Anforderungsanalyse

Requirements Engineering

Dokumente

Methoden

Use Case-Diagramme

Anforderungen

- **Anforderungsanalyse (Requirements Engineering)**
  - Prozess des Auffindens, der Dokumentation und Validierung der **Anforderungen**
  - Ziel: **Spezifikation** des geforderten Systems:
    präzise, vollständige, detaillierte Beschreibung der Anforderung
- **Anforderung**
  - Beschreibung der Leistungen eines Systems
  - Beschreibung von Rahmen- und Nebenbedingungen
  - Beschreibung des Bedarfs des Auftraggebers / der Nutzer

Was soll das System tun?

Arten von Anforderungen

- **funktionale Anforderungen**
  - Dienste **(Funktionen),** die das System erbringen soll
  - Transformationen von Eingabe- in Ausgabe**daten**
  - System**zustände**
- **nicht-funktionale Anforderungen**
  - **„messbare“ Anforderungen**
    - Effizienz
      (Rechenzeit, Speicherplatz, Reaktionszeit)
    - Genauigkeit
    - Robustheit
      (Fehlertoleranz, Wahrscheinlichkeiten, ...)
    - Sicherheit
  - **sonstige Anforderungen**
    - Zuverlässigkeit
    - Bedienerfreundlichkeit
    - Regularien
    - ...

Unser durchgängiges Beispiel

Universitätsbibliotheks-Verwaltungssystem (UBVS)

Die UB hat Bücher und Zeitschriften im Bestand, von den Büchern teilweise mehrere Exemplare. Das System erlaubt den Mitgliedern der Bibliothek, ein Buch auszuleihen. Dabei darf jedes Mitglied höchstens 6 Bücher gleichzeitig ausleihen. Nur Mitarbeiter der Universität (Professoren, wissenschaftliche und technische Angestellte, keine Studenten) dürfen bis zu 12 Bücher gleichzeitig ausleihen. Mitarbeiter dürfen außerdem Zeitschriften ausleihen.

Neue Artefakte werden in den Katalog eingearbeitet, alte aus dem Bestand ausgesondert.

Das System registriert, an wen und wie lange Artefakte des Bestandes ausgeliehen sind. Für jedes Mitglied hält es die Information, wie viele und welche Artefakte es zur Zeit ausgeliehen hat. Das System erinnert automatisch an überfällige Ausleihen.

Mitglieder können mit Hilfe des Systems ferner den Bestand der UB einsehen („Browsing“), ausgeliehene Bücher reservieren und Buchausleihen verlängern, falls keine Reservierung vorliegt.

Anforderungsdokumente (1)

- **User Requirements (Benutzeranforderung)**
  - Beschreibung der *Erwartungen* der Benutzer, *externe Sicht* auf das System
  - oft in natürlicher Sprache (evtl. *intuitive* Diagramme)
  - Fokus auf Gegenstandsbereich (*keine Lösung vordefinierend*)
  - kann als Ausschreibung genutzt werden
  - Grundlage der Machbarkeitsstudie / ersten Planung
  - kann Sammlung von „*User-Stories*“ sein (*agiles Vorgehen*)

Universitätsbibliotheks-Verwaltungssystem (UBVS)
Buch-Ausleihe: *Das System erlaubt den Mitgliedern der Bibliothek, ein Buch auszuleihen. Dabei darf jedes Mitglied höchstens 6 Bücher gleichzeitig ausleihen. Nur Mitarbeiter der Universität (Professoren, wissenschaftliche und technische Angestellte, keine Studenten) dürfen bis zu 12 Bücher gleichzeitig ausleihen.*

Anforderungsdokumente (2)

- **System Requirements (Systemanforderungen)**
  - exakte, detaillierte Beschreibung der Leistungen des Systems
  - legt genau fest, was implementiert werden soll
  - bildet Grundlage für den Systementwurf
  - oft Bestandteil des Vertrages (*verständlich; keine technischen Details*)
  - Verwendung formaler Spezifikationsmittel (Diagramme)

Universitätsbibliotheks-Verwaltungssystem (UBVS)
Buch-Ausleihe: *Eine Person („Ausleiher“) präsentiert ein Buch. Das System prüft,*
1. *ob der Ausleiher Mitglied der Bibliothek ist und*
2. *noch nicht die maximale Anzahl von Büchern ausgeliehen hat. Dieses Maximum ist 6, es sei denn, das Mitglied ist Mitarbeiter der Universität, dann ist es 12.*

Sind beide Eigenschaften erfüllt, registriert das System, dass das Mitglied das präsentierte Buch ausgeliehen hat. Sonst wird die Ausleihe abgelehnt.

Anforderungsdokumente (3)

- Verwendung der Anforderungsdokumente
  - **Kunde/Benutzer** Validierung, Identifizieren von Änderungen/Ergänzungen
  - **Manager** Projektplanung (Zeit, Personal, Kosten, Vorgehen)
  - **Entwickler** Verständnis des zu entwickelnden Systems
  - **Test-Ingenieure** Entwicklung von Validierungstests
  - **Wartungsingenieure** Verständnis des Systems und seiner Rolle

→ in allen Kernphasen der Softwareentwicklung

Lastenheft

- *strukturierte* **Benutzeranforderung**
- **Autor:**
  - Standardsoftware: intern erstellt (z.B. Marketing-Abteilung)
  - Individualsoftware: vom Auftraggeber (*Ausschreibung*) oder Auftragnehmer (*Angebot*) erstellt; *ideal:* gemeinsam (**Workshop**)
  - falls nicht vorgegeben: Ergebnisdokument der Analyse (*Wasserfallmodell*)
- **Adressaten:** Auftraggeber + Auftragnehmer (vollständige Adressen)
- **Zeitpunkt:** erste Dokumentation des zu erstellenden Produkts
- **Umfang:** auf wenige Seiten beschränkt
- **Gliederung:** es existieren verschiedene Gliederungsschemata

Gliederung Lastenheft (Beispiel)

[Balzert: Lehrbuch, Bd. 1]

1. **Zielbestimmung**
   Welches Ziel soll erreicht werden?; Abgrenzung
2. **Produkteinsatz**
   Anwendungsbereiche und Benutzergruppen
3. **Produktfunktionen**
   funktionale Anforderungen, nummeriert: /LF10/, /LF20/, ...
4. **Produktdaten**
   persistente Daten, nummeriert: /LD10/, /LD20/, ...
5. **Produktleistungen**
   messbare nicht-funktionale Anforderungen
6. **Qualitätsanforderungen**
   sonstige nicht-funktionale Anforderungen
7. **Ergänzungen**
   z.B. Barrierefreiheit, Mehrsprachigkeit, Vorbereitung für Ausbaustufen, ...

Pflichtenheft

- offizielles Dokument mit der **Systemanforderung**
  - korrekt, vollständig, präzise
  - Bestandteil des Vertrages
  - Basis für Produktabnahme
- Ergebnisdokument der Anforderungsdefinition

  ***Was**? Nicht wie?*
- **Adressaten:** Auftraggeber + Auftragnehmer (vollständige Adressen)
- **Zeitpunkt:** am Ende der Planungs- und Analysephase
- **Umfang:** ausführlich, detailliert (aber hinreichend abstrakt)
- **Gliederung:** es existieren verschiedene Standards z.B.: ANSI/IEEE Std 830-1998

Wesentliche Inhalte im Pflichtenheft

- Anforderungen an das Produkt
  - Muss-, Kann-, Abgrenzungskriterien (vor allem funktionale Eigenschaften)
- Einsatzgebiet
  - Anwendungsdomäne
  - Benutzergruppe (Spezialisten/Laien/...)
  - Betriebsbedingungen (Einsatzhäufigkeit/Räumlichkeiten/Mobilität?/...)
- Produktumgebung
  - Plattform/Integration in bestehende Systeme/Kompatibilitäten/...
- Produktleistungen
  - Leistungsparameter (nicht-funktionale Eigenschaften)
  - Fertigstellungstermine
- Abnahmetests
- Ergänzende Vereinbarungen/Vertragsbedingungen

Gliederung Pflichtenheft (1. Beispiel)

1. Einleitung
    1. Zweck
    2. Dokument-Vereinbarungen
    3. beabsichtigte Publikum und Lesevorschläge
    4. Produkt-Bereich
    5. Referenzen
2. Gesamtbeschreibung
    1. Produkt-Perspektive
    2. Produkt-Funktionen
    3. Benutzer-Kategorien und Eigenschaften
    4. Betriebsumgebung
    5. Design-und Implementierung Begrenzungen
    6. Benutzer-Unterlagen
    7. Annahmen und Abhängigkeiten
3. Externe Schnittstellen-Anforderungen
    1. Benutzer-Schnittstellen
    2. Hardware-Schnittstellen
    3. Software-Schnittstellen
    4. Kommunikations-Schnittstellen
4. System Eigenschaften
    1. System Eigenschaft 1
        1. Beschreibung und Priorität
        2. Anregung-/ Wartereihenfolgen
        3. Funktionale Anforderungen
    2. System Eigenschaft 2…
5. Andere nicht funktionale Anforderungen
    1. Leistungsanforderungen
    2. Sicherheitsauflagen
    3. Sicherheit Anforderungen
    4. Software-Qualitätsattribute
    5. Geschäftsregeln
6. Andere Anforderungen

Gliederung Pflichtenheft (2. Beispiel)

1. Zielbestimmung
   1. Grenzkriterien
   2. Wunschkriterien
   3. Abgrenzungskriterien
2. Produkteinsatz
   1. Anwendungsbereiche
   2. Zielgruppen
3. Produktkonfiguration Software, Hardware, organisatorische Randbedingungen
4. Produkt-Umgebung Schnittstellen, Betriebsbedingungen
5. Produkt-Funktionen
   1. Funktion 1
   2. Funktion 2 usw.
6. Produkt-Leistungen
7. Benutzerschnittstelle
8. Qualitäts-Zielbestimmung
9. Globale Testfälle
   1. Testfall 1
   2. Testfall 2 usw.
10. Entwicklungs-Konfiguration Software, Hardware, organisatorisches Randbedingungen
11. Ergänzungen

[Balzert: Lehrbuch, Bd. 1]

Dokumente bei Agilem Vorgehen

- schnelle Änderung von Anforderungen erwartet
- *Dokumente veralten schneller, als sie geschrieben werden.*

→ kein formales Anforderungsdokument

→ Benutzeranforderungen *inkrementell* sammeln

  - einzelne User Stories
  - „Karteikarten“
  - häufig strukturiert (*Templates*)
  - Basis für Entwicklung von Entwürfen, Testfällen, Implementierung des nächsten System Releases
- oft kurzes, unterstützendes Dokument
  - geschäftliche Rahmenbedingungen
  - Systemanforderung („System als Ganzes“), Einsatzbedingungen, ...

Dokumente bei Agilem Vorgehen

Buch-Ausleihe/UBVS/x.y.z

Funktion	Ausleihe eines Buch-Exemplars
Beschreibung	Überprüft die Berechtigung zur Ausleihe, registriert die Ausleihe oder weist sie ab.
Eingaben	Daten des Exemplars und des Ausleihers
Ausgaben	Daten des Exemplars und des Ausleihers, aktualisiert
Vorbedingung	1. Anzahl nicht ausgeliehener Exemplare > 0 2. Ausleiher ist als Mitglied registriert 3. Ausleiher hat Maximum an ausleihbaren Artefakten des Bestands nicht erreicht (Maximum: 6, für Mitarbeiter der Universität 12)
Nachbedingung	- Anzahl nicht ausgeliehener Exemplare dekrementiert - Anzahl der durch das Mitglied ausgeliehen Bücher um 1 erhöht - Status „ausgeliehen“, Rückgabedatum und Mitglied als Ausleiher des Exemplars registriert
...	

Analyseprozess

1. **Machbarkeitsstudie**
   - Beitrag zu den Gesamtzielen des Auftraggebers?
   - Erreichbarkeit der Ziele
     - in der vorgegebenen Zeit,
     - mit dem vorgegebenen Budget,
     - mit den vorhandenen Personal- und Organisationsstrukturen,
     - mit der verfügbaren Technologie?
   - Integrierbarkeit in vorhandene Systeme?
2. Iteration der Aktivitäten
   - Identifizieren von Anforderungen
   - Klassifizierung und Organisation der Anforderungen
     - Gruppierung, Dekomposition, Abhängigkeiten
   - Festlegung von Prioritäten
     - Finden und Auflösen von Konflikten
   - Spezifikation der Anforderungen

Identifizieren von Anforderungen (1)

Sammlung von Informationen

- über das zukünftige System
- über bestehende Systeme
- das Anwendungsgebiet (*Domäne*)

- vorhandene **Dokumente** (Systeme, Domäne, Organisation, Prozesse)
- Interaktion mit dem Auftraggeber und künftigen Benutzern!!!
- **Interviews**
  - *Was habt ihr?*
  - *Was wollt ihr?*
  - Vorbereitung: domänenspezifische Kenntnisse
  - offene, vertrauensvolle Atmosphäre
  - gezielte Fragen, gemeinsames Arbeiten

Identifizieren von Anforderungen (2)

- **Szenarien**
  - „real-life“-Beispiele von Sessions
    (*Bedienung des Systems, um ein bestimmtes Ziel zu erreichen*)
  - Benutzer-System-Interaktionen zu einzelnen User-Stories
  - Ziel: Beschreibung der Interaktionen, Finden von Details
    - System- und Benutzer-„Erwartungen“ zu Beginn des Szenarios
    - erwartete Folge von Aktivitäten
    - Bedingungen für das Szenario oder die Aktivitäten
    - Aktivitäten in der Umgebung des Szenarios
    - Systemzustand am Ende des Szenarios
  - Kooperation mit den künftigen Benutzern!!!
  - Beschreibung in Textform (erzählerisch, z.T. strukturiert), Diagrammen, Screenshots etc.

Identifizieren von Anforderungen (2)

- **Szenarien**
  - „real-life“-Beispiele von Sessions
    (*Bedienung des Systems, um ein bestimmtes Ziel zu erreichen*)
  - Benutzer-System-Interaktionen zu einzelnen User-Stories

Universitätsbibliotheks-Verwaltungssystem / Buchausleihe:

Szenario 1: Ein Bibliotheksmitglied leiht ein Exemplar eines Buches, das in der Bibliothek zur Verfügung steht, wenn er noch kein Buch ausgeliehen hat. Das System wird entsprechend aktualisiert.

Szenario 2: Ein Bibliotheksmitglied versucht ein Exemplar eines Buches auszuleihen, das in der Bibliothek zur Verfügung steht. Dieser Versuch schlägt fehlt, weil das Mitglied bereits 6 Bücher ausgeliehen hat, womit es sein Maximum an gleichzeitig ausleihbaren Büchern bereits erreicht hat.

Identifizieren von Anforderungen (3)

- **Use Cases**
  - Darstellung von einzelnen Aufgaben des Systems, die eine externe Interaktion mit dem System erfordern (*Benutzersicht* !)
  - Zusammenfassung von gleichartigen Szenarien (z.B. *„Buchausleihe“*)
    - Szenarien sind „Instanzen“ von Use Cases
  - objektorientierte Sichtweise und Terminologie
  - **Akteure:** Wer interagiert mit dem System?
    - Personen
    - andere Systeme

    } *Rollen als Klassen von Interaktionspartnern*
  - **Interaktionen** (Use Cases): Was tun Akteure mit dem System?
  - UML-Standard zur Darstellung von Use Cases und deren Beziehungen (Use Case-Diagramme)

    Unified Modeling Language

Akteure

1. potenzielle **menschliche Benutzer**
   - **Rollen** bei der Interaktion mit dem System identifizieren

     ⟶ eine Person – verschiedene Rollen

     ⟶ eine Rolle – verschiedene Personen(gruppen)

Unser durchgängiges Beispiel

Universitätsbibliotheks-Verwaltungssystem (UBVS)

Die UB hat Bücher und Zeitschriften im Bestand, von den Büchern teilweise mehrere Exemplare. Das System erlaubt den Mitgliedern der Bibliothek, ein Buch auszuleihen. Dabei darf jedes Mitglied höchstens 6 Bücher gleichzeitig ausleihen. Nur Mitarbeiter der Universität (Professoren, wissenschaftliche und technische Angestellte, keine Studenten) dürfen bis zu 12 Bücher gleichzeitig ausleihen. Mitarbeiter dürfen außerdem Zeitschriften ausleihen.

Neue Artefakte werden in den Katalog eingearbeitet, alte aus dem Bestand ausgesondert.

Das System registriert, an wen und wie lange Artefakte des Bestandes ausgeliehen sind. Für jedes Mitglied hält es die Information, wie viele und welche Artefakte es zur Zeit ausgeliehen hat. Das System erinnert automatisch an überfällige Ausleihen.

Mitglieder können mit Hilfe des Systems ferner den Bestand der UB einsehen („Browsing“), ausgeliehene Bücher reservieren und Buchausleihen verlängern, falls keine Reservierung vorliegt.

Akteure

1. potenzielle **menschliche Benutzer**
   - **Rollen** bei der Interaktion mit dem System identifizieren

     → eine Person – verschiedene Rollen

     → eine Rolle – verschiedene Personen(gruppen)
   - am Beispiel: „Mitglieder“ und „Mitarbeiter“ sind keine *Rollen*
   - *Rollen* über **Interaktionen** identifizieren

Unser durchgängiges Beispiel

Universitätsbibliotheks-Verwaltungssystem (UBVS)

Die UB hat Bücher und Zeitschriften im Bestand, von den Büchern teilweise mehrere Exemplare. Das System erlaubt den Mitgliedern der Bibliothek, ein Buch auszuleihen. Dabei darf jedes Mitglied höchstens 6 Bücher gleichzeitig ausleihen. Nur Mitarbeiter der Universität (Professoren, wissenschaftliche und technische Angestellte, keine Studenten) dürfen bis zu 12 Bücher gleichzeitig ausleihen. Mitarbeiter dürfen außerdem Zeitschriften ausleihen.

Neue Artefakte werden in den Katalog eingearbeitet, alte aus dem Bestand ausgesondert.

Das System registriert, an wen und wie lange Artefakte des Bestandes ausgeliehen sind. Für jedes Mitglied hält es die Information, wie viele und welche Artefakte es zur Zeit ausgeliehen hat. Das System erinnert automatisch an überfällige Ausleihen.

Mitglieder können mit Hilfe des Systems ferner den Bestand der UB einsehen („Browsing"), ausgeliehene Bücher reservieren und Buchausleihen verlängern, falls keine Reservierung vorliegt.

Akteure

1. potenzielle **menschliche Benutzer**
   - **Rollen** bei der Interaktion mit dem System identifizieren

     → eine Person – verschiedene Rollen

     → eine Rolle – verschiedene Personen(gruppen)
   - am Beispiel: „Mitglieder“ und „Mitarbeiter“ sind keine *Rollen*
   - *Rollen* über **Interaktionen** identifizieren

Book Borrower	Bücher ausleihen, reservieren, verlängern
Journal Borrower	Zeitschriften ausleihen
Browser	Katalog einsehen

   - *Erfinde keine Use Cases!!!*

Unser durchgängiges Beispiel

Universitätsbibliotheks-Verwaltungssystem (UBVS)

Die UB hat Bücher und Zeitschriften im Bestand, von den Büchern teilweise mehrere Exemplare. Das System erlaubt den Mitgliedern der Bibliothek, ein Buch auszuleihen. Dabei darf jedes Mitglied höchstens 6 Bücher gleichzeitig ausleihen. Nur Mitarbeiter der Universität (Professoren, wissenschaftliche und technische Angestellte, keine Studenten) dürfen bis zu 12 Bücher gleichzeitig ausleihen. Mitarbeiter dürfen außerdem Zeitschriften ausleihen.

Neue Artefakte werden in den Katalog eingearbeitet, alte aus dem Bestand ausgesondert.

Das System registriert, an wen und wie lange Artefakte des Bestandes ausgeliehen sind. Für jedes Mitglied hält es die Information, wie viele und welche Artefakte es zur Zeit ausgeliehen hat. Das System erinnert automatisch an überfällige Ausleihen.

Mitglieder können mit Hilfe des Systems ferner den Bestand der UB einsehen („Browsing"), ausgeliehene Bücher reservieren und Buchausleihen verlängern, falls keine Reservierung vorliegt.

Akteure

1. potenzielle **menschliche Benutzer**
   - **Rollen** bei der Interaktion mit dem System identifizieren

     → eine Person – verschiedene Rollen

     → eine Rolle – verschiedene Personen(gruppen)
   - am Beispiel: „Mitglieder“ und „Mitarbeiter“ sind keine *Rollen*
   - *Rollen* über **Interaktionen** identifizieren

Book Borrower	Bücher ausleihen, reservieren, verlängern
Journal Borrower	Zeitschriften ausleihen
Browser	Katalog einsehen
Librarian	Katalog aktualisieren

   - *Entdecke versteckte Rollen!!!*

Akteure (2)

2. nicht-menschliche Akteure
   - *andere* Systeme oder Systemkomponenten,
     die mit dem zu entwickelnden System interagieren
   - liefern Eingaben / nehmen Ausgaben entgegen
     und erfüllen eine für das Gesamtsystem relevante Rolle

eher:	Email-System zur Benachrichtigung über überfällige Ausleihen
eher nicht:	Keyboard eines Terminals, Drucker, Massenspeicher

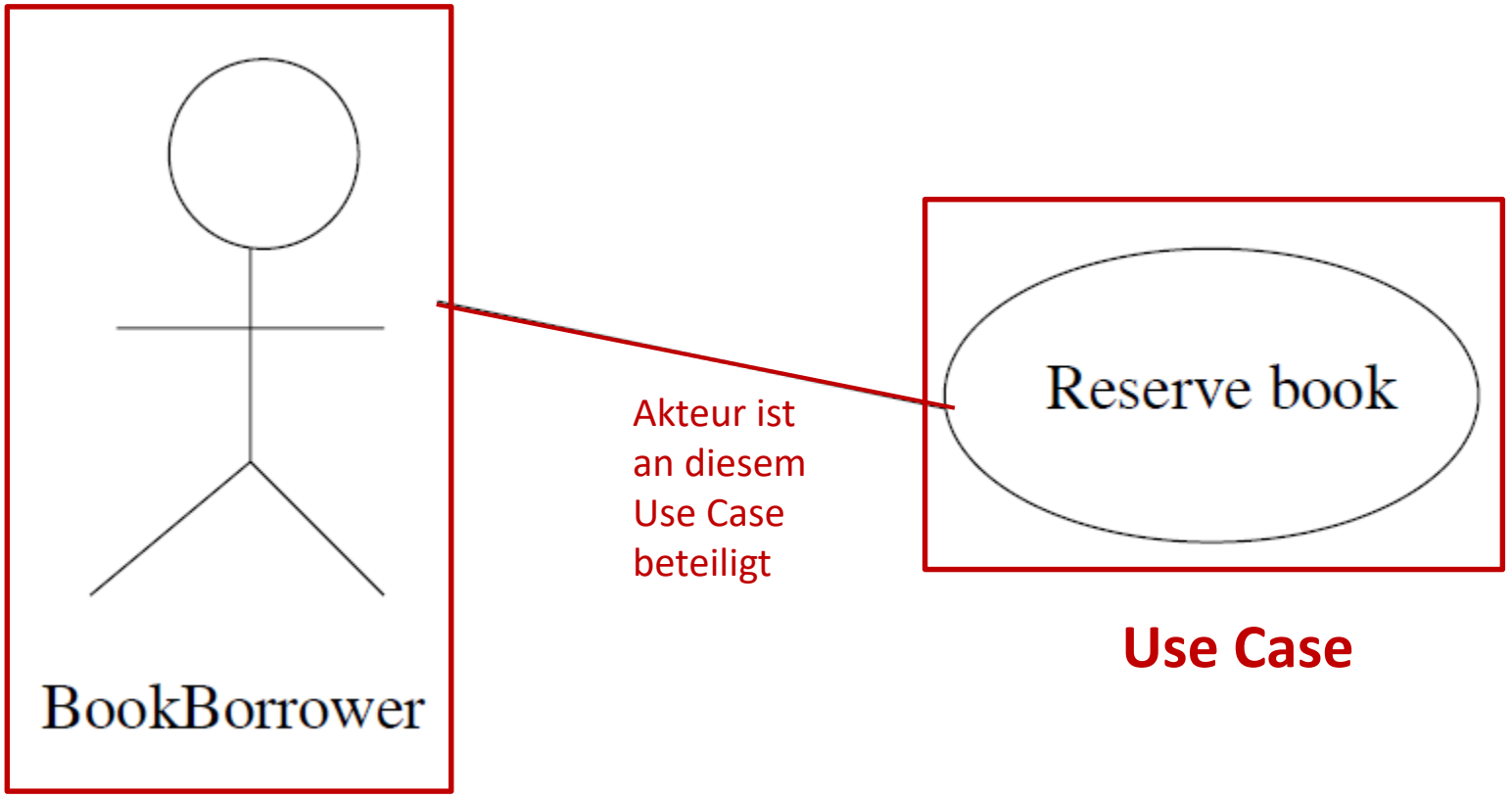
Use Cases

- Mengen von verwandten Szenarien
- Systemaufgaben / Interaktionen des Systems mit Akteuren

Book Borrower	Bücher ausleihen, reservieren, verlängern
Journal Borrower	Zeitschriften ausleihen
Browser	Katalog einsehen
Librarian	Katalog aktualisieren

- Use Cases und Akteure iterativ erschließen

Use Case-Diagramm



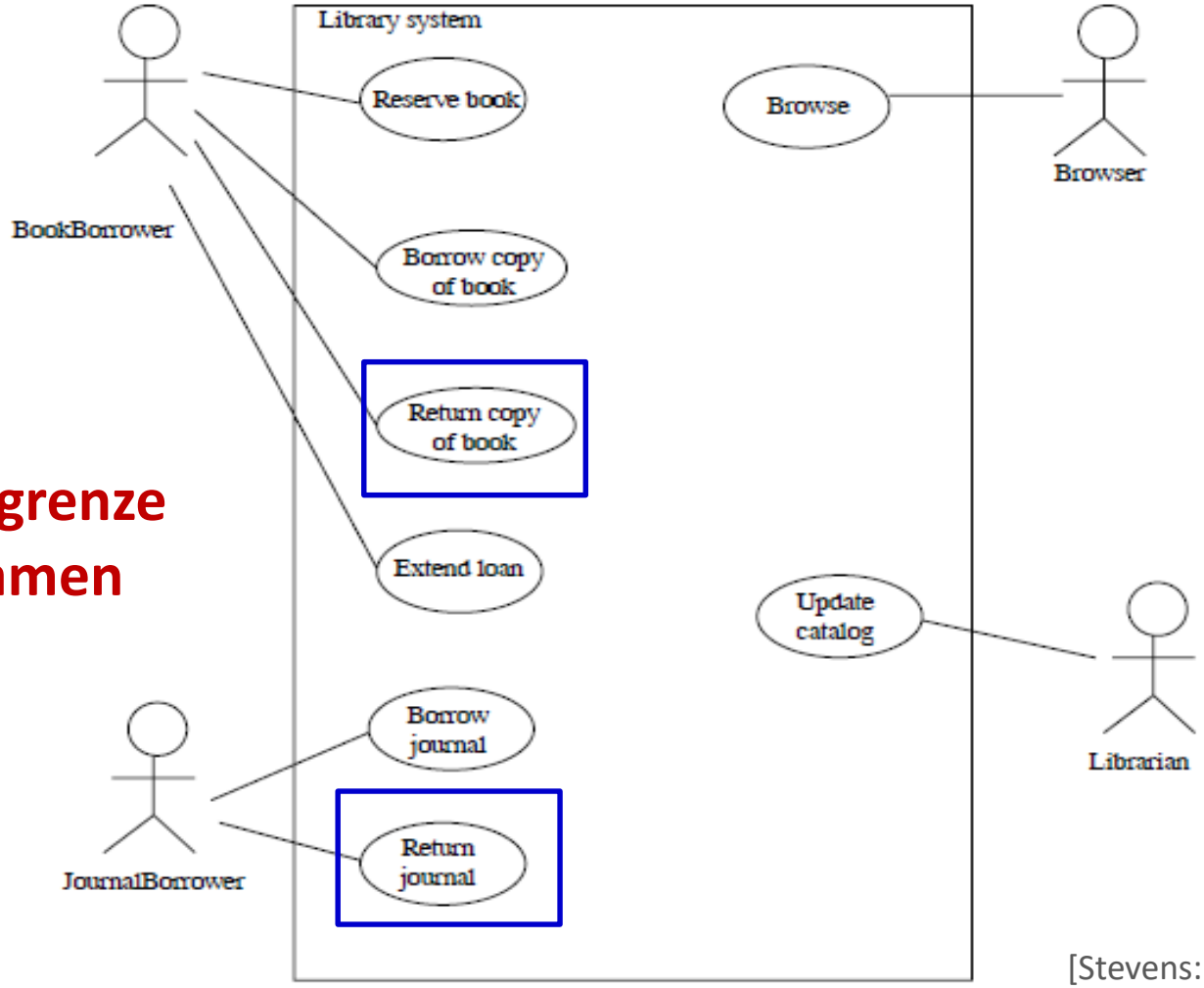
Akteur

Use Case

Keine zeitlichen Abläufe in Use Case-Diagrammen!

[Stevens: Using UML]

Use Case-Diagramm



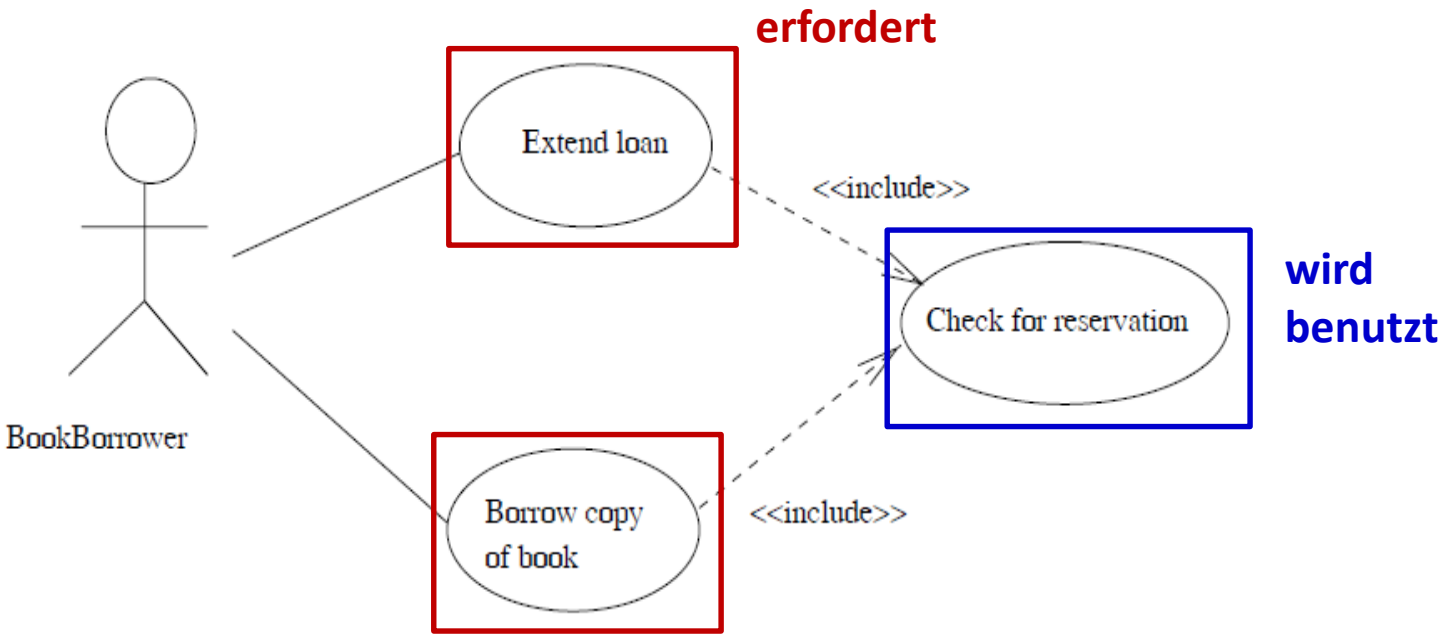
mit Systemgrenze und -namen

[Stevens: Using UML]

Abhängigkeiten von Use Cases

- Verfeinerung der Use Case-Diagramme in späteren Iterationen
  - weitere Use Cases
  - Abhängigkeiten

1. <<include>>

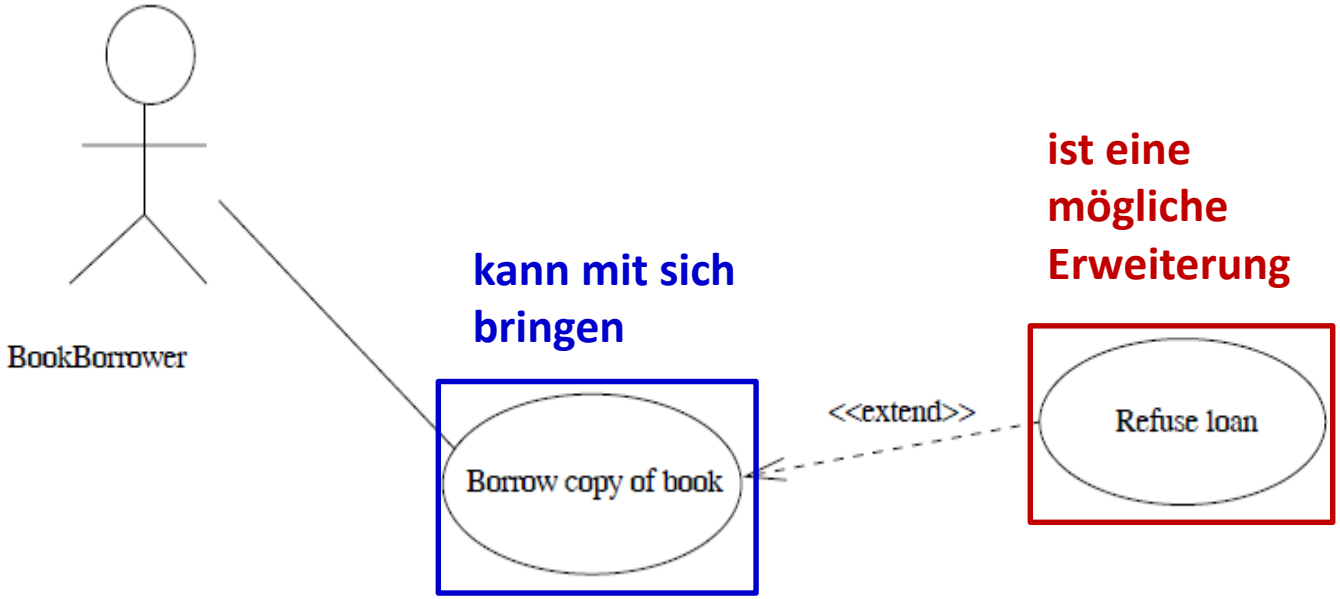


[Stevens: Using UML]

Abhängigkeiten von Use Cases (2)

- Verfeinerung der Use Case-Diagramme in späteren Iterationen
  - weitere Use Cases
  - Abhängigkeiten

2. <<extend>>

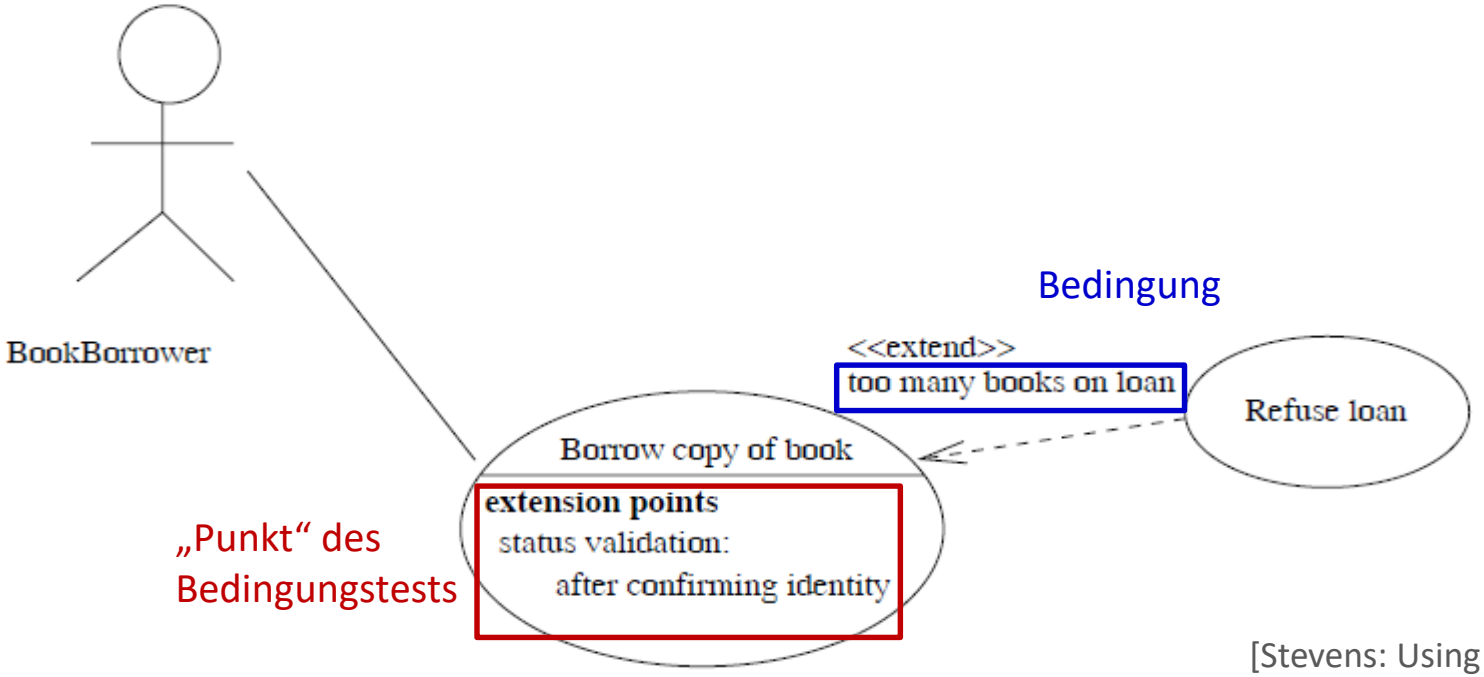


[Stevens: Using UML]

Abhängigkeiten von Use Cases (3)

- Verfeinerung der Use Case-Diagramme in späteren Iterationen
  - weitere Use Cases
  - Abhängigkeiten

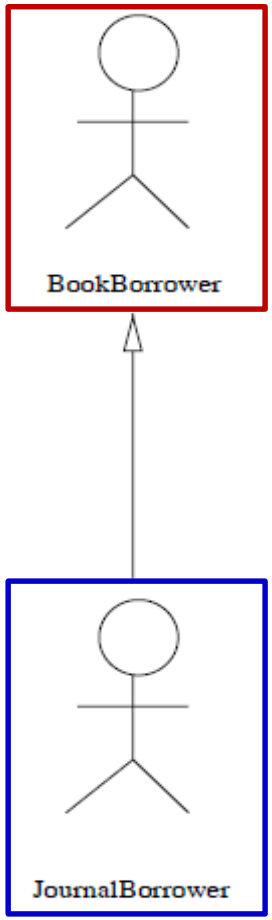
2. <<extend>> - Bedingungen sichtbar machen



[Stevens: Using UML]

Abhängigkeiten zwischen Akteuren

- Verfeinerung der Use Case-Diagramme in späteren Iterationen
  - weitere Akteure
  - Abhängigkeiten: **Generalisierung**



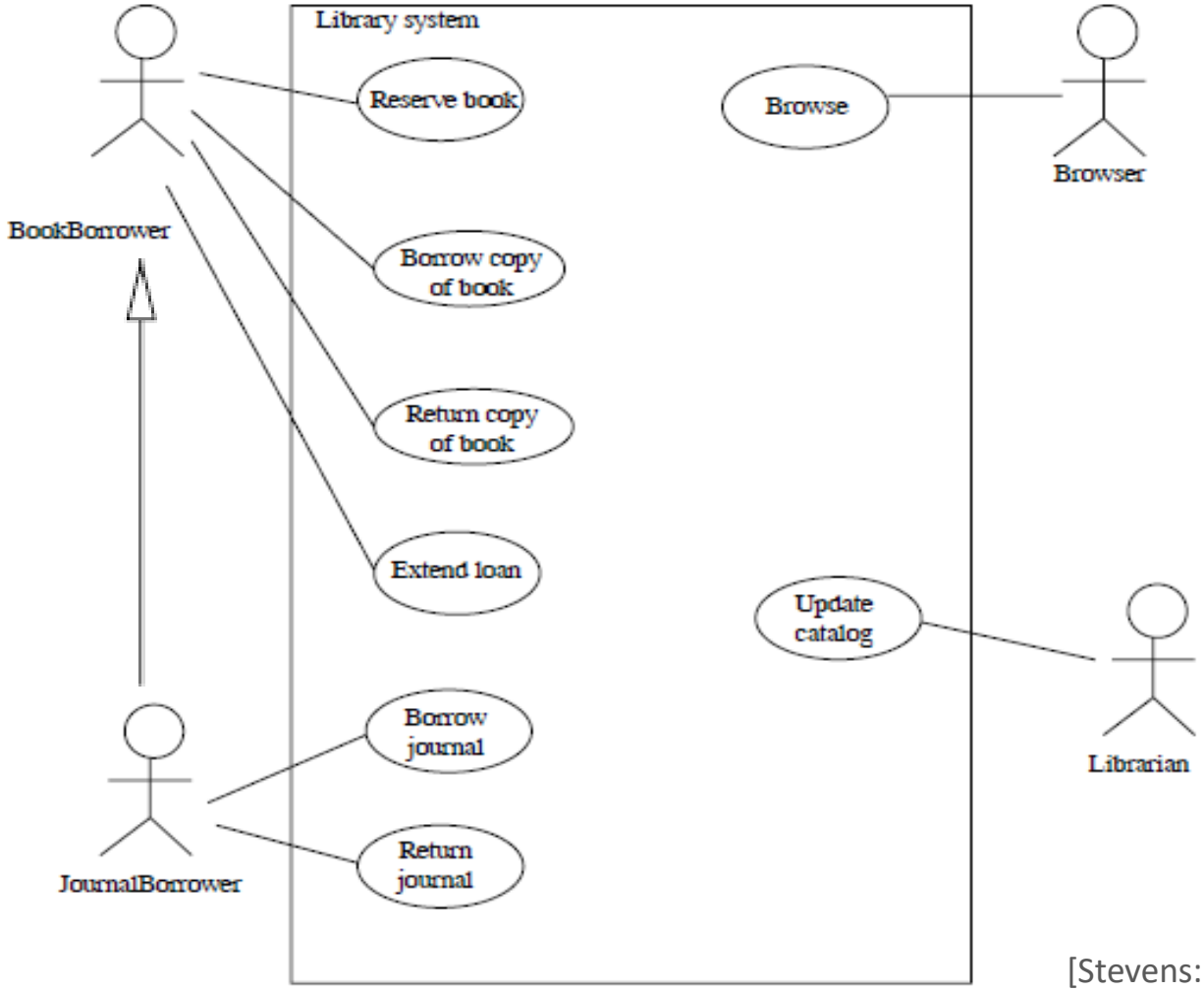
allgemeinerer Akteur (Oberbegriff)

speziellerer Akteur (*mögliche* Spezialisierung)

`JournalBorrower` kann alle Use Cases von `BookBorrower` initiieren

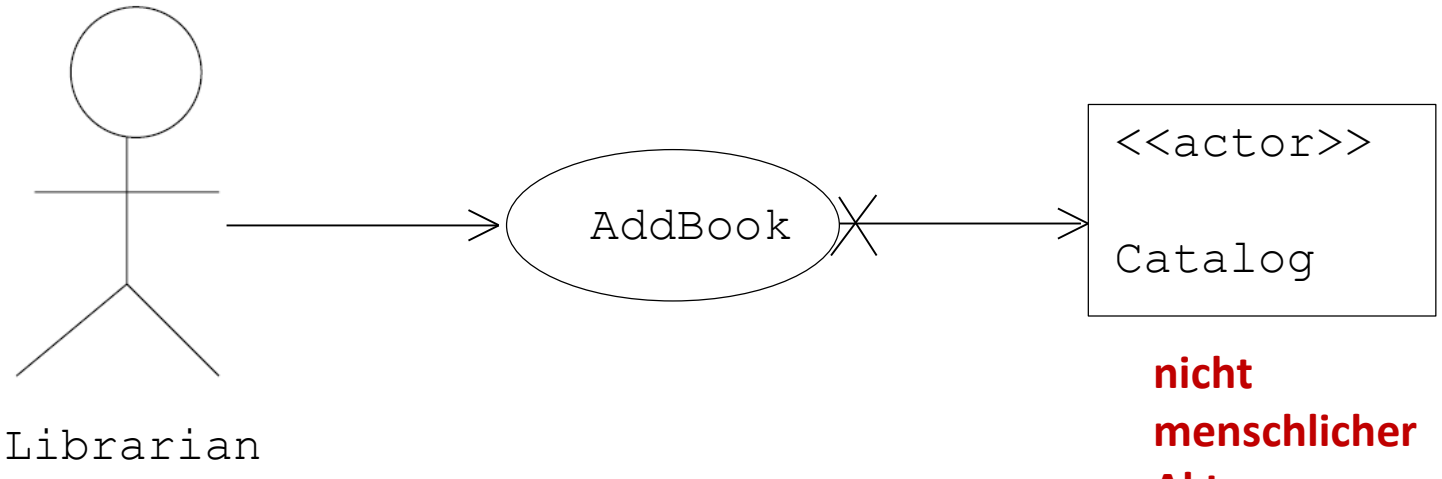
[Stevens: Using UML]

Use Case-Diagramm mit Generalisierung



[Stevens: Using UML]

Navigierbarkeit



nicht menschlicher Akteur

Wofür Use Cases?

Use Cases helfen

- Planungsentscheidungen zu treffen (Kosten, Vorgehen, Risiken, ...)
- Anforderungen zu organisieren und klassifizieren
  → Dekomposition, Abhängigkeiten, ...
- Konflikte zu erkennen und Prioritäten zu setzen
- inkrementell vorzugehen (sinnvolles Prototyping)
  - prioritäre Use Cases zuerst
  - frühe (gröbere aber konsistente) Use Case-Diagramme zuerst
- die Systemanforderung zu validieren
- das System zu validieren
  - Entwicklung von Testfällen

Use Cases sind oft Kernstück der Spezifikation und Ausgangspunkt für die Modellierung.

Dokumentation von Use Cases

- „Notes“ im Use Case-Diagramm
- Beschreibung in Textform außerhalb des Diagramms
  - unstrukturiert
  - strukturiert (→ Templates)
  - andere Diagrammarten

→ **Ablaufdiagramme** zum Darstellen der Szenarien

Use Case ID:	UBVS/xxx.yyy		
Use Case Name:	Borrow Copy of Book		
Autor:	Hans Modellierer	**Datum:**	04.11.2010

Akteure:	BookBorrower
Beschreibung:	Ein Bibliotheksmitglied hat die Möglichkeit, ein Exemplar eines Buches auszuleihen. Dazu muss das Exemplar zu diesem Zeitpunkt zur Verfügung stehen. Ferner darf das Mitglied noch nicht seine maximale Anzahl gleichzeitig ausleihbarer Artefakte ausgeliehen haben.
Trigger:	Das Mitglied legt ein Exemplar zur Ausleihe vor.
Vorbedingung: **Nachbedingung:**	*wie im Beispiel zur „User Story“ (s. Folie 13)*
Normal Flow:	Ein Mitglied legt ein ausleihbares Exemplar vor und hat noch nicht sein Maximum an gleichzeitig ausleihbaren Artefakten erreicht. Das System registriert die Ausleihe: ...
Alternative Flow:	keiner
Exception:	Das Mitglied hat sein Maximum an gleichzeitig ausleihbaren Artefakten erreicht. Die Ausleihe wird zurückgewiesen.

Referenzliste

- Sommerville, I.: *Software Engineering 9.* Pearson, Bosten 2011. (Chapter 4)

- Helmut Balzert. Lehrbuch der Software-Technik (Band 1): Software-Entwicklung. Spektrum Akademischer Verlag Heidelberg 1996.

- Stevens, P.: *Using UML. Software Engineering with Objects and Components.* Addison-Wesley, Pearson Education, Harlow, 2006.