Vorlesung "Softwaretechnik"
Buchkapitel 4
Anwendungsfälle (Use Cases)

Lutz Prechelt
Freie Universität Berlin, Institut für Informatik
http://www.inf.fu-berlin.de/inst/ag-se/

- Was ist ein Use Case?
  - Beispiele
- Wichtige Parameter
  - Bereich ("system under discussion")
  - Detailgrad/Zielniveau
- Schrittweise Präzisierung
  - Gefahren von Präzision
- Use-Case-Hierarchien
  - Überblick, Benutzerziele, Details
  - Warum-Wie-Verbindung
- Checkliste für Use Cases

Wo sind wir?: Taxonomie "Die Welt der Softwaretechnik"

Freie Universität Berlin

Welt der Problemstellungen:

- Produkt (Komplexitätsprob.)
  - Anforderungen (Problemraum)
  - Entwurf (Lösungsraum)
- Prozess (psycho-soziale P.)
  - Kognitive Beschränkungen
  - Mängel der Urteilskraft
  - Kommunikation, Koordination
  - Gruppendynamik
  - Verborgene Ziele
  - Fehler

Welt der Lösungsansätze:

- Technische Ansätze ("hart")
  - Abstraktion
  - Wiederverwendung
  - Automatisierung
- **Methodische Ansätze ("weich")**
  - **Anforderungsermittlung**
  - **Entwurf**
  - **Qualitätssicherung**
  - **Projektmanagement**

Wo sind wir?: Anforderungsermittlung

- Einsicht: Man darf sich nicht auf intuitiven Eindruck darüber verlassen, was gebaut werden sollte
  - sondern sollte die Anforderungen systematisch ermitteln
- Prinzipien:
  - **Erhebung** der Anforderungen bei allen Gruppen von Beteiligten
  - **Beschreibung** in einer Form, die die Beteiligten verstehen
  - **Validierung** anhand der verschriftlichten Form
  - **Spezifikation**: Übertragung in zur Weiterverarbeitung günstige Form
  - **Trennung von Belangen**: Anford. möglichst wenig koppeln
  - **Analyse auf Vollständigkeit**: Lücken aufdecken und schließen
  - **Analyse auf Konsistenz**: Widersprüche aufdecken und lösen
  - **Mediation**: Widersprüche, die auf Interessengegensätzen beruhen, einer Lösung zuführen (Kompromiss oder Win-Win)
  - **Verwaltung**: Übermäßige Anforderungsänderungen eindämmen, Anforderungsdokument immer aktuell halten

Was ist ein Use Case (Anwendungsfall)?

- Rolle:
  - Vereinbarung zwischen den Beteiligten eines Projekts über das Verhalten des Systems
- Ansatz:
  - **Beschreibung** des **Verhaltens** des **Systems**
  - als Antwort auf einen Auftrag eines **Hauptakteurs**,
  - der mit seinem Auftrag ein **Ziel** verfolgt und erreicht
- Ein Use Case bündelt die Beschreibungen für viele einzelne Fälle (Szenarios), die den Zielen nach eng zusammen gehören
  - ist also abstrakt
  - aber nicht sehr

Welche Form hat ein Use Case?

- Normalerweise Text
  - meist strukturierter Text
- Das gleiche Prinzip (siehe später) ist aber auch mit formalen, insbesondere grafischen Notationen anwendbar:
  - Flussdiagramme
  - Zustandsdiagramme
  - Sequenzdiagramme
  - Petrinetze u.a.
- Text ist aber meist die beste Wahl

- Anmerkung: Wer nicht gut Text schreiben kann, kann auch nicht gut Use Cases schreiben

Beispiel: Use Case "Über WWW Aktien kaufen"

Freie Universität Berlin

- **Hauptakteur**: Käufer
- **Anwendungsbereich**:
  - Personal Advisors/Finance package (**PAF**)
- **Niveau**: Benutzerziel
- **Beteiligte und Interessen**:
  - Käufer: Will Aktien kaufen und in sein PAF Portfolio ergänzen
  - Broker: Benötigt vollständige Kaufinformation
- **Voraussetzung**: Benutzer hat PAF gestartet
- **Mindestzusicherung**:
  - Es wird genügend Protokollinformation gesammelt, damit PAF ggf. entdecken kann, wenn etwas schiefgeht um dann beim Benutzer nachzufragen
- **Zusicherung im Erfolgsfall**:
  - Broker-Website hat den Kauf bestätigt; Protokolle u. Portfolio sind aktualisiert

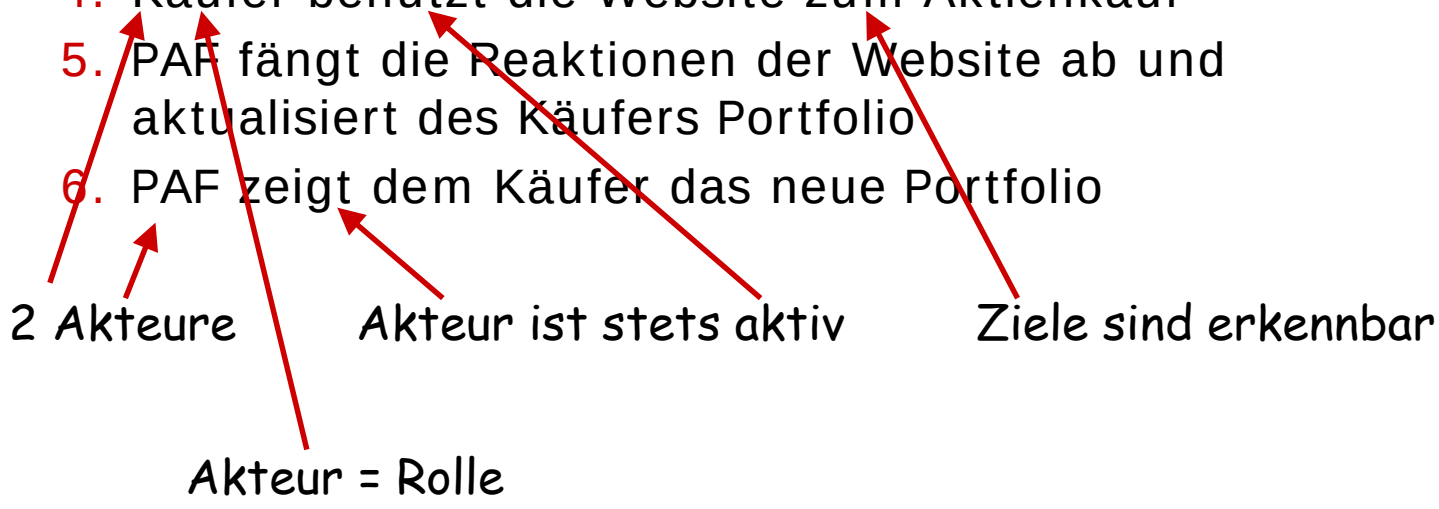
Lutz Prechelt, prechelt@inf.fu-berlin.de

(wird fortgesetzt)

Bsp: Use Case "Über WWW Aktien kaufen" (2)

Freie Universität Berlin

- **Haupt-Erfolgsszenario**:
  1. Käufer wählt "Aktien über das WWW kaufen" aus
  2. PAF fordert Name der Website vom Käufer an
  3. PAF öffnet Website, behält aber die Kontrolle
  4. Käufer benutzt die Website zum Aktienkauf
  5. PAF fängt die Reaktionen der Website ab und aktualisiert des Käufers Portfolio
  6. PAF zeigt dem Käufer das neue Portfolio



Bsp: Use Case "Über WWW Aktien kaufen" (3)

Freie Universität Berlin

- **Erweiterungen**:
  - 3a. Web-Fehler beim Verbindungsaufbau
    - 3a1. PAF meldet Fehler an Benutzer (mit Ratschlag) und geht zurück zum vorherigen Schritt
    - 3a2. Käufer verlässt Use Case oder probiert erneut
  - 4a. Computer stürzt während Transaktion ab oder wird ausgeschaltet
    - 4a1. (Was machen wir da???)
  - 4b. Website bestätigt Kauf nicht sofort, sondern stellt ihn auf Warteliste
    - 4b1. PAF protokolliert dies und setzt einen Terminmerker, um den Käufer später nach dem Ergebnis zu fragen
  - ...
  - 5a. Website liefert zu wenig Information über den Kauf
    - 5a1. PAF protokolliert den Mangel, lässt Käufer Kauf aktualisieren

0, 1 oder mehrere Varianten pro Schritt

Use Cases müssen nicht immer vollständig sein

Verweis auf Unter-Use-Case

Ein realer Use Case

- Die Beschreibung auf den drei vorherigen Folien war übrigens ein realer Use Case aus einem realen Projekt

  Und er ist recht gut gelungen:

  - Enthält genügend detaillierte Information
    - Ziel, Randbedingungen, Schritte, Problemsituationen
  - aber kaum etwas Unnötiges

Beispiel 2: Auto-Unfallschaden ersetzt bekommen

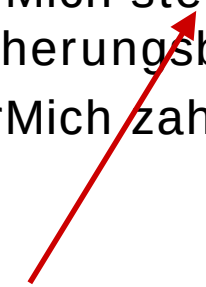
Freie Universität Berlin

- **Hauptakteur**: Geschädigter
- **Anwendungsbereich**: SicherMich AG (Versicherung)
- **Niveau**: Überblick
- **Beteiligte und Interessen**:
  - Geschädigter: Möglichst hohe Erstattung bekommen
  - SicherMich: Möglichst wenig Erstattung bezahlen
  - Aufsichtsbehörde: Einhaltung der Gesetze sicherstellen
- **Voraussetzung**: --
- **Mindestzusicherung**:
  - SicherMich protokolliert Schadensmeldung und alle Aktivitäten
- **Zusicherung im Erfolgsfall**:
  - Geschädigter u. SicherMich einigen sich über Entschädigung und SicherMich zahlt diesen Betrag an Geschädigten
- **Auslöser**: Geschädigter reicht Schadensmeldung ein

(wird fortgesetzt)

Beispiel 2: Auto-Unfallschaden ersetzt bekommen (2)

- **Haupt-Erfolgsszenario**:
  1. Geschädigter reicht Schadensmeldung und Detailinformation ein
  2. SicherMich prüft, dass Geschädigter eine gültige Police hat
  3. SicherMich weist den Fall einem Sachbearbeiter zur Prüfung zu
  4. SicherMich stellt fest, dass der Sachverhalt gemäß der Versicherungsbedingungen versichert ist
  5. SicherMich zahlt an Geschädigten und schließt den Fall



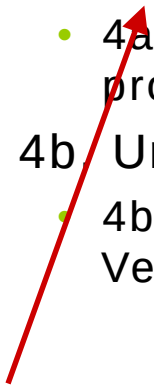
egal ob manuell oder
mittels einer Software

(wird fortgesetzt)

Beispiel 2: Auto-Unfallschaden ersetzt bekommen (3)

Freie Universität Berlin

- **Erweiterungen**:
  - 1a. Eingereichte Daten sind unvollständig
    - 1a1. SicherMich bittet Geschädigten um Auskünfte
    - 1a2. Geschädigter liefert fehlende Informationen
  - 2a. Geschädigter hat keine gültige Police
    - 2a1. SicherMich informiert Geschädigten, weist Antrag zurück, protokolliert all dies und schließt den Vorgang
  - 4a. Unfall verletzt wesentliche Versicherungsbedingungen
    - 4a1. SicherMich informiert Geschädigten, weist Antrag zurück, protokolliert all dies und schließt den Vorgang
  - 4b. Unfall verletzt Details der Versicherungsbedingungen
    - 4b1. SicherMich tritt in Verhandlungen mit Geschädigtem über Verringerung der Zahlung ein



"Unfall" ist kein Akteur! Eigentlich ist gemeint:
"SicherMich stellt fest, dass ..." (Bedingung)
etc.

Zusätzliche Felder

Je nach Projektsituation ist es evtl. sinnvoll, der Use-Case-Schablone weitere Felder hinzuzufügen, z.B.

- Lieferant der Anforderungen
  - Person. Für die Rückverfolgung bei Rückfragen
- Trennung von Erweiterungen, Fehlerfällen, Alternativen
  - Werden ggf. an andere Use-Cases delegiert:
  - Erweiterungen: "extends"-Beziehung
  - Alternativen, Fehlerfälle: "generalizes"-Beziehung
- Besondere Anforderungen
  - z.B. betreffend Datenformate, GUI o.ä. Sparsam einsetzen!
- Offene Fragen
  - Streitpunkte, Zweifelspunkte, erwartete Ergänzungen etc.
- Annahmen
  - z.B. über Kenntnisse des Benutzers

Akteure und Rollen

- Die Akteure werden stets aus einer Perspektive bezeichnet, die zum Use Case gehört
  - z.B. Käufer, Geschädigter
  - jede Bezeichnung führt also eine Rolle ein
- Im Anforderungsdokument muss irgendwo zusätzlich festgelegt werden, welche Beteiligten welche dieser Rollen einnehmen können
  - und unter welchen Umständen
- Diese Festlegungen definieren Rechte, Benutzergruppen, Bündelung von Funktionen im GUI jeder Benutzergruppe etc.

Arten von Use Cases

Use Cases können für viele **Zwecke** eingesetzt werden:

- Beschreibung von **Geschäftsprozesse**n einer Organisation
- Beschreibung der **funktionale**n **Anforderungen** einer Software oder Komponente
- Dokumentation des **Entwurf**s eines Systems
  - textuelle Alternative zu Sequenzdiagrammen
  - dann sind die Akteure Module und die Schritte sind z.B. Methodenaufrufe und deren Wirkungen/Resultate

Und in verschiedenen Projekt-**Kontexte**n, insbesondere:

- Kleine, eng gekoppelte Gruppen von Beteiligten
- Große oder örtlich verteilte Gruppen von Beteiligten

Arten von Use Cases (2)

Deshalb sollte man Use Cases entlang 2 Dimensionen unterscheiden:

- Bezug (Anwendungsbereich, scope, "system under discussion", SuD)
  - eine Organisation — Geschäftsprozess
  - ein Softwaresystem — Interaktion, funktionale Anforderungen
  - eine SW-Komponente — Verhalten einer Schnittstelle

- Abstraktionsgrad (Wie viele Details fehlen?)
  - Überblick — hoher Abstraktionsgrad
  - Benutzerinteraktion — mittlerer Abstraktionsgrad
  - Details — geringer Abstraktionsgrad

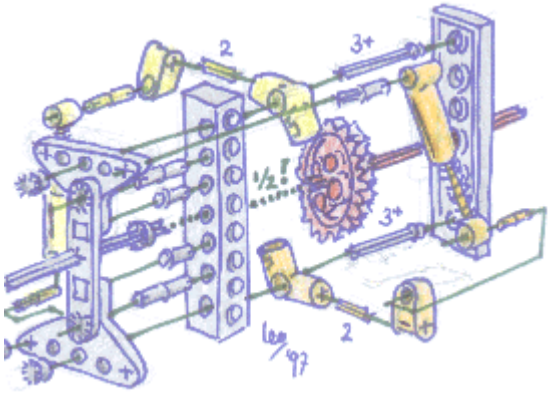
Arten von Use Cases (3)

Anders ausgedrückt: Man muss beantworten

- Von welchem System reden wir hier eigentlich?
  - Der ganzen Firma, dem zu bauenden System, einem Teil des Systems oder noch was anderem
  - Wenn das nicht klar ist, wird der Use Case unverständlich

- Wie genau sollten wir die Abläufe beschreiben?
  - Nur als Hintergrundinformation
  - oder genau genug, um es konstruieren zu können

  - Wenn die Genauigkeit *zu hoch* ist, liest niemand die Beschreibung mehr!



Arten von Use Cases (4)

- (1) Beispiel PAF (Aktien über WWW kaufen):
  - Bezug: Softwaresystem
  - Abstraktionsgrad: Benutzerinteraktion
- (2) Beispiel SicherMich (Auto-Unfallschaden ersetzt bekommen):
  - Bezug: Organisation
  - Abstraktionsgrad: Überblick
- Dies sind auch die beiden häufigsten Sorten:
  - (1) Interaktionsbeschreibungen für SW-Systeme liefern **funktionale Anforderungen**
  - (2) Überblicksbeschreibungen für Organisationen liefern den **Hintergrund für das Verständnis** von Anforderungen

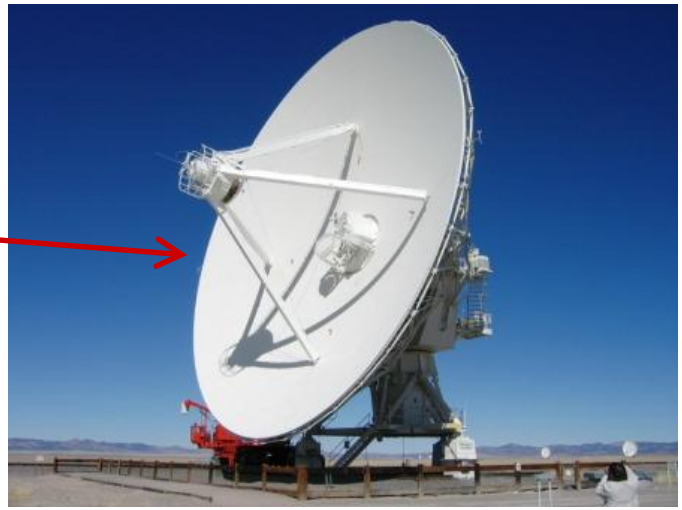
Feinkörnigere Use Cases sind möglich

- Der Hauptakteur muss kein Mensch sein
  - → also kann der Bezug eine Softwarekomponente ohne Benutzerinteraktion sein
  - ist evtl. einfacher aufzuschreiben als ein Sequenzdiagramm
- Ein Use Case kann auch feine Details enthalten
  - Aber: Er sollte stets **Ziele / Zwecke** beschreiben, nicht bloße technische Einzelheiten
- Ein Use Case kann "Innereien" offen legen ("white box")
    - d.h. nicht nur die sichtbare Antwortreaktion beschreiben, sondern auch was dafür intern vorging
  - und beschreibt dann Entwurf, nicht (nur) Anforderungen.
  - Das ist jedoch eine ungewöhnliche Verwendung.
  - Normalerweise bieten Use Cases keinen Einblick ("black box")
    - Beide Beispiele (PAF und SicherMich) waren ohne Einblick

Vorsicht mit Präzision!

- Genauigkeit hat zwei Facetten:
  Präzision (precision) und Richtigkeit (accuracy)

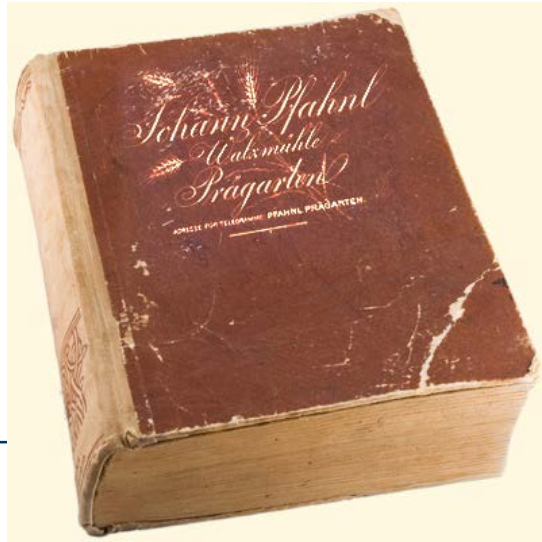
- Pi ≈ 4,141592654
  - sehr präzise, aber wenig richtig
- Pi ≈ 3
  - wenig präzise, aber völlig richtig
- Pi ≈ 3,14
  - ausreichend präzise und völlig richtig
  - Der Fehler durch Präzisionsmangel ist hier schon kleiner als die Ungewissheit durch die Wärmeausdehnung!



Präzisions-Gefahren bei Use Cases

Wer zu früh zu genau spezifiziert, geht Gefahren ein:

- **Gefahr 1: Alles falsch!**
  - Oft übersieht man anfangs etwas Wichtiges total oder sitzt einem wichtigen Missverständnis auf
    - und muss eigentlich alles ganz anders machen
  - Wer dann schon in Details investiert hat, setzt viel Arbeit in den Sand
- **Gefahr 2: Niemand liest's!**
  - Die Beteiligten an einem Projekt sind in der Regel sehr an Informationen, Überblicken etc. interessiert
  - Aber wenn Dokumente zu umfangreich werden, liest niemand sie mehr



- Also: Weniger ist oft mehr
  - Use Cases sollten immer nur gerade *genau genug* sein.

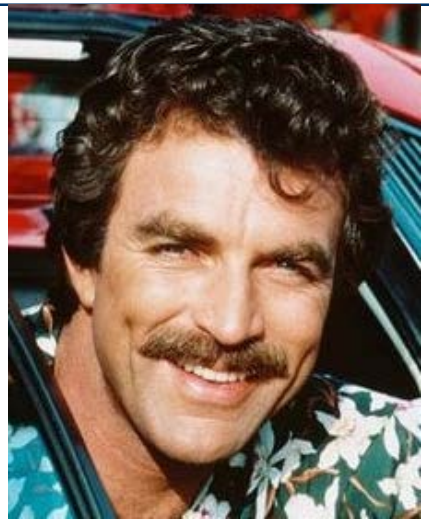
Präzision bei Use Cases: Empfehlung

Use Cases sollte man schrittweise präziser machen:

- Stufe 1: Hauptakteur und Ziel
  - Schon das klärt wichtige Anforderungen und Beteiligte
  - Nun erst mal prüfen, ob alles wirklich stimmt
- Stufe 2: Haupt-Erfolgsszenario
  - Reicht in manchen Fällen schon ganz aus
  - Auch hier prüfen, ob es so akzeptabel ist (z.B. Benutzbarkeit)

- Stufe 3: Erweiterungsbedingungen (Varianten, Fehler)
  - Welche Probleme und Varianten müssen ggf. behandelt werden?
    - "Gefahr erkannt, Gefahr schon halb gebannt"
- Stufe 4: Erweiterungsschritte
  - Wie werden die Probleme/Varianten behandelt?
  - Kann man oft weglassen: Wird dann informell geklärt

Konzentrieren
Sie sich auf das Wesentliche!



Tom Selleck

Saddam Hussein

Präzision der Form

Nicht nur beim Inhalt, auch in der Form der Use Cases gilt die Regel "bitte gerade genau genug":

- Informelle Use Cases sind eventuell reiner Fließtext
  - ganz ohne Abschnittsnamen und feste Struktur
  - Die wichtigsten Angaben (Hauptakteur, Ziel, wichtigste Schritte) müssen natürlich erkennbar sein
    - Meist ist etwas Struktur aber dermaßen hilfreich, dass sich ein tabellarisches Format empfiehlt

Der umgekehrte Fall gilt ebenso:

- Wenn die Anforderungen kritisch sind oder im Team keine hohe Kommunikation herrschen kann
  - z.B. weil es sehr groß oder örtlich verteilt ist
- dann kann verstärkte Struktur nützlich sein
  - z.B. mehr feste Abschnitte oder rigidere Formulierungsweise

Wichtigkeit von Zielen

- Ein gutes Anforderungsdokument benennt Anforderungen nicht einfach, sondern begründet sie
  - damit man Mehrdeutigkeiten leichter ausräumen kann
  - damit man Prioritäten vernünftig setzen kann

- In Use Cases werden die Begründungen durch die Ziele geliefert
  - Ein Ziel beantwortet zu den Schritten die Frage "Warum?"
  - Der Name eines Use Cases sollte ein Ziel bezeichnen
  - In einem umfassenderen Use Case tritt dieses Ziel als Schritt auf

- Ziele-als-Schritte beschreibt eine hierarchische Zerlegung
  - so dass einzelne Use Cases übersichtlich/beherrschbar bleiben
  - Ein guter Use Case hat meistens 3 bis 9 Schritte

Beispiel: Gesund werden (1)

- Use Case: Patient (Hauptakteur) will **gesund werden** (Ziel) und besucht dazu einen Arzt (SuD)

- Schritte:
  - 1. Arzt stellt Diagnose
  - 2. Arzt verordnet Medikament
  - 3. Patient beschafft Medikament (SuD: Apotheke)
  - 4. Patient nimmt Medikament ein

- Varianten:
  - 2a. Arzt überweist an Facharzt
  - 2b. Arzt überweist ins Krankenhaus

Beispiel: Gesund werden (2)

- Use Case:
  Arzt (Hauptakteur) **stellt Diagnose** (Ziel) für Patient

- Schritte:
  - 1. Arzt validiert Krankenversicherung via eGK
  - 2. Arzt befragt Patient nach Symptomen und Historie
  - 3. Arzt untersucht Patient
  - 4. Arzt entscheidet Diagnose(n)
  - 5. Arzt speichert Diagnose auf eGK
- Varianten:
  - 1a. Arzt geht direkt zu Notfallversorgung über (Ende)
  - 1b. eGK oder eGK-Infrastruktur nicht verfügbar
  - 5a. Patient verweigert Speicherung
  - 5b. eGK oder eGK-Infrastruktur nicht verfügbar

Beispiel: Gesund werden (3)

- Use Case: Arzthelfer (Hauptakteur)
  validiert Krankenversicherung (Ziel) für Patient
  über die eGK (SuD)

- Schritte:
  - 1. Arzthelfer legt HBA in eGK-Terminal ein und authentisiert sich
  - 2. Patient legt eGK in eGK-Terminal ein und authentisiert sich
  - 3. Arzthelfer fragt Krankenversicherungsstatus ab

Beispiel: Gesund werden – Beobachtungen

Freie Universität Berlin

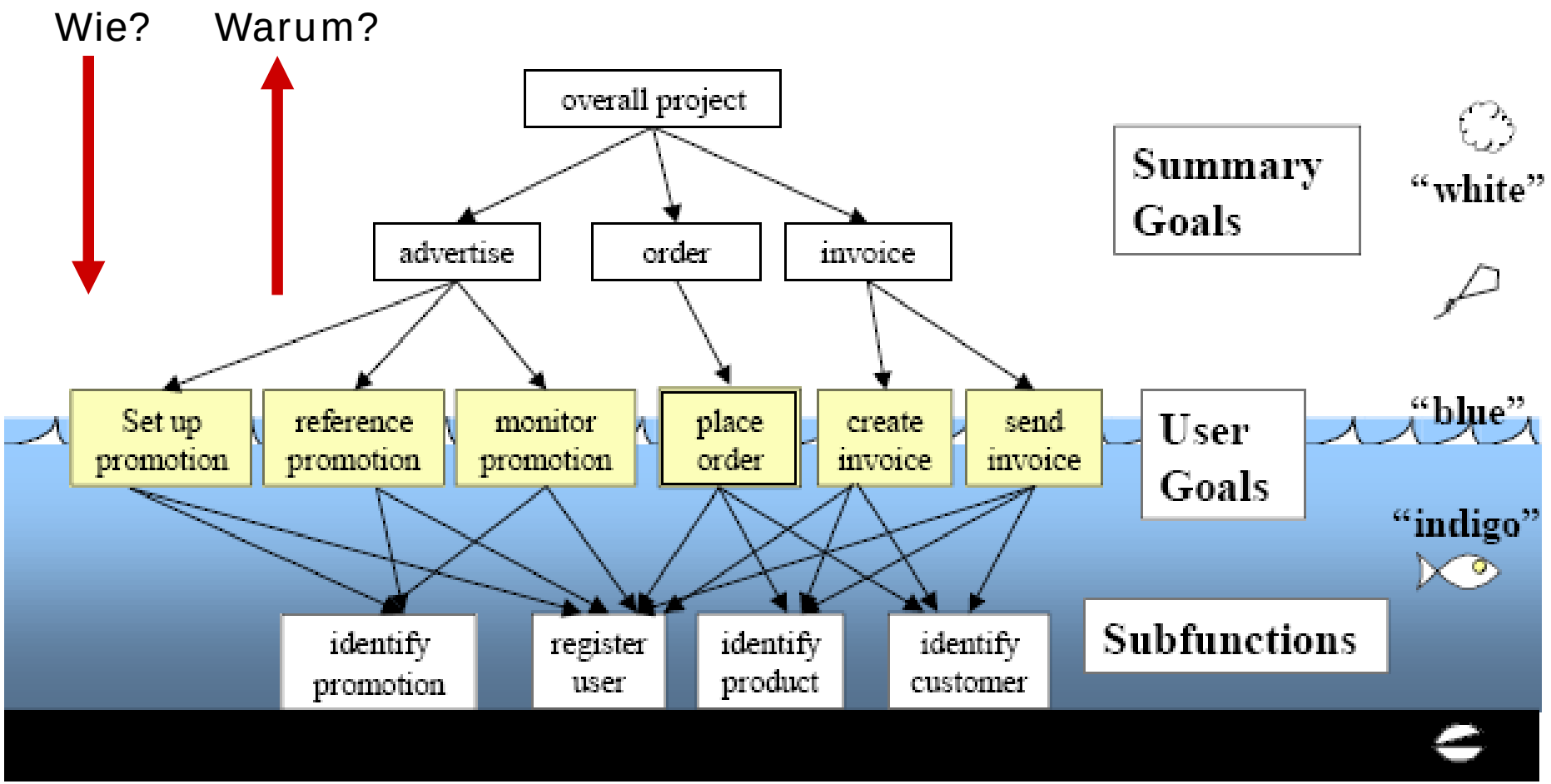
1. Es entsteht eine Hierarchie von Zielen/Zwecken:
   - HBA in eGK-Terminal einlegen hat den Zweck
     Krankenversicherung validieren hat den Zweck
     Diagnose stellen hat den Zweck
     Gesund werden
2. "Arzt" ist gar keine Person, sondern eine Organisation
   - Akteurswechsel von "Arzt prüft" zu "Arzthelfer legt ein"
3. Klare Benennung der Akteure macht auf Probleme aufmerksam:
   - Wer wird am Ende die Diagnose speichern: Arzt oder Arzthelfer?
   - Falls Arzt: Muss sich der Patient dann nochmals anmelden?
   - Allgemein: Wie oft pro Tag melden sich Arzt/Arzthelfer an?
   - etc.
4. Geringe Zahl von Schritten erlaubt jederzeitige Orientierung

Nochmal: Das richtige Detailniveau

Ein gutes Projekt sollte haben:

- 1 Überblicks-Use-Case für das ganze Projekt
- Je 1 Überblicks-Use-Case für jeden Bereich von Zielen
  - meistens 2 bis 4; fast niemals mehr als ein Dutzend
- Je 1 **Überblicks-Use-Case** für Abläufe, die aus mehr als einer Sitzung am Rechner bestehen
  - selten mehr als ein Dutzend; oft Ersatz für obige Bereiche
- Je 1 **Benutzerziel-Use-Case** für jeden Ablauf, der "in einem Rutsch" (1 Sitzung) absolviert wird
  - bei einem großen System eventuell einige Dutzend
  - bei langen Abläufen zerlegt in Teile
- Je 1 **Detail-Use-Case** für besonders komplizierte Teilabläufe
  - Die meist mehrfach wiederverwendet werden
  - Je nach gewünschten Detailgrad viele oder wenige

Beispiel


Wie?
Warum?
overall project
advertise
order
invoice
Summary Goals
"white"
Set up promotion
reference promotion
monitor promotion
place order
create invoice
send invoice
User Goals
"blue"
"indigo"
identify promotion
register user
identify product
identify customer
Subfunctions

Schrittweises Modell für das Gesamtvorgehen

Freie Universität Berlin

Globale Schritte:

- Lege die Grenzen des Systems fest
  - Kontextbild, Liste von Eingaben und Ausgaben
- Finde alle Hauptakteure
  - Vollständigkeit ist essenziell
- Finde alle wichtigen Ziele jedes Hauptakteurs
  - *(Man kann die Leute dafür sogar fragen!)* ☺
- Schreibe wenige, grobe, zusammenfassende Überblicks-Use-Cases, die alle Ziele und Akteure erfassen
  - Das charakterisiert den Zweck und die Prioritäten des Systems
- Analysiere diese: Ziele zufügen, entfernen, vereinigen
  - Befrage dabei die Hauptakteure

Jetzt geht es an die einzelnen Benutzer-Interaktionen:

Schrittweises Modell für das Gesamtvorgehen (2)

Freie Universität Berlin

Für jeden Use Case auf Benutzerebene:

- Wähle nächsten Use Case zum Ausarbeiten aus
- Schreibe das Haupt-Erfolgsszenario
  - möglichst stets 3 bis 9 Schritte
- Finde und beschreibe alle Misslingensfälle und alternative Erfolgsfälle
- Breche Teil-Use-Cases heraus wo nötig

Use Cases und Szenarios

- In Fällen, in denen das Schreiben eines Use Case schwer fällt
  - oder wenn man erwartet, dass er für manche Beteiligte zu schwierig zu verstehen ist (weil zu abstrakt)
- kann man zusätzlich Szenarios verwenden:
    - Meistens kommt man aber ohne Szenarios aus

- Ein Szenario ist eine konkrete(re) Ausprägung eines Use Case
  - Es gibt also keine Varianten, sondern nur einen linearen Ablauf
  - Dieser kann ganz konkrete Datenwerte etc. angeben
  - Der Entwicklungsstil ATDD (Acceptance-test-driven development) würde statt 1 Use Case jeweils N konkrete Szenarios benutzen und diese als automatisierte Testfälle *vor* der SW implementieren

Checkliste für Use-Case-Bestandteile

Bei einem guten Use Case lautet jede Antwort "ja":

- Titel:
  - Aktive Verbalphrase, die das Ziel des Hauptakteurs nennt?
  - Ist das SuD für die Zielerreichung "zuständig"?
- Bereich (SuD):
  - Angegeben?
  - Wenn das SuD entworfen werden muss, müssen (a) alle Teile entworfen werden und (b) nichts außerhalb (Systemgrenze)?
- Detailgrad/Zielniveau:
  - Liegt das Ziel wirklich auf diesem Niveau?
  - Passt der Inhalt zum geplanten Detailgrad?
- Hauptakteur:
  - Hat er/sie/es Verhalten?
  - Hat er/sie/es ein Ziel, das zu einer Dienstzusicherung des SuD passt?

Checkliste für Use-Case-Bestandteile (2)

- Voraussetzungen:
  - Sind sie verbindlich und herstellbar?
  - Werden sie im Use Case nicht mehr überprüft?
- Beteiligte und ihre Interessen:
  - Muss das SuD in diesem Use Case diese Interessen bedienen?
- Mindestzusicherungen:
  - Sind die Interessen aller Beteiligten angemessen geschützt?
- Zusicherungen im Erfolgsfall:
  - Sind die Interessen aller Beteiligten befriedigt?
- Haupt-Erfolgsszenario:
  - Hat es 3 bis 9 Schritte?
  - Beschreibt es den Ablauf vom Auslöser bis zur Erfolgsgarantie?
  - Erlaubt es ggf. geeignete Abwandlungen in der Reihenfolge?

Checkliste für Use-Case-Bestandteile (3)

- Jeder Einzelschritt:
  - Ist er als erreichtes Ziel formuliert?
    - z.B. "validieren", nicht: "prüfen"
  - Treibt er den Prozess sichtbar voran?
  - Ist klar, wer der handelnde Akteur ist?
  - Ist die Absicht des Akteurs klar?
  - Abstrahiert der Schritt von der Bedienschnittstelle?
  - Ist erkennbar, welche Information verarbeitet wird?
- Erweiterungsbedingungen:
  - Kann das SuD diesen Fall entdecken (falls nötig)?
  - Muss das SuD diesen Fall behandeln?
- Inhalt des Use Cases insgesamt:
  - Gegenüber den Beteiligten: *"Ist dies, was Du möchtest?", "Kannst Du später entscheiden, ob es richtig gebaut wurde?"*
  - Gegenüber den Entwicklern: *"Kannst Du das bauen?"*

Und: Anforderungen müssen einleuchten

- Gute, sinnvolle Anforderungen kann man immer in einleuchtender Form beschreiben
  - Evtl. fehlt nur ein Überblick
- Hinterfragen Sie also uneinleuchtende!

Wie geht's dann weiter?

- Use Cases sind aus Benutzersicht wunderbar
  - klar, prüfbar, einleuchtend
- Auch als Werkzeug für die Anforderungserhebung sind sie günstig
  - einfach zu schreiben, einfach mit Beteiligten abzugleichen

- Aus Entwicklersicht lassen sie aber zu wünschen übrig
  - Sie beschreiben nicht, welche **Funktionen** die SW braucht
  - Und auch nicht, was für **Daten** im System vorhanden sind
  - Es bleibt unklar, wie man die Entwicklung auf Personen verteilen kann
- Deshalb müssen als nächstes Daten und Funktionen identifiziert werden
  - Anforderungs**analyse**
  - Siehe nächste Vorlesungen

Warnung: Use Cases sind nur ein Teil!

Freie Universität Berlin

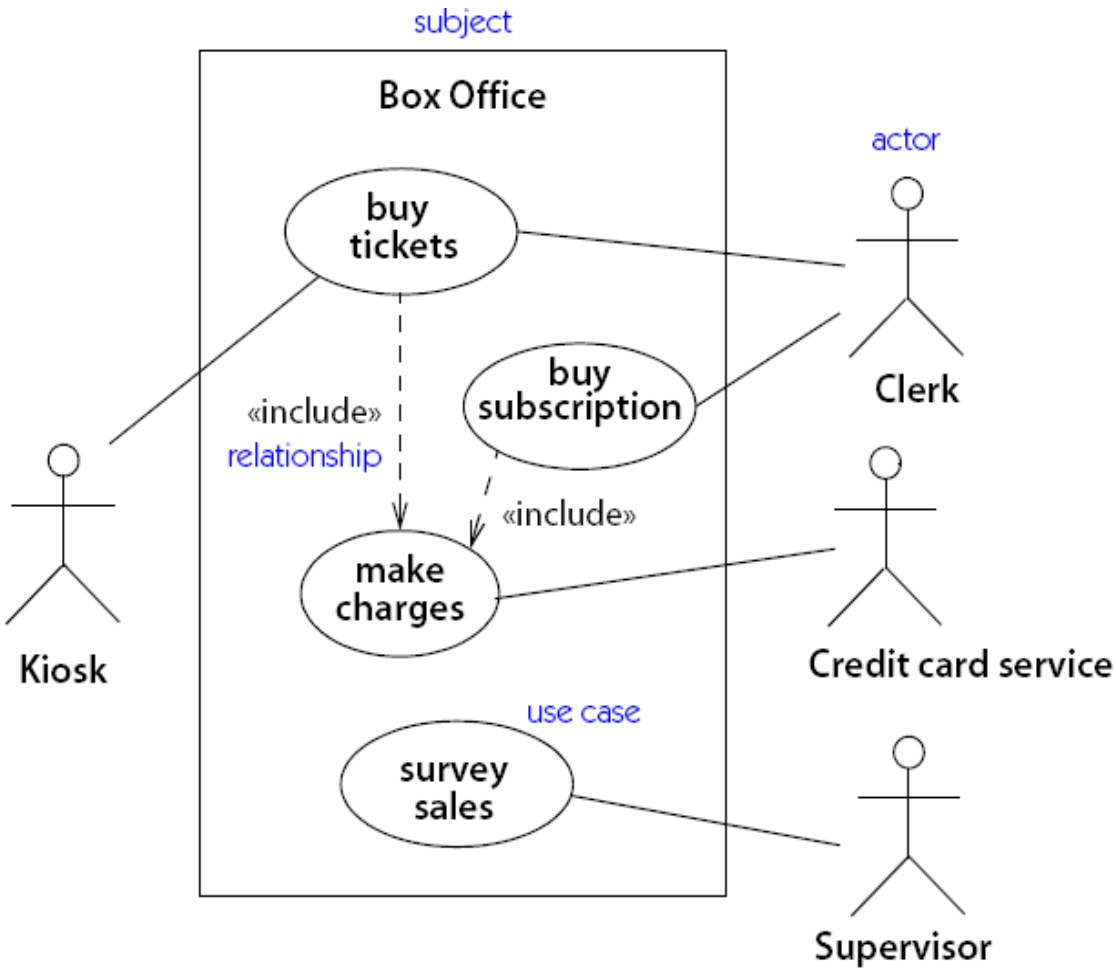
- Use Cases decken nur Verhaltensanforderungen ab!

- Sie beschreiben z.B. nicht
  - Geschäftsregeln (globale Festlegungen);
  - nichtfunktionale Anforderungen;
  - ob bestimmte konkrete Funktionen gewünscht werden;
  - Domäneneigenschaften und Annahmen;
  - Datenmodell, technische Schnittstellen u.ä.;
  - technische, organisatorische, rechtliche u.a. Randbedingungen

- Das Anforderungsdokument muss also mehr als nur Use Cases enthalten
  - aber Use Cases bilden einen guten Rahmen für den Rest

Warnung: Use-Case-Diagramme

- UML-Use-Case-Diagramme enthalten im Gegensatz zu anderen UML-Diagrammen kaum Information
  - Sind nur nützlich, um Übersicht über die Zusammenhänge zwischen mehreren Use Cases zu fördern
    - «include» war bei uns Unterstreichen
  - Bitte nie das Diagramm mit dem Use Case verwechseln!

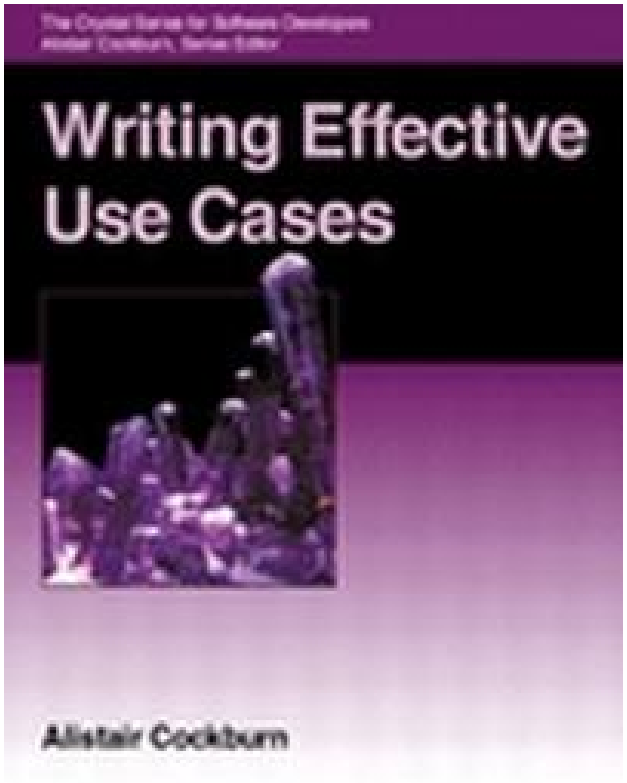


Zusammenfassung

- Was ist ein Use Case?
  - Beispiele
  - Felder
  - Arten von Use Cases
- Wichtige Parameter
  - Bereich ("system under discussion", SuD)
  - Detailgrad/Zielniveau
- Schrittweise Präzisierung
  - Gefahren von Präzision
- Gesamtvorgehens-Schritte
- Checkliste für Use Cases

Quelle

- Alistair Cockburn: "Writing Effective Use Cases", Addison-Wesley 2001
  - http://alistair.cockburn.us/crystal/books/weuc/weuc0002extract.pdf



Danke!