Karlsruher Institut für Technologie

Kapitel 1 – Die Planungsphase

SWT I – Sommersemester 2021

Walter F. Tichy, Christopher Gerking, Tobias Hey

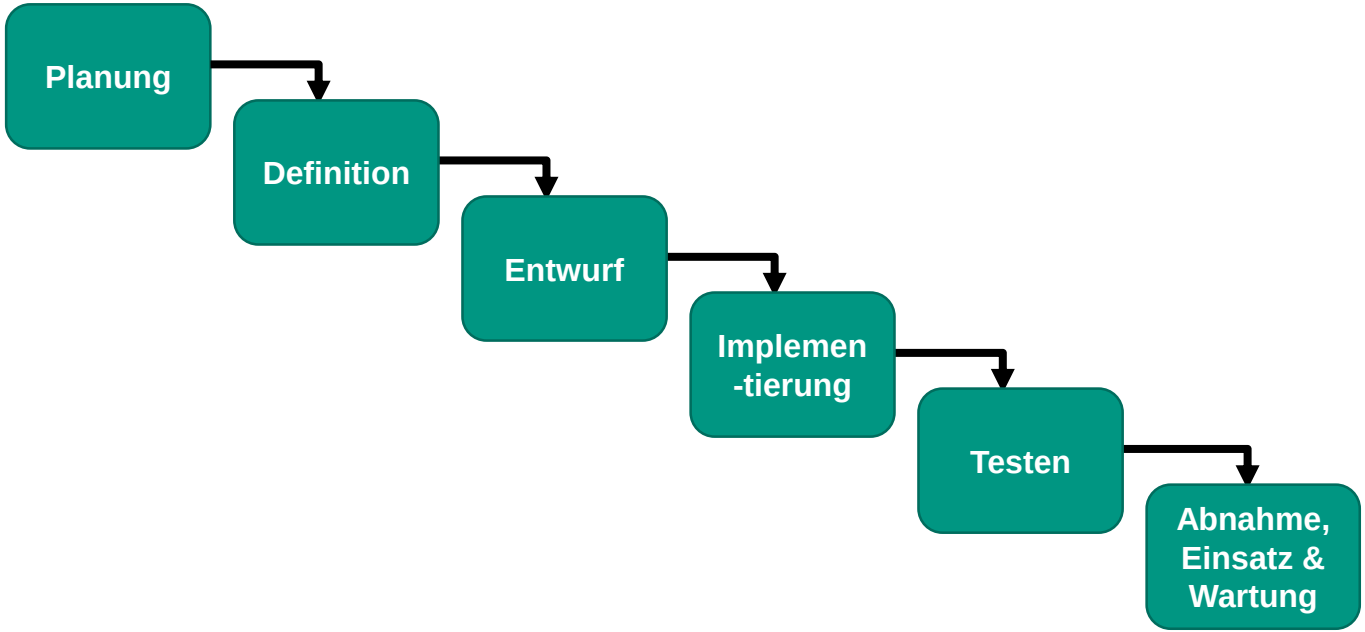
Literatur

- Diese Vorlesung orientiert sich an Abschnitt 2.4.1 **Use Case Diagrams** und Kapitel 4 **Requirements Elicitation** aus

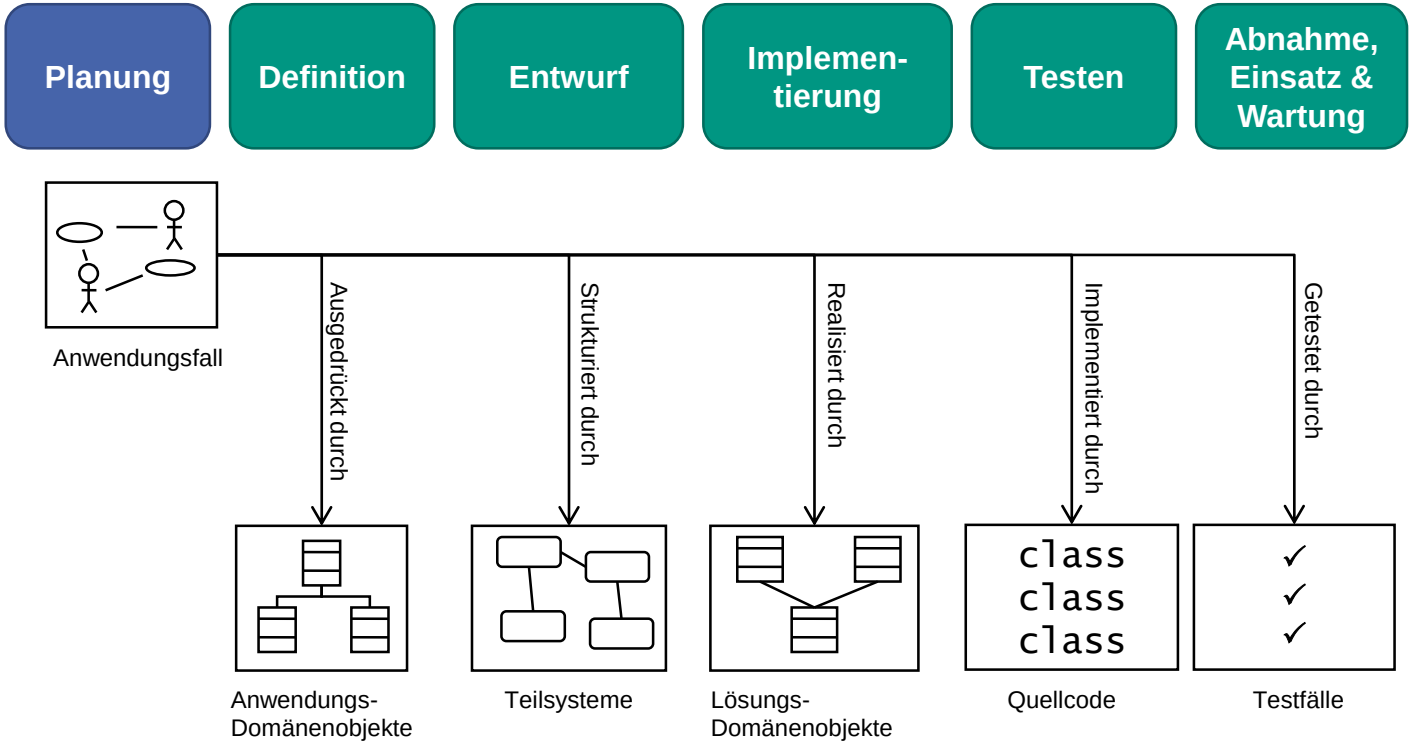
  B. Bruegge, A.H. Dutoit, **Object-Oriented Software Engineering: Using UML, Patterns and Java**, Pearson Prentice Hall.

- **Lesen!**

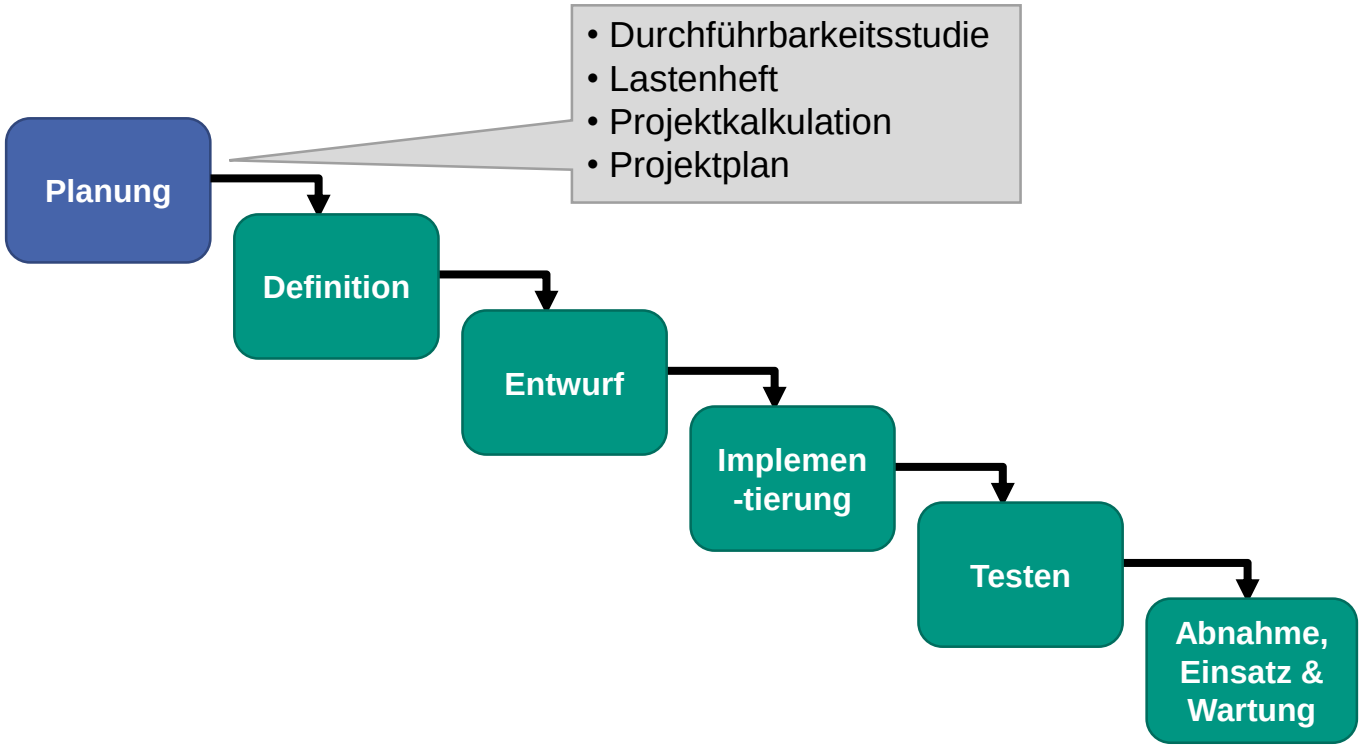
Das Wasserfallmodell – Überblick



Das Wasserfallmodell – Phasen und Modelle



Das Wasserfallmodell – Überblick



Planungsphase

- Die Planungsphase hat das Ziel, in einem Lastenheft das zu entwickelnde System **in Worten des Kunden** zu beschreiben und die Durchführbarkeit des Projektes zu überprüfen.

- In der englischsprachigen Literatur wird diese Phase **Requirements Elicitation** genannt, auf deutsch: Anforderungserhebung oder –ermittlung.

Was ist eine Anforderung?

- Der IEEE Standard 610.12-1990 definiert eine **Anforderung** (engl. *requirement*) wie folgt:

  "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 document.
  The set of all requirements forms the basis for subsequent development of the system or system component".

Die Planungsphase – Erste Schritte

- Die Planungsphase beginnt mit der Auswahl des Produktes:
  - Auftrag eines Kunden
  - Forschungsergebnisse
  - Vorentwicklungen
  - Trendstudien
  - Marktanalysen
  - Eine Idee!

Anforderungen an das Produkt

- Nach der Auswahl des zu entwickelnden Produktes müssen die Hauptanforderungen an das Produkt ermittelt werden.
- Hierfür können verschiedene Techniken eingesetzt werden:
  - Fragebögen (engl. *questionnaires*)
  - Interviews
  - Aufgaben-Analyse, Dokument-Analyse (engl. *task analysis*)
  - Szenarien (engl. *scenarios*)
  - Anwendungsfälle (engl. *use cases*)

Szenarien (1)

- Ein Szenario
  - Ist die Beschreibung eines Ereignisses oder einer Folge von Aktionen und Ereignissen.
  - Ist die Beschreibung der Verwendung eines Systems in Textform **aus Sicht eines Benutzers**.
  - Kann Texte, Bilder, Videos und Ablaufpläne enthalten, sowie Details über den Arbeitsplatz, das soziale Umfeld und Einschränkungen, die die Ressourcen betreffen.

Beispielszenario „Brennende Lagerhalle“

- Während Bob mit seinem Einsatzwagen die Hauptstraße entlangfährt, bemerkt er, dass Rauch aus einem Lagerhaus aufsteigt. Seine Kollegin, Alice, meldet den Notfall vom Fahrzeug aus.
- Alice gibt die Adresse der Lagerhalle in ihren mobilen Rechner ein, eine kurze Beschreibung der Lage (z.B. nord-westliche Ecke) und eine Priorität.
- Sie bestätigt ihre Eingabe und wartet auf eine Bestätigung.
- John, der Disponent auf der Leitstelle, wird durch ein Piepen seines Rechners auf den Notfall aufmerksam gemacht. Er analysiert die Informationen, die Alice ihm geschickt hat, und bestätigt die Meldung. Er alarmiert die Feuerwehr und gibt die erwartete Ankunftszeit an Alice weiter.
- Alice empfängt die Bestätigung und die erwartete Ankunftszeit.

Bemerkungen zu „Brennende Lagerhalle“

- Es ist ein spezielles Szenario
  - Es beschreibt eine einzelne Instanz vom Melden eines Feuers.
  - Es beschreibt nicht alle möglichen Situationen, in welchen ein Feuer gemeldet werden kann.
- Teilnehmende Akteure:
  - Polizisten, Disponent (Bob, Alice, John)

Szenarien (2)

- Szenarien können auch in der Testphase und der Auslieferung eingesetzt werden.
- Werden Szenarien in der Entwurfsphase eines Systems eingesetzt, spricht man auch von „Szenario-basiertem Entwurf“.

Szenarien (3)

- Die Szenario-basierte Anforderungs-Ermittlung erfolgt iterativ, wobei jedes Szenario als Arbeitspaket angesehen wird.
- Jedes dieser Arbeitspakete wird iterativ erweitert und überarbeitet, sobald sich die Anforderungen ändern.
- Szenario-basierter Ermittlung basiert auf konkreten Beschreibungen und nicht auf abstrakten Ideen.
- Szenario-basierter Ermittlung ist informell und ohne festes Ende.

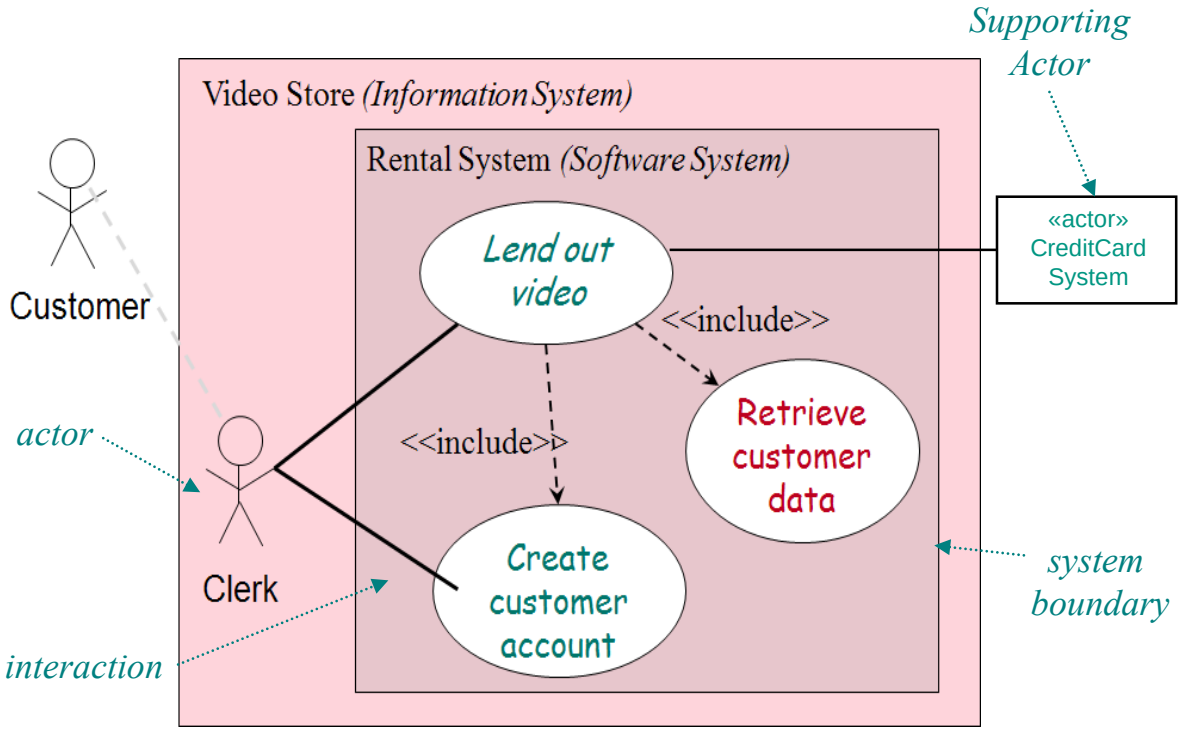
Wie findet man Szenarien?

- Erwarte keine wörtlichen Anweisungen des Kunden, solange das System nicht existiert.
  - Der Kunde versteht zwar die Domäne des Problems, nicht jedoch die Domäne der Lösung.
- Versuche, mit dem Kunden zu kommunizieren:
  - Helfe dem Kunden, Anforderungen zu formulieren.
  - Der Kunde hilft, die Anforderungen zu verstehen.
  - Die Anforderungen werden sich ändern, während das Szenario entwickelt wird.

Tipps zum Finden von Szenarien

- Stellen Sie sich oder dem Kunden folgende Fragen:
  - Was sind die Hauptaufgaben, welche das System erfüllen soll?
  - Welche Daten wird der Benutzer erzeugen, speichern, ändern, löschen oder in das System eingeben?
  - Über welche Änderungen außerhalb des Systems muss das System Bescheid wissen?
  - Über welche Änderungen oder Ereignisse muss der Benutzer des Systems informiert werden?
- Aber: Verlassen Sie sich nie auf Fragen und Befragungen alleine.
- Setzen Sie Beobachtungen ein, sofern das System bereits existiert! (Z.B. als Satz von Formularen, die auszufüllen sind)

Anwendungsfall-Diagramme (Use Case Diagrams)



<<include>> ~ sub use case "call"

UML-Anwendungsfalldiagramm

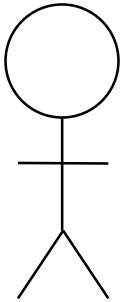
- Anwendungsfalldiagramme werden während der Planungsphase verwendet, um das von außen sichtbare Verhalten des Systems darzustellen.
- Ein **Akteur** (*actor*) spezifiziert eine Rolle eines Benutzer oder eines anderen Systems, welches mit dem von uns geplanten System interagiert.
- Ein **Anwendungsfall** (*use case*) stellt eine Klasse von Funktionen dar, welche das System anbietet.
- Ein **Anwendungsfalldiagramm** (*use case model*) ist die Menge aller Anwendungsfälle, welche die gesamte Funktionalität des Systems beschreiben.

UML-Anwendungsfalldiagramm

- Ein Akteur ist ein Modell für eine externen Einheit, welche mit dem System interagiert:
  - Administrator, Endnutzer, Umwelt, externe Systeme,…
- Ein Akteur besitzt einen eindeutigen Namen sowie optional eine Beschreibung.
- Beispiel:
  - **Student**: Eine studierende Person
  - **Zufallszahlengenerator**

Name

Optionale Beschreibung

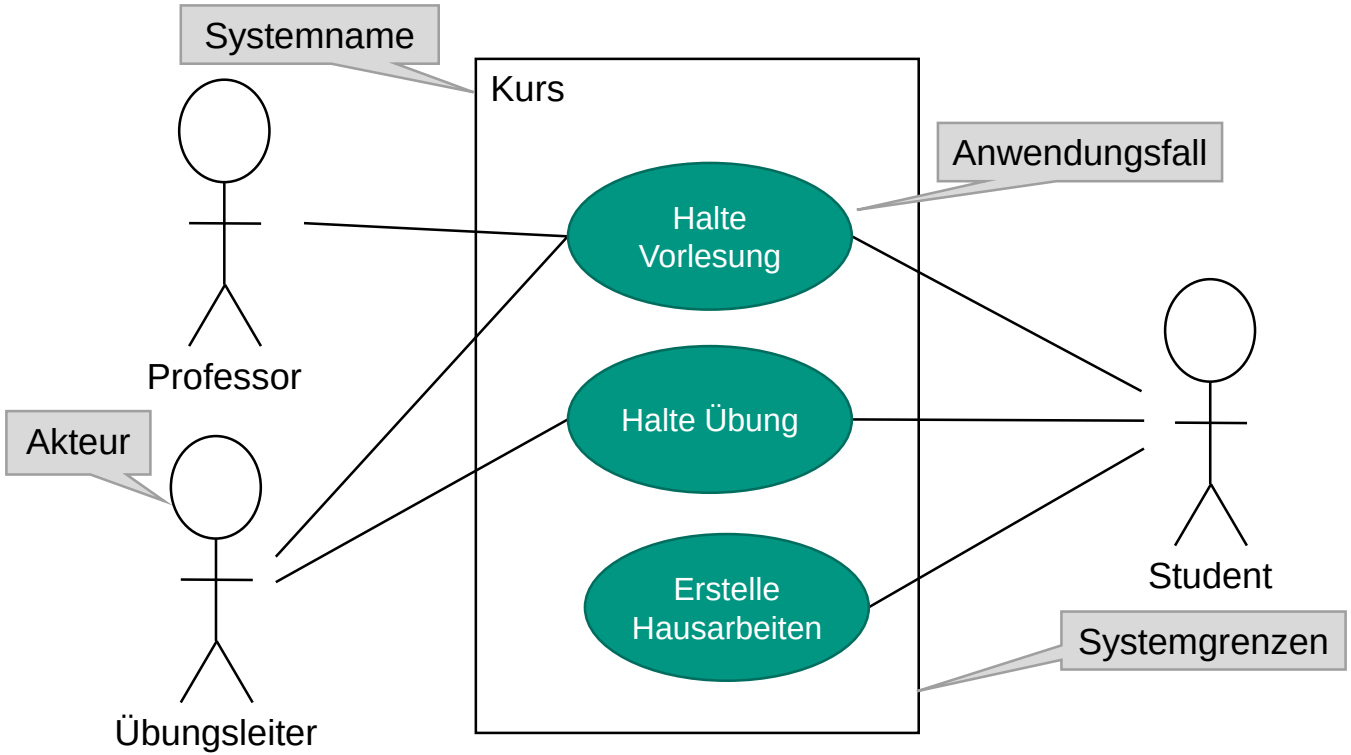


Student

UML-Anwendungsfalldiagramm

- Anwendungsfälle können mit Text beschrieben werden, mit dem Schwerpunkt auf dem Interaktionsfluss zwischen Akteur und System.
- Die Beschreibung eines Anwendungsfalles mit Text besteht aus 6 Teilen:
  - Eindeutiger Name
  - Teilnehmende Akteure
  - Eingangsaktionen
  - Ausgangsaktionen
  - Ereignisfluss
  - Spezielle Anforderungen

UML-Anwendungsfalldiagramm

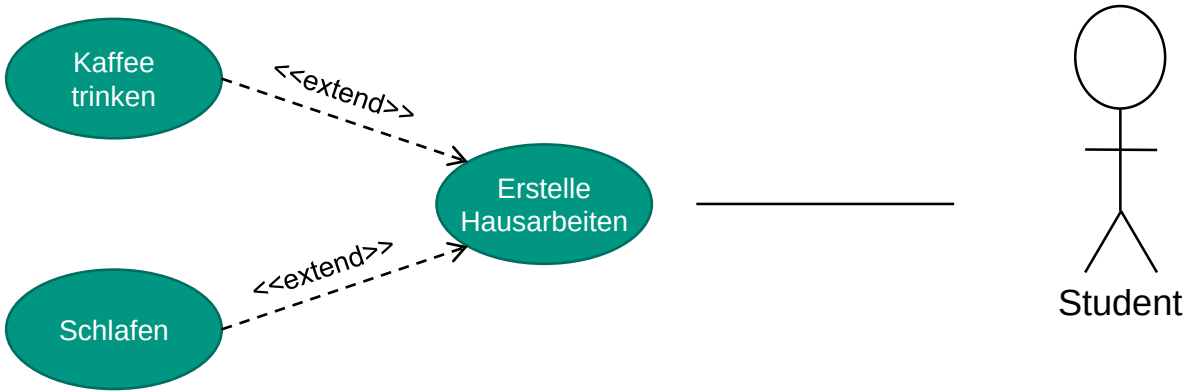


Beziehungen in Anwendungsfällen

- Anwendungsfälle können untereinander in Beziehung stehen.
- **Erweiternde** Beziehung (engl. *extend relationship*):
  - Stellt selten aufgerufene Anwendungsfälle oder außergewöhnliche Funktionalität dar.
- **Einschließende** Beziehung (engl. *include relationship*):
  - Stellt Funktionalität dar, welche von mehr als einem Anwendungsfall verwendet wird.

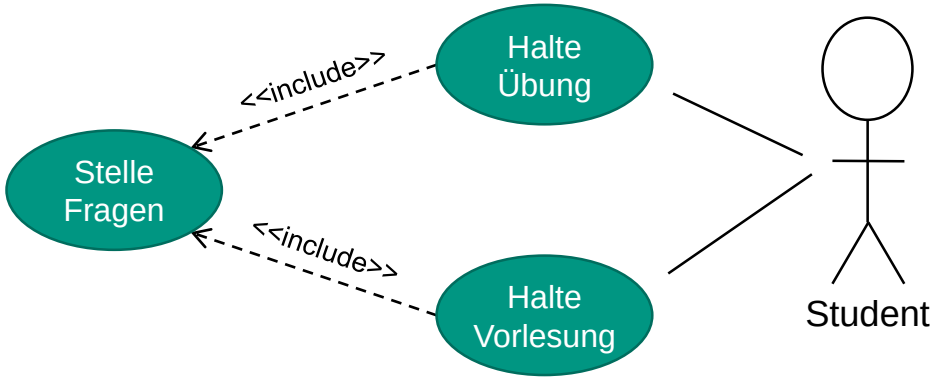
Die <<*extend*>>-Beziehung

- Außergewöhnliche Ereignisflüsse werden aus dem Hauptereignis der Übersichtlichkeit wegen herausgezogen.
- Die Richtung einer <<*extend*>> Beziehung ist zum erweiterten Anwendungsfall (eine Art Unterklassen-Beziehung).
- Anwendungsfälle, die außergewöhnliche Flüsse darstellen, können mehr als einen Anwendungsfall erweitern.



Die <<*include*>>-Beziehung

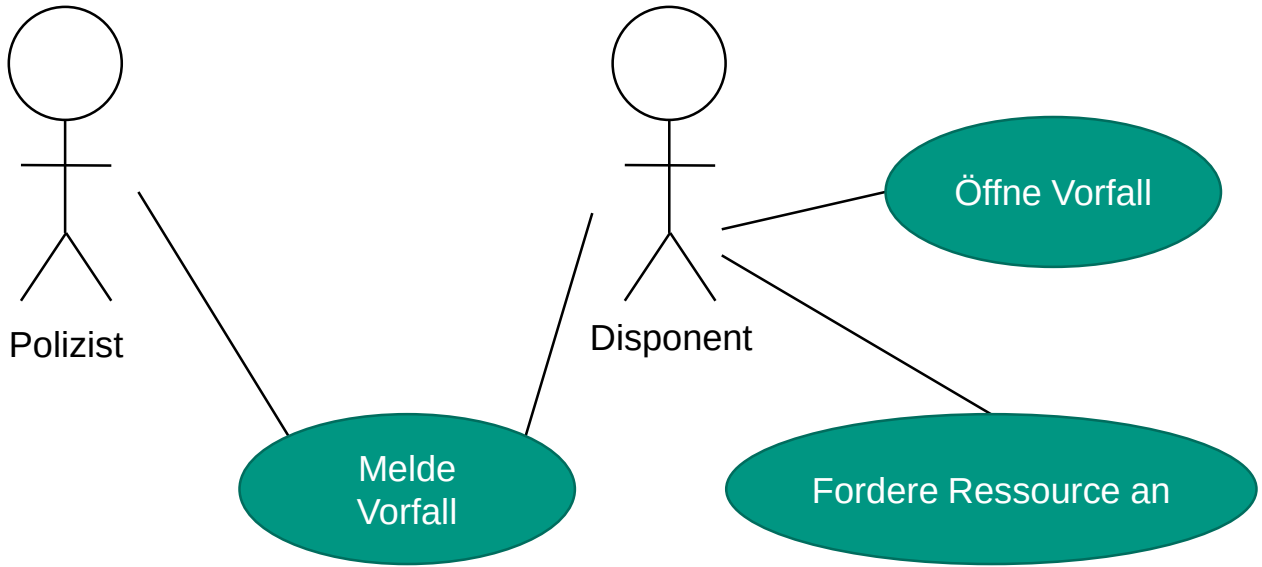
- <<include>>-Beziehungen stellen allgemeine Funktionen dar, welche in mehr als einem Anwendungsfall verwendet werden. (z.B. eine Aufrufsbeziehung)
- <<include>>-Verhalten wird aus Gründen der Wiederverwendbarkeit herausfaktorisiert.
- Die Richtung einer <<include>>-Beziehung geht zum verwendeten Anwendungsfall.



Beschreibung in Textform

- **Name:**
  - Erstelle Hausarbeit
- **Teilnehmender Akteur:**
  - Student
- **Eingangsbedingung:**
  - Student erhält Übungsblatt
  - Student ist gesund
- **Ausgangsbedingung:**
  - Student gibt Lösung ab
- **Ereignisfluss:**
  - Student holt aktuelles Übungsblatt
  - Student liest sich die Aufgaben durch
  - Student löst die Aufgabe und gibt sie in den Rechner ein
  - Student druckt die Lösung aus
  - Student gibt die Lösung ab
- **Spezielle Anforderungen:**
  - Keine

Anwendungsfallmodell für die Vorfall-Verwaltung



Kurze Fragerunde…

- Wahr oder falsch?

- Szenarien können nach Abschluss der Planungsphase „gelöscht“ werden, da sie in Anwendungsfalldiagrammen festgehalten sind.
- Der szenariobasierte Entwurf kann nur eingesetzt werden, wenn man einen Kunden hat (diskutieren, etc. ist sonst unmöglich).

Wie findet man Anwendungsfälle?

- Wähle einen begrenzten, vertikalen Ausschnitt des Systems (z.B. ein Szenario)
  - Diskutiere diesen im Detail mit dem Benutzer, um den bevorzugten Interaktionsstil des Benutzers zu ermitteln
- Wähle einen horizontalen Ausschnitt (z.B. viele Szenarien) um einen größeren Bereich des Systems zu definieren.
  - Diskutieren den Bereich mit dem Benutzer
- Benutze aussagekräftige Prototypen (engl. *prototype*, *mock-ups*) zur visuellen Unterstützung.
- Finde heraus, was der Benutzer tut
  - Beobachtung (Gut)
  - Befragung (oftmals ungenaue Antworten)

Anwendungsfall aus Szenario

- Suche alle Anwendungsfälle im Szenario, die alle Instanzen angeben, wie man ein Feuer melden kann.
  - Beispiel: „Melde Notfall“ im ersten Absatz der Szenarien ist ein Kandidat für einen Anwendungsfall
- Beschreibe jeden dieser Anwendungsfälle so genau wie möglich:
  - Teilnehmende Akteure
  - Beschreibe deren Eingangsaktionen
  - Beschreibe deren Ereignisfluss
  - Beschreibe deren Ausgangsaktionen
  - Beschreibe Ausnahmen
  - Beschreibe nicht-funktionale Anforderungen

Anwendungsfall: „MeldeNotfall“

- Name des Anwendungsfalls: MeldeNotfall
- Teilnehmende Akteure:
  - Polizist (Bob und Alice in diesem Szenario)
  - Disponent (John in diesem Szenario)
- Ausnahmen:
  - Der Polizist wird sofort benachrichtigt, wenn die Verbindung zwischen Terminal und Zentrale abbricht.
  - Der Disponent wird sofort benachrichtigt, wenn die Verbindung zwischen einem Polizisten und der Zentrale abbricht.
- Ereignisfluss: siehe nächste Folie
- Nichtfunktionale Anforderungen:
  - Die Meldung des Polizisten wird innerhalb von 30 Sekunden bestätigt. Die Antwort kommt nicht später als 30 Sekunden nach dem Absenden durch den Disponenten beim Polizisten an.

Anwendungsfall: „MeldeNotfall“

- Der Polizist aktiviert die „Melde Notfall“ Funktion auf seinem Terminal. FRIEND antwortet, indem es ein Formular anzeigt.
- Der Polizist füllt das Formular aus, indem er die Notfall-Stufe, die Einsatzart, die Adresse sowie eine kurze Beschreibung der Situation eingibt. Der Polizist beschreibt ebenfalls eine Reaktion auf die Notfallsituation.
- Der Disponent erstellt einen Vorfall in der Datenbank durch das Aufrufen des Anwendungsfalls „Öffne Vorfall“. Er wählt eine Reaktion und bestätigt die Meldung.
- Der Polizist empfängt die Bestätigung und wählt die Reaktion.

Weiteres Beispiel: „Fordere Ressource an“

- Akteure:
  - Einsatzleiter: Der Verantwortliche beim Einsatz
  - Ressourcen-Anforderer: Ist verantwortlich für die Anforderung und Freigabe von Ressourcen, welche vom FRIEND-System verwaltet werden.
  - Disponent: Gibt Vorfälle ein, aktualisiert und löscht Vorfälle im System. Er ist auch für das Schließen von Vorfälle verantwortlich.
  - Polizist: Meldet Vorfälle.

Weiteres Beispiel: „Fordere Ressource an“

- Name des Anwendungsfalls: Ressourcenanforderung
- Teilnehmende Akteure:
  - Polizist (Bob und Alice in diesem Szenario)
  - Disponent (John in diesem Szenario)
  - Ressourcen-Anforderer und Einsatzleiter
- Eingangsaktionen:
  - Der Ressourcen-Anforderer hat eine verfügbare Ressource ausgewählt
- Ereignisfluss:
  - Der Ressourcen-Anforderer wählt einen Vorfall
  - Die Ressource wird dem Vorfall zugewiesen
- Ausgangsaktionen:
  - Der Anwendungsfall ist fertig, wenn die Ressource zugewiesen wurde.
  - Die ausgewählte Ressource ist nicht verfügbar für andere Anfragen.
- Besondere Anforderungen:
  - Der Einsatzleiter ist verantwortlich für die Verwendung der Ressourcen

Formulieren von Anwendungsfällen

1. Name des Anwendungsfalls:
   - Z.B. Melde Notfall
2. Finde die Akteure:
   - Verallgemeinere die konkreten Namen („Bob“) zu teilnehmenden Akteuren („Polizist“)
   - Teilnehmende Akteure:
     - Polizist (Bob und Alice in diesem Szenario)
     - Disponent (John in diesem Szenario)
3. Finde den Ereignisfluss:
   - In natürlicher Sprache beschrieben

Probleme bei der Anforderungsermittlung

- Seichtes Bereichswissen
  - Verteilt über viele Quellen
    - selten explizit festgehalten
  - Die verschiedenen Quellen werden sich widersprechen
    - Betroffene haben unterschiedliche Ziele
    - Betroffene haben unterschiedliches Problemverständnis
- Stillschweigendes Wissen
  - Es ist schwierig, Wissen korrekt zu beschreiben, das man regelmäßig nutzt.
- Beschränkte Beobachtbarkeit
  - Betriebsblindheit
  - Gegenwart des Beobachters verändert Verhalten
- Verzerrung
  - Betroffene dürfen evtl. nicht sagen, was benötigt ist
    - Politisches Klima, organisatorische Faktoren
  - Betroffene wollen evtl. nicht sagen, was benötigt ist
    - Angst vor Wegrationalisierung
    - Betroffene versuchen, den Anforderungsermittler für ihre Zwecke zu beeinflussen (verdeckte Ziele)

Beispiel für eine ungenaue Spezifikation

- Während eines Laser-Experiments wurde ein Laserstrahl von der Erde zu einem Spiegel auf der Raumfähre Discovery geschickt.
- Es wurde erwartet, dass der Strahl zu der Spitze eines Berges in der Höhe von 10.023 Fuß reflektiert wird.
- Der Verantwortliche gab als Höhe „10023“ ein.
- Der Laserstrahl traf nie die Spitze des Berges.
  Was war das Problem?
- Der Rechner interpretierte die Daten als Meilen.

Beispiel für ein unbeabsichtigte Funktion

- Aus den Nachrichten: In London verlässt eine U-Bahn die Station ohne Zugführer.
- Was ist passiert?
  - Eine Fahrgasttür hing fest und wollte sich nicht mehr schließen
  - Der Zugführer verließ den Zug, um die Tür zu schließen:
    - Er ließ die Fahrertür offen.
    - Er verließ sich auf die Spezifikation, dass der Zug mit geöffneter Tür nicht losfährt.
  - Als er die Fahrgasttür geschlossen hatte, fuhr der Zug ohne den Zugführer los
    - Die Fahrertür wurde im Steuerprogramm nicht als Tür berücksichtigt

Anforderungsspezifikation vs. Analysemodelle

- Beide haben die Anforderungen an das Systems aus dem Blickwinkel des Benutzers als Ziel
  - Die Anforderungsspezifikation verwendet natürliche Sprache (abgeleitet von der Art des Problems)
  - Das Analysemodell verwendet formale oder semi-formale Notationen.
  - In dieser Vorlesung wird UML verwendet

Arten von Anforderungen

- Funktionale Anforderungen (*functional requirements*)
  - Beschreiben das Verhalten, die Reaktionen des Systems auf Eingaben und Daten, oder die Interaktionen zwischen dem System und der Systemumgebung, unabhängig von der Implementierung.
    „Ein Polizist muss in der Lage sein, Ressourcen anzufordern.“
- Qualitäts- Anforderungen (*quality requirements*)
  - Qualitätseigenschaften der Funktionen
    „Die Antwortzeit muss weniger als eine Sekunde betragen.“
- Einschränkungen (*constraints*)
  - Sind durch den Kunden oder die Umgebung vorgegeben
    „Die Implementierung muss in Java erfolgen.“
- (Qualitäts-Anforderungen und Einschränkungen heißen auch nicht-funktionale Anforderungen)

Funktionale vs. Nichtfunktionale Anforderungen

Funktional

- Beschreiben Benutzer-Aufgaben, die das System unterstützen muss
- Werden als Aktionen formuliert
  - „Benachrichtige Interessenten“
  - „Erstelle eine neue Tabelle“

Nichtfunktional

- Beschreiben Eigenschaften des Systems oder der Domäne
- Werden als Einschränkungen oder Zusicherung (*assertion*) formuliert
  - „Jede Benutzereingabe muss in weniger als einer Sekunde erkannt werden“
  - „Ein Systemabsturz darf nicht zu Datenverlust führen“

Arten nichtfunktionaler Anforderungen

Qualitäts- Anforderungen

- Benutzbarkeit (*usability*)
- Zuverlässigkeit (*reliability*)
- Robustheit (*fault tolerance*)
- Sicherheit (*security*)
- Leistungsfähigkeit (*performance*)
  - Antwortzeit
  - Durchsatz
- Skalierbarkeit (*scalability*)
- Verfügbarkeit (*availability*)
- Wartbarkeit (*maintainability*)
  - Anpassbarkeit
  - Erweiterbarkeit
- Portierbarkeit (*portability*)
- …

Einschränkungen

- Implementierung
- Schnittstellen
- Einsatzumgebung
- Lieferumfang
- Rechtliches
  - Lizenzen
  - Zertifikate
  - Datenschutz
- …

Definitionen einiger qualitativer Anforderungen

- Benutzbarkeit (auch Gebrauchstauglichkeit, engl. *usability)*
  - Die Leichtigkeit, mit welcher die Akteure mit einem System eine Funktion ausführen können
  - Benutzbarkeit ist einer der am häufigsten falsche verwendeten Begriffe
  - Benutzbarkeit muss messbar sein. Sonst ist es Marketing.
    - Beispiel: „Anzahl der Schritte um eine Internet-Bestellung im Browser aufzugeben.“
- Robustheit (engl. *fault tolerance* oder *robustness*)
  - Die Fähigkeit eines Systems, die Funktion fortzusetzen, wenn
    - Der Benutzer eine falsche Eingabe macht (Fehlbedienung)
    - Fehler auftreten
    - Betriebsbedingungen nicht eingehalten werden
      - „Betriebstemperatur -10°C bis +50°C.“
      - „maximale Anzahl an Anfragen ist 2000/s.“
- Verfügbarkeit (engl. *availability)*
  - Das Verhältnis von störungsfreier Betriebszeit zu Gesamtzeit
  - Beispiel: „Das System ist pro Woche weniger als 5 Minuten nicht verfügbar“

Beispiele für nichtfunktionale Anforderungen

- Zuschauer müssen in der Lage sein, Wettkämpfe ohne vorherige Registrierung und ohne Vorwissen zu sehen.
  → Benutzbarkeit
- Das System muss 10 gleichzeitige Wettkämpfe unterstützen.
  → Leistungsfähigkeit
- Der Verwalter muss in der Lage sein, einen Wettkampf hinzuzufügen, ohne bereits existierende ändern zu müssen
  → Benutzbarkeit

Kurze Fragerunde…

- Wahr oder falsch?
  - Funktionale Anforderungen kommen immer vom Kunden.
  - Der Kunde darf keine nichtfunktionalen Anforderungen stellen.
  - Der Kunde darf die Entwicklungsumgebung und –methoden vorschreiben
- Wie misst man
  - Antwortzeit,
  - Durchsatz,
  - Benutzbarkeit?

Was gehört nicht zu den Anforderungen?

- Systemstruktur, Implementierungsdetails
- Entwicklungsmethoden
- Entwicklungsumgebung
- Meistens auch nicht vorgeschrieben: Programmiersprache, Wiederverwendbarkeit

- Es ist klar, dass keiner der obigen Punkte Einschränkungen vom Kunden sind

Validierung von Anforderungen

- Die Validierung von Anforderungen ist ein Schritt zur Qualitätssicherung, welcher normalerweise nach der Planungsphase oder Definitionsphase durchgeführt wird
- Korrektheit
  - Die Anforderungen stellen die Sicht des Kunden korrekt dar.
- Vollständigkeit
  - Alle Situationen, in denen das System benutzt werden kann, sind beschrieben, einschließlich Fehler und Fehlbedienung.
- Konsistenz
  - Keine funktionalen oder nichtfunktionalen Anforderungen widersprechen sich.

Validierung von Anforderungen

- Eindeutigkeit
  - Anforderungen können nur auf eine Art interpretiert werden
- Realisierbarkeit
  - Anforderungen können erfüllt und geliefert werden.
- Verfolgbarkeit
  - Es wird möglich sein, jede Systemfunktion einer oder einer Menge von Anforderungen zuzuordnen, die die Funktion benötigen.
- Probleme mit der Validierung
  - Anforderungen ändern sich während der Planungsphase
  - Inkonsistenzen können bei jeder Änderung auftreten
  - Werkzeugunterstützung ist erforderlich!

Anforderungen an die Anforderungsverwaltung

- Funktionale Anforderungen
  - Speichere die Anforderungen in einem gemeinsamen Verzeichnis, mit Konfigurationsmanagement (GIT, etc.)
  - Biete Mehrbenutzer-Unterstützung für den Zugriff auf die Anforderungen
  - Erstelle das Gesamt-Spezifikationsdokument automatisch aus dem gemeinsamen Verzeichnis
  - Biete Verfolgbarkeit der Anforderungen

Werkzeuge für die Anforderungsermittlung

- DOORS (Telelogic)
  - Werkzeug zur Anforderungsverwaltung für mehrere Plattformen, wobei die Entwickler-Teams am selben Ort arbeiten. DOORS XT bietet die gleichen Funktionen wie DOORS, ist jedoch für verteilte Teams gedacht.
- RequisitePro (IBM/Rational)
  - Integriert sich in Microsoft Word

- Aktuelle Forschung: Texte inhaltlich nach Schwächen analysieren, z.B. Nominalisierungen, Passiv, u.a.
  - „Nach der Anmeldung wird der Kunde auf die Startseite weitergeleitet...“ Wer meldet an? Wie?
  - „Transport der Palletten...“ Wer transportiert? Wie? Von wo nach wo?

Arten der Anforderungsermittlung

- Entwicklung auf der grünen Wiese
  - Entwicklung beginnt von null. Es existiert kein System, auf welchem man aufbauen kann. Anforderungen kommen von den Benutzern und Kunden.
  - Wird durch die Wünsche des Kunden ausgelöst
- Re-Engineering
  - Neuentwurf und/oder Neuimplementierung eines existierenden Systems mit Verwendung neuerer Technologien.
  - Ausgelöst durch neue Technologien oder neue Anforderungen
- Schnittstellen-Entwicklung
  - Bereitstellung von existierenden Diensten in einer neuen Umgebung
  - Ausgelöst durch neue Technologien oder den Marktbedarf

Gliederung eines Lastenheftes

1. Zielbestimmung
2. Produkteinsatz
3. Funktionale Anforderungen
4. Produktdaten
5. Nichtfunktionale Anforderungen
6. Systemmodelle
   a) Szenarien
   b) Anwendungsfälle
7. Glossar (Begriffslexikon zur Beschreibung des Produktes)

Beispiel: Seminarorganisation

1. Zielbestimmung
   - Die Firma Teachware soll durch das Produkt in die Lage versetzt werden, die von ihr veranstalteten Seminare rechnerunterstützt zu verwalten

Version	Autor	QS	Datum	Status	Kommentar
2.1	**Balzert**		**3/05**	**akzeptiert**	
2.2	**Balzert**		**10/05**	**akzeptiert**	**/F115/**

Vgl. „Lehrbuch der Softwaretechnik“, Balzert, 2000

Beispiel: Seminarorganisation

2. Produkteinsatz
   - Das Produkt dient zur Kunden- und Seminarverwaltung der Firma Teachware.
   - Zielgruppe: die Mitarbeiter der Firma Teachware.
   - Plattform: PC mit aktueller Version von MS Windows

Beispiel: Seminarorganisation

Anforderungen wurden aus Platzgründen stark vereinfacht

3. Funktionale Anforderungen

/FA10/
Ersterfassung, Änderung und Löschung von Kunden (Teilnehmer, Interessenten)

/FA20/
Benachrichtigung der Kunden (Anmeldebestätigung, Abmeldebestätigung, Änderungsmitteilungen, Rechnung, Werbung)

/FA30/
Ersterfassung, Änderung und Löschung von Seminarveranstaltungen und Seminartypen

/FA40/
Ersterfassung, Änderung und Löschung von Dozenten sowie Zuordnung zu Seminarveranstaltungen und Seminartypen.

Beispiel: Seminarorganisation

/FA50/
Ersterfassung, Änderung und Löschung von Seminarbuchungen

/FA60/
Erstellung von Rechnungen

/FA70/
Erstellung verschiedener Listen (Teilnehmerliste, Umsatzliste, Teilnehmerbescheinigungen)

/FA80/
Anfragen der folgenden Art sollen möglich sein:
Wann findet das nächste Seminar X statt? Welche Mitarbeiter der Firma Y haben das Seminar X besucht?

Beispiel: Seminarorganisation

4. Produktdaten

/PD10/
Es sind relevante Daten über die Kunden zu speichern

/PD20/
Falls ein Kunde zu einer Firma gehört, dann sind relevante Daten über die Firma zu speichern

/PD30/
Es sind relevante Daten über Seminarveranstaltungen, Seminartypen und Dozenten zu speichern

/PD40/
Bucht ein Kunde eine Seminarveranstaltung, dann sind entsprechende Buchungsdaten zu speichern.

Beispiel: Seminarorganisation

5. Nichtfunktionale Anforderungen

/NF10/
Die Funktion /FA80/ darf nicht länger als 15 Sekunden Interaktionszeit benötigen, alle anderen Reaktionszeiten müssen unter 2 Sekunden liegen

/NF20/
Es müssen maximal 50.000 Teilnehmer und maximal 10.000 Seminare verwaltet werden können.

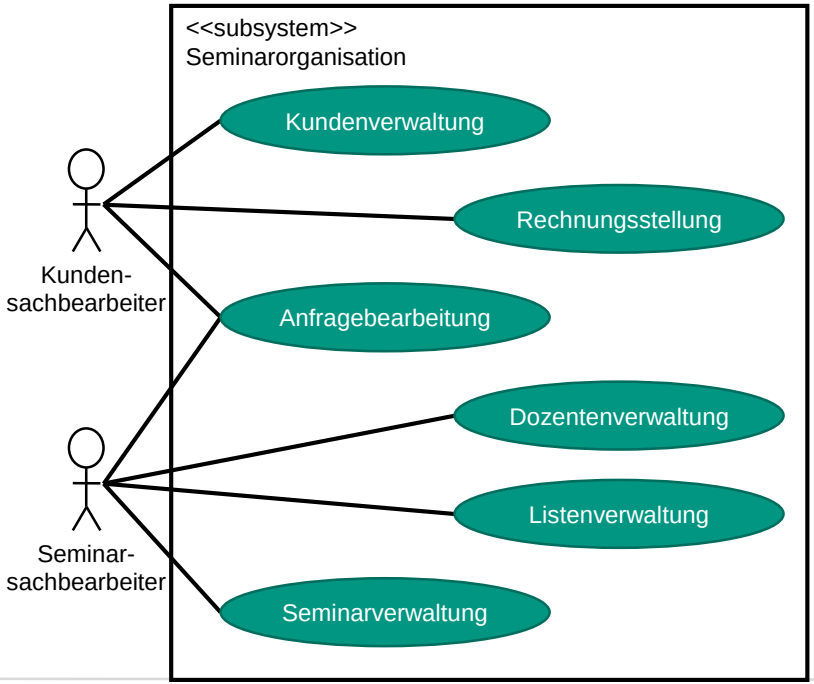
Beispiel: Seminarorganisation

- Zuverlässigkeit: Nicht mehr als ein Ausfall pro Woche. Zu normalen Geschäftszeiten ist das System nicht mehr als 1 h pro Monat nicht verfügbar.
- Benutzbarkeit: Nach Einführung von 4h begehen Nutzer nicht mehr als 2 Fehler pro Tag.
- Übertragbarkeit: Produkt muss den Datenbankstandard ABC benutzen und auf dem Nachfolger des Betriebssystems XYZ lauffähig sein.
- Änderbarkeit: Benutzerschnittstellen, Datenbank sind austauschbar. Das Produkt enthält nicht mehr als 0,1% plattformspezifischer Anweisungen.

Beispiel: Seminarorganisation

6. Systemmodelle – Anwendungsfall: „Seminarorganisation“

- Akteure:
  - Kundensachbearbeiter
  - Seminarsachbearbeiter
- Anwendungsfälle:
  - Kundenverwaltung
  - Rechnungsstellung
  - ...
- Textuelle Beschreibung des Anwendungsfalles fehlt in diesem Beispiel



Beispiel: Seminarorganisation

7. Glossar
   - Dozent

     Leiter eines oder mehrerer Seminartypen
   - Kunde

     (Zahlende) Teilnehmer einer oder mehrerer Seminarveranstaltung/en
   - Seminartyp

     Typ einer Lehrveranstaltung (z.B. „Schöner Malen – Anfängerkurs“)
   - Seminarveranstaltung

     Tatsächlich stattfindende Lehrveranstaltung (z.B. „Schöner Malen – Anfängerkurs, Sommer 2014“)
   - usw.

Durchführbarkeitsuntersuchung

- Anschließend gilt es, die Durchführbarkeit des Projektes zu überprüfen
  - Prüfen der fachlichen Durchführbarkeit
    - softwaretechnische Realisierbarkeit
    - Verfügbarkeit Entwicklungs- und Zielmaschinen
  - Prüfen alternativer Lösungsvorschläge
    - Beispiel: Kauf und Anpassung von Standardsoftware vs. Individualentwicklung
  - Prüfen der personellen Durchführbarkeit
    - Verfügbarkeit qualifizierter Fachkräfte für die Entwicklung
  - Prüfen der Risiken

Durchführbarkeitsuntersuchung

- Durchführbarkeit prüfen (Fortsetzung):
  - Prüfen der ökonomischen Durchführbarkeit
    - Aufwands- und Terminschätzung
    - Wirtschaftlichkeitsrechnung.
  - Rechtliche Gesichtspunkte
    - Datenschutz
    - Zertifizierung
    - Relevante Standards