Software Engineering

Lesson 02
Business Processes / Use Cases
v1.0d

Uwe Gühl

Fall 2007/ 2008

Contents

Requirements analysis
Differentiation of requirements

- Required Functionality
  - What provides the system?
- Required Constraints (Non-functional requirements)
  - General quality requirements
    - Performance (response time of the system?)
    - Reliability (which downtime is allowed?)
    - Testability (how is it possible to prove functionality with Test Cases for acceptance?)

Requirements analysis Differentiation of requirements

- Required Constraints
(Non-functional requirements)
  - Special quality requirements
    - System platform
    - Systems to connect to
    - Interfaces
    - Legal requirements (data protection, accountancy rules)

Requirements analysis Differentiation of requirements

- Required Constraints (Non-functional requirements)
  - Operational quality requirements
    - Easy installation
    - Low operability costs
    - Simple administration
    - highly customisable
    - Monitoring
    - Easy to update

Often underestimated and too late considered

Requirements analysis Importance

- Why is requirements analysis important?
  - Software is meanwhile critical for success in business
  - If an IT project is not successful, a crisis of the company is possible
  - Quality of requirements is necessary, so that there will be no gap between definition and realization of the software
- Too few experts master subject area, analysis methods, and IT at the same time.
- That's why the requirements analysis is often the least element in the chain

Requirements analysis Importance

- Typical problems: Requirements are
  - not structured
  - not complete
  - inconsistent
  - Imprecise

  ...especially in the beginning of a project
- Causes: Different interests of different project members, language, politics, missing professional know-how
- Requirements analysis helps

Requirements analyst

- A requirements analyst is between users and developers of a software system and should describe the functional requirements and the technical specification
- Why is a requirements analyst needed?
  - Users are overstrained by the complexity of a system
  - Users as members in the project team have to do their regular work as well
  - Users do not see the big picture
  - Human weakness and misunderstanding in language
  - Thinking in processes is important, but often missed

Requirements analyst

- The ideal requirements analyst
  - moderates with
    - customer
    - potential users of the specialist division and
    - developers
  - obtains trust of the users, is doing interviews and incorporates into the specialist area
  - collects and structures the information methodically, and describes the requirements
  - masters the specialist area, analysis, and IT

Requirements analyst

- In general: What helps?
  - Concrete measurable acceptance criteria:
    Not: The system must be fast
    But: The system has to fulfil a general request in 5 seconds, for this specific action the system has to respond in 10 seconds: a) ... b) ... c) ...
  - Idea: avoids different interpretations of requirements
  - Acceptance criteria are basic for testing

Business Processes

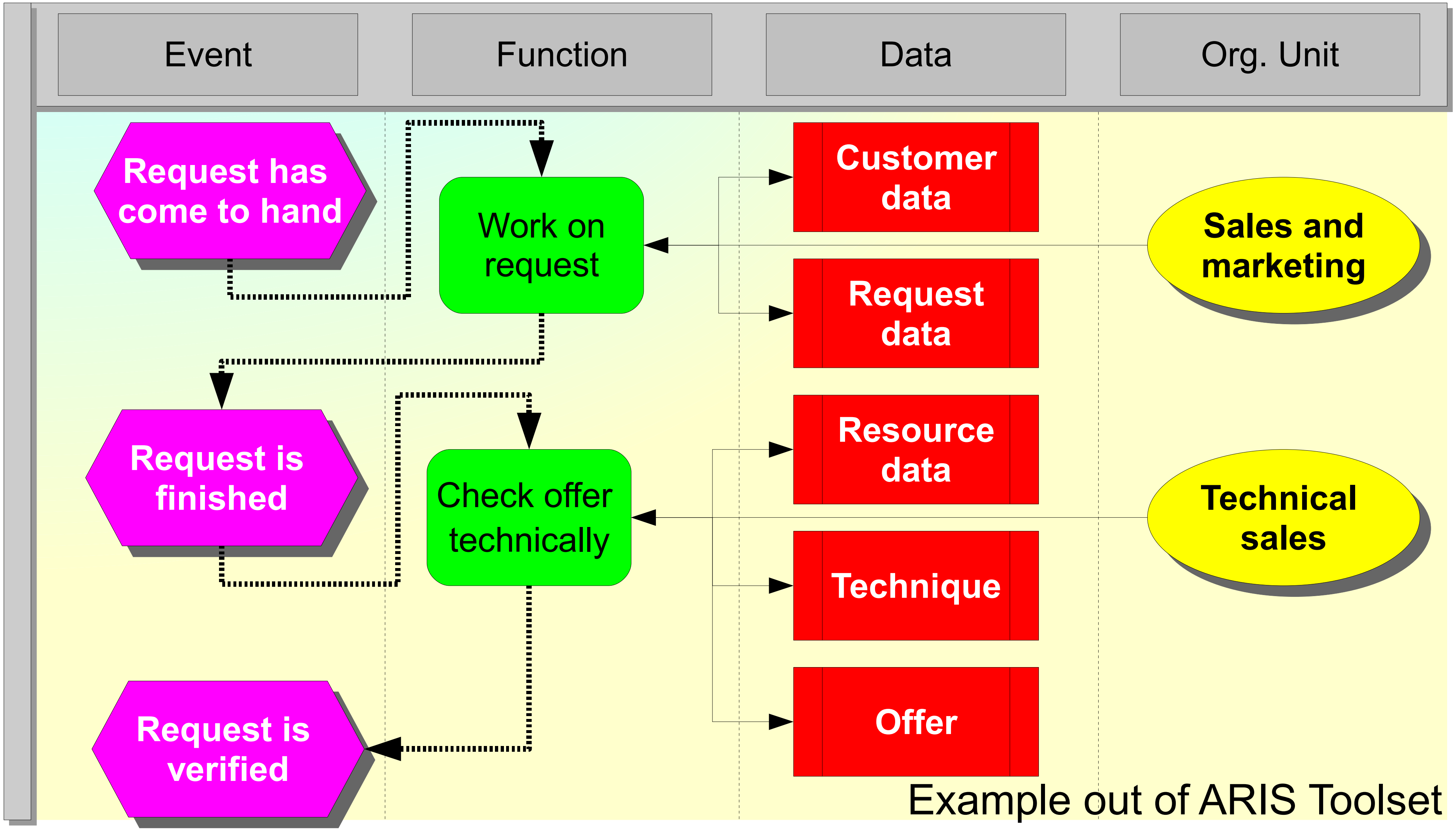
- Business processes show complete activities which are relevant to run the business
  - Specific agents are involved
  - From start until the end
  - Including all activities, independent if they are part of the system to be realized or not
- To set the boundaries of a system do
  - Business Process Analysis – to identify the current activities
  - Business Process Modelling – to define new activities
  - Business Process Redesign – to update current activities

Business Processes

- Business Process Modelling with Event Driven Process Chains (*German: Ereignisgesteuerte Prozessketten (EPK))*
- ARIS Toolset from IDS Scheer, notations are the EPK
- Modelling elements of EPK for Business Processes are
  - Functions (*German: Funktionen*) as relevant activities for business
  - Events (*German: Ereignisse*) as trigger and results of functions
  - Organisational units (*German: Organisationseinheiten*) are taking part in functions
  - Data (*German: Daten*)

Business Processes

- Business Process Modelling with Event Driven Process Chains

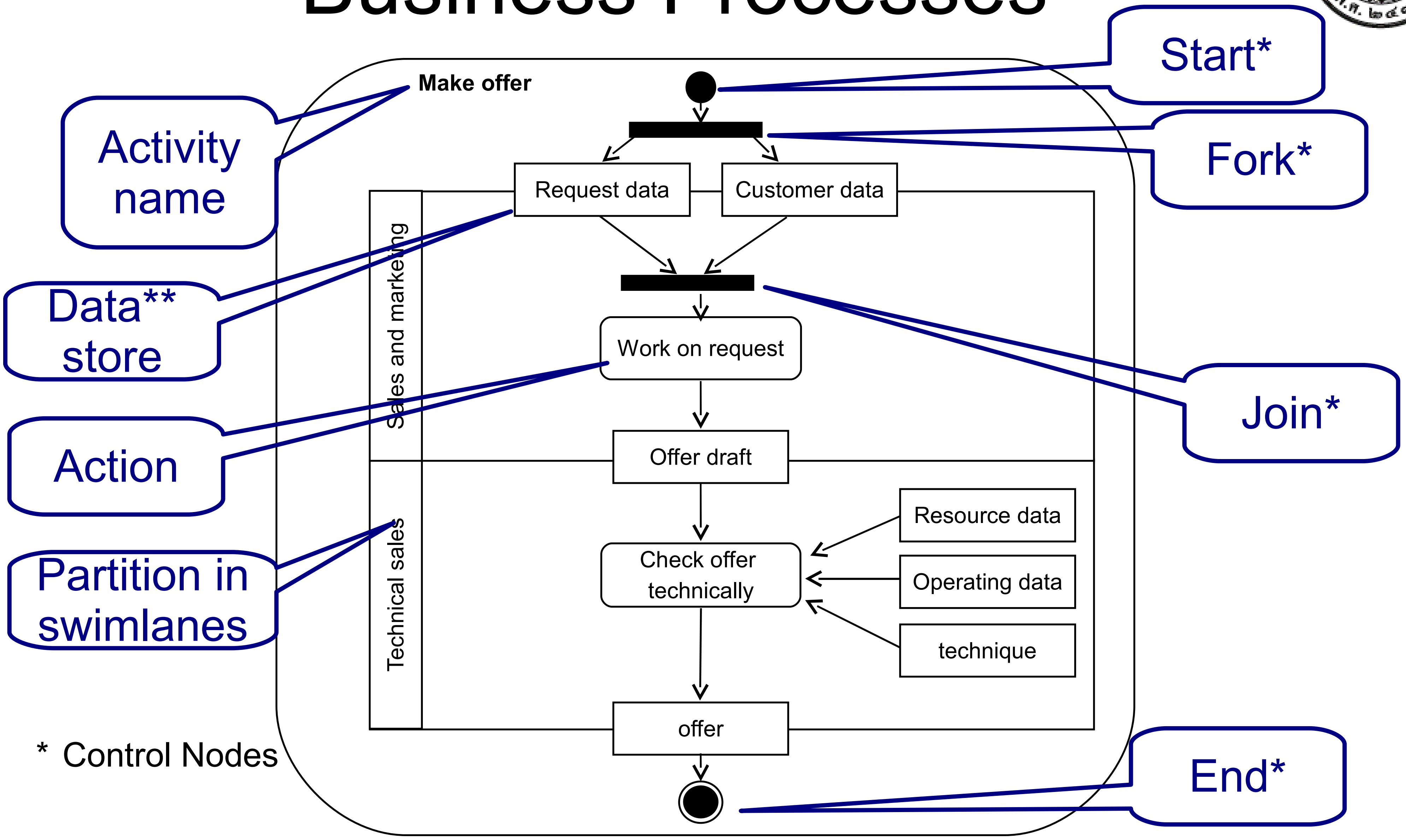


Example out of ARIS Toolset

Business Processes

- Business Process Modelling with UML Activity Diagrams
  - Activity diagrams describe possible activities of a system
  - An activity is a defined step in a processing procedure, it contents
    - an internal action and a
    - at least one resulting transition

Business Processes



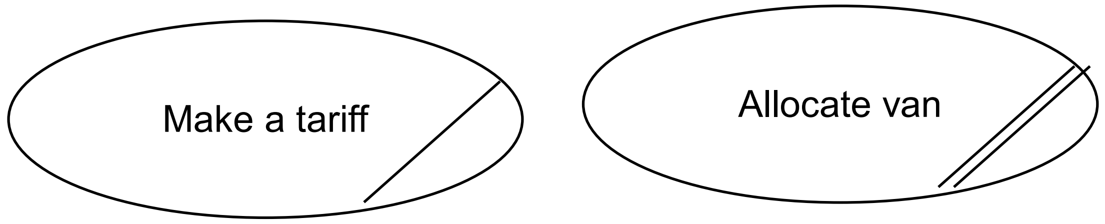
* Control Nodes

** Object Nodes

Example using an activity diagram

Business Processes

- Business Process Modelling with extension of the UML with Business Use Cases [Oes06]
  - A Business Use Case describes a commercial activity to achieve directly or indirectly business value
  - Business Use Cases could help in Business Process Modelling, which should be done before software development
  - Business Use Cases are not necessary to develop software



Notation for Business Use Cases

(on the right Core Business Use Case)

Use Cases

- Overview
  - First descriptions of the functional behaviour of a system from the outside started already end of 1970s and beginning of 1980s
  - Use Cases help to structure functional requirements
  - Use Cases usually find their origin in Business Processes, and after the decision, which activities get part of the system and which not
  - Use Cases were described first in the book of Ivar Jacobson: „OOSE - A Use Case driven approach“ [JCJO92]

Use Cases

- Goals
  - Use Cases should help to collect functional requirements in a structured way
  - Use Cases must be understood by the subject area specialists and the developers
  - Motto: „What“ instead of „How“
    - Example: “Display Web-Site”
      A Use Case does not show the classes and operations involved

Use Cases

- Links
  - [Coc] http://alistair.cockburn.us/index.php/Resources_for_writing_use_cases
  - [Pol] http://www.pols.co.uk/use-case-zone contents Use Case papers and links

Use Cases
Terms

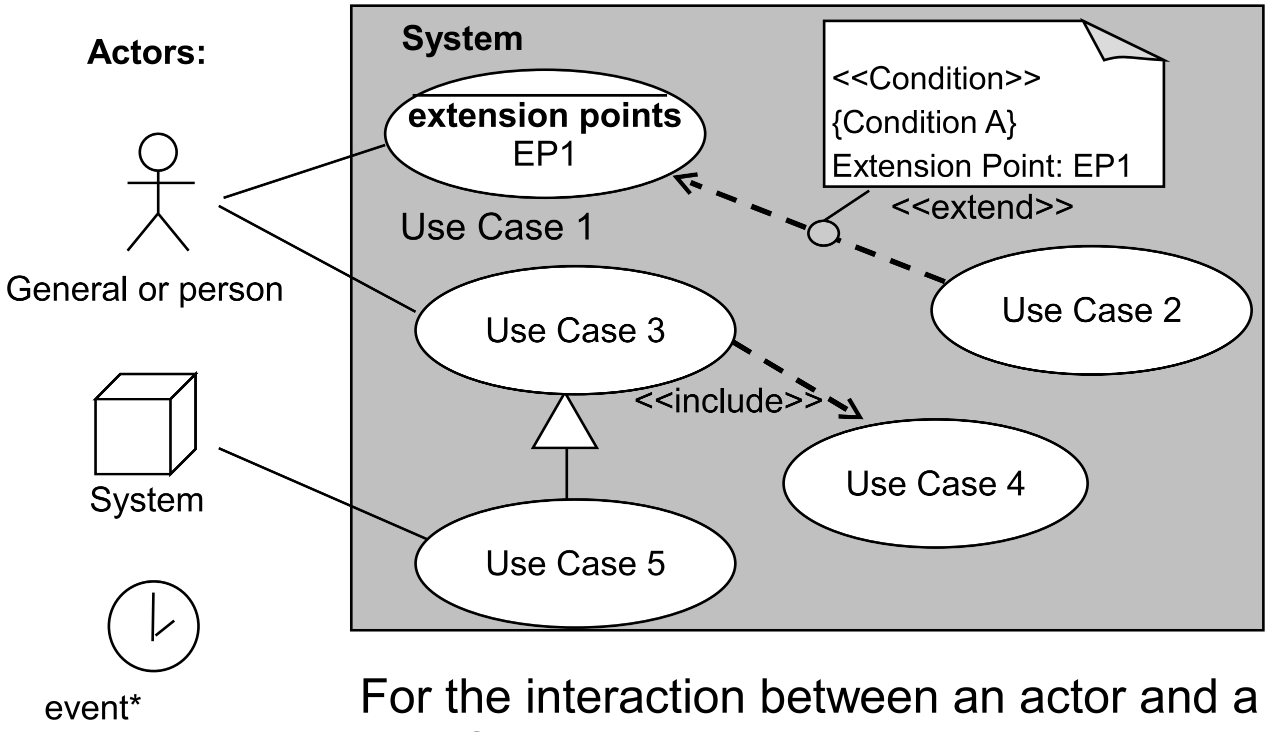
- Use Case Diagram
  - shows relations between Actors and Use Cases
  - shows relations between Use Cases
- Actor – acts with the system but is not part of it
  - User
  - Another system
  - Event
- Use Case – models continuous activities of an Actor with a system to get a professional added value

Use Cases
Terms

- Use Case Description – describes in text typical activities and alternatives as an exception
- Use Case Model – contents
  - one or more Use Case Diagrams (overview)
  - Use Case Description (detail)

Use Cases
Use Case diagrams

- Basic elements



For the interaction between an actor and a Use Case one uses a line.
An additional arrow could show, who is starting the interaction.

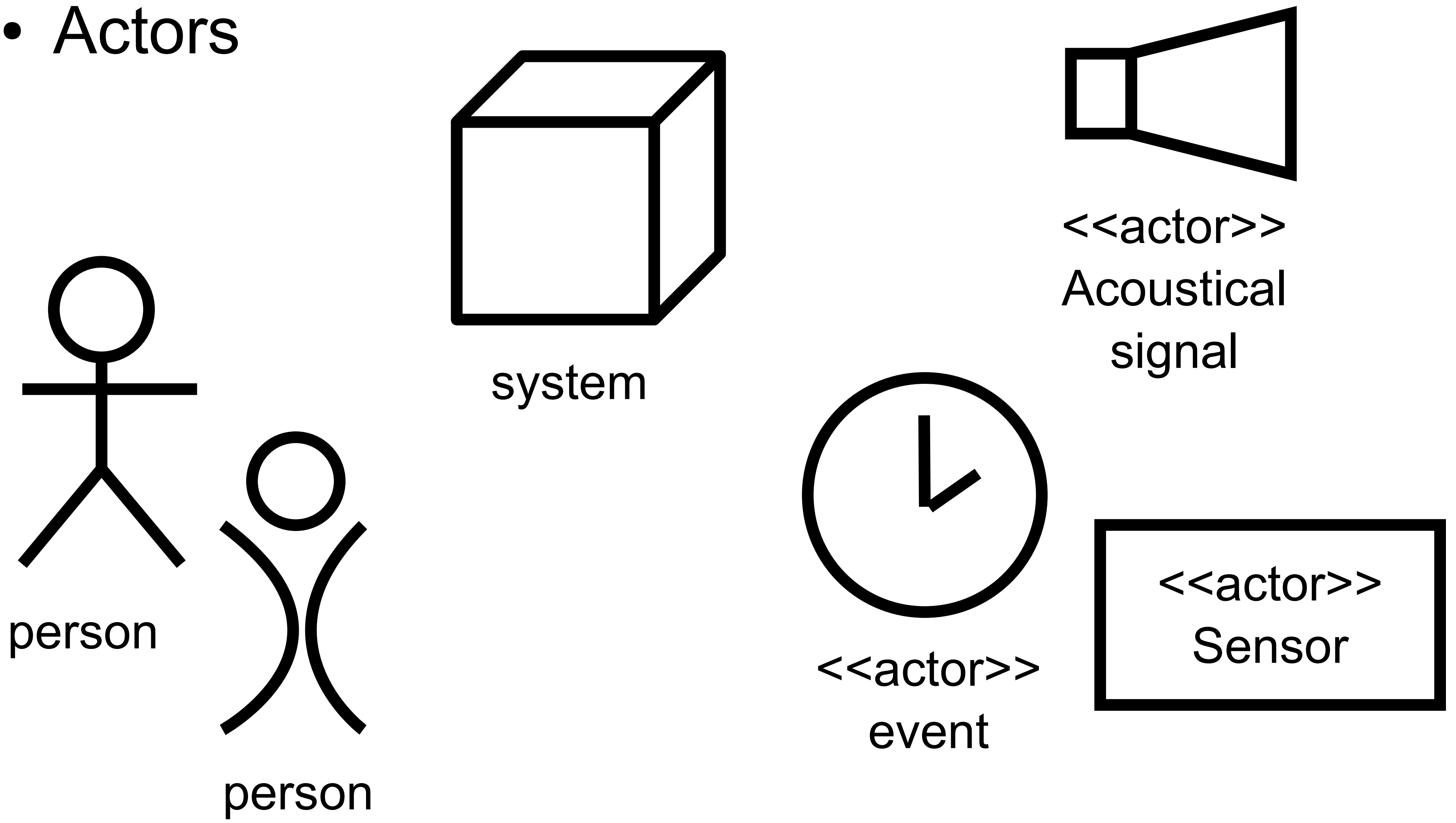
* Attention: Events are more problematical: For whom has an event a professional added value?

Use Cases
Use Case diagrams

- Actors
  - are typical stick-figures
    But you may use own symbols
  - could be signed optional with the stereotype <<actor>>
  - are always out of the system

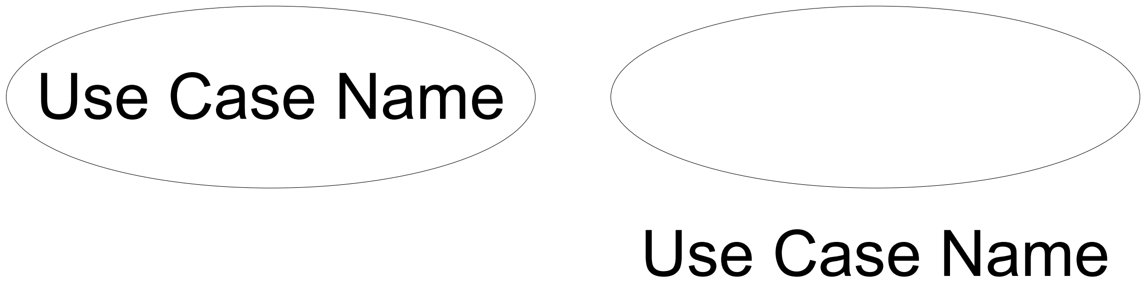
Use Cases
Use Case diagrams

- Actors

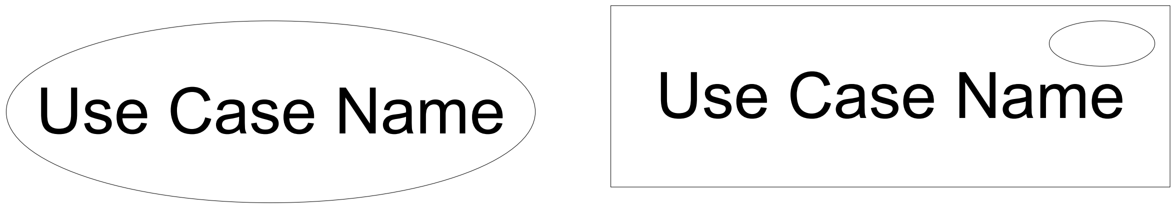


Use Cases
Use Case diagrams

- Use Case
  - is described with an ellipse, the name is inside or outside



  - could be drawn alternatively as a rectangle with a small ellipse in the right top corner

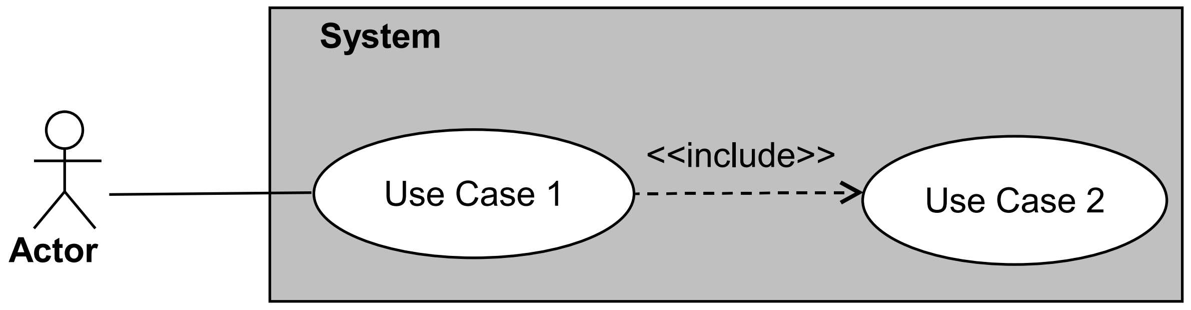


Use Cases
Use Case diagrams

- Relationships
  - An association is demonstrated by a line
  - Optional an arrow shows who is starting the interaction
  - Distinguish
    - <<include>>-Relationship
    - <<extend >>-Relationship
    - Specialization / Generalization
  - Relationships could help but are not necessary Don't do a too detailed decomposition

Use Cases
Use Case diagrams

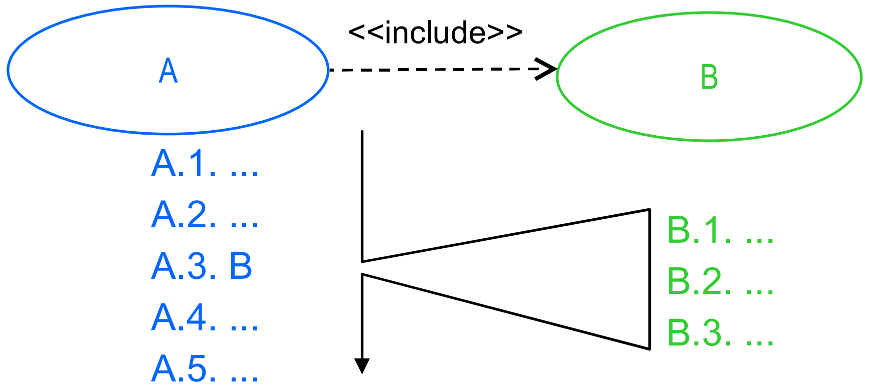
- <<include>>-Relationship
  A Use Case utilizes another Use Case
  - Why? To avoid redundancy
  - Cyclic dependencies are not allowed



Use Case 1 needs obligatory Use Case 2

Use Cases
Use Case diagrams

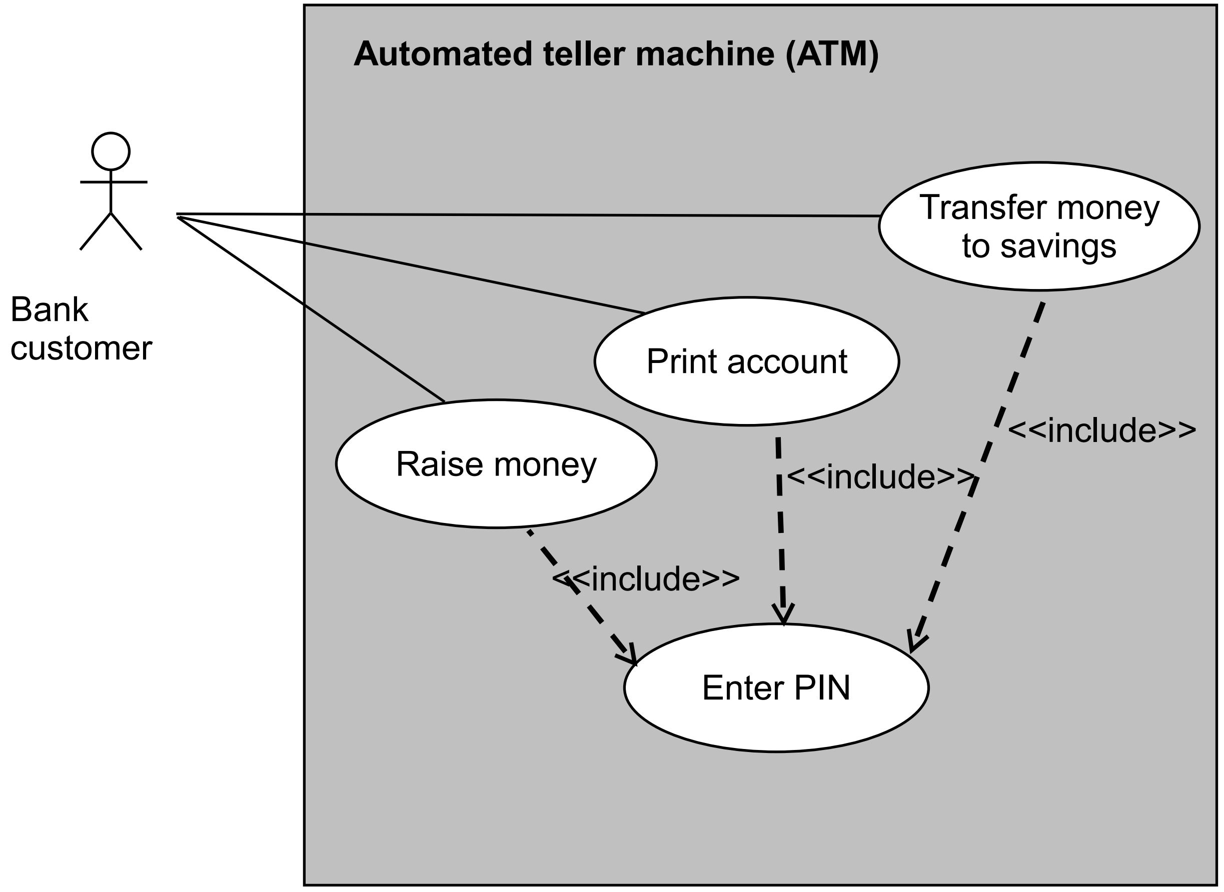
- <<include>>-Relationship – example of activities



Activities of Use Case A:
A.1
A.2
B.1.
B.2.
B.3
A.4
A.5

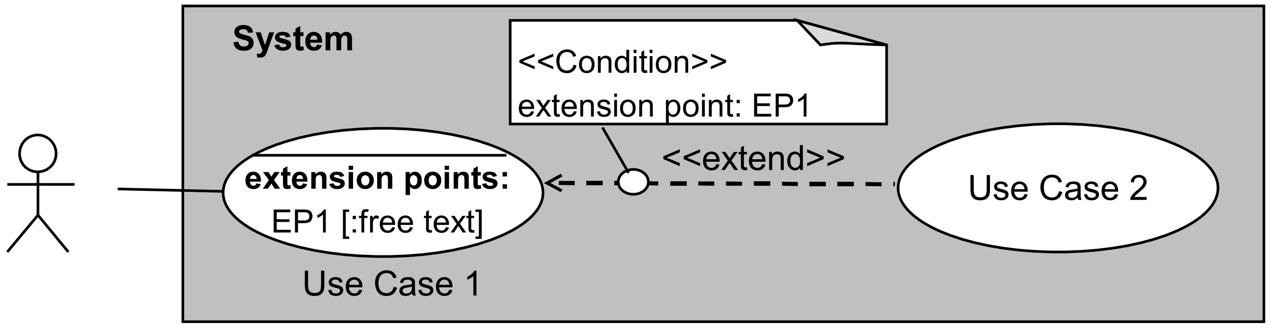
Use Cases
Use Case diagrams

- <<include>>-Relationship – example



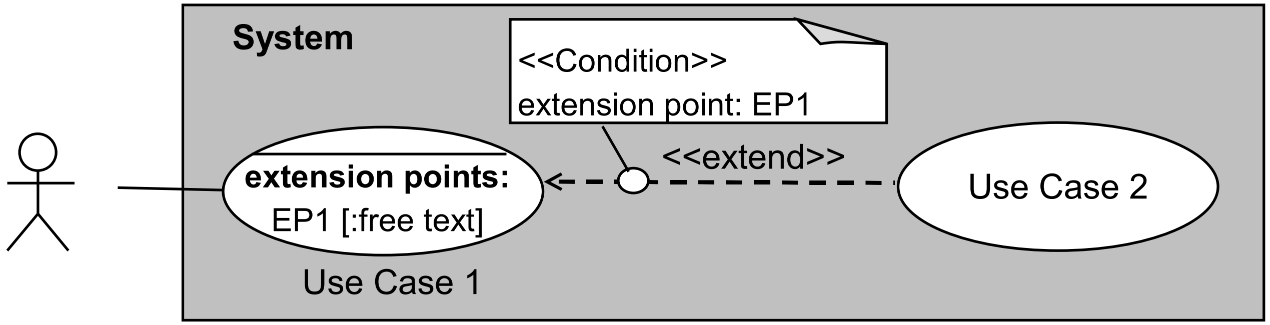
Use Cases
Use Case diagrams

- <<extend >>-Relationship
  A Use Case is extended optional through another Use Case if a specific condition matches
  - to model optional behaviour
  - Declaration of condition is always necessary



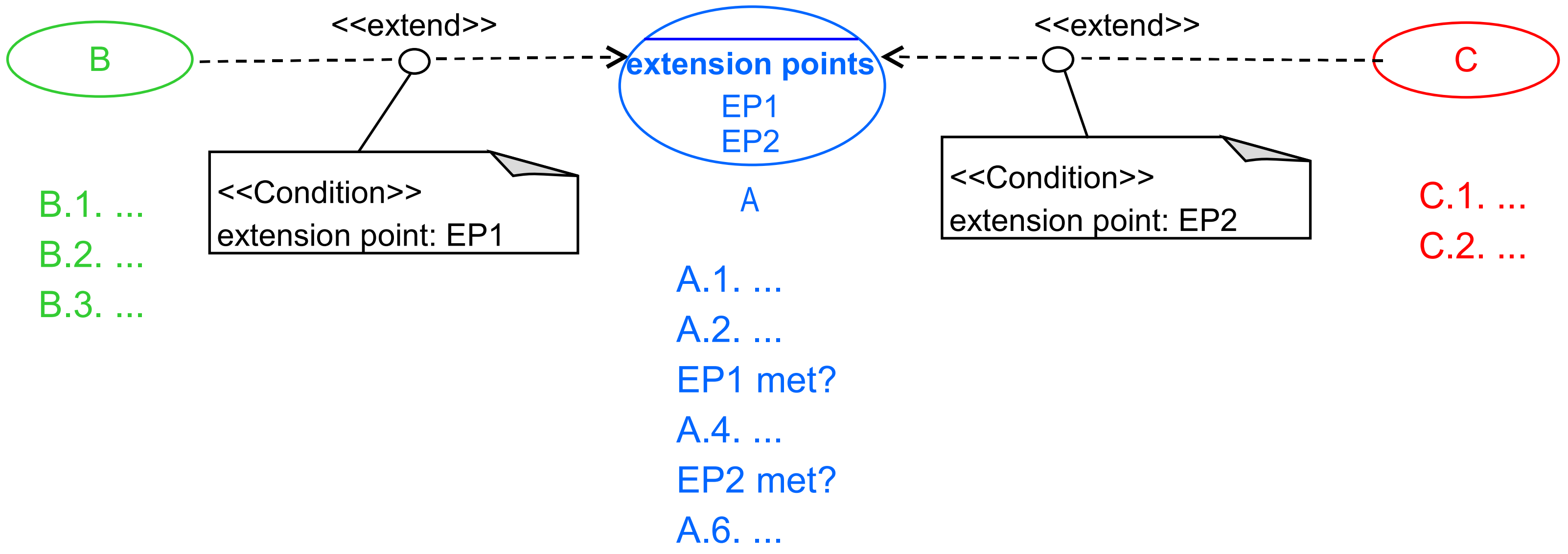
Use Cases
Use Case diagrams

- <<extend >>-Relationship
  The <<extend >> stereotype describes the <<extend >>-Relationship
- Additionally there are extension points, showing the possible extensions. The condition could be defined as comment.



Use Cases
Use Case diagrams

- <<extend >>-Relationship

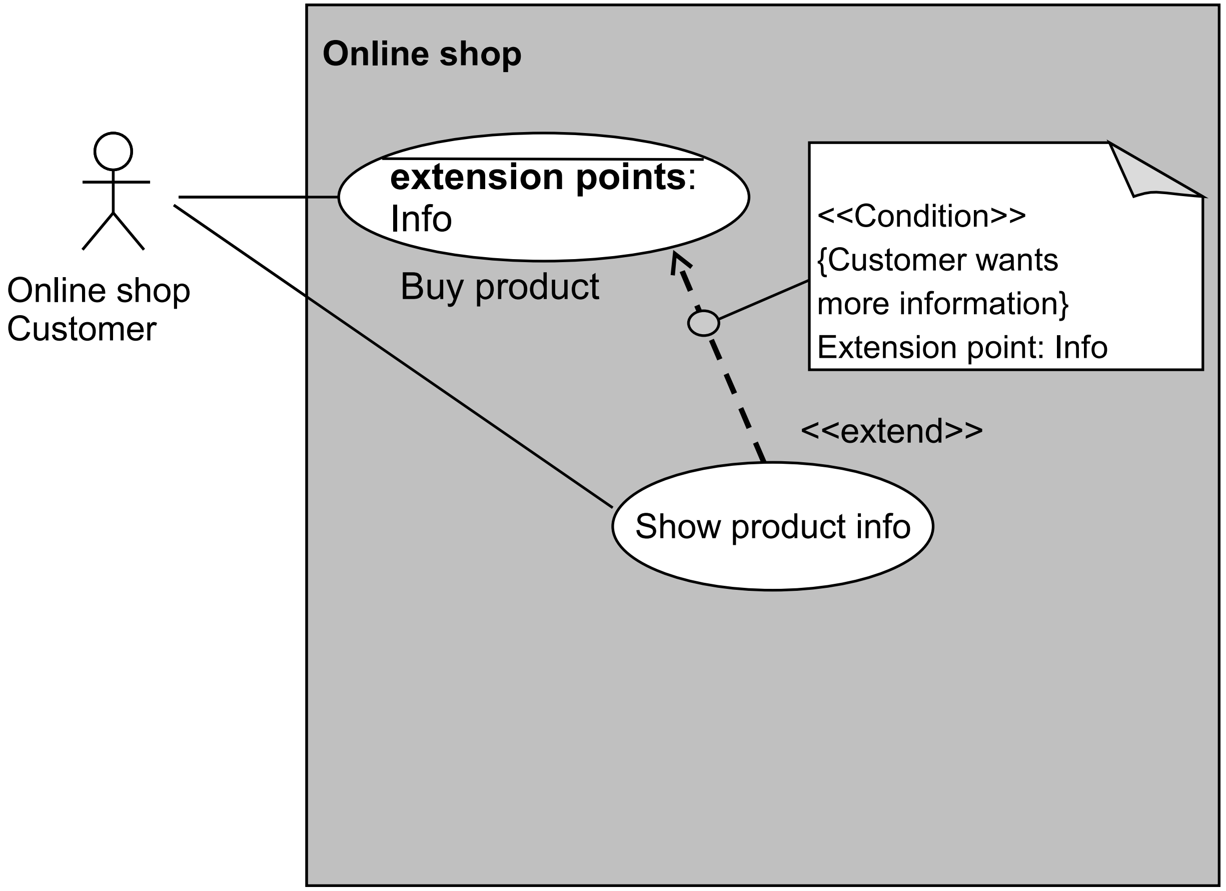


Activities of Use Case A could be:

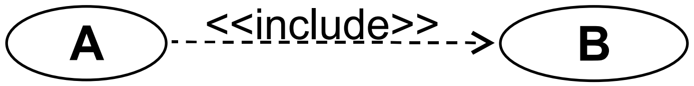
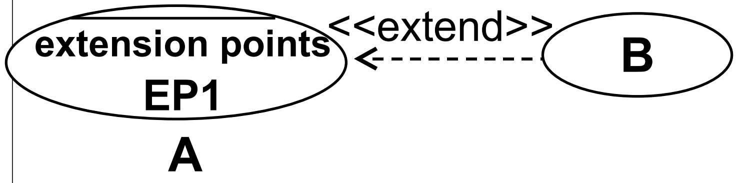
- *Alternative A*: Preconditions of EP1 and EP2 are met:
  A.1, A.2, B.1, B.2, B.3, A.4, C.1, C.2, A.6
- *Alternative B*: Precondition of EP1 is met but not of EP2:
  A.1, A.2, B.1, B.2, B.3, A.4, A.6
- etc.

Use Cases
Use Case diagrams

- <<extend >>-Relationship – example



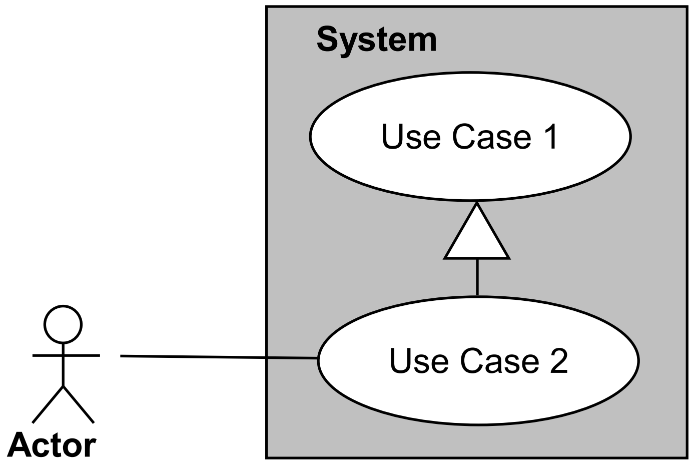
Use Cases
Use Case diagrams

	<<include>> Relationship	**<<extend>> Relationship**
		
Meaning	Activity of A always includes activity of B	Activity of A could, but does not have to be extended by activity B
When to use the relationship?	Activity of B could be used in various Use Cases. Hierarchical, functional decomposition	A has regular behavior but additional special cases
Meaning for Modelling	A is mostly incomplete and only the inclusion of B makes it complete. Typically B is artificial to avoid redundancy.	A is mostly complete and may be extended by B. B is typically complete as well.
Dependencies	A has to consider B in modelling. B gets modelled independent from A, so it could be used by other Use Cases as well. B itself has not to be complete.	A has to be prepared for extension with B in indicating extension points. B gets modelled complete and independent from A.

Source [JRH04]

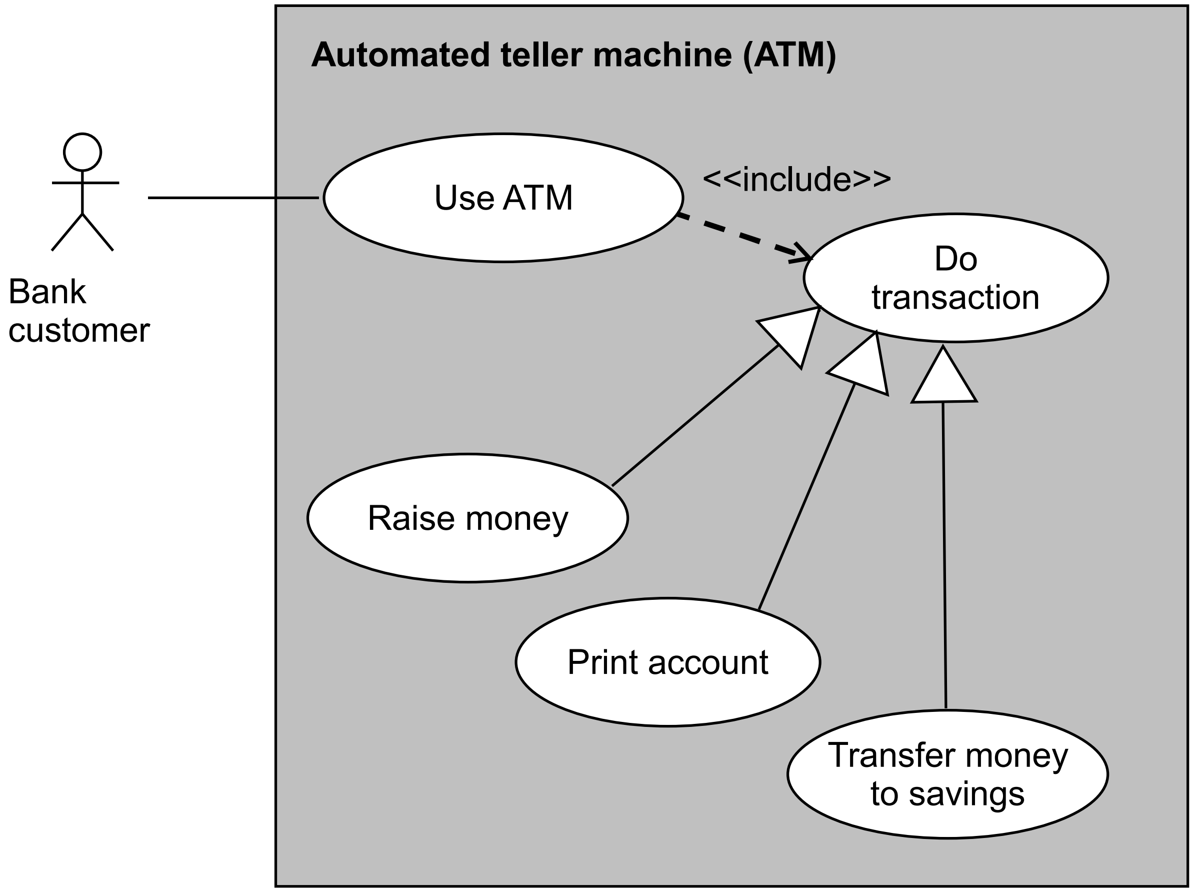
Use Cases
Use Case diagrams

- Specialization / Generalization
  - Derivation of a Use Case from another Use Case
    - Inheritance of behaviour and meaning



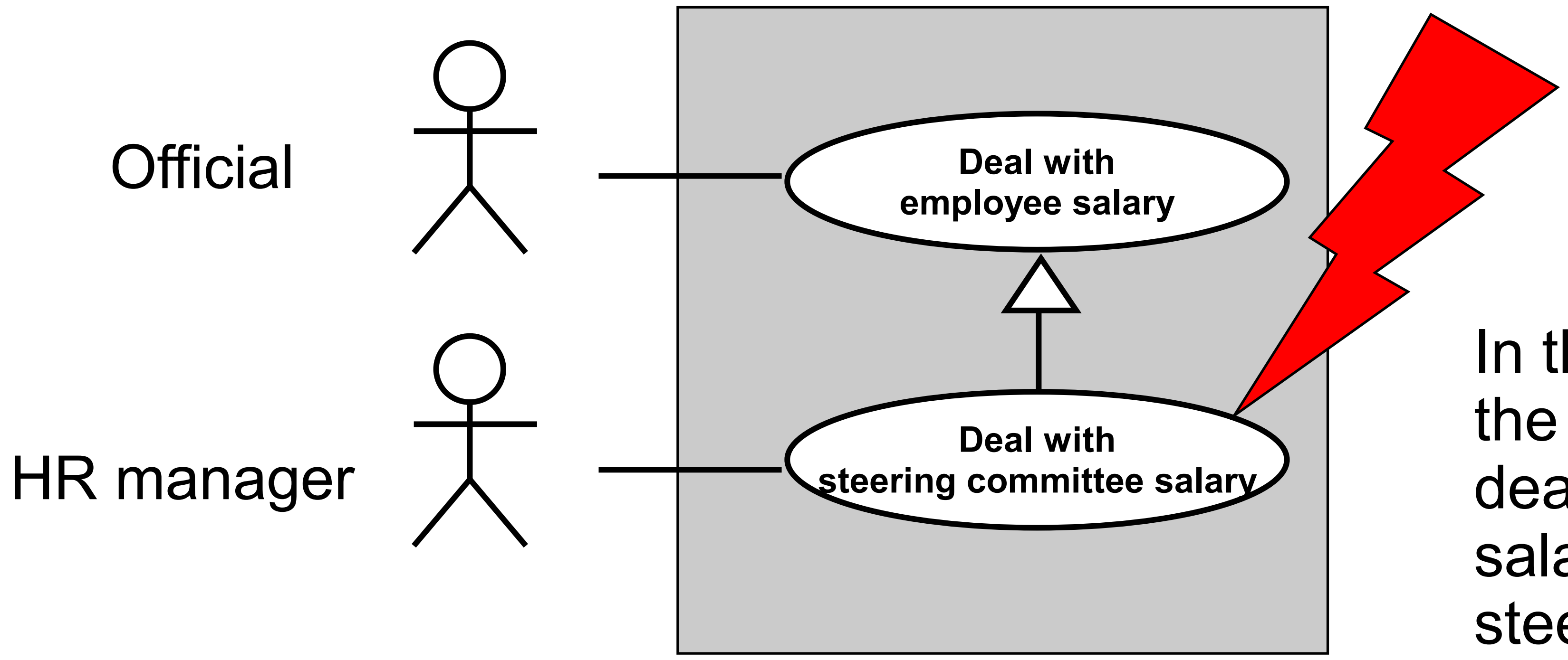
Use Cases
Use Case diagrams

- Specialization / Generalization – example



Use Cases
Use Case diagrams

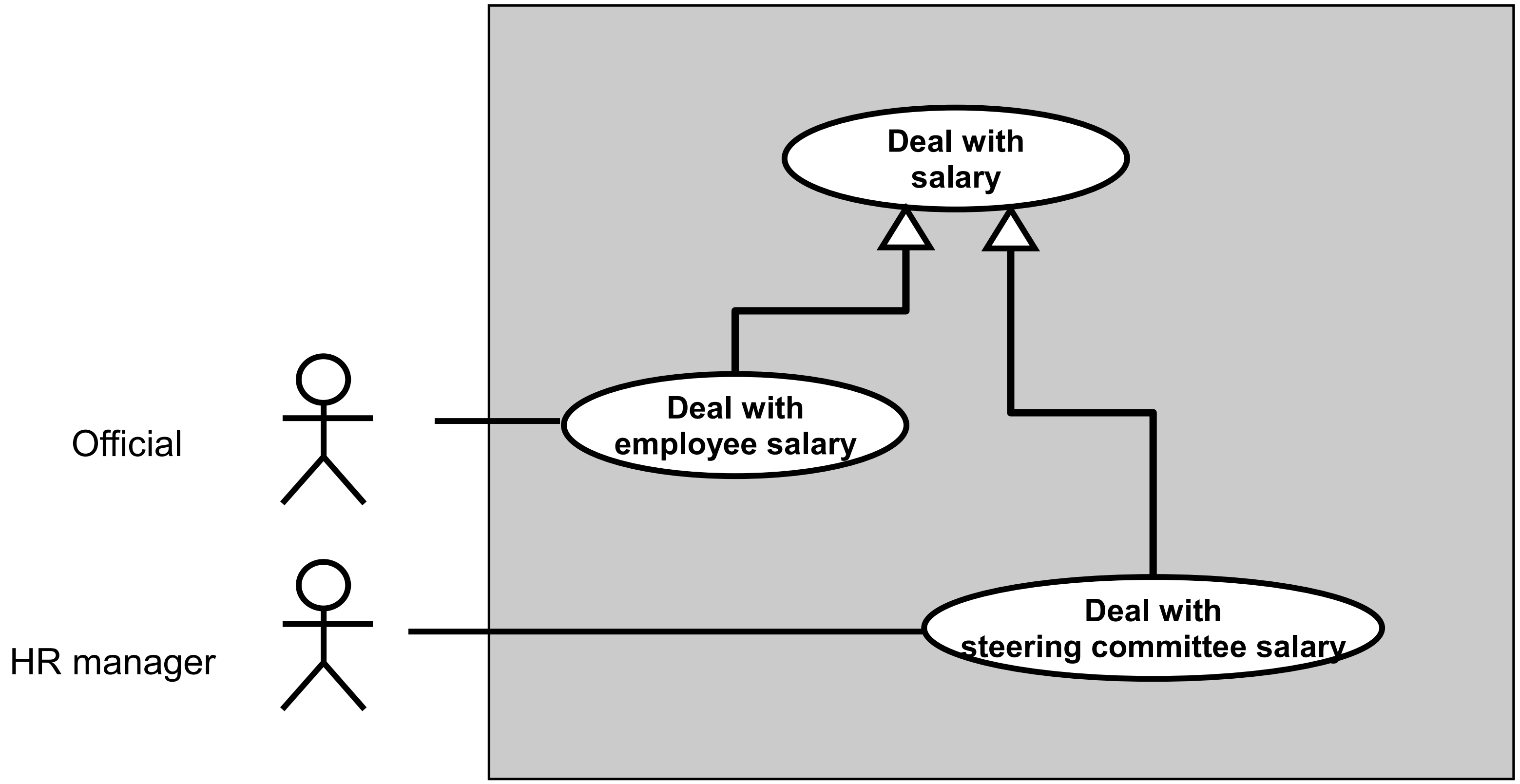
- Specialization / Generalization – Attention:
  - A specialized Use Case may be used always instead of the generalized


Official
HR manager
Deal with employee salary
Deal with steering committee salary

In this example the official may deal with the salary of a steering committee!

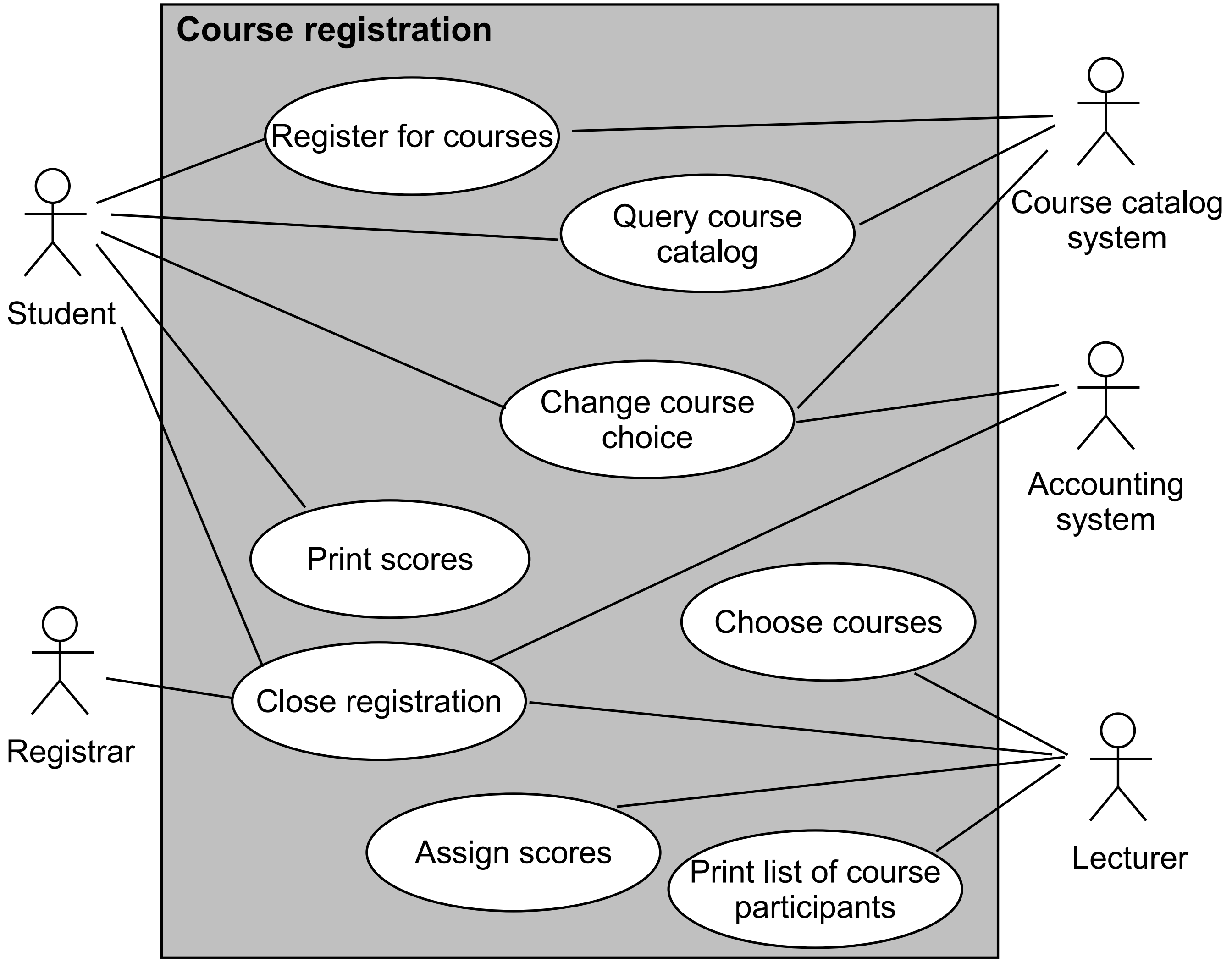
Use Cases
Use Case diagrams

- Specialization / Generalization
  Possible solution


Deal with salary
Official
Deal with employee salary
HR manager
Deal with steering committee salary

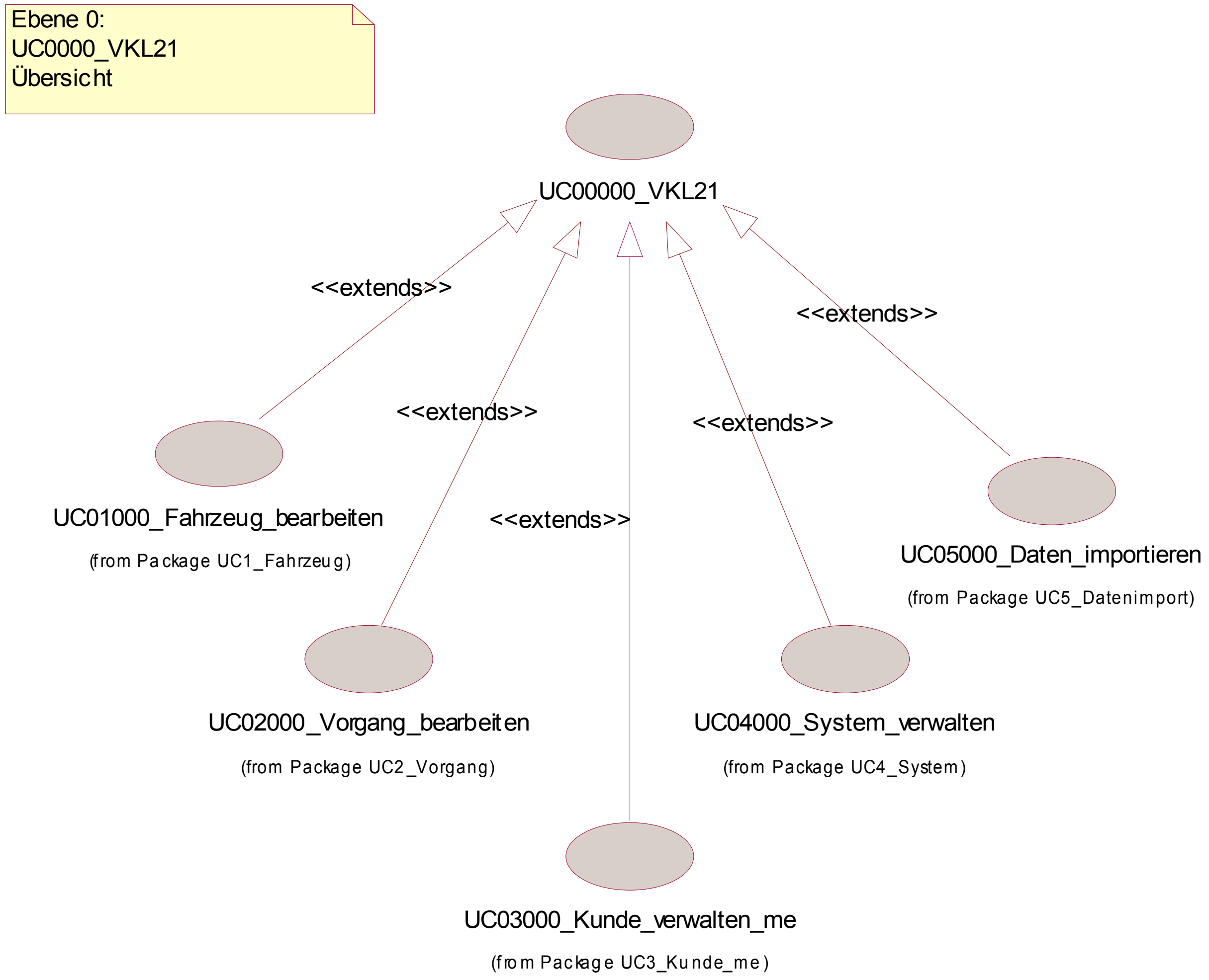
Use Cases
Use Case diagrams

- Example



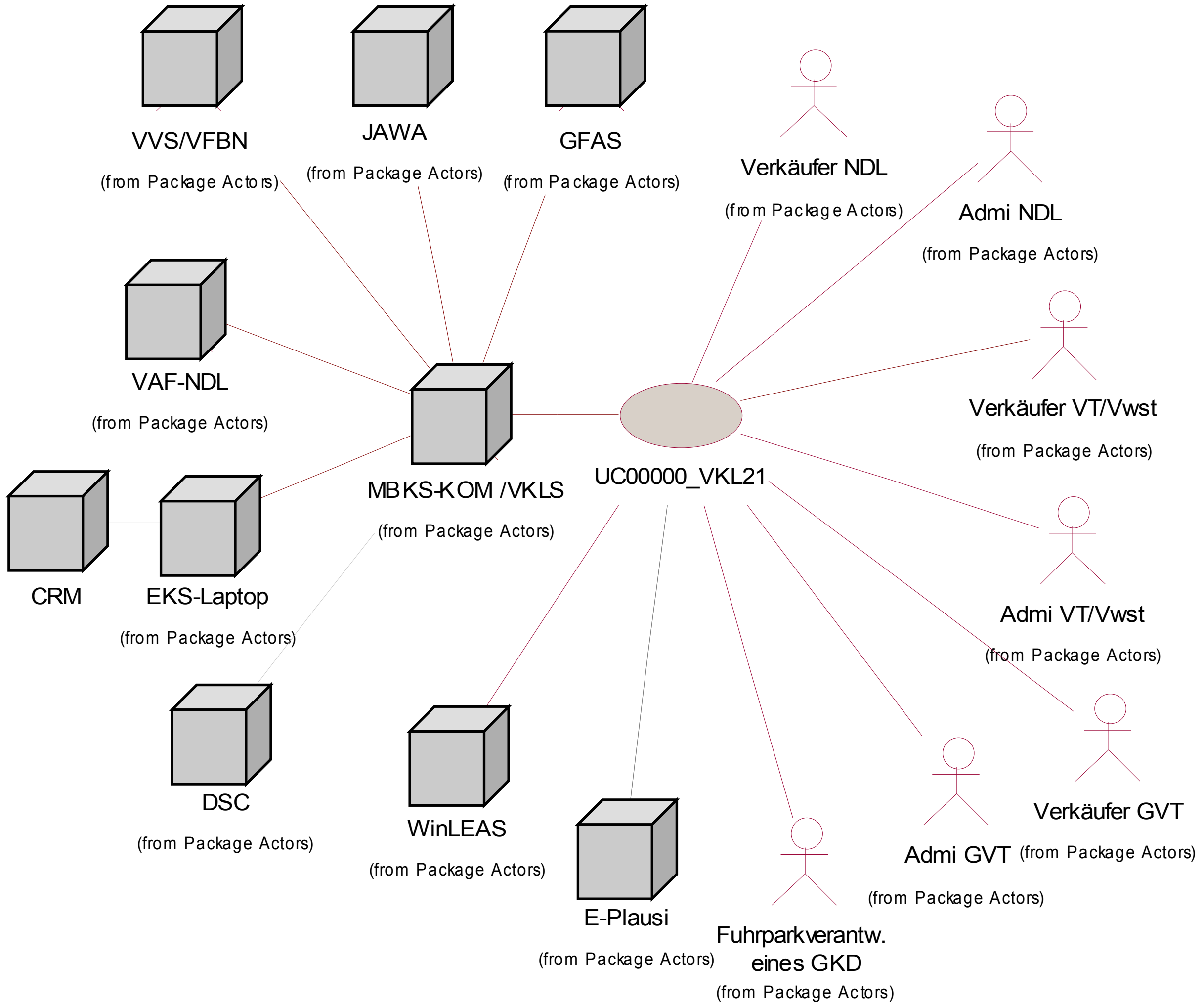
Use Cases
Use Case diagrams

- Example out of practice


Ebene 0:
UC0000_VKL21
Übersicht
UC00000_VKL21
<<extends>>
<<extends>>
<<extends>>
<<extends>>
<<extends>>
UC01000_Fahrzeug_bearbeiten
(from Package UC1_Fahrzeug)
UC05000_Daten_importieren
(from Package UC5_Datenimport)
UC02000_Vorgang_bearbeiten
(from Package UC2_Vorgang)
UC04000_System_verwalten
(from Package UC4_System)
UC03000_Kunde_verwalten_me
(from Package UC3_Kunde_me)

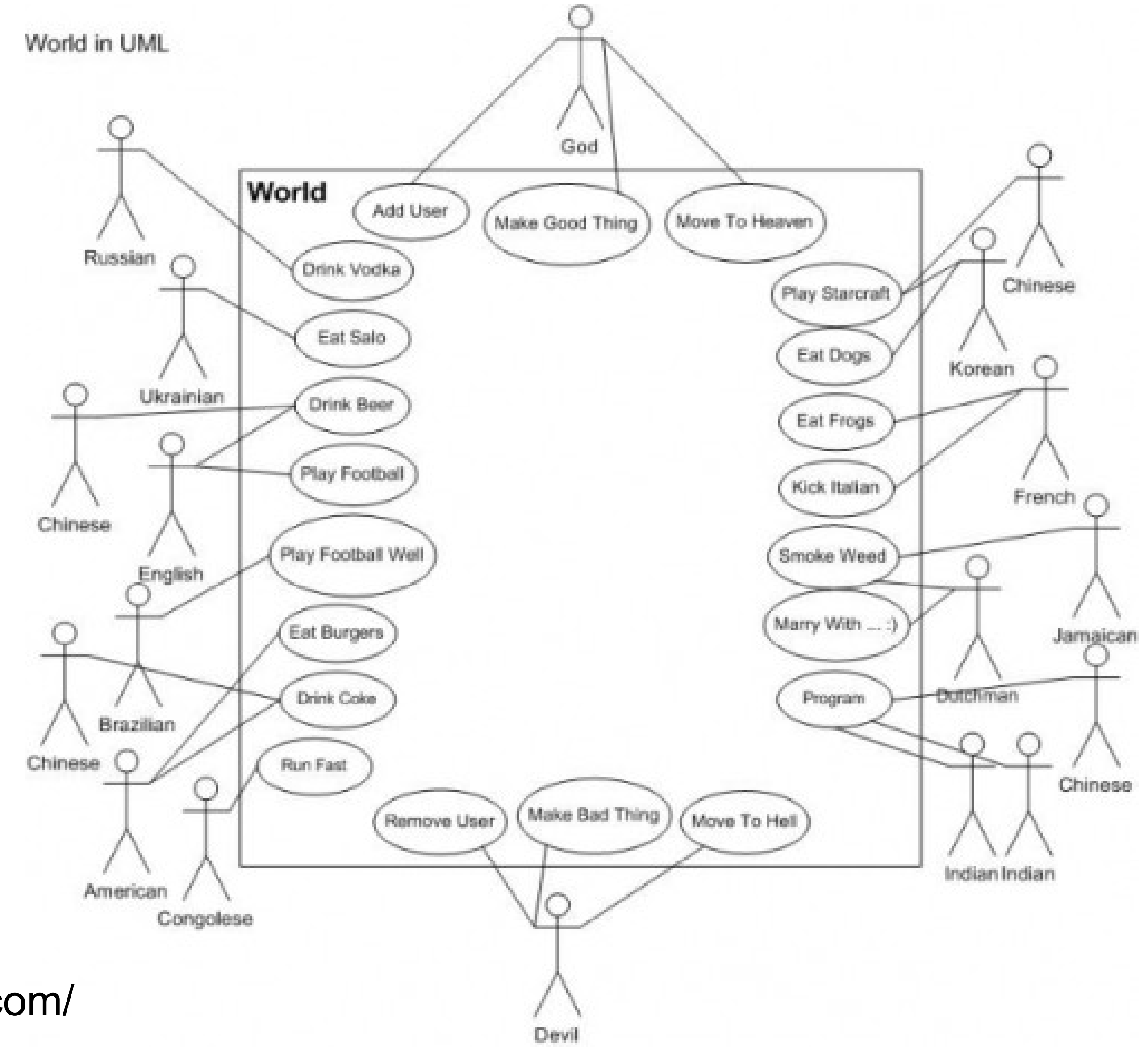
Use Cases
Use Case diagrams

- Example out of practice



Use Cases
Use Case diagrams

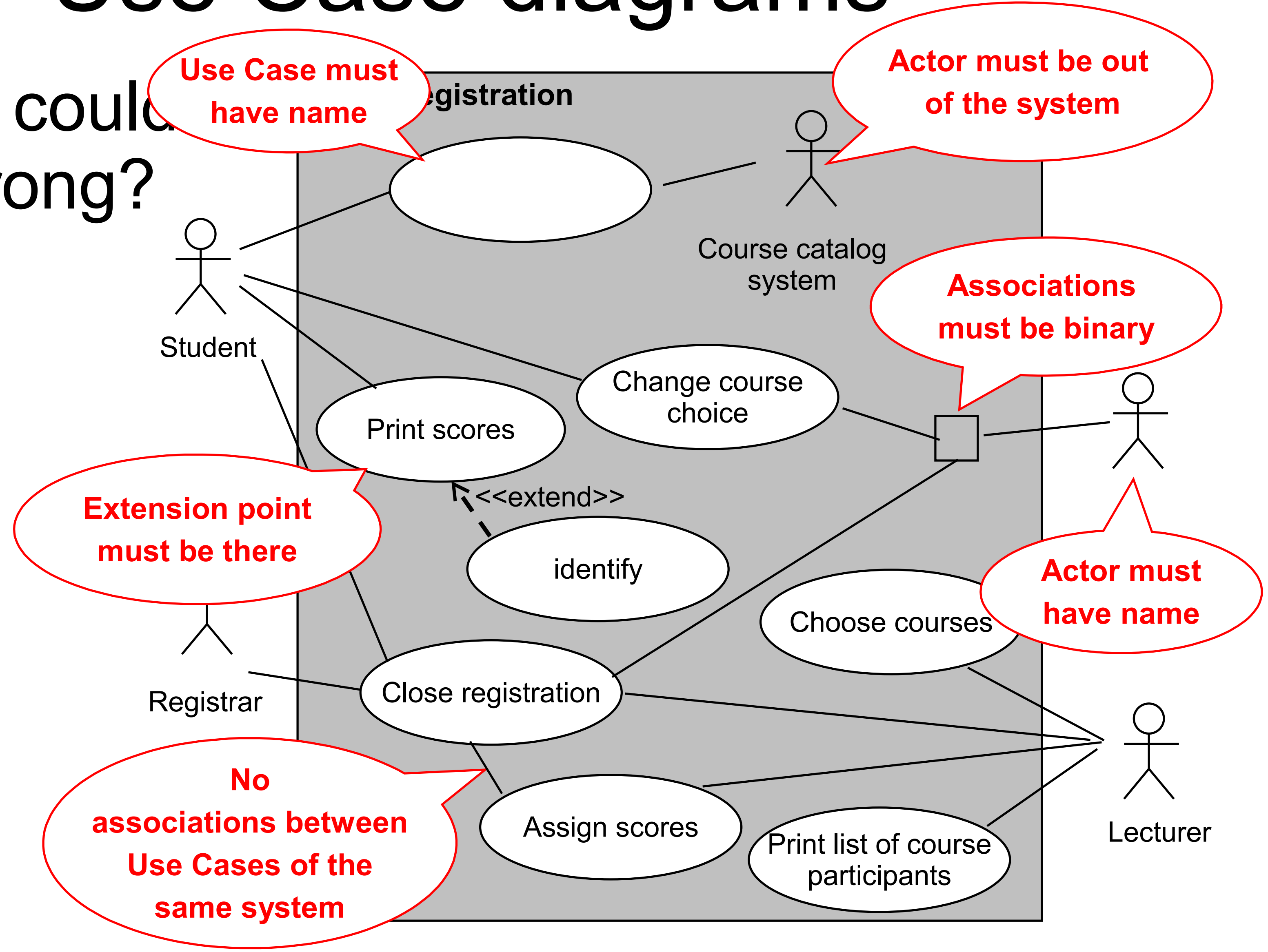
- ... and more



Source: http://www.umljokes.com/

Use Cases
Use Case diagrams

- What could go wrong?



Source [JRHZ03]

Use Cases
Use Case description

- Contents of a Use Case description [pp. 109 Oes01]
  - Id
  - Name (Active formulation: „Verb Object“)
  - Short description and goal
  - Actor
  - Used (incoming) information
  - Results and output information
  - Pre conditions and post conditions
  - General activities (about 1-2 pages common speech) with numbered steps
  - Exceptional activities
  - Cross reference to other Use Cases concerning extend and include relationships

There is no ultimate valid list!

Use Cases
Use Case description

- Describe for every Use Case the activities of the main scenario:
  - As abstract and short as possible
  - As concrete and extensive as necessary
  - Numbering of each step
  - Identify for every step in the main activities every possible professional exception;
    - Example for professional exception „Article not in stock“
    - System problems like „Harddisk overflow“ are not to be considered (non functional!!!)

Use Cases
Use Case description

- Describe for every Use Case the activities of the main scenario:
  - Describe every professional exception with alternative activities
  - If necessary links to other Use Cases which are in connection to the original Use Case with <<include>> or <<extend>> relationship

Use Cases
Use Case description

- Example

Id / Name	214 / Rent a car
Short description	A customer comes to the car rental agency and chooses a car which he rents for a fixed period
Actors	Customer, agent
Trigger	Customer asks agent
Pre condition	The rental system is ready to get customer data and to realize a lease contract
Result	Leasing is done, and the customer has signed the contract
Post condition	The rental system is ready to get customer data and to realize a lease contract
Activities	1. Enter customer data. If customer is yet not registered ⇨ UC *12 Register customer.* 2. Enter desired car category 3. Enter desired leasing period 4. If a car is available in the desired period: 1. Reserve a car 2. Enter credit card information 3. Print contract and sign Otherwise: Adapt item 2. or 3., if possible

Use Cases Proceeding

- How to find Use Cases?
  1. Identify Actors
     - ➢ Who uses the system?
     - ➢ Who gets information out of the system?
     - ➢ Who provides information to the system?
     - ➢ In which environment the system is used
     - ➢ Who operates the system?
     - ➢ Which other system is using the system?

Attention:
A stakeholder* is not an actor!

*Stakeholder = Everyone who is involved in the project (Customer, work council)

Use Cases Proceeding

- How to find Use Cases?

  2.Identify goals of Actors

  - ➢ What should the system do for me (as Actor)?
  - ➢ A goal describes the headline of a Use Cases
  - ➢ Active Form from the point of view of the actor („Verb noun“)
    - ◆ Examples: „Reserve car“, „Buy article“, ...

Use Cases Proceeding

- How to find Use Cases?

  3. Describe main scenario
     - Number every step
  4. Every step could cause a failure
     - Error handling means alternative scenarios
     - We talk about functional errors like “article not on stock”
     - We don't talk about technical problems like “hard disk full”

Use Cases Proceeding

- How to find Use Cases?
  - List of events (Tool of Hroschka) – A resource to find Use Cases
    - All events of the real world in the system environment gets collected in a “List of events”
    - External events occur, if an actor does something (Who is doing what?)
    - Time events with observing of clocks or system internal data memory (It's time to do ...)
    - Every event is an independent Use Case

Use Cases Proceeding

- Challenges
  - Scope of a Use Case - Example: Bottle automat: 1 or 2 Use Cases?
    - Alternative A: 1) Exchange bottle with receipt
    - Alternative B: 1) Enter bottle 2) Get receipt
  - Idea of Alistair Cockburn "Coffee break test": Could a user do a coffee break during interactions of a Use Case?
    - No?
    - Seems to be good granularity

Source: http://www.anker-andersen.com/images/produkter/retursystemer/retur_flaskeautomat.jpg

Use Cases Proceeding

- Challenges
  - Completeness
    - Use Cases describe functional requirements („Required Features“)
    - Use Cases themselves don't describe a complete specification, the non-functional requirements are missing („Required constraints”)

Use Cases
Faults – Traps

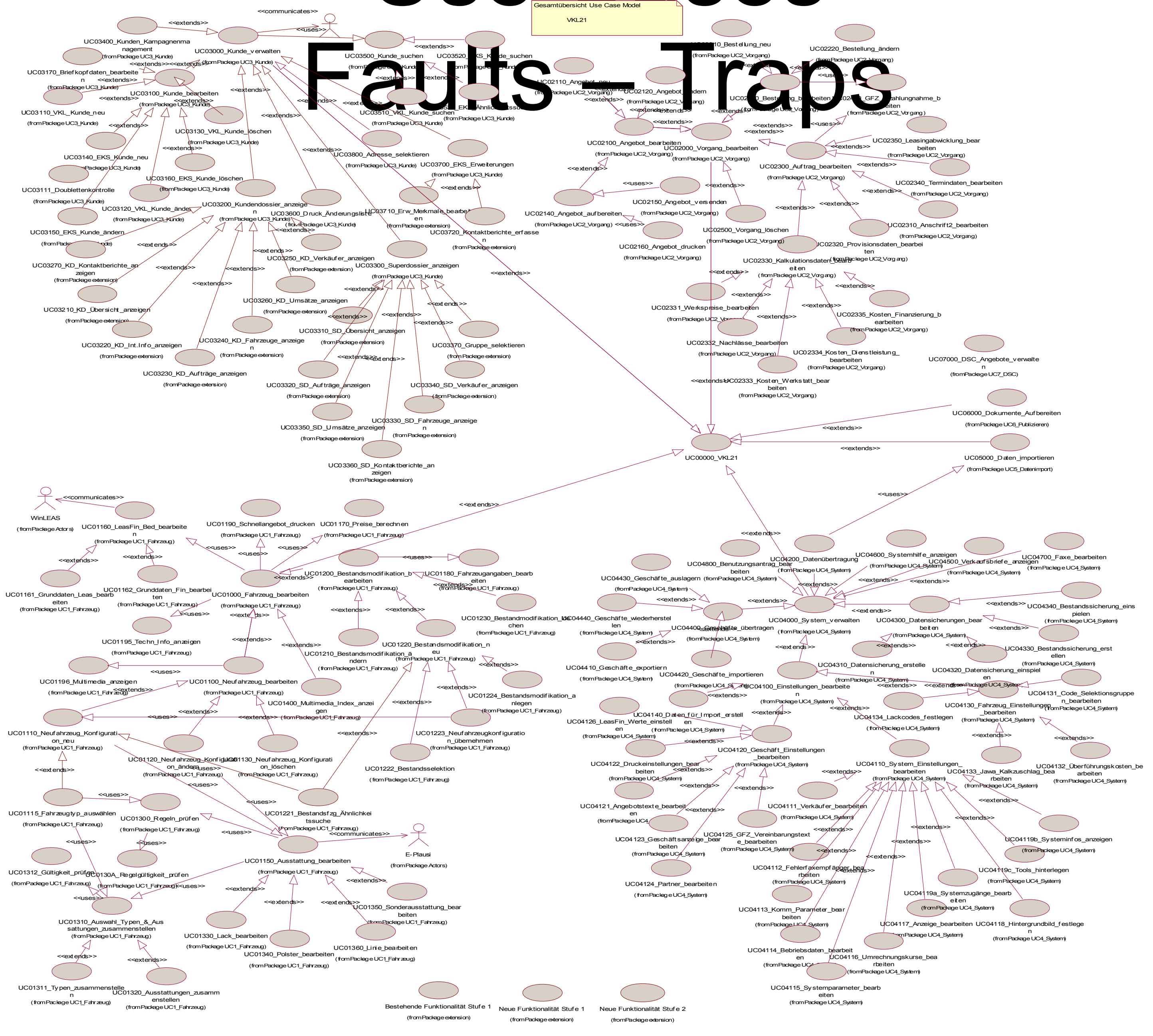
- Too marginal formulation
  - A Use Case description should describe all fundamental activities. Use Case Title in the diagram is not sufficient
  - All Use Cases together should describe the complete functional requirements
- System boundaries
  - E. g. Internet customer, Phone customer, call center agent
    Phone customer talks with call center agent – who is the (real) actor?

Use Cases
Faults – Traps

- Relationships between Use Cases (include / extend / generalization) reduce readability. That's why to be used thriftily
- Extreme breakdown of Use Cases causes lost of clear view

Use Cases

Faults - Traps


Gesamtübersicht Use Case Model
VKL21
UC00000_VKL21
UC03000_Kunde_verwalten
UC02000_Vorgang_bearbeiten
UC01000_Fahrzeug_bearbeiten
UC04000_System_verwalten
WinLEAS
E-Plausi
Bestehende Funktionalität Stufe 1
Neue Funktionalität Stufe 1
Neue Funktionalität Stufe 2

Use Cases
Faults – Traps

- Too much formalization
  - Pseudo Code, IF...THEN-Formulation
  - Both the customer and developer respective software architect have to understand the Use Cases
  - Ideally the customer writes the Use Case Description in his own words („not filtered“)
- Too early definition of solutions („How“)
  - GUI-Specification
- Bad choose of Use Case Names
  - Imprecise, ambivalent, too general formulation make it more difficult – like not necessary plural “rent cars”

Use Cases
Hints – Tricks

- Use Cases are good master for Test Cases
  - Concrete Test Cases out of abstract Use Cases
- Create glossary to ensure consistent use of terms (No synonyms, no homonyms)
- Multiple iteration with Use Cases to get more and more detailed
- Use Case description could additionally content diagrams (e.g. state diagrams) and spread sheets, as long as all involved people understand!

✓

Use Cases
Hints – Tricks

- Collect Use Cases in a database
  - Administration easier
  - Different reports for different audience possible
    - Management summary
      (diagrams, names, short descriptions)
    - Specification (diagrams and description)
    - Complete information (specification and more like GUI-model, object model)

✓

Differentiation
Business Processes / Use Cases

- Business Process
  - A Business Process is a collection of Business Use Cases
  - A Business Process contents interrelated activities to generate a benefit for the company
- Use Case
  - describes interactions with a system
  - is initiated by an actor