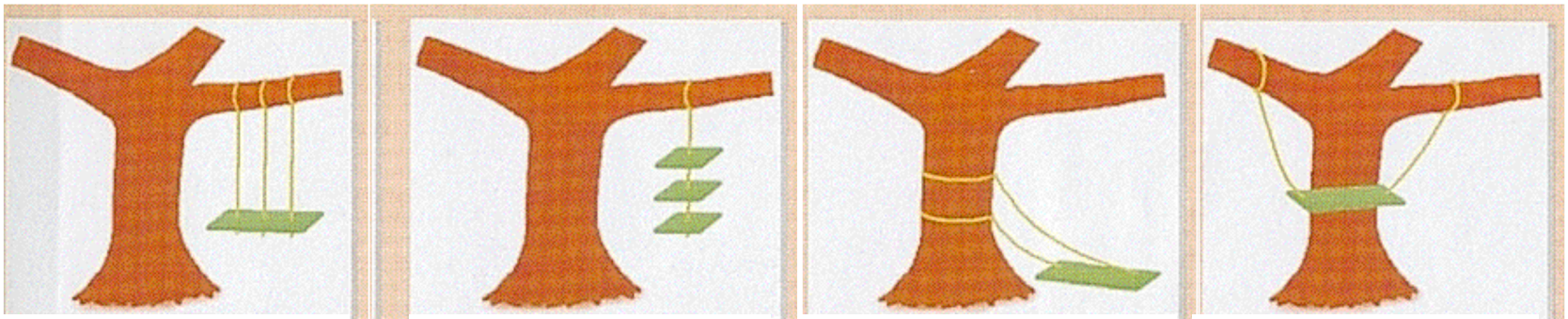
Requirements Engineering

Tutorial outline

- Requirements engineering
  - Basic concepts
- Requirements engineering process for ARENA
- REQuest: Live Demonstration
- REQuest: Guidelines
- Summary & next steps

Problem

Lifecycle of a software project



1.The client's needs

2. Requirements Analysis

3. After system design

4. After implementation

- Clients and developers often have different backgrounds and "speak another language"
- Requirements have to be fulfilled to make a project successful

Solution

Requirements Engineering

Clients

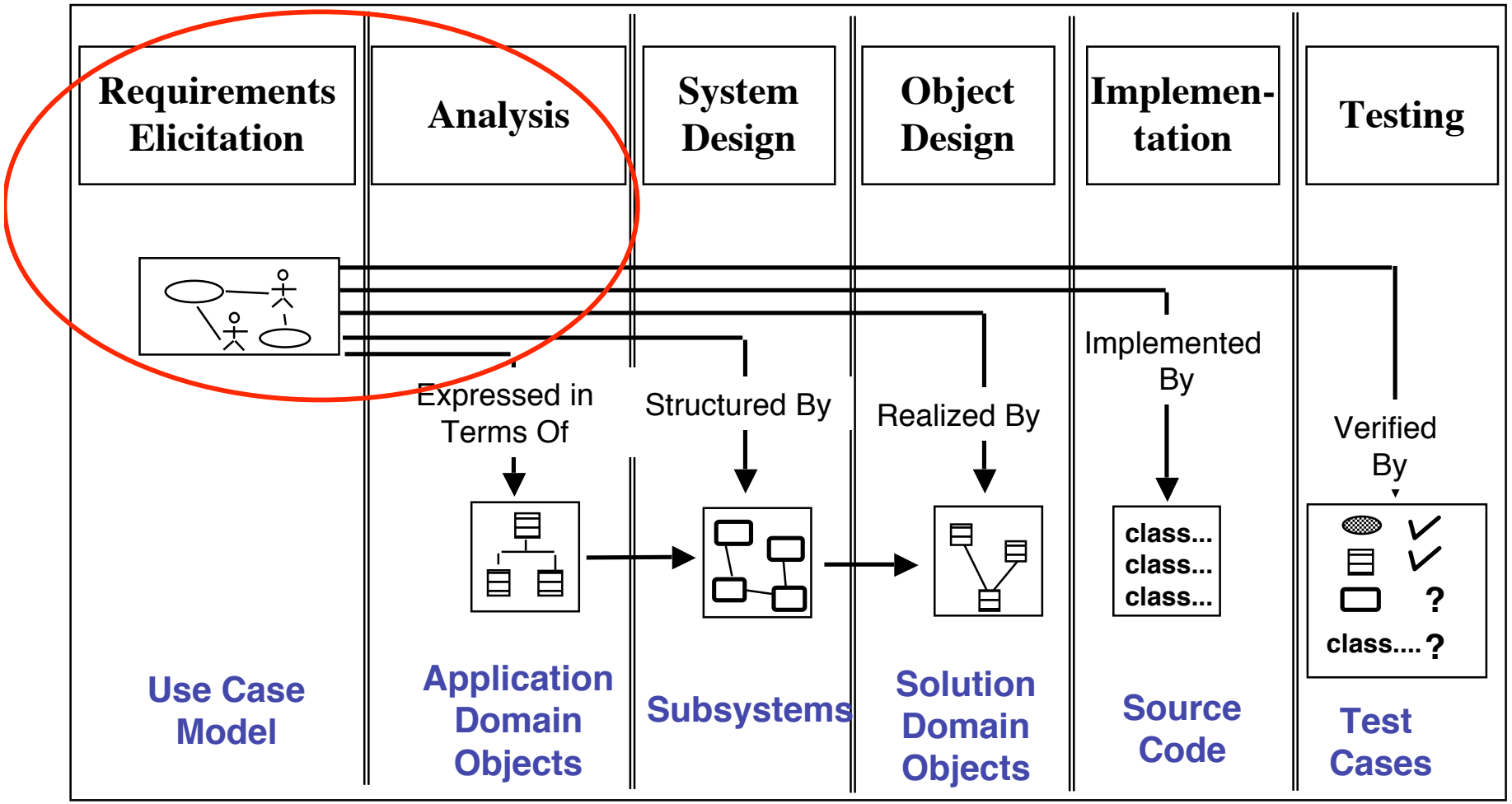
Requirements Analysis Document

Developers

Requirements Analysis Review

… can solve the communication problem …

Requirements Engineering



Requirements Analysis Document

A RAD includes 3 descriptions:

Requirements Elicitation:

Use case model

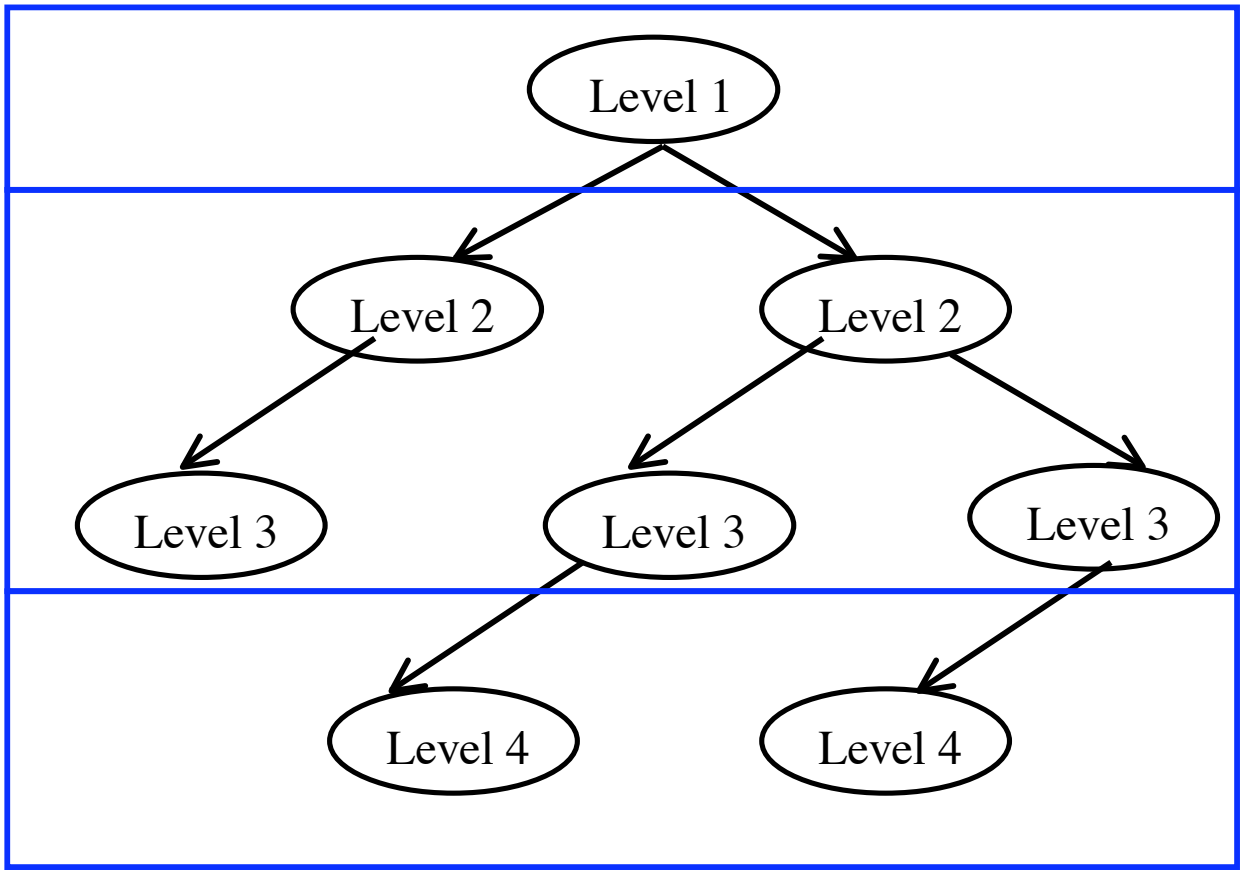
- *Requirements:* What do users do?
- *Interactions:* How do users use the system?

Analysis:

Requirements analysis model (object model)

- *Specification:* What does the system do?

RAD: Levels of descriptions



User tasks
describe domain

Use Cases
describe interactions

Services
describe system

Tutorial outline

- Requirements engineering
  - Basic concepts
- Requirements engineering process for ARENA
- REQuest: Live Demonstration
- REQuest: Guidelines
- Summary & next steps

Process for ARENA (1)

- REQuest for the requirements specification
  - Web-based tool
  - Actors, User Tasks, Use Cases, & Services
  - Constraints & Glossary

- REQuest for review and negotiation
  - Questions, Options, Criteria, Assessments
  - Discussion
  - What's new, what's revised, conflict detection

Process for ARENA (2)

Instructors/coaches

Oct 23: RAD v.0 from coaches

- Actors, user tasks & use cases

Nov 3: Feedback from coaches

Nov 6: Requirements review

Teams

Before Oct 31:

- Questions to coaches

Oct 31: RAD v.1 due

- Use cases, Services, Constraints
- Glossary

Nov 6: RAD v.2 due

- Options, assessments
- Decisions
- Revised requirements elements
- *Analysis Object Models and sequence diagrams (Together)*

Requirements elicitation activities (1)

- Define the boundary of the system:
  - Identify and describe actors
- Define the needs of the user
  - Describe one or more user tasks per actor
- Describe the interactions between the actors and the system
  - Describe one or more use cases per user task
  - Exceptions & nonfunctional constraints

Requirements elicitation activities (2)

- Describe the functionality of the system
  - Identify all services needed to realize the use cases
  - Each use case uses one or more services
  - Each service can be used by one or more use cases
- Review the system specification with the client

Tutorial outline

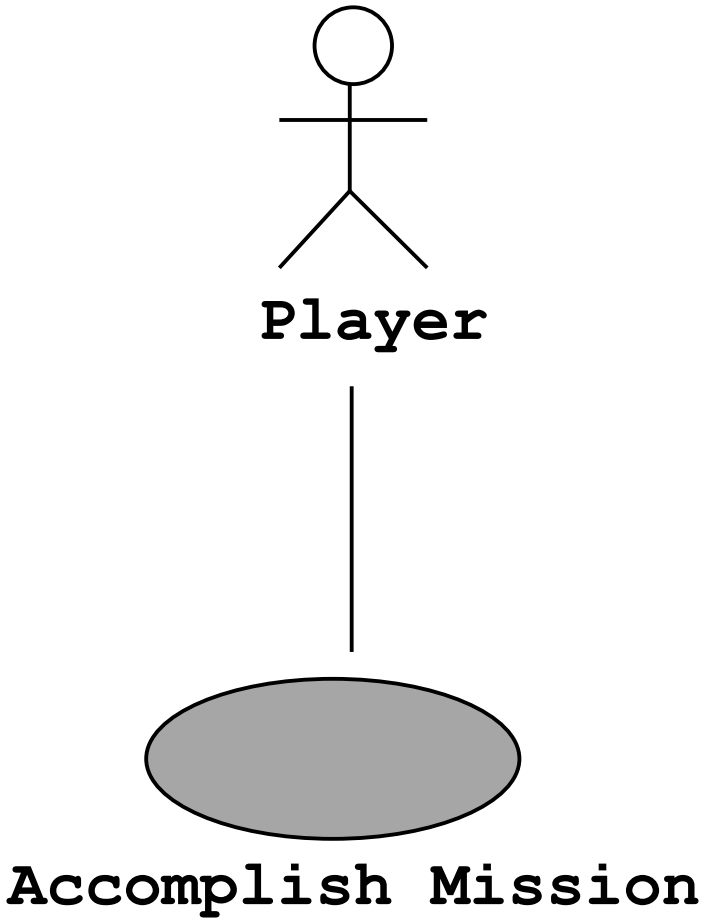
- Requirements engineering
  - Basic concepts
- Requirements engineering process for ARENA
- REQuest: Live Demonstration
- REQuest: Guidelines
- Summary & next steps

Live Demonstration

- First step: login to REQuest

http://sysiphus.in.tum.de:8080/arena02/servlet/SYSLogin

Requirements: What do users do? (1)



- Brief high-level descriptions
- ***Actors*** represent roles, that is, a type of user of the system
  - Player
- ***User tasks*** represent activities accomplished by the user, independently of the system.
  - Accomplish Mission

Requirements: What do users do? (2): Examples

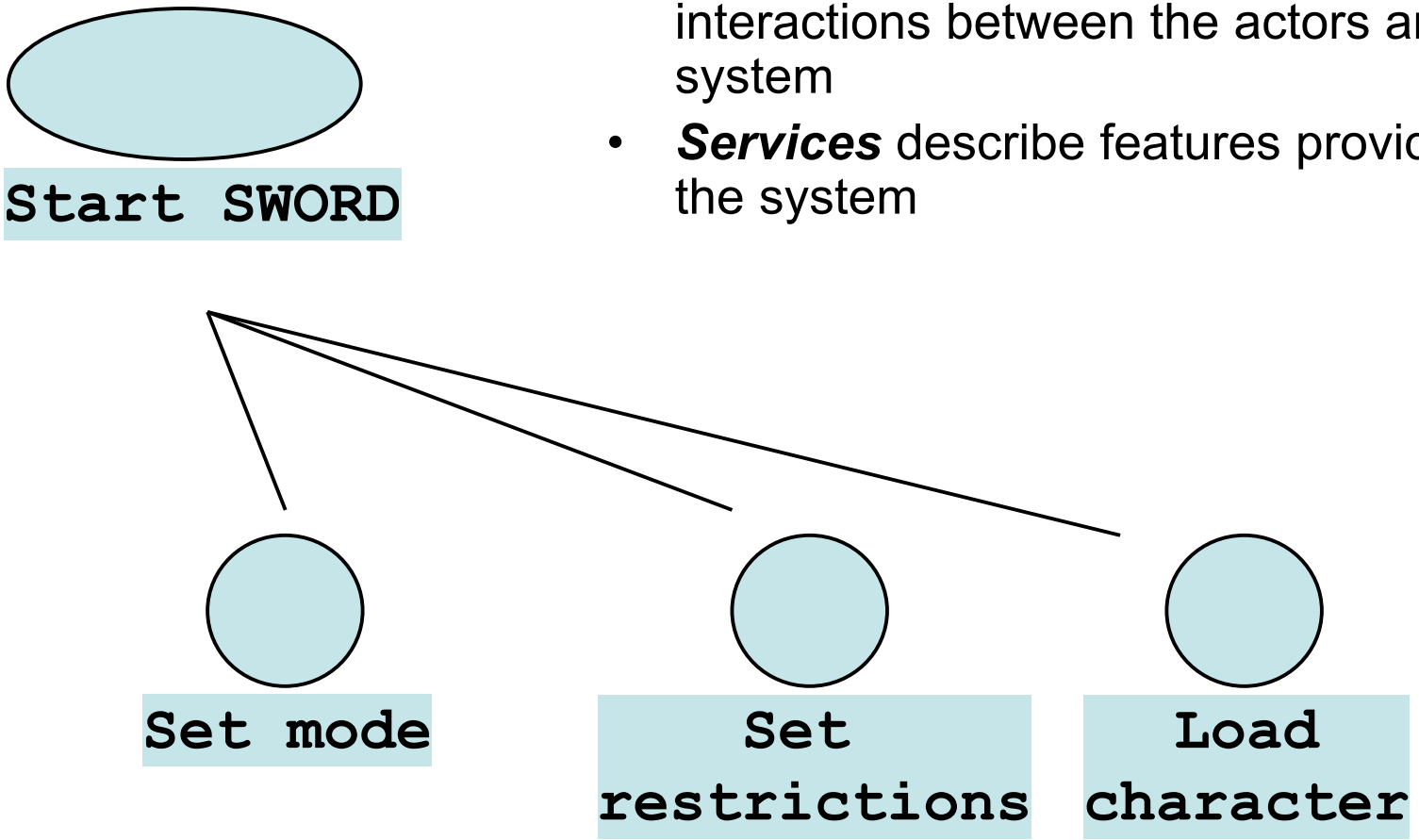
Actor **Player**

- Person who is able to play one or more games.

User Task **Accomplish Mission**

- The Player starts the game.
- The Player sets her/his preferences.
- The Player receives a certain mission.
- The Player completes the mission.

Specification (1) : What does the system do?



- ***Use cases*** describe sequences of interactions between the actors and the system
- ***Services*** describe features provided by the system

Specification (2): Example of use case attributes

Use Case **Start SWORD**

Initiatiating actor:

- Player

Preconditions:

- Player has installed SWORD on her/his computer.

Postconditions:

- Player is able to enter the game.

Specification (3): Example of use case flow of events

Actor steps	System steps
1. The Player double clicks the SWORD icon on her/his computer	
	2. SWORD asks for the preferred game mode
3. The Player chooses the stop-watch mode and sets a deadline	
	4. SWORD asks if the player wants to set any restrictions
5. The Player restricts the game to her/his buddy list	
	6. SWORD loads the Player's character
7. The Player can enter the game	

Specification (4): Example services

Service **Set mode**

- Inputs: one game mode and deadline
- Output: message asking for restrictions

Service **Set restrictions**

- Input: one or more players (from menu)
- Output: message that restrictions are set

Specification (5): Exceptions

Actor steps

1. The Player double clicks the SWORD icon on her/his computer.

3. The Player chooses the stop-watch mode and sets a deadline. **[invalid format]**

5. The Player restricts the game to her/his buddy list. **[no buddy list defined]**

7. The Player can enter the game

[invalid format]
SWORD displays a message box and asks to use the valid format for setting deadlines.

[no buddy list defined]
SWORD announces the failure and offers the possibility to set the buddy list now as well as canceling this step. If the Player chooses the first option, a window will pop up so that the Player can compile her/his buddy list.

Nonfunctional requirements

Domain constraints

- Domain facts
- Applicable to user tasks

Global functional constraints

- Functionality that is easier to describe in terms of constraints
- Applicable to use cases

Quality constraints

- Constraint on the attribute of a user task, use case, or service.

Tutorial outline

- Requirements engineering
  - Basic concepts
- Requirements engineering process for ARENA
- REQuest: Live Demonstration
- REQuest: Guidelines
- Summary & next steps

Guidelines for use cases (1)

Name

- Use a verb phrase to name the use case.
- The name should indicate what the user is trying to accomplish.
- Examples:
  - “Request Meeting”, “Schedule Meeting”

Length

- A use case should not exceed 2 A4 pages. If longer, use *include* relationships.
- A use case should describe a complete set of interactions.

Guidelines for use cases (2)

Flow of events

- The active voice should be used. Steps should start either with “The Actor …” or “The System …”.
- The causal relationship between the steps should be clear.
- All flow of events should be described (not only the main flow of event).
- The boundaries of the system should be clear. Components external to the system are described as such.
- Define important terms in the glossary.

Guidelines for use cases (3)

Exceptions

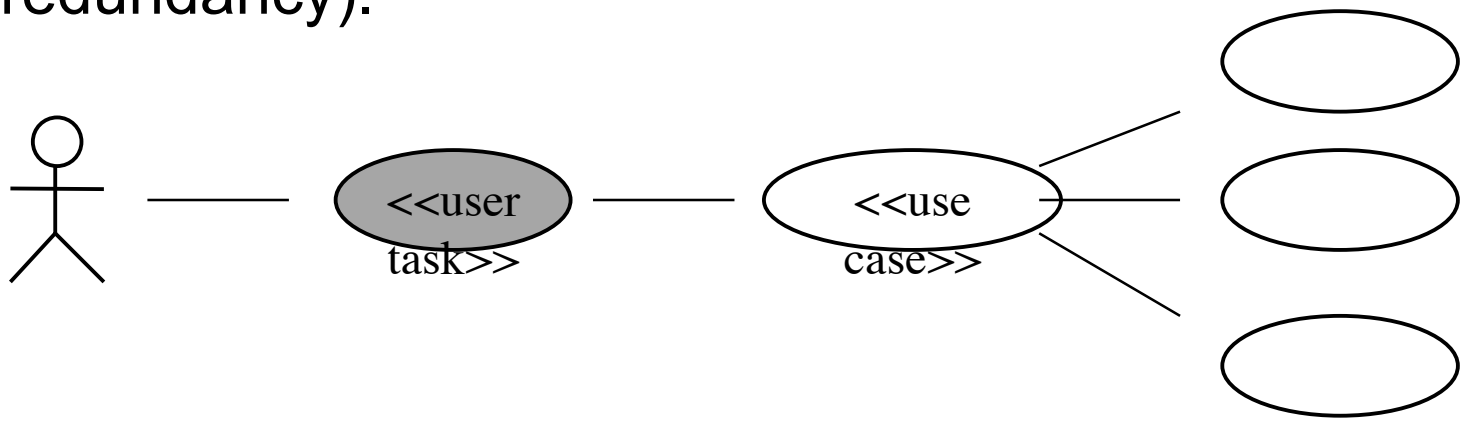
- Exceptions should be attached to the step where they are detected.
- If an exception can occur in any step, describe it only in the exception section.
- Exception handling is described as flow of events.
- At the end of the exception handling, it should be clear what happens next (if the use case is terminated or if it is resumed in a particular step).

Preconditions

- If a case is excluded with a precondition, then it should not be handled as an exception.

Guidelines for use cases (4)

- Write one high-level use case per user task
- If a use case includes only one or two steps, it should probably be a service, not a use case.
- If a sequence of steps is identical in several use cases, it should be factored out into a separate use case and included in the original use cases (eliminate redundancy).



General guidelines: Use Rationale (1)

Question: Which restrictions are possible?

References: Service: Set restrictions

Decision: Buddy list + single persons

	Criteria 1: Flexibility	**Criteria 2:** User Friendliness
Opt. 1: Buddy list	–	+
Opt. 2: Single persons	+	–
Opt. 3: Buddy list + single persons	+	+

Use Rationale (2)

Questions can be used to:

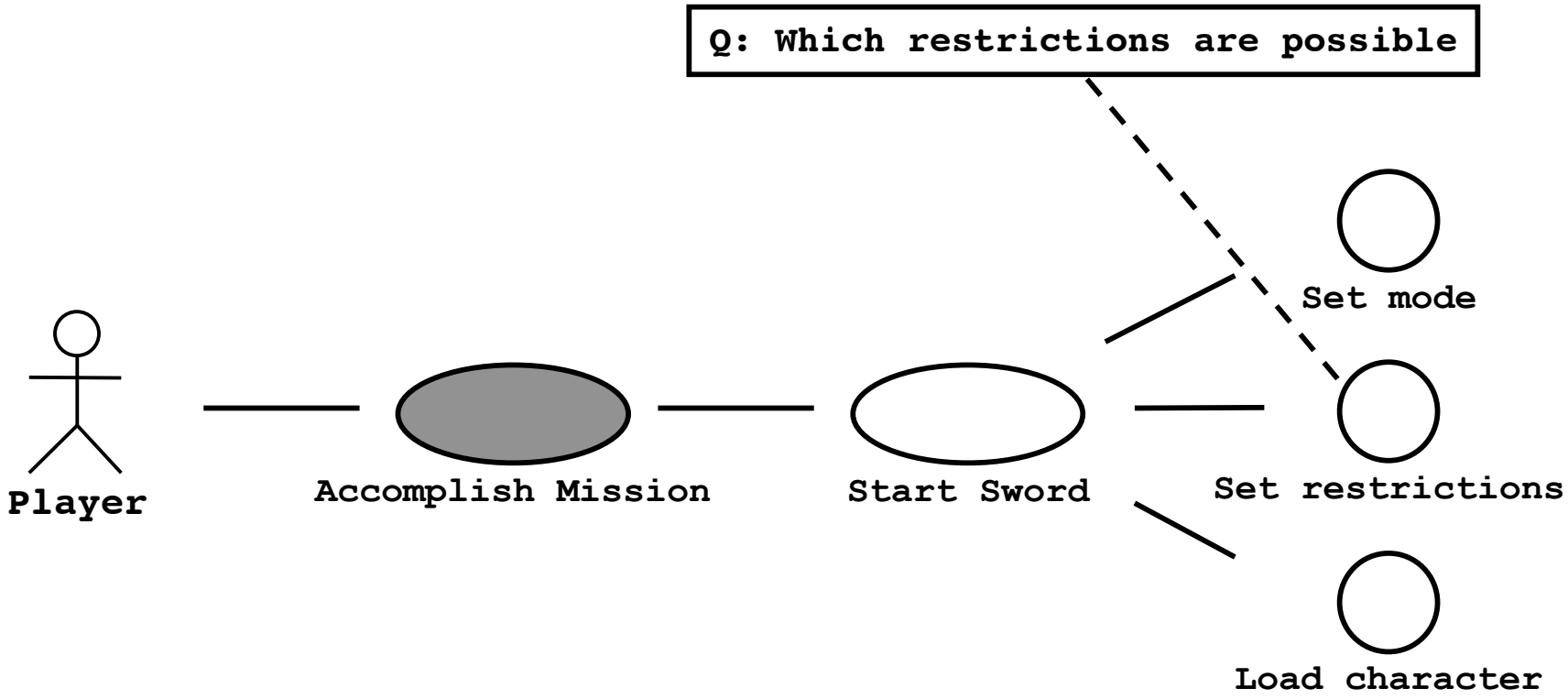
- Request a clarification — *How can a Player restrict a game to her/his buddy list?*
- Indicate a defect — *Isn't a second game mode missing?*
- Justify a use case or service — *Which solution is the best?*

Questions are asked during review and consolidated into justifications during revisions.

Tutorial outline

- Requirements engineering
  - Basic concepts
- Requirements engineering process for ARENA
- REQuest: Live Demonstration
- REQuest: Guidelines
- Summary & next steps

Putting everything together



Summary

- REQuest supports
  - Definition of requirements specification
  - Questions about the requirements elements
  - Discussion, negotiation, and resolution of questions
  - Finding out what others have done

- ARENA deadlines
  - RAD v.1 October 30
  - RAD v.2 November 6

Important: Process for ARENA (1)

Instructors/coaches

Oct 23: RAD v.0 from coaches

- Actors, user tasks & use cases

Nov 3: Feedback from coaches

Nov 6: Requirements review

Teams

Before Oct 31:

- Questions to coaches

Oct 31: RAD v.1 due

- Use cases, Services, Constraints
- Glossary

Nov 6: RAD v.2 due

- Options, assessments
- Decisions
- Revised requirements elements
- *Analysis Object Models and sequence diagrams (Together)*

Next steps

- Meet with your development team as soon as possible
- Login to REQuest with your Lotus Notes account
- Add actors, use cases and services
- Discuss, negotiate and resolve questions
- Write one complete scenario per team (that is one possible example of the system in use)

Further readings

- Bruegge B., Dutoit A.: Object-Oriented Software Engineering: Conquering Complex and Changing Systems, Prentice Hall, 2000
- REQuest online help

Questions?