

# Use Case Diagrams

Mohammad Mousavi

Formal Systems Analysis (FSA)  
Model-Driven Software Engineering  
Department of Computer Science  
TU/Eindhoven

November 14, 2011

## Definition [Cockburn'01]

Specification of a *contract* between the stakeholder of a system about its *behavior*.

## Application

- 1 the first step into the development process
- 2 classifying user **requirements**
- 3 **guiding** the rest of the development process
- 4 base for **test-cases**

## Common Mistakes

- 1 individual **methods** as use cases
- 2 internal **events** as use cases

Use cases are usually large end-to-end processes comprising several (inter)actions

## Running Example: A File System

In the file system, users can create new files, execute, display (on different output devices) and delete existing files. There is a special type of delete, which removes the file permanently from the file system. The file system makes use of an access right system which specifies who the owner of each file is and what operations are allowed by which users. The owner of each file may change the access rights to the file and give or take other people's permissions to access the file. In addition to the person who creates the file, the administrator is considered the owner of all files.

# Use Cases

## File System: Raw Use Cases

**Execute File**

**Change Access Rights**

**Create File**

**Display File**

**Delete File**

**Permanent Delete**

# Use Case Description

- 1 pre-condition: when a use case is available to its user,
- 2 trigger: what interactions initiate the use case,
- 3 guarantee (post-condition): what the use case achieves through this use case,
- 4 scenarios:
  - 1 main scenario: typical course of actions
  - 2 alternatives: re-consider each step, what can go wrong

## Example: Change Access Rights

- 1 pre-condition: the user "U" is logged in
- 2 trigger: the user issues a command to change the access right of a given file
- 3 guarantee: the access rights of the file are changed as specified by the user
- 4 scenarios:
  - 1 main scenario:
    - 1 the user issues a command to change the access right of file "A" for user "U" to "AR"
    - 2 the owner of the file "O" is retrieved
    - 3 if  $U=O$  or  $U=Admin$ , and the new access rights do not remove the access right of the owner or the admin, the change is made
    - 4 the user is notified about the successful outcome
  - 2 alternatives:
    - 3.1 if the file does not exist, an error is issued
    - 3.2 if U is neither admin nor O, an error is issued
    - 3.3 if the changes remove the access right of the owner or admin, an error is issued

## Sub-Scenarios

- Extra scenarios that are repeated in several use-cases.
- Make **abstract use-cases** out of them.
- Reference them in use case descriptions (instead of repeating them).

# Actor

## Definition

A class of entities (human or computer), falling beyond the system boundaries, interacting with the system.

## Running Example: A File System

In the file system, **users** can create new files, execute, ... The **owner** of each file may change the access rights to the file ... In addition to the person who creates the file, the **administrator** is considered the owner of all files.

## File System: Actors



Administrator



Owner



User

# Extract Use Cases

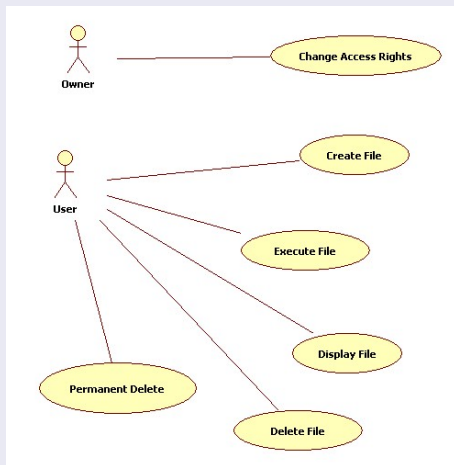
- 1 Identify system-boundaries and actors
- 2 What are the goals of each actor w.r.t. the system?
- 3 What are the external events that the system should react to?

# Association

## Definition

An actor interacts with the system for the specified use cases.

## File System: Some Associations



# Generalization

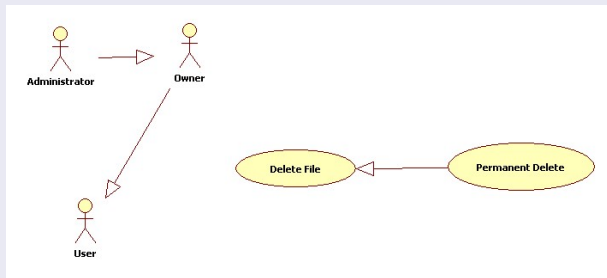
## Definition

A use case / actor is a special case of another one.

## Substitutability Principle

A specialized class (use case, actor) can always replace the general one.

## File System: Generalization Relations

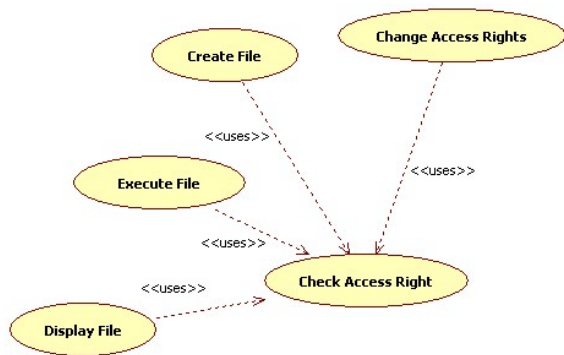


# Dependency (with <<uses>> stereotype)

## Definition

A use case depends on another use case for realizing its goal. The target use case is referenced by the source.

## File System: Some Dependencies



# Assemble It Into One Piece

