Object oriented analysis and modeling
Requirements modeling

Robert Lukoťka
lukotka@dcs.fmph.uniba.sk
www.dcs.fmph.uniba.sk/~lukotka

M-255

Functional requirements

- Identify actors
- Identify use-cases (btw. when do we use which one?)
  - Use case
  - Scenario
  - User story
- Define system boundary

Functional requirements

DRY - Do not repeat yourself

- Parts of use cases / scenarios are often repetitive
- includes
  - The included use case is an integral part of the including use case
  - Used to avoid text duplication
- extends
  - Optional
  - Extended use case is meaningful on its own
  - Extending use case typically defines optional behavior that is not necessarily meaningful by itself.
- inheritance (between use cases and/or actors)

Use case diagrams

When do we use use case diagrams?

- Business modeling
- Requirement documentation/analysis

Purposes:

- To identify actors, use-cases, system/subsystem boundary
- To capture relationship between use cases

Use case diagrams

Actors

- Primary - initiates the use case
- Secondary - is (may be) necessary to complete the use case
- Primary + secondary
- We can use inheritance on actors when appropriate - less lines = less mess

UML use case diagrams - how to draw them

Use case diagrams

Use cases, system boundary

- Assign use cases to primary and secondary actors
- Inheritance of use cases may be natural
- Child actor/use case inherits associated use cases from the parent
- Includes extends relationships between the use cases
- Define system boundary (some use cases may be outside of the system boundary)

UML use case diagrams - how to draw them

What to focus on?

To identify actors, use-cases, system/subsystem boundary

- System boundary
- Primary / secondary actors
- Actor / use case inheritance

UML use case diagrams - how to draw them

What to focus on?

To capture relationship between use cases

- Includes and extends relationships between use cases
- Actor/use case inheritance

How to model use cases themself?

This is only an issue for the use cases

- If the use case does not have complex structure, then a numbered list is often sufficient
- Complex stuff
  - Many alternative paths
  - Complex control structures
  - Concurrency
  - Events affect the use case execution

Uml activity diagram