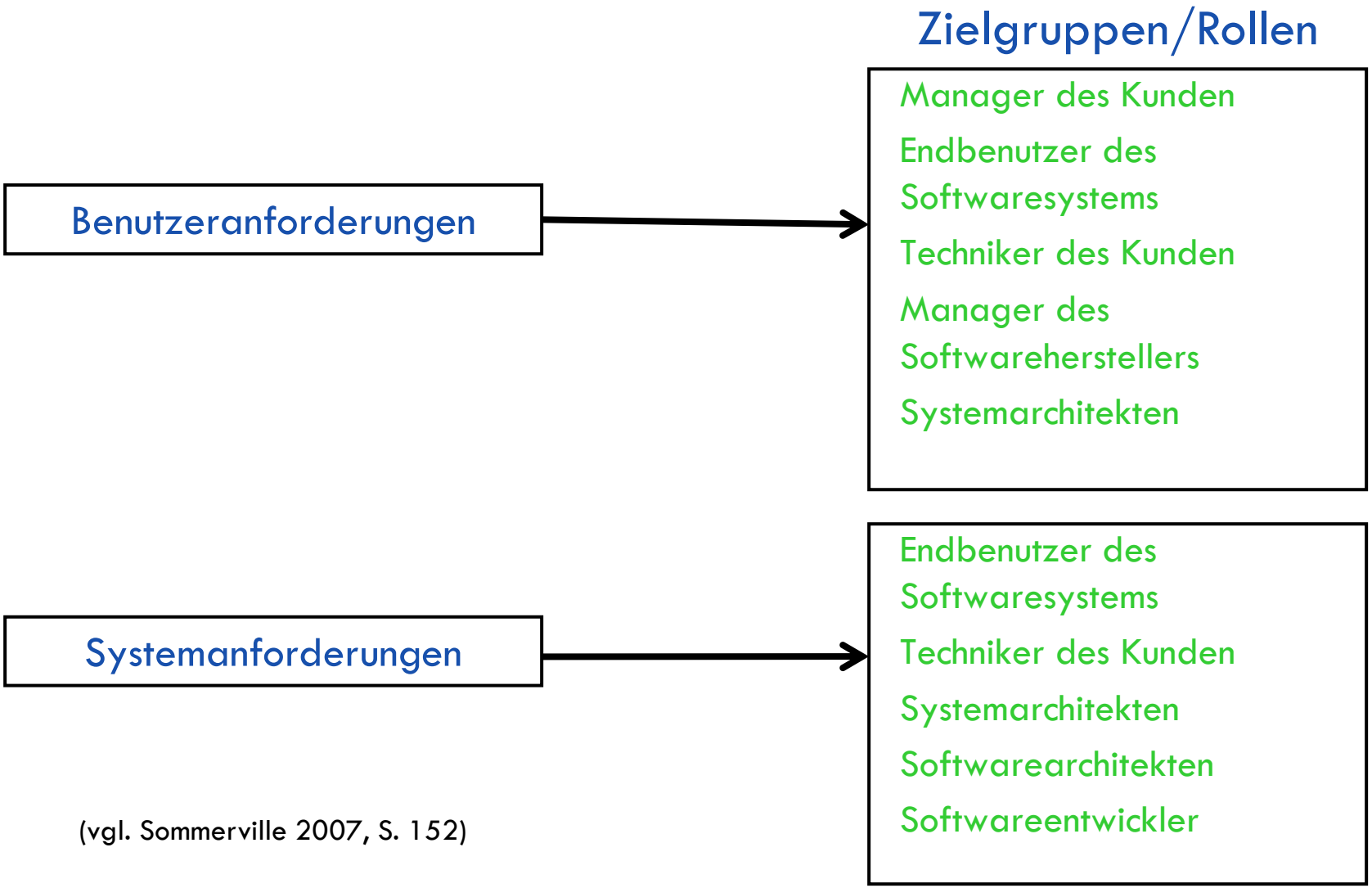
- Man unterscheidet hierbei
  - **Benutzeranforderungen** und
  - **Systemanforderungen**

Funktionale Anforderungen

- Benutzeranforderungen
  - Sind Aussagen in natürlicher Sprache
  - Intuitiv verständliche Diagramme zur Beschreibung der Funktionen und Dienste, die das spezifizierte Softwaresystem leisten soll
  - die Randbedingungen, unter denen es betrieben wird

- Systemanforderungen
  - Legen die Funktionen, Dienste und Beschränkungen detailliert und möglichst präzise fest (=> Quantifizierbarkeit)
  - Es muss genau spezifiziert werden, welche Anforderungen zu implementieren sind
  - Nah an der Entwicklungsebene

Funktionale Anforderungen



(vgl. Sommerville 2007, S. 152)

Funktionale Anforderungen

- Beispiel
  - Benutzeranforderung
    - Als Arzt möchte ich die Anamnese meiner Patienten dokumentieren, damit ich diagnostische und therapeutische Entscheidungen treffen kann.
  - Systemanforderung
    - Die Erhebung einer Anamnese soll auf Basis eines definierten Formulars erfolgen.
    - Das Anamnese-Formular hat definierte Pflichtfelder.
    - Sind die Pflichtfelder nicht ausgefüllt, soll der Benutzer eine Fehlermeldung erhalten.

Nichtfunktionale Anforderungen

- **Eigenschaften** eines **Softwaresystems**
- Anforderungen, die **NICHT** die durch das Softwaresystem bereitzustellenden Funktionen bzw. zu leistenden Dienste betreffen
- Sind selten an einzelne Systemfunktionen gebunden
- Oftmals sind einzelne nichtfunktionale Anforderungen deutlich relevanter als einzelne funktionale Anforderungen
- Es können hierbei unter anderem auch:
  - Vorgehensmodelle der Softwareentwicklung
  - Programmiersprachen und/oder
  - Entwicklungswerkzeuge
- festgelegt werden

Nichtfunktionale Anforderungen

- Typische nichtfunktionale Anforderungen
  - Zuverlässigkeit, Fehlertoleranz
  - Aussehen und Handhabung („Look and Feel“)
  - Benutzbarkeit, Verständlichkeit, Bedienbarkeit
  - Performanz, Durchsatz, Antwortzeiten, Reaktionszeiten
  - Hardware-Anforderungen, Ressourcenbedarf
  - Betrieb und Umgebungsbedingungen
  - Portierbarkeit, Installierbarkeit,
  - Skalierbarkeit, Verfügbarkeit, Erweiterbarkeit
  - Sicherheitsanforderungen: Vertraulichkeit, DSDS
  - Korrektheit: fehlerfreie Ergebnisse, Fehlertoleranz
  - Unterstützung von Standards, Normen und Vorschriften
  - Einzusetzende Sprachen, Methoden und/oder Werkzeuge

Nichtfunktionale Anforderungen

- Beispiele
  - „Das zu entwickelnde Softwaresystem muss hochverfügbar (nach AEC-4) betreibbar sein.“
  - „Das zu entwickelnde Softwaresystem muss mindestens 85.000 Datensätze zur eGK-Personalisierung innerhalb von 24 Stunden produzieren können.“
  - „Der Software- und Systementwicklungsprozess folgt dem Vorgehen und den Ergebnissen des V-Modell XT.“
  - „Das Softwaresystem soll den Benutzern des Systems keine persönlichen Informationen über Kunden preisgeben, abgesehen von der eindeutigen Referenznummer.“
  - „Bei 90% der Benutzerinteraktionen, darf eine Antwortzeit von einer Sekunde nicht überschritten werden. “

Abgrenzung funktionale & nichtfunktionale Anforderungen

Funktionale Anforderungen	Nichtfunktionale Anforderungen
WAS soll das System leisten	**WIE** soll das System oder eine Einzelfunktion arbeiten
Anforderungen an Dienste und Verarbeitung	Anforderungen an Qualität wie Performanz oder Zuverlässigkeit
Anforderungen an Verhalten	Anforderungen an Benutzbarkeit

- Anforderungsmanagement

  - Einleitung
  - Anforderungstypen
  - Anforderungserhebung
  - Anforderungsmanagement in der agilen Softwareentwicklung
  - Anforderungsänderungen

Stakeholderanalyse

- Die **Anforderungserhebung** wird mit partizipativer Beteiligung der **Stakeholder** und **späteren Benutzer/innen** des Softwaresystems durchgeführt
- Somit sind **initial** die relevanten Stakeholder zu **identifizieren**
  - beeinflussen die Softwareentwicklung, aber auch die Einführung und den Betrieb des entwickelten Softwaresystems
- Liefern direkt oder indirekt **Informationen** über
  - die Ziele,
  - die Rahmenbedingungen,
  - den Kontext,
  - die Anforderungen und auch
  - die Risiken des zu entwickelnden Softwaresystems

Stakeholderanalyse

- Stake (Anspruch, Anteil)
- Holder (Besitzer, Eigentümer)
- Stakeholder sind
  - Interne und externe **Personen** oder **Personengruppen**,
  - Von **Aktivitäten** direkt oder indirekt betroffen oder haben ein konkretes Interesse an den Aktivitäten
  - natürliche oder juristische Personen
- Typische Stakeholder:
  - Kunden und Benutzer eines (zu entwickelnden) Softwaresystems
  - Bearbeiter nachgelagerter Entwicklungsphasen, bspw. Integration oder Qualitätssicherung
  - Product Owner
  - Lieferanten oder Partner
  - Kapitalgeber, Eigentümer, Aktionäre

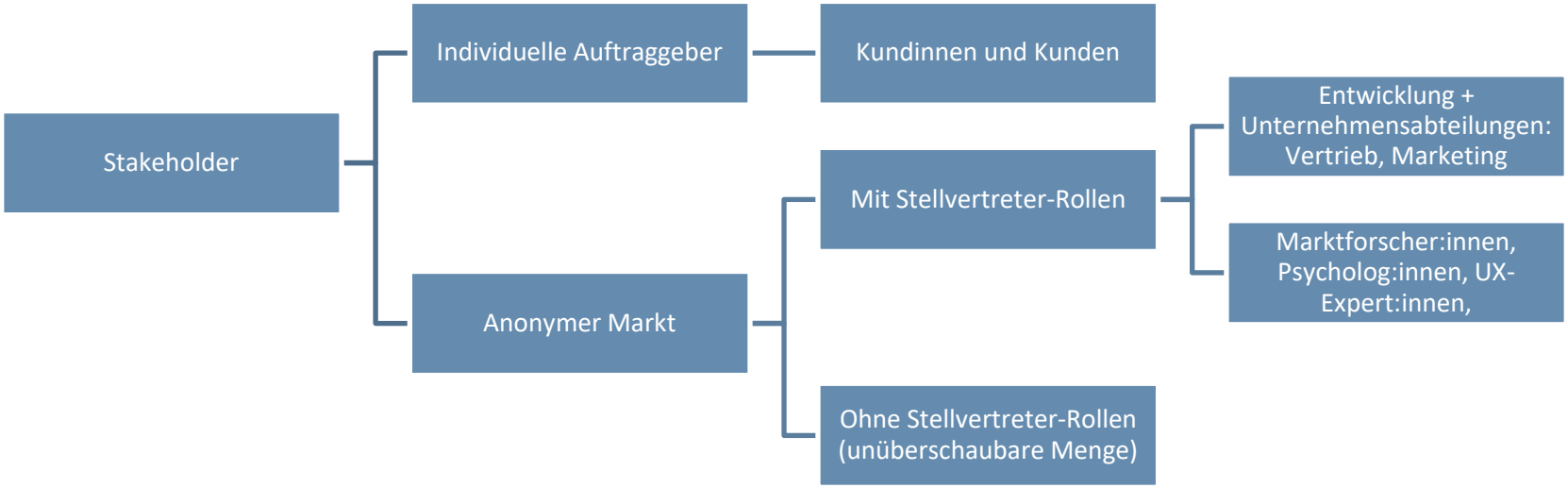
Stakeholderanalyse

- **Wichtig**: Stakeholder **frühzeitig einbinden**
- Beim Start des Requirements-Engineering-Prozesses werden oftmals noch nicht alle Stakeholder gefunden
- Kontinuierlich darauf achten, ob es weitere Stakeholder gibt
- Werden wichtige Stakeholder **übersehen**, können z.B. bei der Inbetriebnahme massive **Probleme** und **Anforderungsänderungen** entstehen, die zeitaufwändig und kostenintensiv zu beheben sind
- Daher:
  - wichtige Stakeholder frühzeitig in die Kommunikation und Prozesse einbinden
  - Stakeholdern Verantwortung und Aufgaben übertragen
  - Stakeholder zu wichtigen Besprechungen einladen

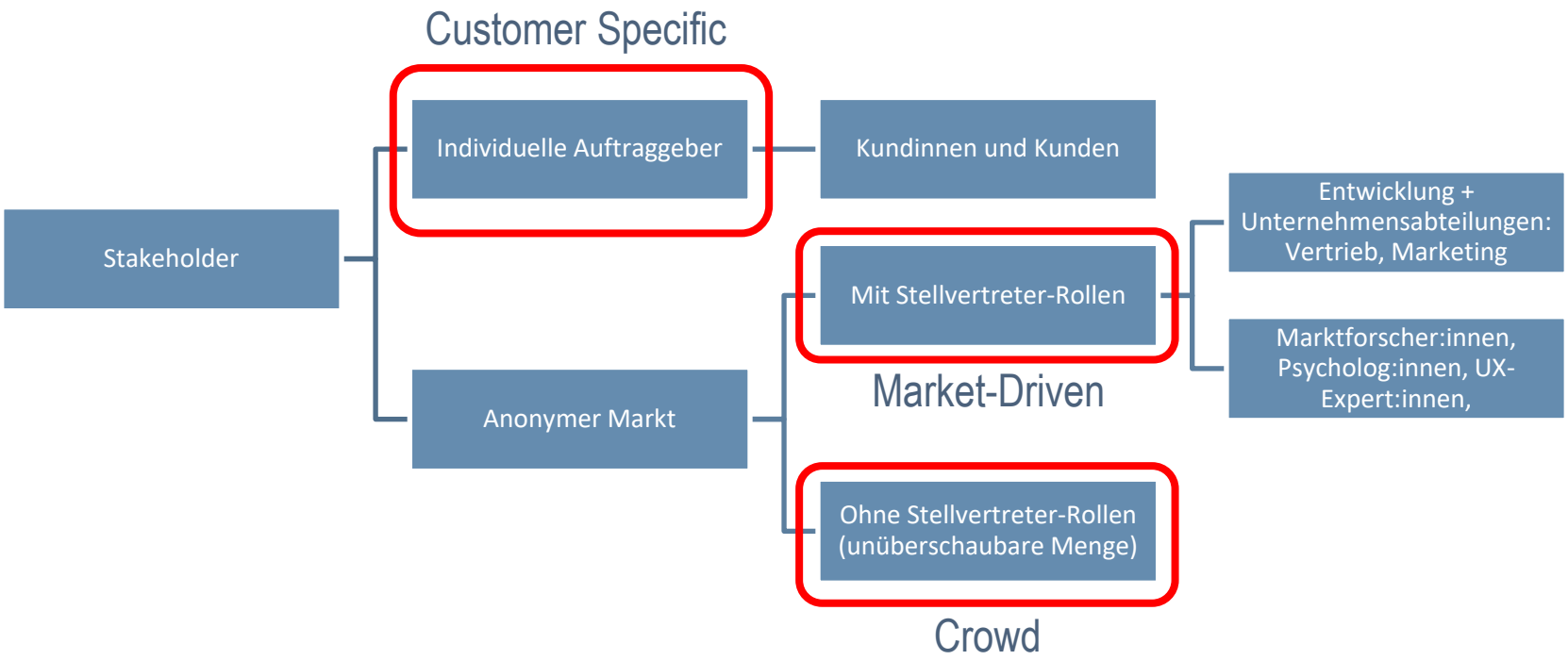
Stakeholderanalyse

- Ergebnis der Stakeholder-Analyse ist ein **Dokument** mit
  - einer Liste der Namen der relevanten Stakeholder
  - mit Kontaktdaten (Tel, Mobil, E-Mail) und individueller Verfügbarkeit
  - mit den jeweiligen Verantwortungsbereichen und Wissensgebieten
  - mit den jeweiligen Vertretern
  - mit einer kurzen textuellen Begründung, warum der- bzw. diejenige als Stakeholder eingestuft wird

Stakeholder-Typen



Stakeholder-Typen



Customer Specific Requirements Engineering

Merkmale

- Individuelle Anforderungen von Auftraggeberinnen und Auftraggebern
- Individuelle Anforderungen werden mit Auftraggebern erarbeitet
  - Anforderungsdokument / Backlog
- und getestet
  - Bspw. Akzeptanztests von Releases vor Produktivnutzung
- → kooperativer Prozess
- Direkte Zusammenarbeit in allen Phasen des Entwicklungszyklus möglich
- Alle relevanten Anforderungen sollen bekannt und im erforderlichen Detaillierungsgrad verstanden sein
- Anforderungsquellen: Stakeholder, Dokumente & Artefakte, Systeme im Betrieb

Geeignete Erhebungstechniken

- Gruppen- & Kreativitätstechniken, z.B. Fokusgruppen
- Befragungstechniken, z.B. Interviews und Fragebögen
- Beobachtungstechniken, z.B. Apprenticing
- Dokument-/Artefaktzentrierte Techniken, z.B. Systemarchäologie

Market-Driven Requirements Engineering

Fachhochschule Dortmund
University of Applied Sciences

Merkmale

- Keine direkten Auftraggeberinnen oder Auftraggeber
- Keine direkte Zusammenarbeit mit Kundinnen und Kunden oder Benutzerinnen und Benutzern möglich
- Stattdessen: stellvertretende Akteure wie Vertriebs- oder Marketing-Expertinnen & -Experten, die gemeinsam mit dem Entwicklungsteam Anforderungen definieren
- Skalierbarkeit von Anforderungen: möglichst Relevanz für viele Käuferinnen und Käufer statt individuell relevante Anforderungen
- Ziel: Gewinnmaximierung durch hohe Verkaufszahlen eines Produkts

Geeignete Erhebungstechniken

- Marktanalysetechniken
- Befragungstechniken mit Stellvertreterinnen und Stellvertretern, z.B. Interviews
- Gruppen- und Kreativitätstechniken mit Stellvertreterinnen und Stellvertretern, z.B. Design Workshops
- Dokument-/Artefaktzentrierte Techniken, z.B. Systemarchäologie

Crowd Requirements Engineering

Merkmale

- Zielgruppe unbekannt
- Fokus auf Einsatz von Techniken zur Sammlung und Auswertung von Daten über Nutzung, Skalierbarkeit oder Relevanz eines Produkts
- Daten müssen zunächst erhoben und gesammelt werden
  - Ziel: heterogene und repräsentative Datensammlung
- Gewinnung von Anforderungen durch
  - Crowd Sourcing: Auslagerung von Aufgaben an Crowd → Daten-, Ideen- und Lösungssammlung durch Crowd
  - geeignete Analysetechniken gesammelter Daten
- Zentrale Voraussetzung: Motivation der Mitglieder der Zielgruppe (Crowd)

Geeignete Erhebungstechniken

- Befragungstechniken (i.d.R. nicht adressiert), z.B. Fragebögen, Feedback
- Dokument-/Artefaktzentrierte Techniken,
  - z.B. Implementierung einer Protokollfunktion zur Sammlung (anonymer) Protkolldaten über Nutzungsverhalten zu einem Produkts
- Crowd Sourcing

Wesentliche Unterschiede

Charakteristik	Customer Specific RE	Anonymer Markt
Zentrale Ziele	Compliance zur Anforderungsdefinition	Time-to-Market, Wettbewerbsvorteil
Erfolgs-kriterien	Kundenzufriedenheit, Akzeptanz	Verkaufszahlen, Marktanteil, Gewinn, Produkt-Reviews
Kundinnen und Kunden	Wenige Kundinnen & Kunden, enger Kontakt	initial unbekannt, anonym oder stellvertretend, wenige bis viele
Erhebung	Anforderungen mit Kund:innen/Zielgruppen erhoben, traditionelle Erhebungstechniken	Häufig durch Entwicklungsteam erhoben für erstes Release des Produkts
Spezifikation	Anforderungsdokument als Vertrag	Weniger formale Spezifikation
Validierung	Zusammen mit Kund:innen vor Auslieferung	Nach Auslieferung auf den Markt

Methoden zur Anforderungserhebung und -ermittlung

- Kreativitätstechniken 
- Beobachtungstechniken 
- Befragungstechniken 
- Artefaktbasierte Techniken 

- [Rupp, C. (2014) : Requirements-Engineering und –Management, Aus der Praxis von klassisch bis agil. 6. Aufl., München: Hanser.]

Methoden zur Anforderungsermittlung
Kreativitätstechniken

Fachhochschule Dortmund
University of Applied Sciences

- Brainstorming
  - 2 Phasen: Generation & Evaluation Phase
  - Hilft bei der kreativen Entwicklung von Lösungen für spezifische Probleme
- Methode 6-3-5
  - Kreativitätstechnik zur Ideenfindung, bspw. im Rahmen von Design Thinking
  - 6 Teilnehmer, je 3 Ideen, 5mal Weitergeben
- Storyboard
  - Dient der Strukturierung von Inhalten
  - Hilft bei der Fokussierung auf Kerninhalte oder Strukturierung einer Argumentation
  - Bspw. Customer Journey
- Rollenspiele
  - Bsp. Walt-Disney-Methode: Problembetrachtung durch Einzelpersonen aus 4 Blickwinkeln: Träumer, Realist, Kritiker, Neutraler

Methoden zur Anforderungsermittlung
Beobachtungstechniken

- Problem: Stakeholder können ihr Wissen nur schwer weitergeben und/oder haben nur begrenzte Zeit zur Verfügung
  - ⇒ hier helfen Beobachtungstechniken
- Requirements-Engineer beobachtet einen Stakeholder, meistens einen Anwender des Systems, bei dessen Arbeit
- Hierbei werden Arbeitsschritte erfasst und dokumentiert
  - Daraus können relevante Arbeitsabläufe analysiert werden
- Methode funktioniert nur, wenn ein (Software-) System vorliegt, bei dem Abläufe auch tatsächlich gut beobachtet werden können
- Ziele:
  - Identifikation von Arbeitsschritten oder Prozessen
  - Bestimmung der Funktionsweise von Teams oder Prozessen
  - Erhebung von Arbeitsaufwänden einzelner oder sequentieller Arbeitsschritte
  - Identifikation von Interaktionen zwischen Akteuren

Methoden zur Anforderungsermittlung
Beobachtungstechniken

Fachhochschule Dortmund
University of Applied Sciences

- Feldbeobachtung (Contextual Inquiry)
  - Dauerbeobachtung: Stakeholder wird über definierten Zeitraum beobachtet. Die Dokumentation erfolgt schriftlich oder per Video- oder Tonaufnahme.
  - Multimomentaufnahme: Stakeholder werden in vorab definierten Zeitabständen beobachtet. Gewonnene Informationen werden anschließend hinsichtlich Signifikanz analysiert.
- Apprenticing
  - Requirements Engineer erlernt die Tätigkeiten des Stakeholders und führt sie selbstständig durch
  - Optimierung von Erkenntnisgewinn durch selbstständiges Ausführen der Tätigkeiten

Methoden zur Anforderungsermittlung
Befragungstechniken

- Häufig eingesetzte Ermittlungs- und Erhebungsmethode, um Erwartungen, Wünsche und Bedürfnisse von Stakeholdern zu erfragen
- So können vor allem Anforderungen ermittelt werden, die den Stakeholdern sehr präsent bzw. bewusst sind
- Erfolg hängt sehr stark von den sprachlichen Fähigkeiten der Beteiligten ab
- Nichtfunktionale Aspekte lassen sich nur schwer in Worte fassen
- es gibt sowohl mündlich als auch schriftlich durchgeführte Befragungstechniken:
  - Interview
  - Fragebogen

Methoden zur Anforderungsermittlung
Befragungstechniken

Fachhochschule Dortmund
University of Applied Sciences

- Standardisiertes Interview:
  - Reihenfolge und Formulierung der Fragen ist vorher festgelegt
  - Keine Zusatzfragen möglich
  - Antwortmöglichkeiten können im Vorfeld festgelegt werden (geschlossene Fragen)

- Teil-standardisiertes Interview:
  - Interview besteht aus einem vorgegebenen sowie einem offenen Teil
  - Offener Teil des Interviews erlaubt freie Fragen und Antworten
  - Zusatz- und Verständnisfragen möglich
  - Fragen können in unterschiedlicher Reihenfolge gestellt werden

- Nicht-standardisiertes Interview:
  - es wird lediglich ein Thema oder ein grober Leitfaden festgelegt
  - es können bestehende Fragensammlungen hinzugezogen werden

Methoden zur Anforderungsermittlung
Artefaktbasierte Techniken

- Es kann sein, dass nur ein (Alt-) System selbst und seine Dokumentation als Anforderungsquellen zur Verfügung stehen
- Mit artefaktbasierten Techniken lässt sich ermitteln, wie die Funktionalitäten eines Systems umgesetzt sind
- Diese werden oft (stillschweigend) vorausgesetzt und werden in Befragungen nicht genannt
- Legen Sie vor dem Einsatz von artefaktbasierten Techniken fest,
  - welche Teile des Altsystems ins neue Softwaresystem übernommen werden und
  - welche Teile neu zu entwickeln sind

Methoden zur Anforderungsermittlung
Artefaktbasierte Techniken

- Systemarchäologie
  - Analyse von unzureichend dokumentierter Software
  - Techniken nutzen Tools und beinhalten bspw.
    - Reverse-Engineering
    - API documentation
    - Durchsuchen von Quellcode nach Schlüsselwörtern
    - Debugging und Tracing
    - Software-Visualisierung
- Reuse
  - Wiederverwendung bereits entwickelter, validierter und verifizierter Elemente (Artefakte)
  - Bspw. bereits entwickelter Software-Komponenten oder -Module

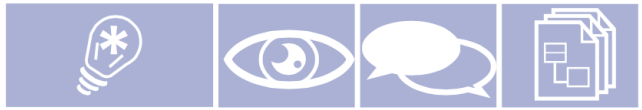
Methoden zur Anforderungsermittlung
Methodenauswahl

- **Auswahl** der **geeigneten Ermittlungsmethode** ist nicht trivial, da sie von vielen Faktoren beeinflusst wird
- Auswahlmatrix
  - Entscheiden Sie im Vorfeld, **welche Art von Informationen** Sie erheben möchten => Kreativitäts-, Beobachtungs-, Befragungs- oder artefaktbasierte Techniken
  - Analysieren Sie die vorliegenden Einflussfaktoren und markieren Sie die drei bis vier am **stärksten ausgeprägten Einflussfaktoren** in der Matrix
  - Suchen Sie nun die Ermittlungstechniken aus der in Schritt 1 gewählten Gruppe aus, die für Sie die beste Bewertung bezüglich der Einflussfaktoren aus Schritt 2 besitzen
- Meist ist ein **Mix an Ermittlungstechniken** sinnvoll bzw. erforderlich

Auswahlmatrix für Ermittlungstechniken

Legende:

–	nicht empfohlen
0	kein Einfluss=> ist anwendbar
+	empfohlen
++	sehr empfohlen



	Brainstorming	Methode 6–3–5	Wechsel der Perspektive	Feldbeobachtung	Apprenticing	Fragebogen	Interview	Systemarchäologie	Reuse
Menschliche Einflussfaktoren									
Geringe Motivation der Stakeholder (aktiv mitzuwirken)	–	–	–	+	–	0	+	++	++
Schlechte kommunikative Fähigkeiten	–	–	–	++	++	–	+	++	++
Geringes Abstraktionsvermögen	–	–	–	++	++	0	+	++	++
Viele verschiedene Meinungen	+	++	+	++	++	+	0	0	0
Machtgefälle zwischen beteiligten Personen	–	+	–	0	0	0	0	0	0
Problematische Gruppendynamik	–	+	+	0	0	0	0	0	0

Auswahlmatrix für Ermittlungstechniken

Legende:

–	nicht empfohlen
0	kein Einfluss=> ist anwendbar
+	empfohlen
++	sehr empfohlen

	Brainstorming	Methode 6–3–5	Wechsel der Perspektive	Feldbeobachtung	Apprenticing	Fragebogen	Interview	Systemarchäologie	Reuse
Organisatorische Einflussfaktoren									
Entwicklung für den komplexen Markt	++	+	+	–	–	++	0	+	0
Fixiertes, knappes Projektbudget	++	++	+	+	–	–	+	–	++
Hohe örtliche Verteilung der Stakeholder	–	0	–	0	0	++	0	0	0
Schlechte zeitliche Verfügbarkeit der Stakeholder	+	+	–	+	–	+	++	++	++
Hohe Anzahl der Stakeholder	+	–	+	0	–	++	0	0	0

Auswahlmatrix für Ermittlungstechniken

Legende:

–	nicht empfohlen
0	kein Einfluss=> ist anwendbar
+	empfohlen
++	sehr empfohlen

	Brainstorming	Methode 6–3–5	Wechsel der Perspektive	Feldbeobachtung	Apprenticing	Fragebogen	Interview	Systemarchäologie	Reuse
Fachlich/inhaltliche Einflussfaktoren									
Hohe Kritikalität des Sachverhalts	0	0	+	++	–	+	+	++	+
Großer Systemumfang	0	0	0	+	–	–	+	++	+
Keine Erfahrung im Fachgebiet	0	0	0	–	+	–	+	++	+
Grobe Anforderungen gesucht	++	++	+	+	0	+	++	–	0
Detaillierte Anforderungen gesucht	+	+	+	+	++	–	+	++	+
Nicht–funktionale Anforderungen gesucht	0	0	0	0	+	–	+	+	+
Hohe Komplexität des Sachverhalts	0	0	0	+	–	–	+	+	+

- Anforderungsmanagement

  - Einleitung
  - Anforderungstypen
  - Anforderungserhebung
  - Anforderungsmanagement in der agilen Softwareentwicklung
  - Anforderungsänderungen

Anforderungsmanagement in Scrum

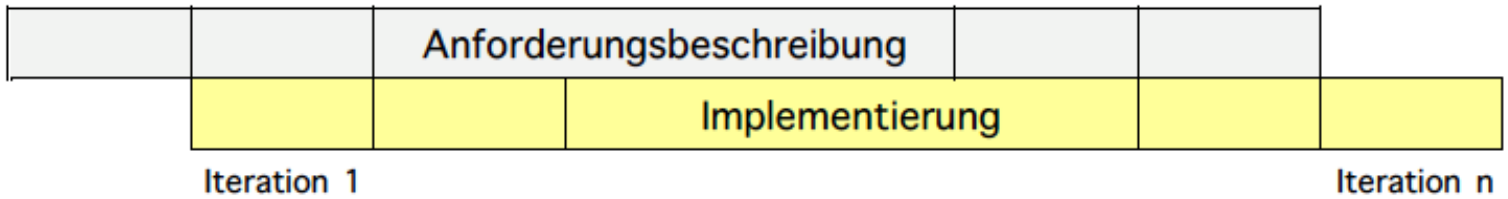
- **Weicht** deutlich von traditionellen Vorgehensweisen zur Beschreibung und zum Management von Anforderungen ab
- Product Owner:
  - entscheidet über die Anforderungen einer Software-Version
  - Beschreibt die Anforderungen in Kooperation mit dem Team, dem Kunden und anderen Stakeholdern
- Zentrales Instrument und Dokument zum Erfassen und Verwalten der Anforderungen ist das **Product Backlog**
- Einträge des Product Backlogs bilden die **Basis** für den Releaseplan und den Input für den jeweiligen Sprint

Anforderungsmanagement in Scrum

- Das **frühe vollständige und genaue** Beschreiben der Anforderungen
  - ist bedingt durch traditionelle Arbeitsorganisations- und Planungsverfahren aus der Fertigungsindustrie
  - Hilft bei einer genauen und umfassenden Schätzung in **Festpreisprojekten**
  - Ist bei komplexer und innovativer Softwareentwicklung **nicht immer effizient und effektiv**
- Probleme:
  - Aufbau eines umfangreichen Anforderungsinventars
  - Informationsverlust durch Übergaben
  - Überproduktion von Funktionalität
  - Unausgeglichener Arbeitsanfall und Überlastungen

Anforderungsmanagement in Scrum

- Daher werden Anforderungen in Scrum nicht nur einmal zu Projektbeginn erhoben und beschrieben
  - In Scrum „verschwimmen“ die Phasen
  - Es gibt **keine separate Definitionsphase** (für die Anforderungen) und Implementierungs- bzw. Umsetzungsphase
  - Anforderungsbeschreibung und Umsetzung erfolgen **zeitnah und überlappend**:



Anforderungsmanagement in Scrum

- Das **Product Backlog** enthält **alle bekannten Anforderungen** und Arbeitsergebnisse, die zur Erreichung des Projektziels umgesetzt werden müssen
- Dazu zählen **funktionale und nichtfunktionale** Anforderungen sowie Anforderungen an die grafische Benutzerschnittstelle
- Product Backlog kann auch Arbeitsergebnisse wie
  - Das Aufsetzen der Test- und Entwicklungsumgebung
  - Das Beseitigen von Bugs
- enthalten
- **Product Owner** ist für das Product Backlog verantwortlich und verfeinert das Product Backlog dabei kontinuierlich

Anforderungsmanagement in Scrum

- Product Backlog
  - Enthält die gesamte Menge der (zu einem bestimmten Zeitpunkt) bekannten Anforderungen an ein Softwaresystem
- Sprint Backlog
  - Enthält den Ausschnitt des Product Backlogs, für den ein Commitment zur Umsetzung mit der nächsten Iteration erfolgt ist
- Product Backlog ist ein „**lebendes Dokument**“ mit **hoher Änderungsdynamik**:
  - Anforderungen verändern sich oder fallen weg
  - Neue Anforderungen kommen hinzu
  - Unvollständige Anforderungen werden verfeinert
  - Priorität von Anforderungen kann sich verändern

Product Backlog

- Erkenntnisse aus der Praxis:
  - Nicht nur die Software sondern auch das Product Backlog wächst iterativ-inkrementell
- Scrum nimmt dediziert auf Anforderungsänderungen Rücksicht
- Daher kennt Scrum auch **kein Change-Request-Verfahren**

Product Backlog

- Das Product Backlog sollte so weit wie **nötig** zu Beginn des Projekts aufgefüllt werden
  - **Alle wichtigen und wesentlichen** Anforderungen (ggf. Alleinstellungs-merkmale) müssen aufgelistet sein
  - Häufig sind **nichtfunktionale** Anforderungen wie Skalierbarkeit, Robustheit, Performanz und GUI-Anforderungen entscheidend
  - Product Owner ist für frühzeitige Identifizierung genau dieser Anforderungen verantwortlich

Anforderungsworkshop

- Teilnehmer: Product Owner, Team, Stakeholder (bspw. potentielle Benutzer, Marketing, Vertrieb, Service)
- Anforderungen werden gemeinsam identifiziert, diskutiert und beschrieben
- Workshops
  - Stellen sicher, dass alle Beteiligten gemeinsames Verständnis der Anforderungen haben und „eine gemeinsame Sprache sprechen“
  - Stellen sicher, dass Anforderungen adäquat beschrieben sind (idealerweise keine zeitaufwändigen Reviews mehr notwendig)
  - Mehrere Anforderungsworkshops vor dem erstem Sprint sind hilfreich
  - Ziel: Auffüllen des Product Backlog
  - Auf Karteikarten werden die Anforderungen in Form von User Stories festgehalten
  - Auch während der Sprints finden Anforderungsworkshops statt, um neue Anforderungen zu identifizieren, alte zu eliminieren und relevante zu verfeinern

Product Backlog

- Anforderungen sollten zunächst schriftlich auf Karteikarten erfasst werden und dann tabellarisch in elektronischer Form zusammen-gefasst werden
  - Karteikarten sind insbesondere für Workshops mit Stakeholdern und Endanwendern deutlich praktischer als eine elektronische Form

Priorität	Thema	Beschreibung	Akzeptanzkriterien	Aufwand
1	Kalender	Als Basisanwender möchte ich eine Besprechung anlegen.	Teste das Eingeben ungültiger Werte, z.B. Endzeit liegt vor Startzeit.	1
2	Kalender	Als Basisanwender möchte ich eine Besprechung stornieren.	Teste, dass dieselbe Besprechung nicht zweimal gelöscht werden kann.	3
3	Kalender	Als Basisanwender möchte ich eine Besprechung ändern.	Teste, ob die Änderungen persistiert wurden.	2
...				

Product Backlog

- Priorisierung
  - **Alle Einträge** im Product Backlog sind **priorisiert**
  - **Product Owner** entscheidet, wonach und wie die Anforderungen priorisiert werden
  - **Das Team** unterstützt den Product Owner insbesondere bei der Identifizierung und Berücksichtigung technischer Risiken
  - Hochpriorisierte Anforderungen werden **präziser und umfassender** beschrieben, als niedrigpriorisierte Anforderungen

Product Backlog

- Gründe für die Priorisierung des Product Backlogs:
  - **Wichtige** Anforderungen werden **als erstes umgesetzt**
  - => **Frühzeitige Rückmeldung** seitens des Kunden/Endanwender
  - Zeit und Aufwand wird für wichtige Anforderungen geblockt
  - Team kennt die für den Projekterfolg **essenziellen** Anforderungen
  - Product Owner kann die Umsetzung der wichtigen Anforderungen besonders gut im Auge behalten

Product Backlog

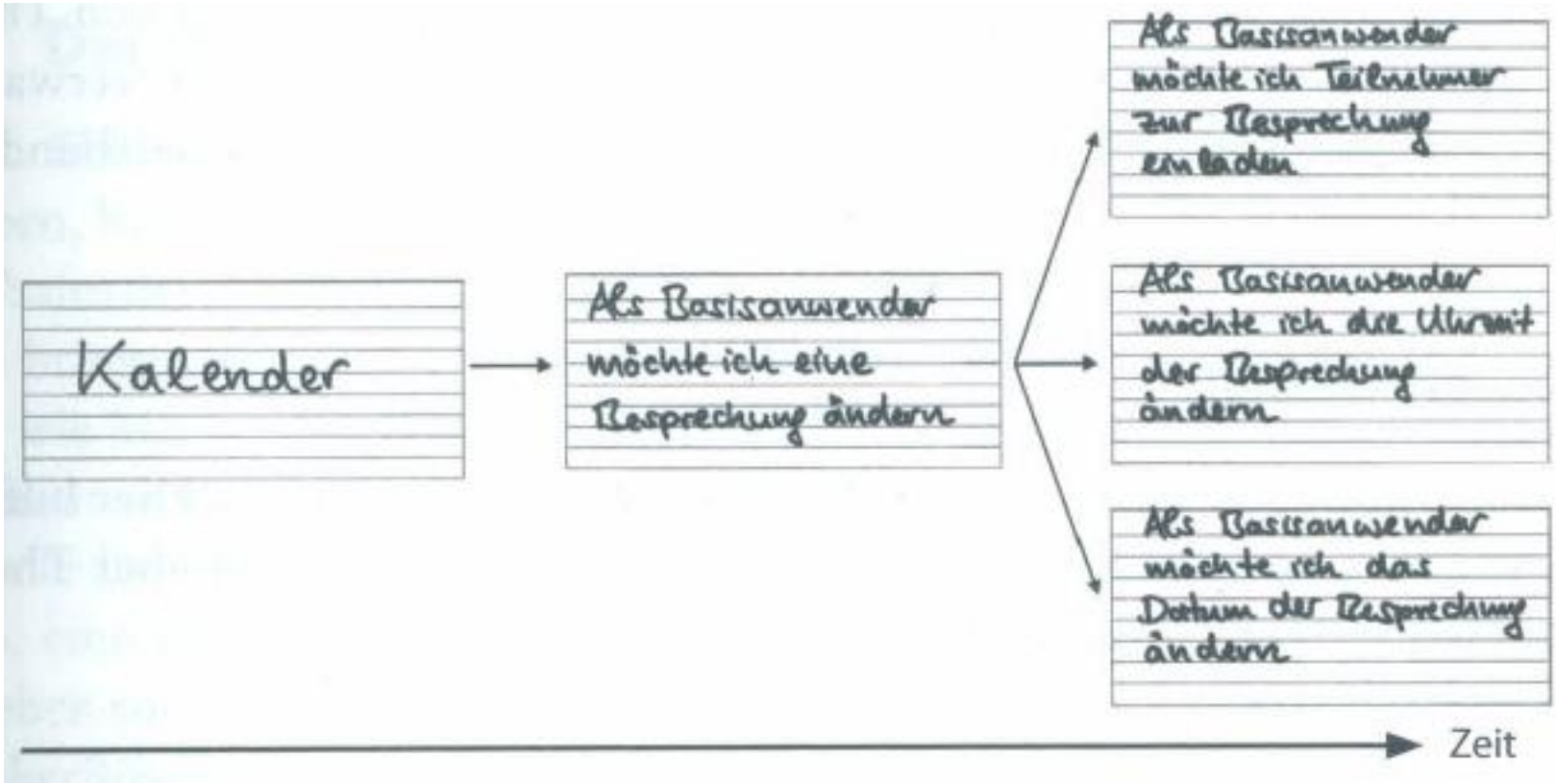
- Kriterien für die Priorisierung des Product Backlogs:
  - Wert/Nutzen:
    - Welchen Mehrwert schafft die Realisierung der Anforderung?
    - Welches relevante Wissen generiert die Implementierung der Anforderung und welche Auswirkung hat dies auf den weiteren Projektverlauf?
  - Risiko:
    - Welches Risiko wird durch die Umsetzung der Anforderung beseitigt?
  - Kosten:
    - Welcher Aufwand bzw. welche Kosten fallen bei der Realisierung der Anforderung an?

Product Backlog

- Alle Einträge im Sprint Backlog sind hinsichtlich ihres Aufwands geschätzt
  - Das ermöglicht es dem Product Owner eine Kosten/Nutzen-Analyse für die Anforderungen umzusetzen
  - Das Team, das für die Umsetzung der Anforderungen verantwortlich ist, schätzt den Aufwand zur Realisierung ab
  - Metriken und Verfahren: siehe LE04

Product Backlog

- Verfeinerung
  - Product Owner ist für die systematische Verfeinerung der im nächsten Sprint umzusetzenden Anforderungen verantwortlich:



Product Backlog

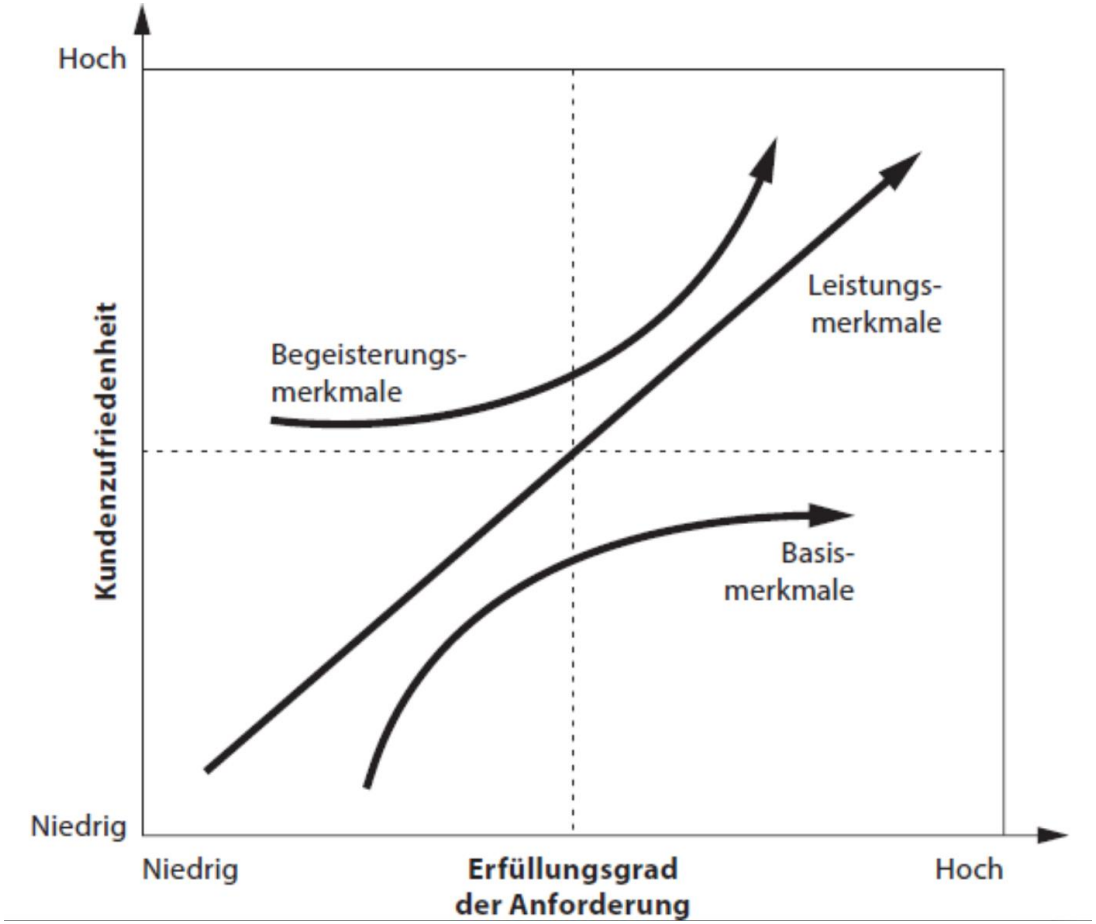
- Vorgehen zur schrittweisen Verfeinerung der Anforderungen:
  - Product Owner füllt das Product Backlog mit dem **Epic** „Kalender“
  - Hinzufügen einer **zielorientierten User Story**
  - **Detaillierung** der User Story in mehrere User Stories
  - Bevor die Anforderung vom Team in einem Sprint Planning eingeplant werden kann, muss sie weiter präzisiert und mit **Akzeptanzkriterien** versehen werden
  - Verfeinerung der Anforderungen spätestens vor dem Sprint Planning
  - Einträge im Product Backlog werden zu Epics gruppiert („Kalender“)
  - Praktische Erfahrung: Meist stellen nicht einzelne Anforderungen, sondern komplett umgesetzte Themen einen Mehrwert für den Endanwender dar
  - Häufig ist es einfacher, ein bestimmtes Thema (Epic) statt einer bestimmten Anforderung bzw. User Story zu priorisieren

Product Backlog

- Tipps und Tricks:
  - Es gibt Anforderungen mit hohem Risikopotenzial
  - Es gibt Anforderungen, über die nur wenig Wissen im Team verfügbar ist
  - Stellen Sie als Product Owner sicher, dass genau diese Anforderungen möglichst detailliert und präzise im Product Backlog beschrieben werden
  - Das Fokussieren der wichtigen Anforderungen und das grobe Skizzieren aller anderen Anforderungen ist zunächst sehr ungewohnt
  - Setzt voraus, dass die Kundenwünsche im Detail verstanden werden
  - Führt dazu, dass Team und Product Owner über das gesamte Projekt eng zusammenarbeiten

Kano-Modell

- Zur Bestimmung des Nutzens/Mehrwerts aus Kundensicht
- Product Owner ist für Bestimmung des Nutzens einer Anforderung verantwortlich



Kano-Modell

- Kano-Modell unterteilt Anforderungen in **Basis-, Leistungs- und Begeisterungsmerkmale**
- Kano-Modell anhand eines Beispiels:
  - Basismerkmale eines Mountain-Bikes sind zum Beispiel:
    - Funktionaler Rahmen
    - Laufräder
    - Bremsen
  - Leistungsmerkmale eines Mountain-Bikes sind zum Beispiel:
    - Federweg (je mehr, je besser)
    - Gewicht (je weniger, je besser)
  - Begeisterungsmerkmale sind zum Beispiel:
    - Kontrolle über die Federgabel während des Fahrens
    - Farbe und Qualität der Lackierung

Kano-Modell

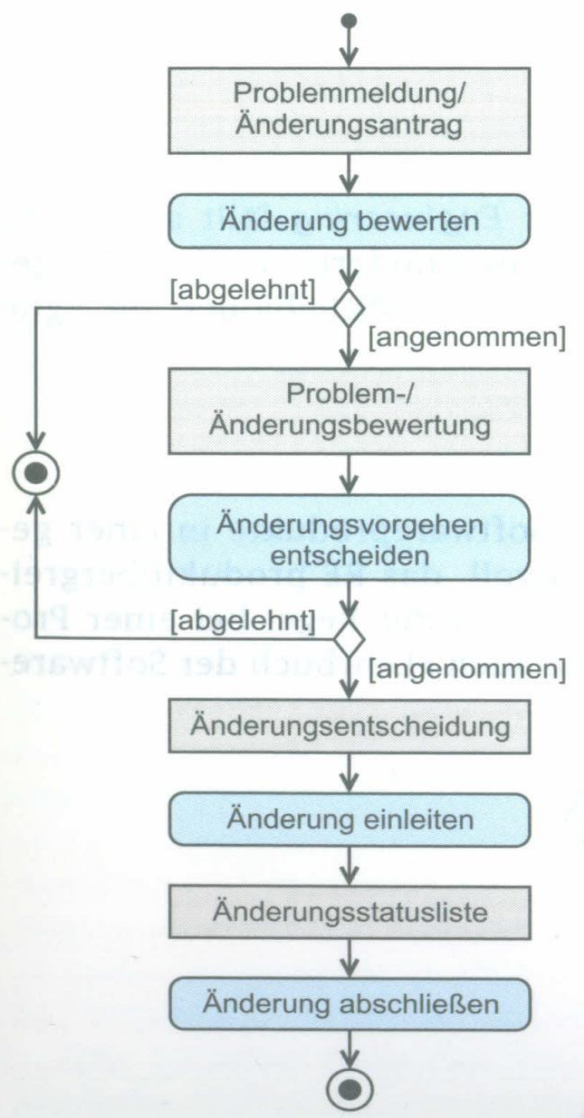
- **Basismerkmale** sind also **essenziell notwendig**, um die Software einsetzen und vertreiben zu können
  - Aber: Kano-Modell sagt voraus, dass die Basisanforderungen rasch zu einer Stagnation der Kundenzufriedenheit führen!
- **Leistungsmerkmale** führen zu einem **linearen Anstieg der Kundenzufriedenheit** nach dem Motto „je mehr, je besser“
- **Begeisterungsmerkmale** bewirken **hohe Kundenzufriedenheit**
- Ziel: für jedes Release eine ideale Kombination von Anforderungen aus den drei Kategorien entwickeln

- Anforderungsmanagement
  - Einleitung
  - Anforderungstypen
  - Anforderungserhebung
  - Anforderungsmanagement in der agilen Softwareentwicklung
  - Anforderungsänderungen

Einleitung

- Erkenntnis: das **Anforderungsdokument** weist immer nur eine **temporäre Aktualität und Gültigkeit** auf
- Umgang mit sich **verändernden** Anforderungen
  - Ist eine der größten **Herausforderungen** in der Softwareentwicklung
  - **Späte Anforderungsänderungen** sind oftmals nur **zeitintensiv** zu bewerkstelligen
  - **Späte Anforderungsänderungen** sind in der Regel **teuer**
  - Ist eine der zentralen Aufgaben der Softwareentwicklung

Change Request Prozess



Change Request Prozess

- Muss in **Festpreisprojekten immer** definiert werden
- Auch in Projekten mit einer Vergütung „Preis nach Aufwand“ (**Time and Material**, T&M) gibt es trotzdem oftmals einen **formal einzuhaltenden CR-Prozess**
- Klassischerweise Teil des Anforderungsmanagements und der dafür zuständigen Rolle des Anforderungsmanagers

Validierung

- **Validierung** soll zeigen, dass die Anforderungen **genau** das Software-System definieren, das der Kunde wünscht, benötigt und erwartet.
- Hierbei sollen
  - **Fehler und Inkonsistenzen** im **Anforderungsdokument** und
  - **zeit- und kostenintensive Nacharbeit** in späteren Phasen der Software-Entwicklung
- **vermieden** werden.
- Es gibt **Prüfungen** und **Methoden** zur Validierung der Anforderungen

Prüfungen zur Validierung

- Gültigkeitsprüfungen
  - Wurden die sinnvollen und notwendigen Funktionen für das Software-System ausgewählt und spezifiziert?
- Konsistenzprüfungen
  - Widersprüchliche Beschreibungen und/oder Beschränkungen für die gleiche Systemfunktionalität müssen identifiziert und korrigiert werden
- Vollständigkeitsprüfungen
  - Wurden auch tatsächlich alle Anforderungen spezifiziert?
- Realisierbarkeitsprüfungen
  - Können die spezifizierten Anforderungen mit den verfügbaren Technologien realisiert und umgesetzt werden?
  - Ist die Umsetzung in der vorgesehenen Zeit und mit dem bereitgestellten Budget möglich?
- Verifizierbarkeitsprüfungen
  - Sind die Anforderungen so formuliert, dass sie objektiv überprüfbar (verifizierbar) sind?

Methoden zur Validierung

1. Anforderungsreviews
   - Spezifizierte Anforderungen werden nach entsprechender Vorbereitung systematisch durch ein Gutachter-Team analysiert

2. Prototyping
   - Endbenutzern und Kunden wird funktionsfähiges Modell des Systems zur Verfügung gestellt
   - Modell kann ausprobiert werden und mit dem Modell experimentieren, um zu sehen, ob es den tatsächlichen Anforderungen entspricht

Methoden zur Validierung

3. Testfallgenerierung
   - Anforderungen müssen im Rahmen des Abnahmetests zwischen Auftraggeber und Auftragnehmer abgenommen werden können
   - Werden die Tests als ein Teil des Validierungsprozesses definiert, offenbart dies oftmals Probleme in den spezifizierten Anforderungen.
   - Können Tests nicht oder nur unter großem Aufwand durchgeführt werden ist eine Überarbeitung der Anforderungen zwingend erforderlich!

Ergänzende Quellen & Literatur

- C. Alves, S. Pereira, J. Castro, „A Study in Market-Driven Requirements Engineering“
- H. Balzert, *Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering*

Herzlichen Dank für Ihre Aufmerksamkeit !