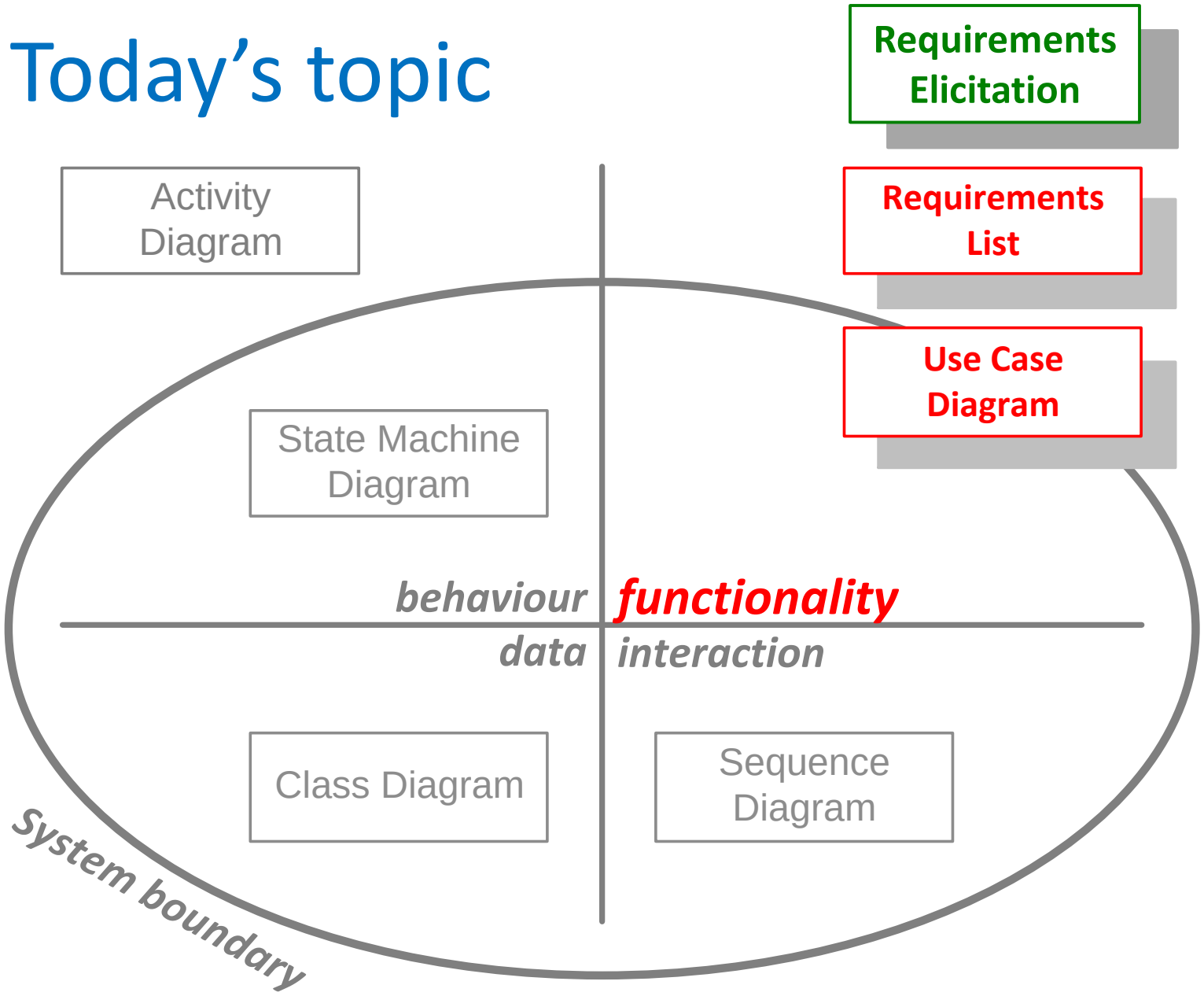
Software Systems

Design – 1B (part 1)
Requirements modelling

Joke van Staalduinen

Credits to Klaas Sikkel

Today's topic



Contents

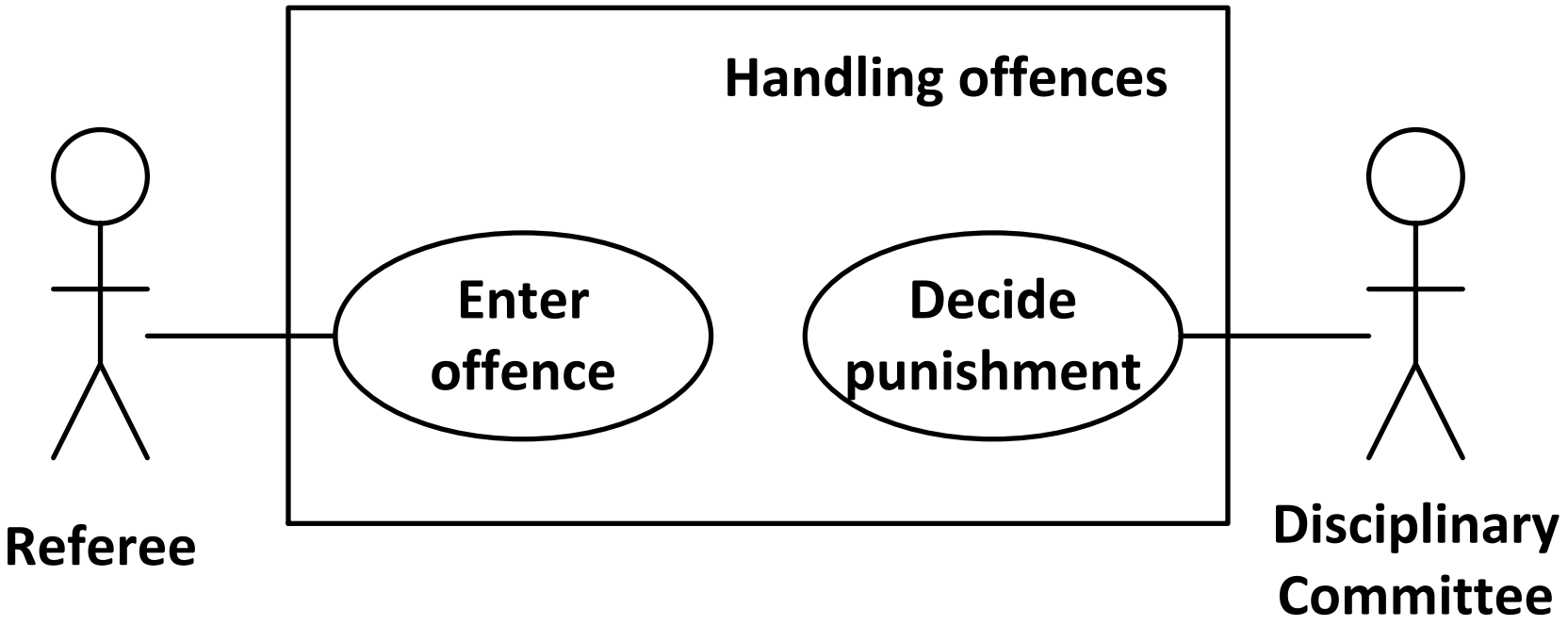
First hour

1. Use Case Diagram (basics)
2. Use Case Model
   - Glossary, Requirements list, Actor list
   - Use case diagram(s)
   - Use case descriptions (short + extended)
3. Use Case Diagram (some extensions)
4. Some pitfalls / guidlines for usage

Second hour

- Interviewing stakeholders

1. Use case diagram (basics)



(Case study from last lecture)

The Dutch Sports Association wants a system for registration of offences committed by players.

1. **When a player commits an offence during a match, the referee can give a warning or show a yellow / red card.**
2. **When a player gets a yellow / red card, after the match the referee enters the offence and the type of card into the system. Warnings are not registrered.**
3. **The Disciplinary Committee decides on a punishment** (and sends letters to the player and the club).

Use case diagram

- Top-level description of system functionality
- Easy enough to be understood by non-IT staff

- UCD contains
  - Actors
  - Use cases
    - Described as imperative sentences
    - Linked to a single actor

2. Use Case Model

- A Use Case Diagram gives a schematic overview of the system on the highest level
- However, additional descriptions are needed
  - to show that the requirements are met by the proposed use cases
  - to correctly interpret the diagram

2. Use Case Model

A Use Case Model consists of

- Glossary
- Requirements list
- Actor list
- Use case diagram
- Use case descriptions (short) of all use cases
- Use case descriptions (extended) as appropriate

2.1 Glossary

Term	*Description*
Offence	**Act of a player in violation of the rules**
Warning	**For a minor offence, a player can be reprimanded (not punished). Warnings are not recorded**
Red card	**Signals a major offence. The player must leave the match immediately and will receive punishment**
Yellow card	**Signals a minor offence. With one yellow card a player can continue the match, but the player will receive punishment**

2.2. Requirements list

Nr	*Requirement*	*Use case(s)*
1	***As a referee I want ...*** **To enter details about red / yellow card that I gave**	**Enter offence**
2	***As the Disciplinary Committee I want...*** **To decide which punishments are given for which offences**	**Decide punishment**

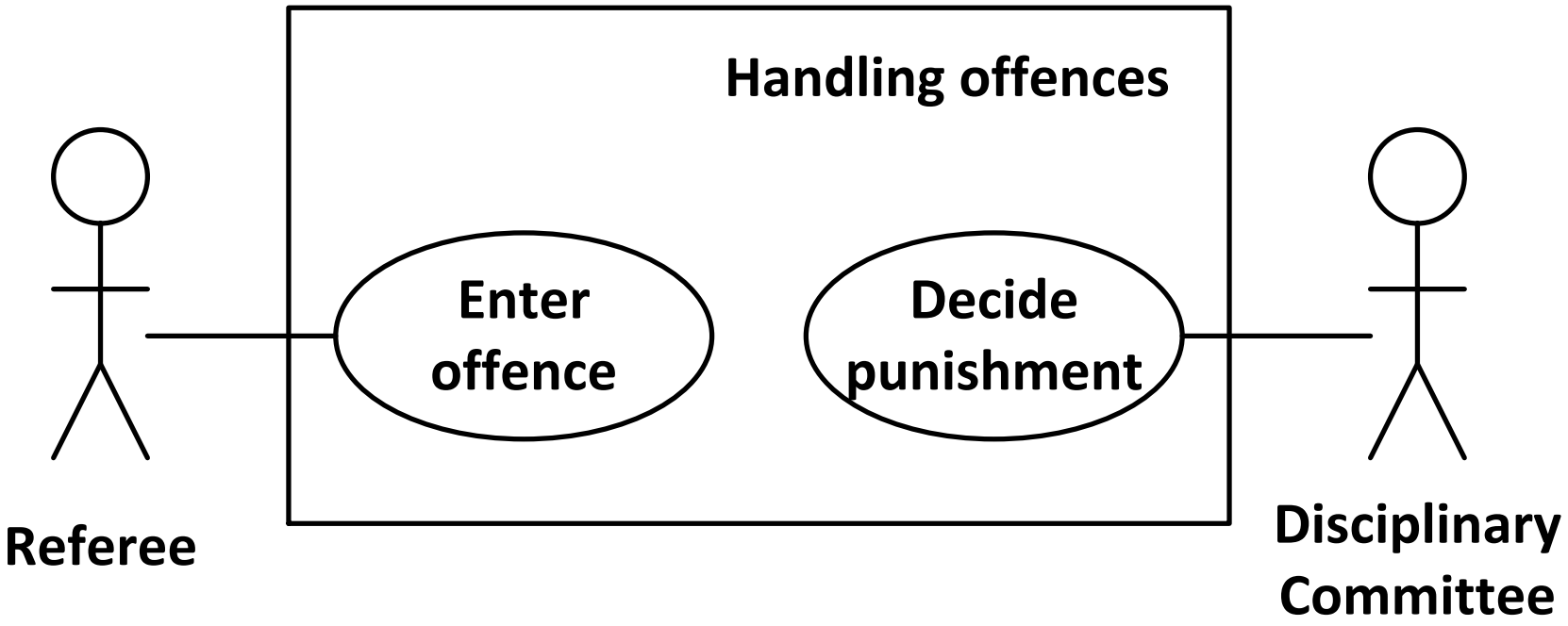
2.2. Requirements list

- The requirements list states requirements and their corresponding use cases
  - Please note how the requirements are written
    - Starting with an infinitive verb
    - Can be turned into a user story by adding
      "**As a <role> I want …**" as a prefix
  - (This is a widely used convention.
    If you want to write complete user stories, that is fine as well)

2.3. Actor list

Actor	*Description*
Referee	**Referees matches, gives warnings and yellow or red cards**
Disciplinary Committee	**Determines the punishment which a player will get after receiving a card, taking into account the player's previous behaviour**

2.4. Use case diagram



2.5. Use case descriptions (short)

Use case	*Description*
Enter offence	**After the match the referee enters a description of the offence and the type of card**
Decide punishment	**The Disciplinary Committee, taking the circumstances into account, decides on a suitable punishment for the offence**

2.6. Use case descriptions (extended)

The extended use case description (*see next slide*) gives a step-by-step description of how a use case is executed

short use case description:

Use case	*Description*
Enter offence	**After the match the referee enters a description of the offence and the type of card**

2.6. Use case descriptions (extended)

Use case description: **Enter offence**

Actor Action		*System Response*	
1	**Start application**	**2**	**Show data entry fields**
3	**Enter name of player and club**	**4**	**Show all details of player**
5	**Enter match details, Description of offence, and type of card**	**6**	**Save and send notification to the Disciplinary Committee**

Alternatives:

Step 4: Show list if it matches more than one person

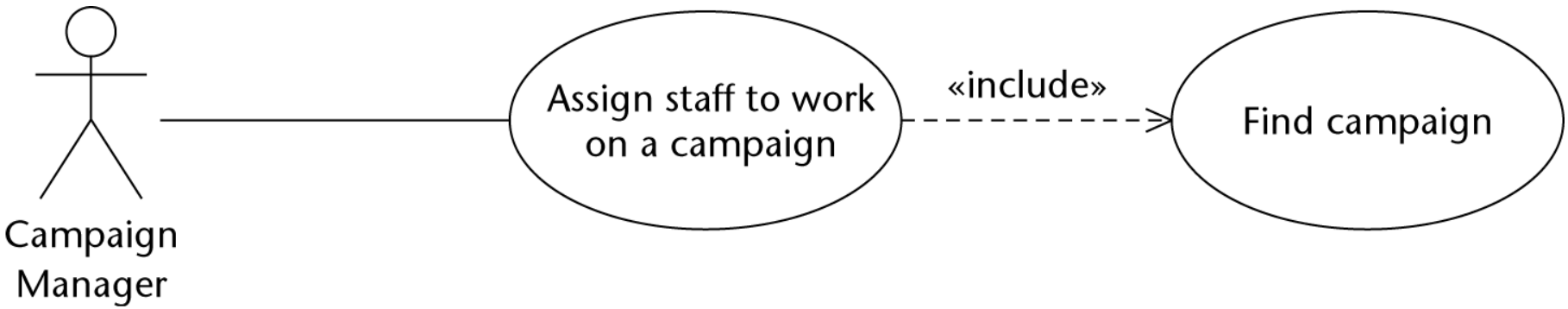
Stap 4A: Select person from the list

3. Use Case Diagram (extensions)

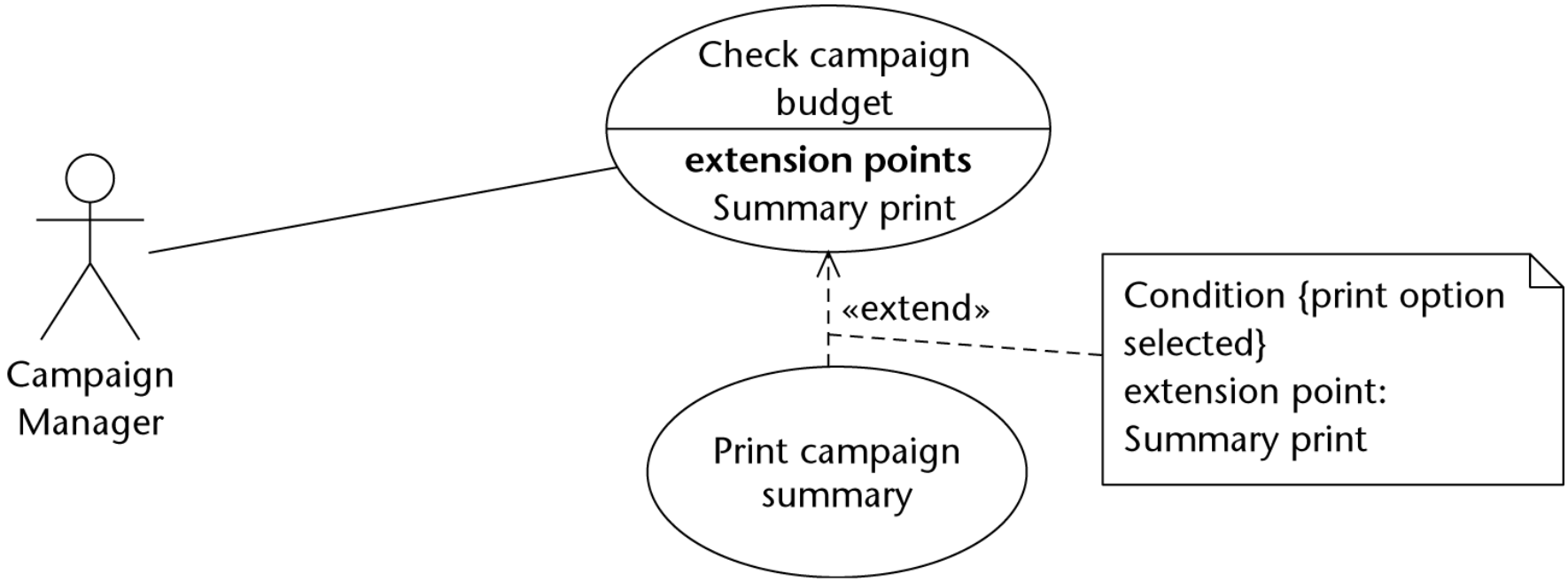
- Inclusion
- Extension
- Generalization
- (multiple use case diagrams for one system)

Examples from case study in Advertising

Inclusion

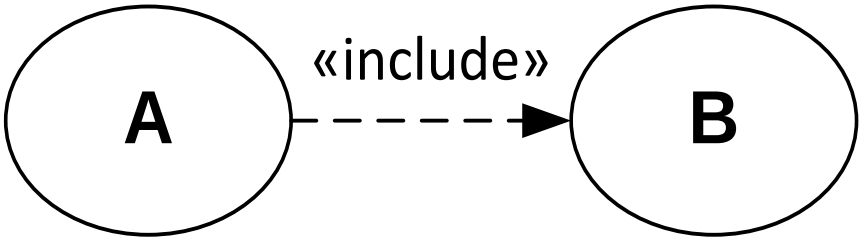

Campaign Manager
Assign staff to work on a campaign
«include»
Find campaign

Extension

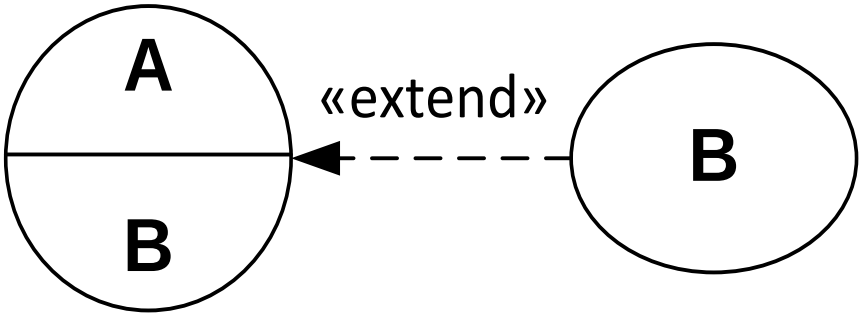

Check campaign budget
extension points
Summary print
Campaign Manager
«extend»
Condition {print option selected}
extension point:
Summary print
Print campaign summary

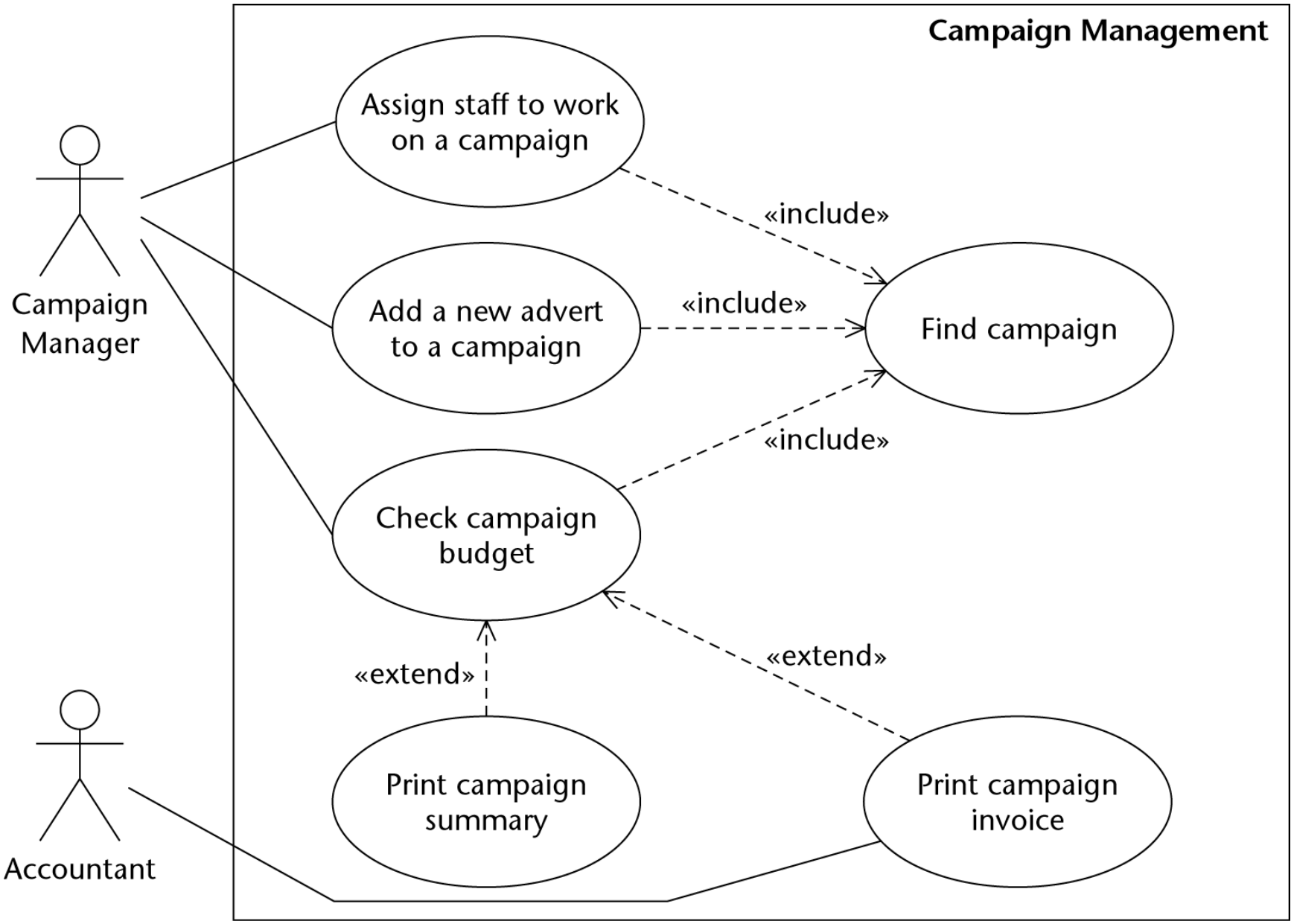
«include» and «extend»

- «include»:
  executing A always
  includes B

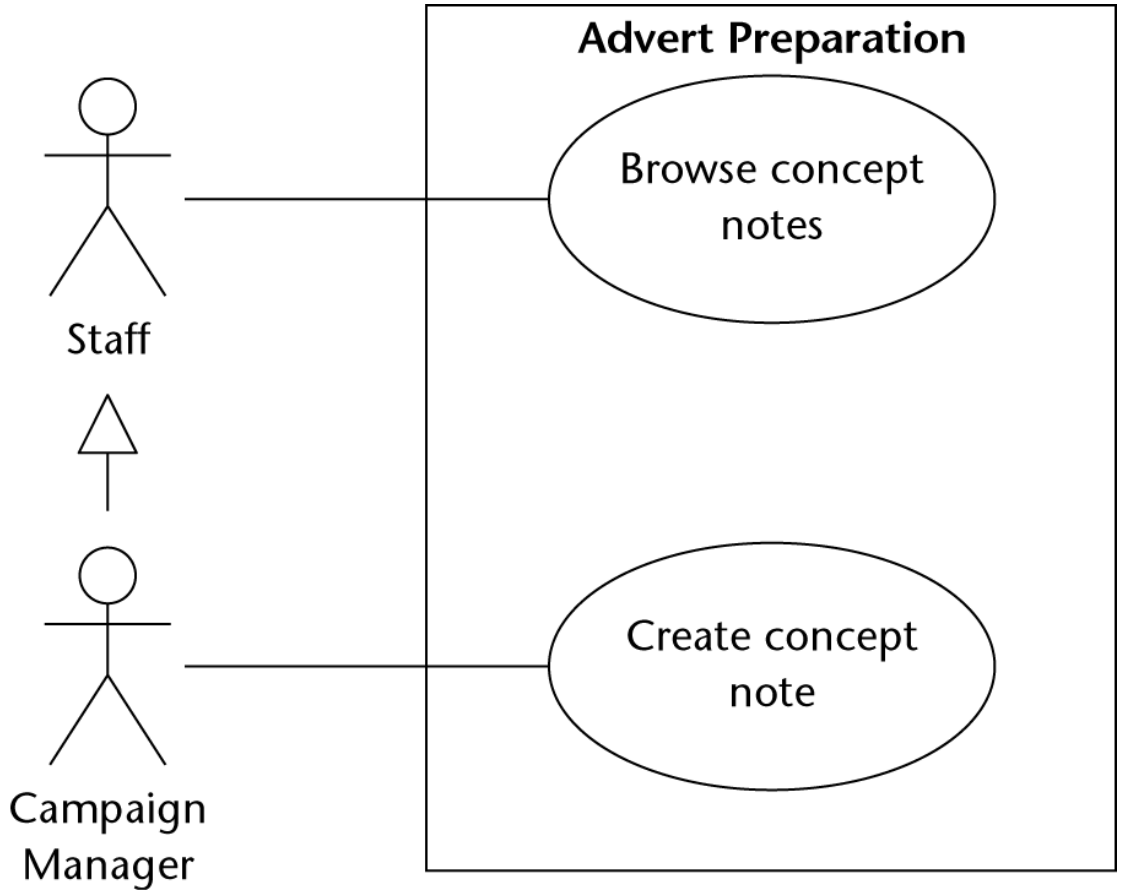


- «extend»:
  executing A could
  (but does not have to) include B




Campaign Management
Assign staff to work on a campaign
Add a new advert to a campaign
Check campaign budget
Print campaign summary
Print campaign invoice
Find campaign
«include»
«include»
«include»
«extend»
«extend»
Campaign Manager
Accountant

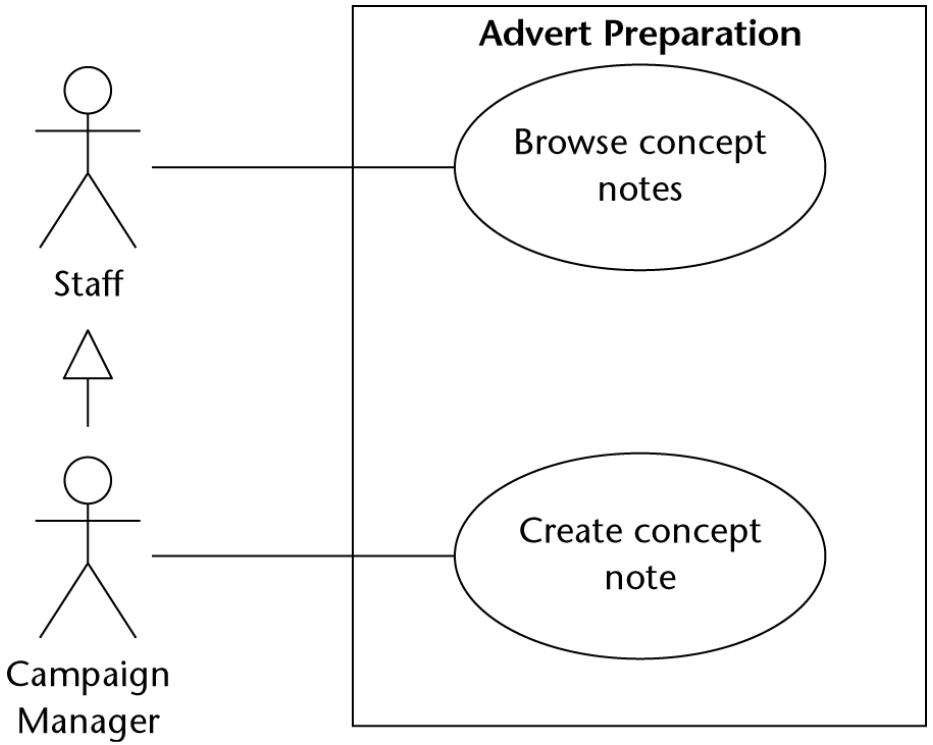
Generalization (if one actor does everything another actor does, plus more)



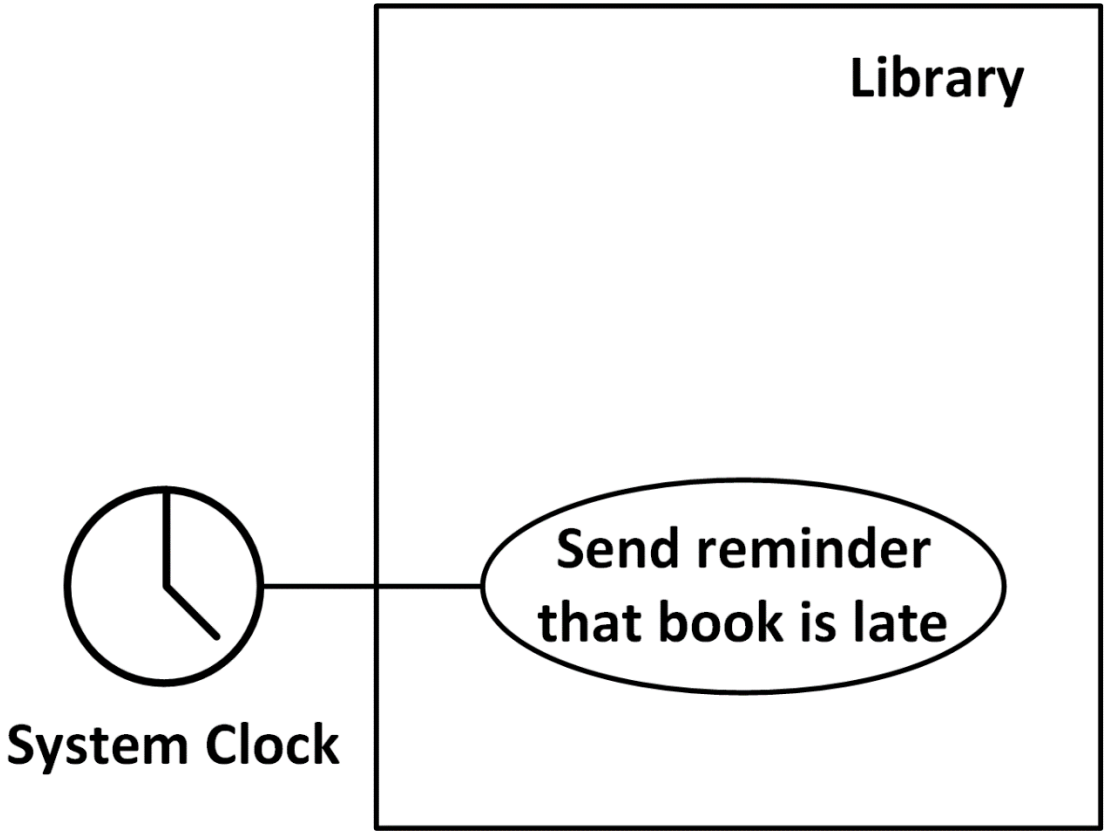
Generalization

- Every campaign manager is a staff member

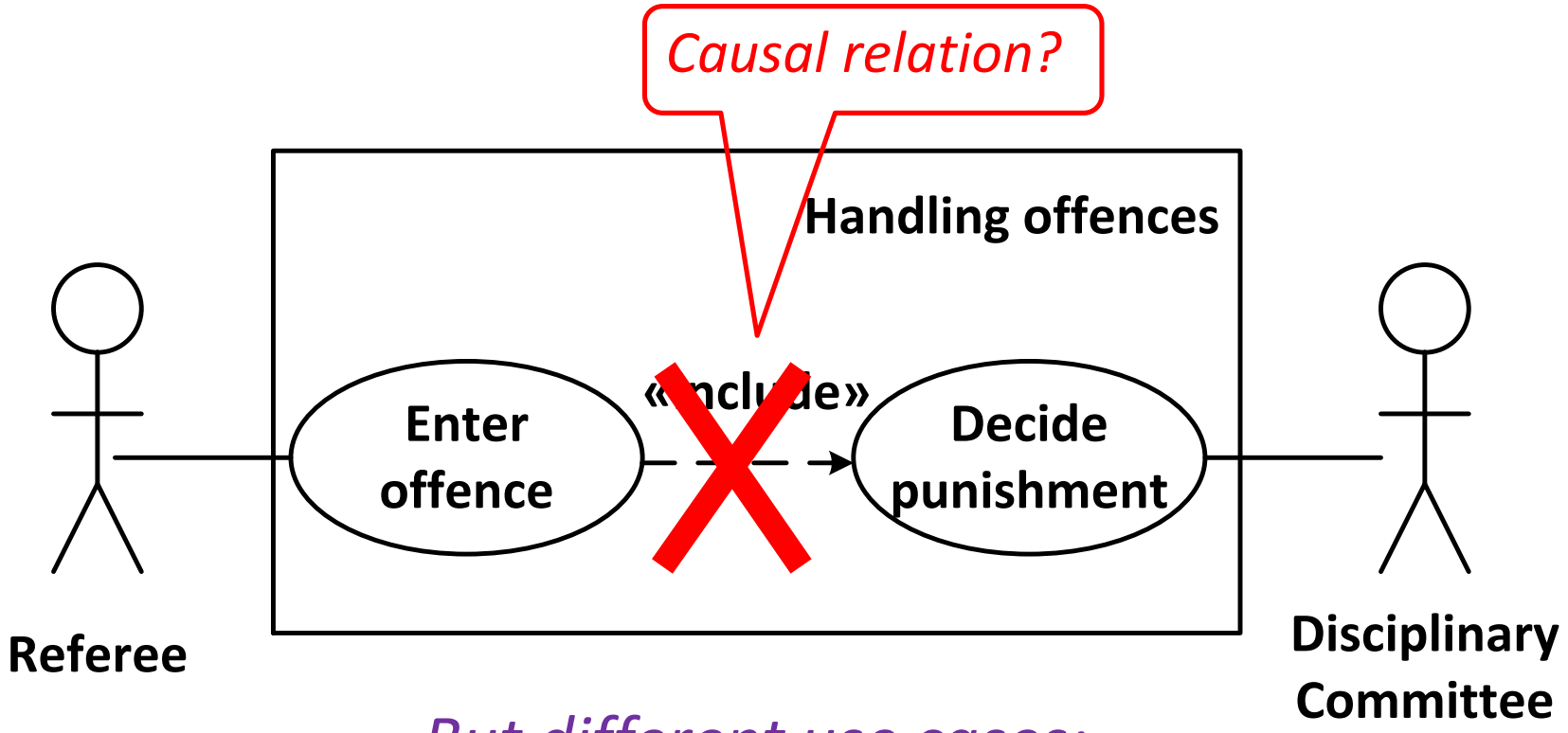
  (but not every staff member is a campaign manager)



Every use case has at least one actor



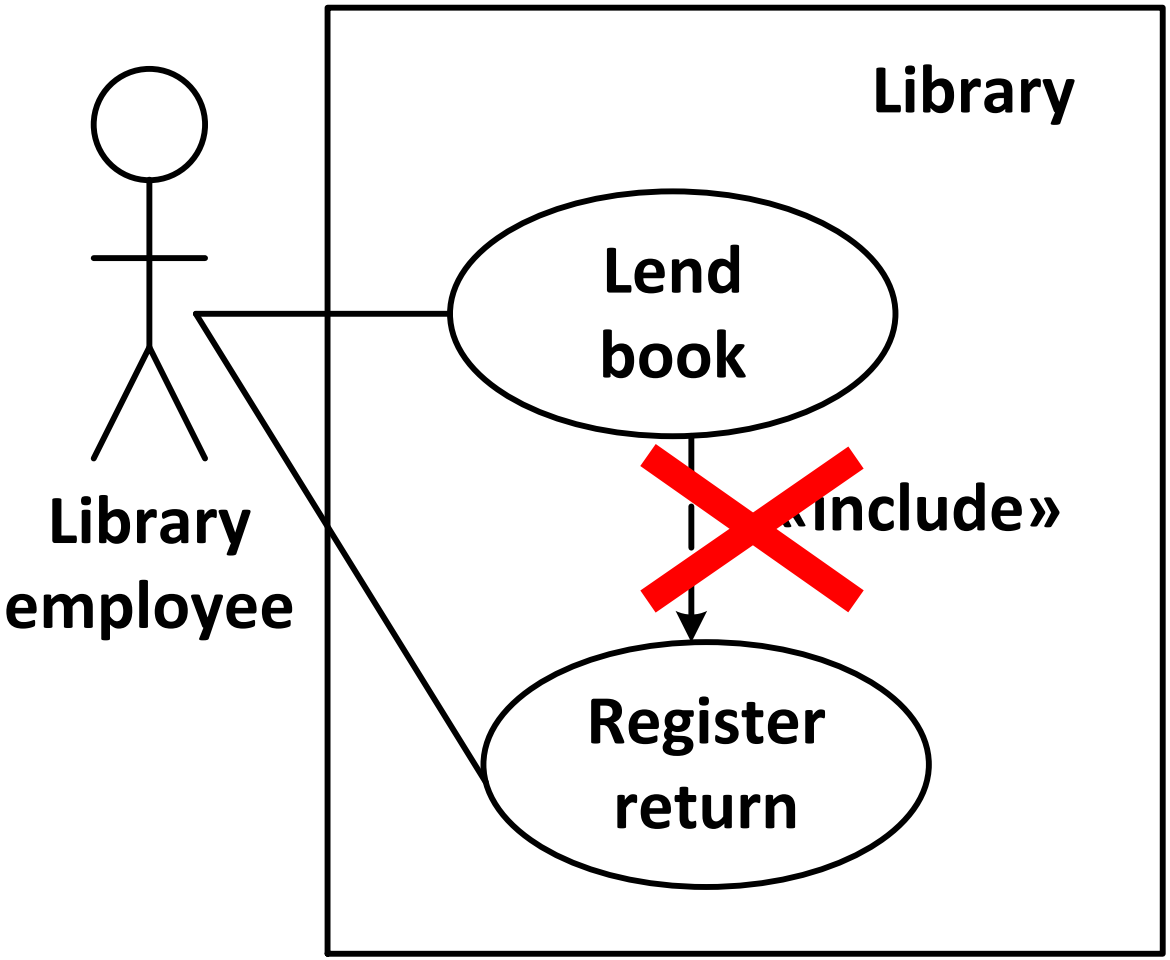
4. When not to use include/extend



But different use cases:

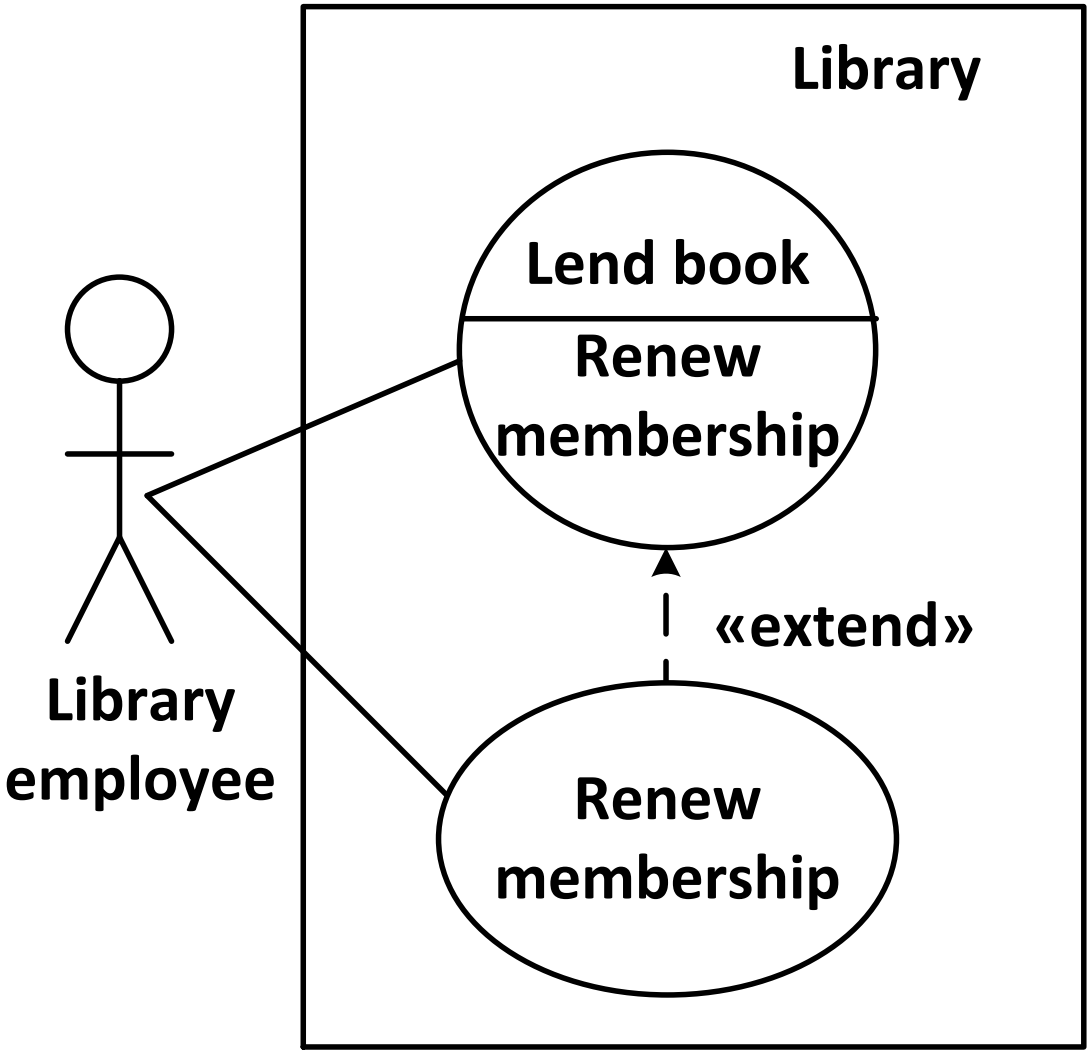
- *Different actor*
- *Different time*

4. When not to use include/extend



Idem

When it's OK to use include/extend



When during lending it turns out that membership has expired, it can be renewed immediately, so that lending can proceed

Guidelines for usage

- «include» and «extend» are used to indicate that executing one system function is / can be part of executing another system function

- «include» and «extend» are **not** used to describe causal relations between use cases (*such as: if interaction A takes place then interaction B must logically follow*)
  - these should be part of a **process model** (activity diagram) **not** of a functional model

Guidelines for usage

A use case describes one (set of) interaction(s) with the system, in a limited time frame.

- If the same process / object needs another interaction at a later time, that will be a different use case
  (or another invocation of the same use case)

Lab session today

- Exercise with <<include>> and <<extend>>

- Theatre tickets: basic version
  - Glossary, Requirements list, Actor list
  - Use case diagram

- Theatre tickets: extended version
  - Actor list, Requirements list
  - Use case diagram