Requirements specification, Use case diagram

PB007 Software engineering I

Marián Macik
originally by Stanislav Chren

Week 02

Functional requirements

Functional requirements focus on **WHAT** the system should do (or should not do), i.e. it is related to the system's functionality.
It is recommended to follow the form:

- <**id**><**system**><**function**>

Examples:

- 1. The ATM verifies the validity of the inserted card.
- 2. The ATM verifies PIN provided by the customer.
- 3. The ATM does not allow to withdraw more than 10000 CZK for a given card over 24 hours.

Non-functional requirements

Non-functional requirement is a constraint imposed on the system. The constrained is often related to qualitative attributes, such as performance, availability, reliability, security, etc.

Examples:

- 1. The ATM will be programmed in C++ language.
- 2. The ATM will use 256-bit cryptography in communication with the bank.
- 3. The ATM will verify the card validity in less than 3 seconds.

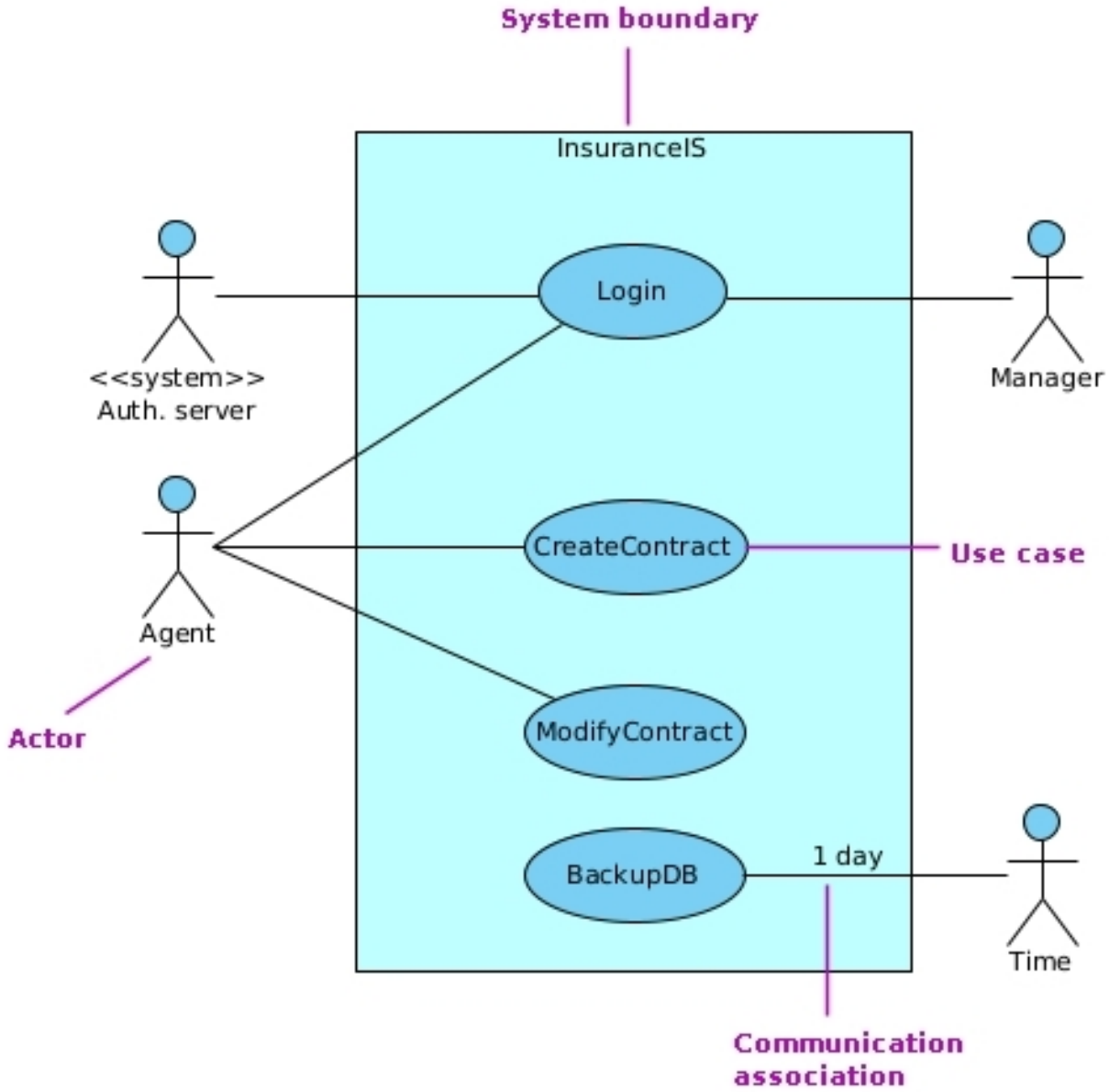
Use case diagram

Use case diagram is a method to capture the functional requirements of a software system.

It consists of:

- **System boundary, Subject**
- **Actors**
- **Use cases**
- **Relationships**

Use case diagram example



Actors I

Actor is a role that represents an external entity (user, another system, device, time), which directly communicates with the system.

- Actors are always external to the system (*with respect to the specified system boundary*).
- Actor communicates wit the system directly.
- Actors represent roles. They are not specific people or objects.
- Each person or object can have multiple roles at the same time. The role assignment can change over the time.
- Each actor must have an appropriate name.
- Each actor should have a short description.

Actors II

Identification of actors:

- Who/what uses the system?
- What role do they have in the interaction with the system?
- Are there any other systems communicating with our system?
- Who/what obtains/sends information from/to the system?
- Are there any events occurring periodically or at a fixed time?

Use cases I

Use case describes an interaction between the system and the external actors. These are the actions the actors need to perform in the interaction with the system.

- The use cases always begin with some action by an actor (*primary actors*). Afterwards, other actors can join the interaction (*secondary actors*).
- Use cases are expressed from the actor's point of view.
- The name should represent an activity or behaviour (i.e. it should be a verb phrase).

Use cases II

Identification of use cases:

- What system features are required by a particular actor?
- Does the system retrieve and store any information? If yes, which actors trigger this such behaviour?
- What happens then the system state changes? Are actors notified about the change?
- Are there any external events that affect the system? What alerts the system about these events?

Modelling procedure

To model the use case diagram, the following steps are recommended:

1. Define the system boundary
2. Find actors
3. Find use cases
4. Determine relationships between actors and uses cases
5. Specify use cases in detail

Information sources

- www.agilemodeling.com/artifacts/useCaseDiagram.htm
- http://www.agilemodeling.com/style/useCaseDiagram.htm
- www.andrew.cmu.edu/course/90-754/umlucdfaq.html
- www.drdobbs.com/top-ten-use-case-mistakes/184414701

Catalogue of common mistakes

- Available in the interactive syllabus

Tasks

- Create a new project in Visual paradigm.
  - Title: lastname1-lastname2-lastname3.
  - Language: UML 2.x.
- Create a numbered list of (non-)functional requirements in the documentation of the UC diagram. Order the functional requirements according to the user roles.
- Based on the functional requirements, create an initial UC diagram (add actors, use cases and communication associations between them).
- Look for gaps in the project specification and ask about information you are missing.
- Generate a **PDF report** and upload it to the homework vault (**Week 02**).

Rules for report submission

1. Submit the PDF report, not the VP source file and not an exported image.
2. PDF report must be created using the procedure shown on the seminars including the report settings.
3. The name of the PDF report file should be *lastname1-lastname2-lastname3* of the team members.
4. PDF report must contain all diagrams modelled until now.
5. PDF report must be uploaded to the homework vault by the specified deadline.
6. PDF report must be uploaded to the correct homework vault. The name of the homework vault is always specified on the slides.
7. Each team uploads only a single PDF report for the whole team.
8. Submitted diagrams must be clear and readable.
9. Submitted diagrams should not contain serious mistakes. At least, they should not contain mistakes mentioned in the *Catalogue of common mistakes*.

VP report settings