Specification, Part 2

Use Cases

Introduction to System Engineering

AARHUS UNIVERSITY
DEPARTMENT OF ENGINEERING

Agenda

- What is a Use Case
- Why use Use Cases
- Use Case diagrams
- Actors
- Scenarios : Format and styles
- Guidelines: Finding and describing actors and UC's
- Good and Bad Use Cases

Agenda

- What is a Use Case
- Why use Use Cases
- Use Case diagrams
- Actors
- Scenarios : Format and styles
- Guidelines: Finding and describing actors and UC's
- Good and Bad Use Cases

What is a Use Case (1/2)

A use case is a specific way of using the system by performing some part of the functionality. Each use case constitutes a complete course of events initiated by an actor, and it specifies the interaction that takes place between an actor and the system

- A textual description of events between a user and a system
- In order to discover and describe requirements
- Are described from a users view or the environment (not from the systems view)

What is a Use Case (2/2)

"If you design a new house and you are reasoning about how you and your family will use it, this is use case-based analysis. You consider the various ways in which you'll use the house, and these use cases drive the design."

– Booch

What is a Use Case (3/3)

- Defined by Ivar Jacobson, Ericsson
- Originally named: Usage cases
  - Swedish: användningsfall
  - Danish: brugssituation / brugstilfælde
    - Normally ”Use case” is used
- Key people
  - Ivar Jacobson, Craig Larman, Alistair Cockburn

What are they used for

- Used to specify the functional requirements
  - In FURPS+, they give emphasis to the F
- Outside-in approach, where the functionality is described from the users viewpoint
  - Not from the developers

Specification and Use Cases

- A use case describes behavior
- Requirements describes a law that directs behavior

Use Case Example

Buy Product

1. *Customer* browses through catalog and selects items to buy
2. *Customer* goes to check out
3. *Customer* fills in shipping information
4. *Customer* fills in credit card information
5. *System* authorizes purchase

5a [Authorization fails]
Customer may go to 4. or Cancel

Martin Fowler, UML Distilled (2nd Edition)

Use Case terminology

Buy Product

Actor

Reference

Scenario

1. *Customer* browses through catalog and selects items to buy
2. *Customer* goes to check out
3. *Customer* fills in shipping information
4. *Customer* fills in credit card information
5. *System* authorizes purchase

5a [Authorization fails] *Customer may go to 4. or Cancel*

Extensions/ Alternate Flows

Use Case terminology

- A use case either reach it's goal or fails;
- If only a main success scenario is given, then success is assumed
- Use Case naming rule:
  Described in terms of obtaining a goal for a given actor

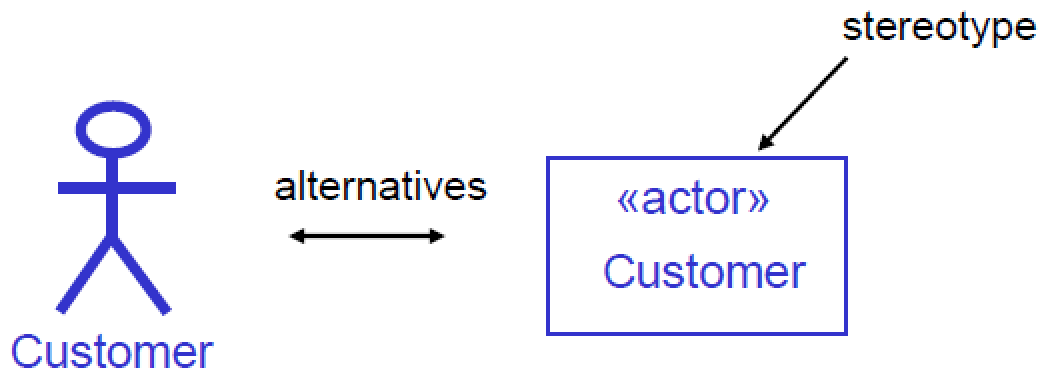
  Examples:
  Withdraw Money
  Check Balance

Why use Use Cases

- Capturing requirements of a system
- Validating systems (all use cases are realized)
- Can drive implementation and tests
- Use Case Modeling is
  - A simple concept
  - User-friendly (allow customers to contribute)
  - Effective
- Discovery and definition are the goals of Use Cases

Actors

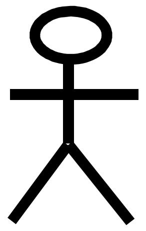
- Role – more precise translation from swedish
- An actor describes an object, external to the system, who interacts with the system
- Actors represents:
  - Persons
  - Other systems
  - HW devices
- UML notation:



Actors

- Person actors:
  Describes the role played by a person who interacts with the system
  The same person can play different roles over time
- Other systems:
  Describes the external systems who communicates with the system under development
- HW devices:
  A concrete HW unit can also be subdivided in different logical roles played by the HW unit or device

Actor type



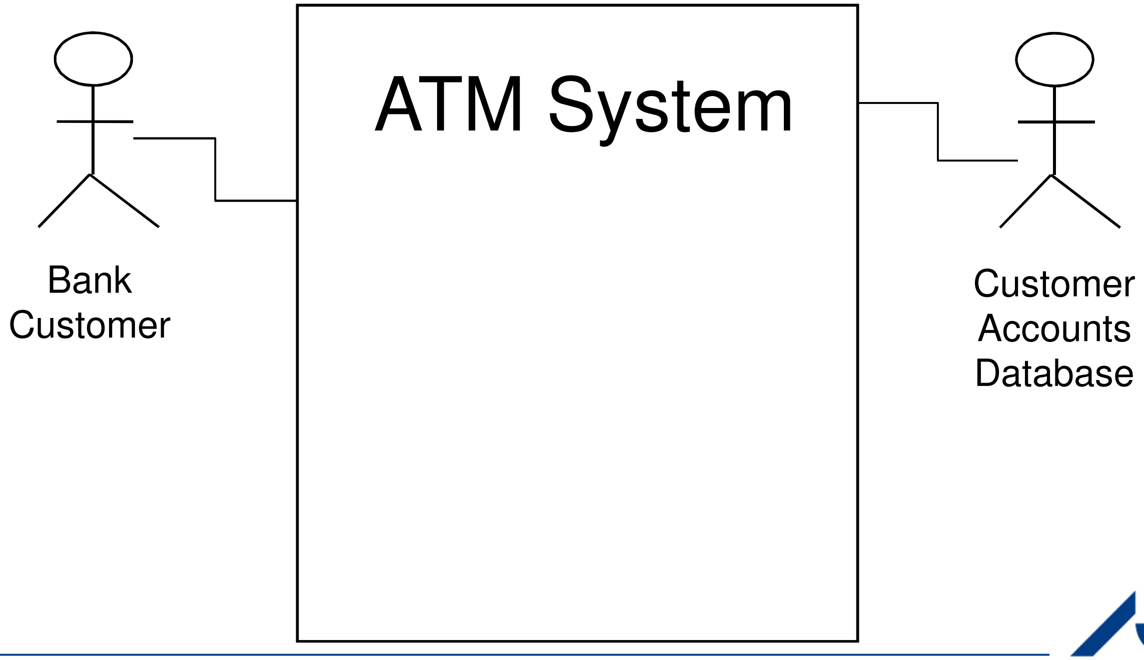
- Primary Actor
  Has goals to be fulfilled by system
- Supporting Actor (alternatively Secondary Actor)
  - Provides service to the system
- Offstage Actor
  - Interested in the behavior, but no contribution

Actor description

Name of actor:	Shopper
Alternate references	Customer
Type:	Primary
Description:	The Shopper is the sole end user of the system … Wants to …

Actor-Context diagram

- Supplies overview of the actors in relation to the system boundary
- Essentially a diagram with actors, the system boundary and no use cases.



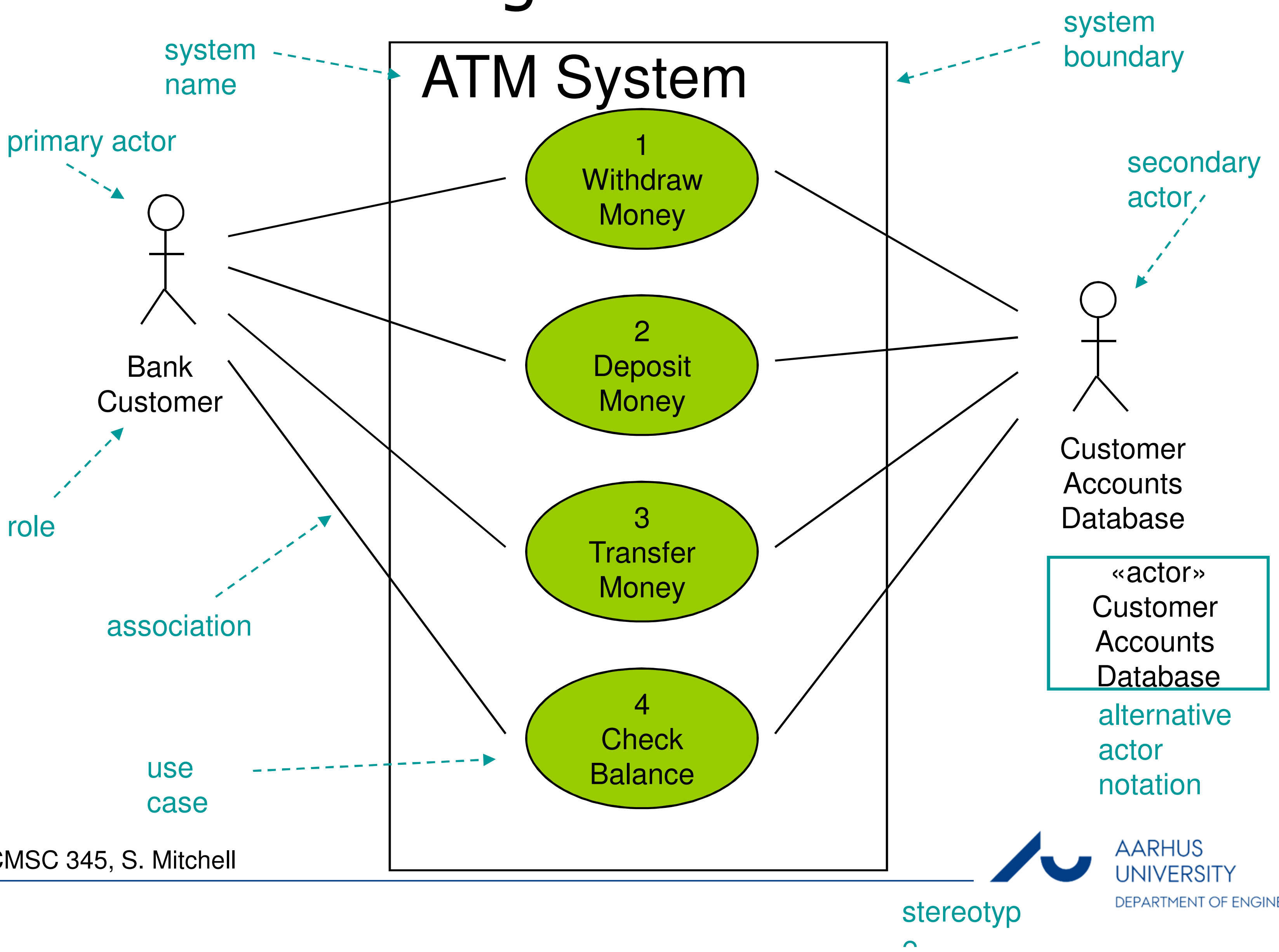
System Boundary

- Boundary between the inside and the outside of the system
- Determining the level of abstraction
- There may be systems within systems
  - One system may be made up of several subsystems, which interact with each other
- Identify interactions between the system and its environment (stimuli and responses)

Why money is not an actor

- No interest in the scenario
- A means to fulfil the scenario

Use Case diagrams



Exercise 1

- Bomanlæg
  - Find Boundary
  - Actors
- Ismaskine
  - Find Boundary
  - Actors

Use Case diagrams

- A way of visualizing the relationships
  - between actors and use cases
  - among use cases
- A graphical table of contents for the use case set

Use Case diagrams (Parkerings-automat)

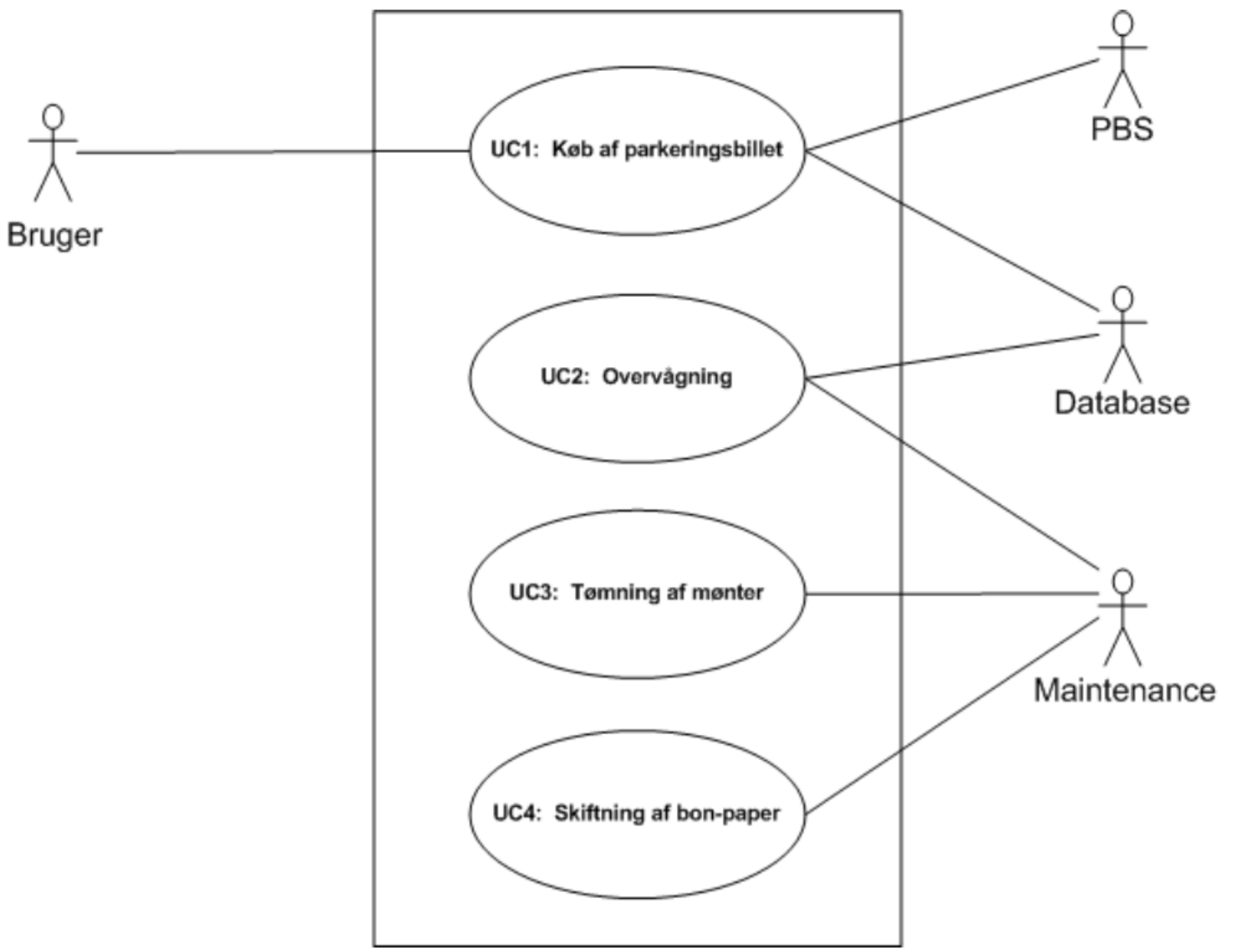


Diagram Purpose

- A Use Case diagram is an overview of Use case scenarios
- Diagrams shall only contain Use Cases that actually exists
- Do not join several Use Cases into one in a diagram

Exercise 2

- Medical Consultation System
- Draw a Use Case diagram of the following:

	Patient	Doctor	EPR/EPJ	National Board of Health
Make Appointment	Primary	Secondary	-	-
Cancel Appointment	Primary	Secondary	-	-
Access Patient Record	-	Primary	Secondary	Offstage

Scenarios

- The Main Success Scenario is the scenario where the actor obtains its goal
  - - is also called
    - Sunshine scenario
    - Happy scenario
- Extensions / Alternative Flows
  - The alternative flows can be more comprehensive than the main scenario

Styles

- Essential:
  - Focus is on intent
  - Free of technology and mechanisms
  - Avoid making user interface decisions
- Concrete:
  - UI decisions are embedded in the use case text
    - e.g. “Admin enters ID and password in the dialog box, (see picture X)”

Scenarios : Formats

- Different ways of structuring use cases:
  - Brief
  - (Casual)
  - Fully Dressed

Scenarios : Formats : Brief

- 1-6 sentence description of behavior
- Mention only most significant behavior and failures
- Short enough to put many on a page
- Used to
  - Estimate complexity
  - To get a sense of subject and scope

Scenarios : Formats : Brief : Example

Process Sale:

A customer arrives at a checkout with items to purchase. The cashier uses the POS system to record each purchased item. The system presents a running total and line-item details. The customer enters payment information, which the system validates and records. The system updates inventory. The customer receives a receipt from the system and then leaves with the items.

Applying UML and Patterns, C. Larman 2005

Scenarios : Formats : Fully dressed

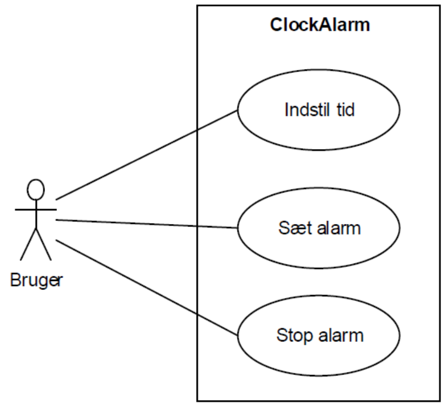
- Paragraphs written in a numbered form
- Includes all step and alternate flows in detail
- Normally follows a defined template of supporting sections

Fully-dressed Template

Name	Name of UC (Start with verb)
Goal	What is achieved by the UC
Initiation	Actor, System initiation
Actors and Stakeholders	List of actors Actor role (type)
References	Other use cases referenced
Number of concurrent occurrences	2, 10, none
Precondition	What must be true on start
Postcondition	What is true on completion
Main Scenario	Happy path scenario
Extension	Alternate flows
Data Variations List	e.g. Data Formats

Fully-dressed elements

- Actors and Stakeholders
  - list of stakeholders and their key interests in the use case
- Pre/Post conditions
  - assumptions before and success guarantees
- Data Variations List
  - technical variations in how data is defined



Fully-dressed example (Alarm Clock)

Navn:	Sæt alarm
Mål:	Bruger ønsker at sætte alarmtiden.
Initiering:	Bruger trykker på ALARM knappen
Aktører:	Bruger - primær
Samtidige forekomster:	1
Prækondition:	Uret er tændt og operationel
Postkondition:	Alarmen er sat til den ønskede tid

Hovedscenarie:

1. Bruger trykker på ALARM
2. Urets display viser tidligere alarm
   [Extension 1a: Ingen tidligere alarm]
3. Bruger trykker på henholdsvis HOUR og MIN
4. Uret optæller time og minut visningen for alarm
5. Bruger trykker på ALARM for at afslutte indstillingen
6. Uret skifter tilbage til at vise klokken

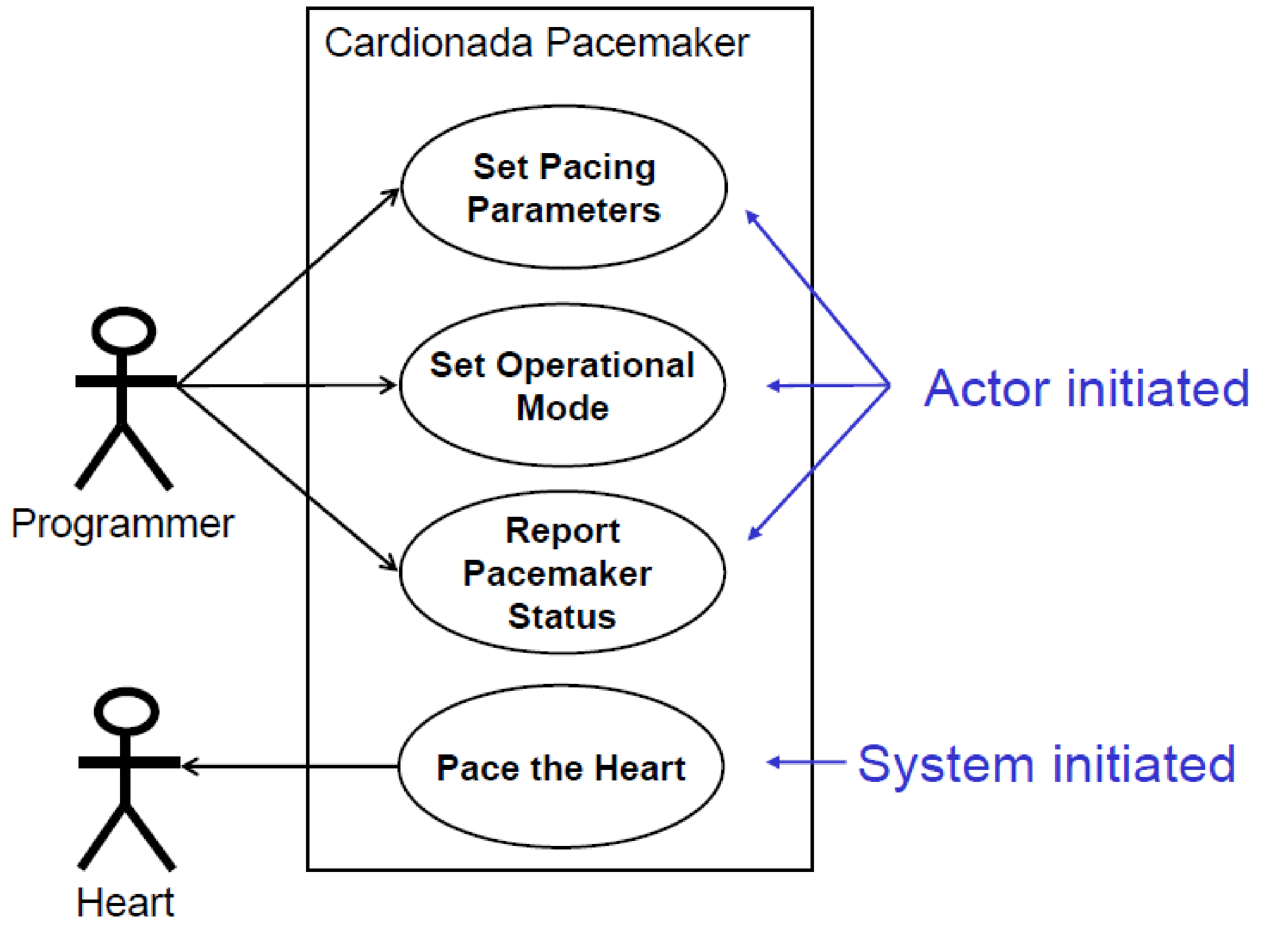
Udvidelser/undtagelser:

[Extension 1a: Ingen tidligere alarm]
Alarm indstillingen starter ved 00:00.

Use Case activation

- Actor initiated:
  - Actor takes initiative to activate a Use Case
- System initiated:
  - Use Cases activated by the system, is very common in technical systems
    - Periodic activated Use Cases
    - Aperiodic activated Use Cases, where a Use Case is started, when a given condition is true

Use Case activation



Guidelines: Building a System in UC's

1. Name the system scope
2. Brainstorm and list the primary actors
3. Brainstorm and exhaustively list user goals for the system.
4. Select one use case to expand
5. Write the main success scenario
6. Brainstorm and exhaustively list the extension conditions
7. Write the extension-handling steps

Writing Effective Use Cases, A. Cockburn, 2000

Finding and describing actors

- Identify
    - Ask the End-Users
    - Documentation
- Issues
    - Roles Vs. Job Titles
    - Time

Finding and describing use cases

- Scenario Driven
  - Find measurable value
  - Business events
  - Services actor needs / supplies
  - Information needed
- Actor/Responsibility
- Mission decomposition

Testing use cases : Boss test

- The Boss Test
  - the boss ask questions about the Use case scenarios to ensure they fulfills the needs.
  - Your Boss asks, “What have u been doing all the day?” You reply: “Logging in”..
  - Is your boss happy?

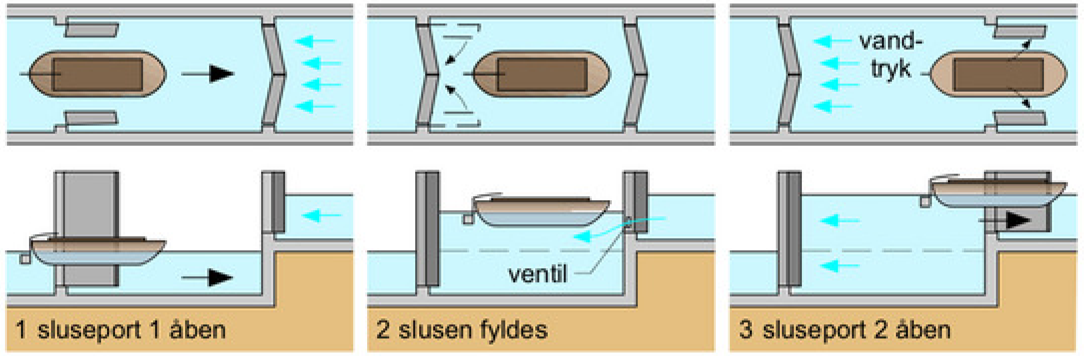
Exercise 3

- Service station
  UseCaseOvelse.pdf

AARHUS UNIVERSITY
DEPARTMENT OF ENGINEERING

Mandatory Exercise A

- Kaffeautomat
- Slusesystem



- Requirements, Use Cases and Accept Test

Extra topics for Use Cases

AARHUS UNIVERSITY
DEPARTMENT OF ENGINEERING

Developing a Use Case

Start out by answering these questions:

- Who are the primary and secondary actors?
- What are each (primary) actor's goals?
- What preconditions must exist before the story begins?
- What main tasks or functions are performed by each actor?
- What exceptions will need to be considered as the story develops?
- What variations in the actors' interactions are possible?
- What system information will each actor acquire, produce, or change?
- What information does each actor need from the system?

Developing a Use Case

- Use case names should start with a verb.
  - **DO:** Rent Items
  - **DON'T:** Item Rental
- Actor names should be capitalized.
- Use cases should be written in the active voice, using actors.

  DO: Customer arrives with videos to rent.

  DON'T: Videos are brought to the cash register by a Customer.
- Be as terse as possible while still being clear.

  DO: Clerk enters..., System outputs....

  DON'T: The Clerk enters..., The System outputs...

Good Use Cases

- Keep it simple
- Use present tense
- Subject should be primary actor, system under design and secondary actors
- Must provide a meaningful result to primary actor
- Verb should be what actor does to successfully move the use case forward
- Avoid GUI: write in terms of goals, not details of the GUI

Applying UML and Patterns, C. Larman 2005
Writing Effective Use Cases, A. Cockburn, 2000

Relationships between Use Cases

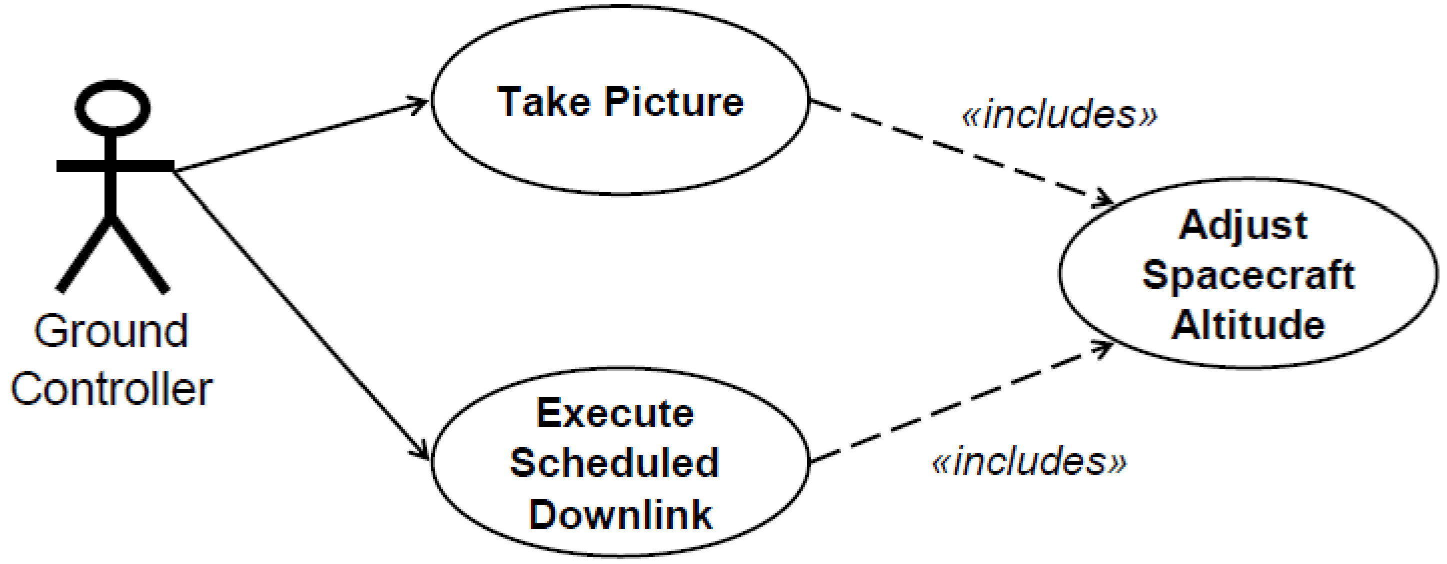
- You have three types of relationships:
  - Include
  - Extends
  - Generalization

- Use of these features will typically make a use case diagram more complex to read.
- So they should be used with caution.

Includes

- An *include* relation is a structuring mechanism
- Used primarily to avoid redundancy in the specification
  - An include use case can be used and reused in many situations

Includes



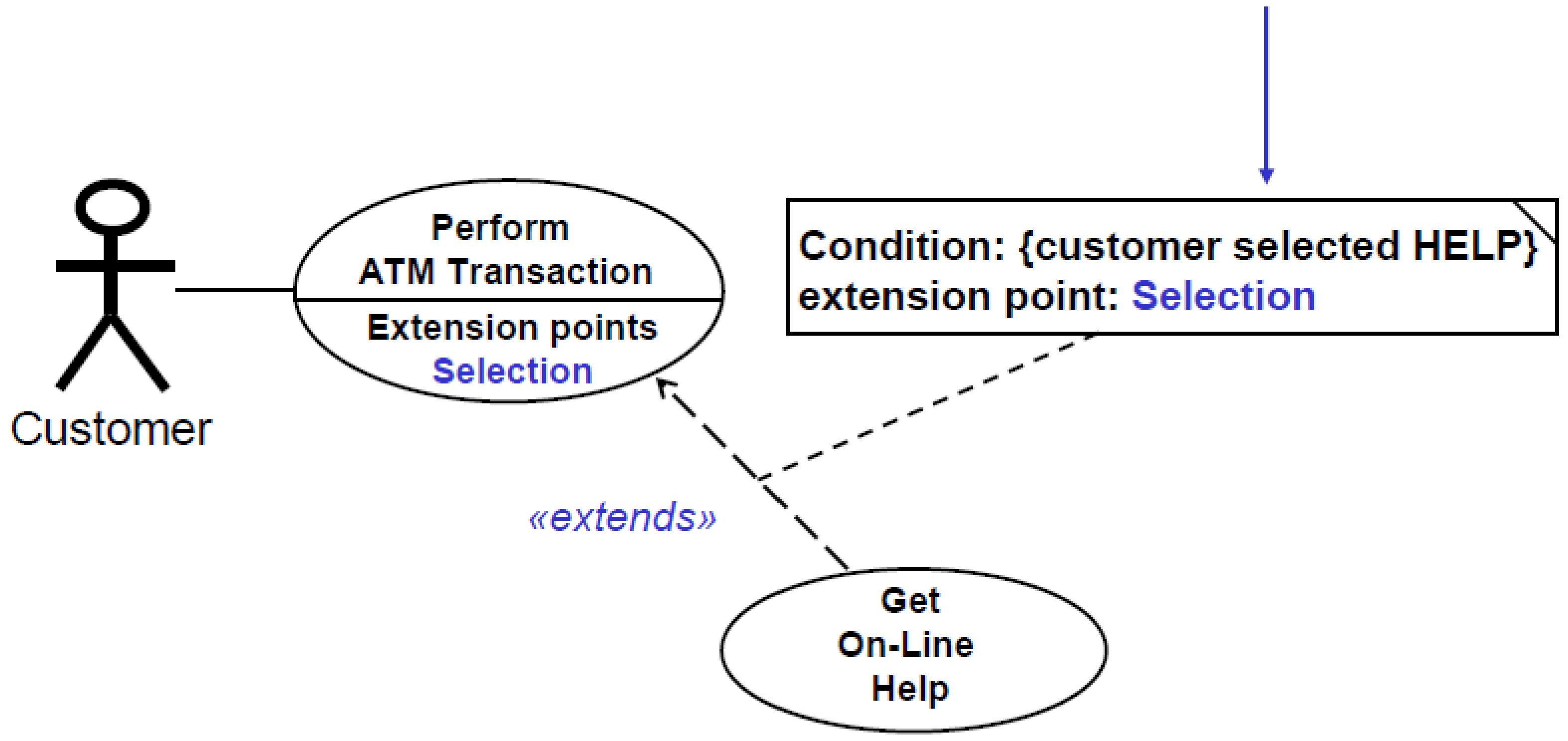
Common functionality is here moved out and described by its own Use Case

Extends

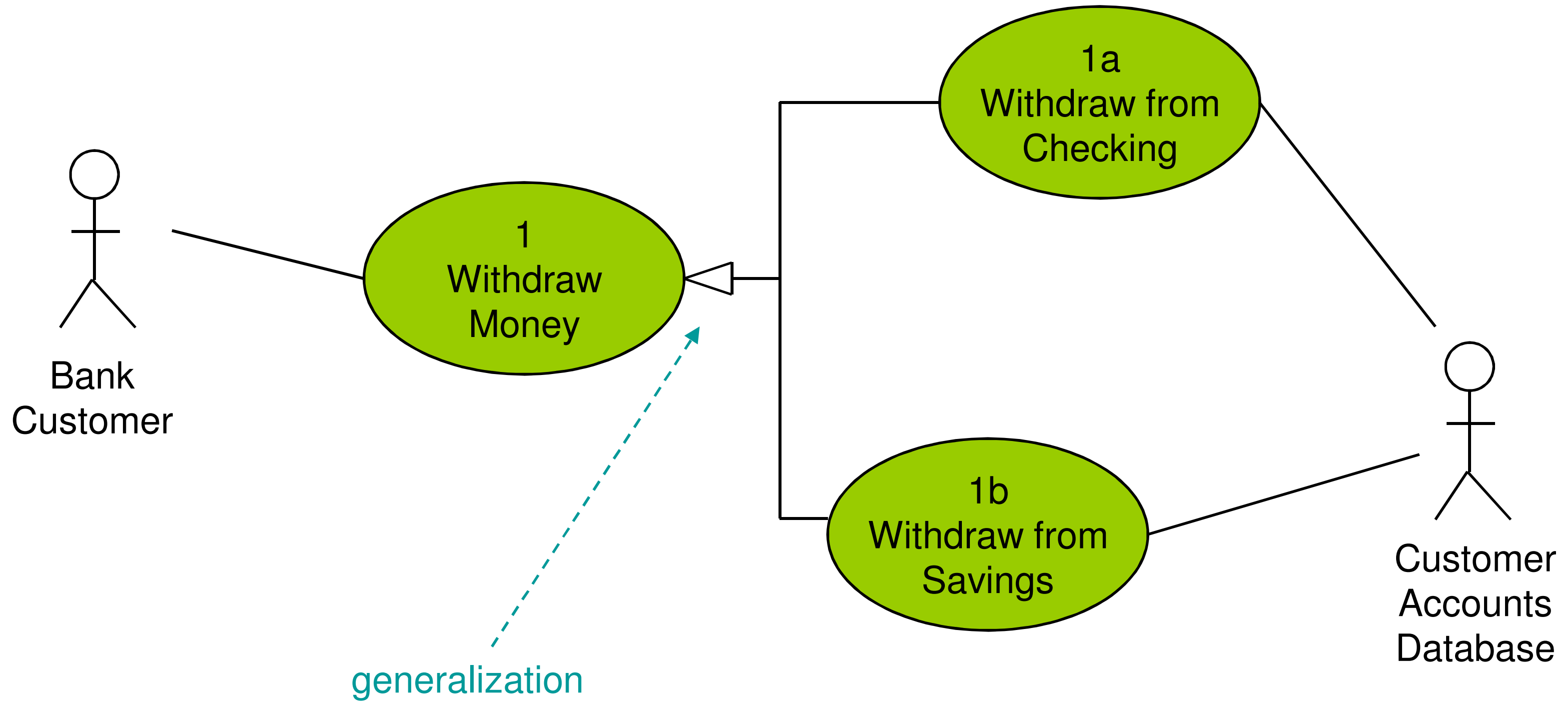
- An extends relation is a structuring mechanism:
  - Used to describe optional extensions
  - Used to describe special situations for example errors or other exceptional scenarios
- Can be used late in the development cycle – as a way to add functionality in a structured way, without disturbing the other use cases

Extends

Example of an extension with a condition



Generalization



Questions?

AARHUS UNIVERSITY
DEPARTMENT OF ENGINEERING