Software Engineering with UML

Modeling systems with the UML

Requirement modeling with UML Use Case Diagrams

Organisation of the course (1/4)

- Week 1 : Introduction to SE with UML
  - Part 1 : Course
  - Part 2 : Exercises/ Lab work
- Week 2 : Requirement analysis: Use case diagrams
  - Part 1 : Course
  - Part 2 : Exercises/ Lab work
- Week 3 : Structural modeling class diagrams + design pattern
  - Part 1 : Course
  - Part 2 : Exercises/ Lab work
- Week 4: Structural modeling : Lab work and work on project.
  - Part 1 : Exercises/ Lab work
  - Part 2 : Work on the project.

•

Organisation of the course (2/4)

- Week 5 : Sequence diagrams + state machines
  - Part 1 : Course
  - Part 2 : Exercises/ Lab work
- Week 6 : Sequence diagrams + state machines: Lab work
  - Part 1 : Exercises/ Lab work
  - Part 2 : Work on the project
- Week 7 : Code generation & reverse engineering
  - Part 1: Course
  - Part 2: Exercises/ Lab work
- Week 8 : Software Testing
  - Part 1 : Course
  - Part 2 : Exercises/ Lab works

Organisation of the course (3/4)

- Week 9 : Software Product Lines
  - Part 1 : Course
  - Part 2 : Exercises/ Lab work

Organisation of the course (4/4)

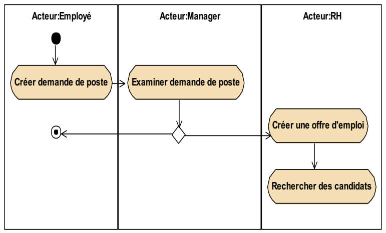
Evaluation

- Intermediate Written Test (Week October 16) ==> (20%)
- Long Mini Project (30%)
- Final Exam (50%)

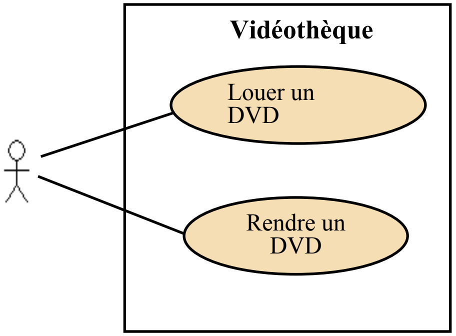
Long Mini Project (using the Modelio UML tool)

- Applying the concepts to develop a system InterimECE (the specification document is available on the campus web page).
- Groups of 2 students (maximum 3).
- One final report (deadline before Week 8).

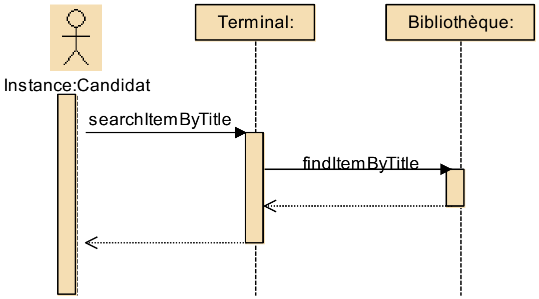
The UML Is a Language for Documenting



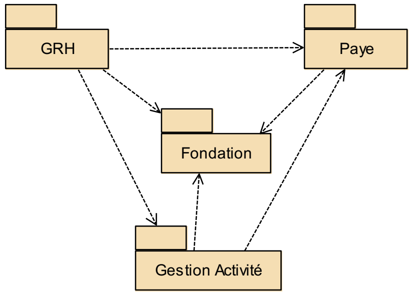
Activity diagrams



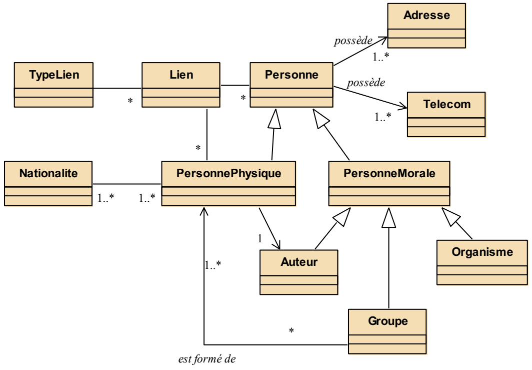
Use case diagrams



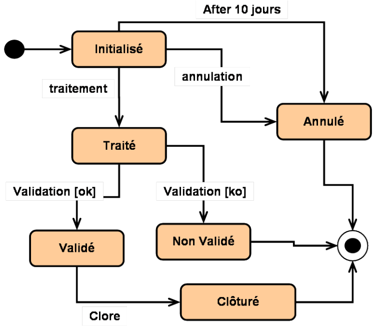
Sequence diagrams



Package Diagrams



Class Diagrams



State machine diagrams

Contents

- Introduction to requirements modeling
- Use case modeling – UML Use case diagrams
  - Writing use cases

Requirements modeling

The purpose of Requirements modeling is to:

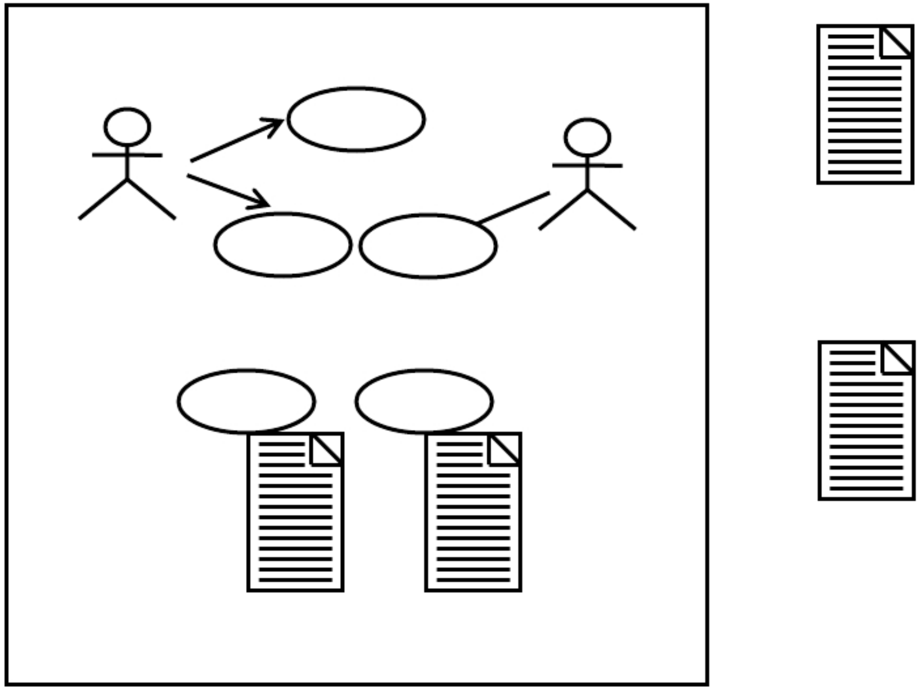
- Establish and maintain agreement with the customers and other stakeholders on what the system should do.

- Delimit the system.

- Define the functionalities of the systems.

- Define a user interface of the system.

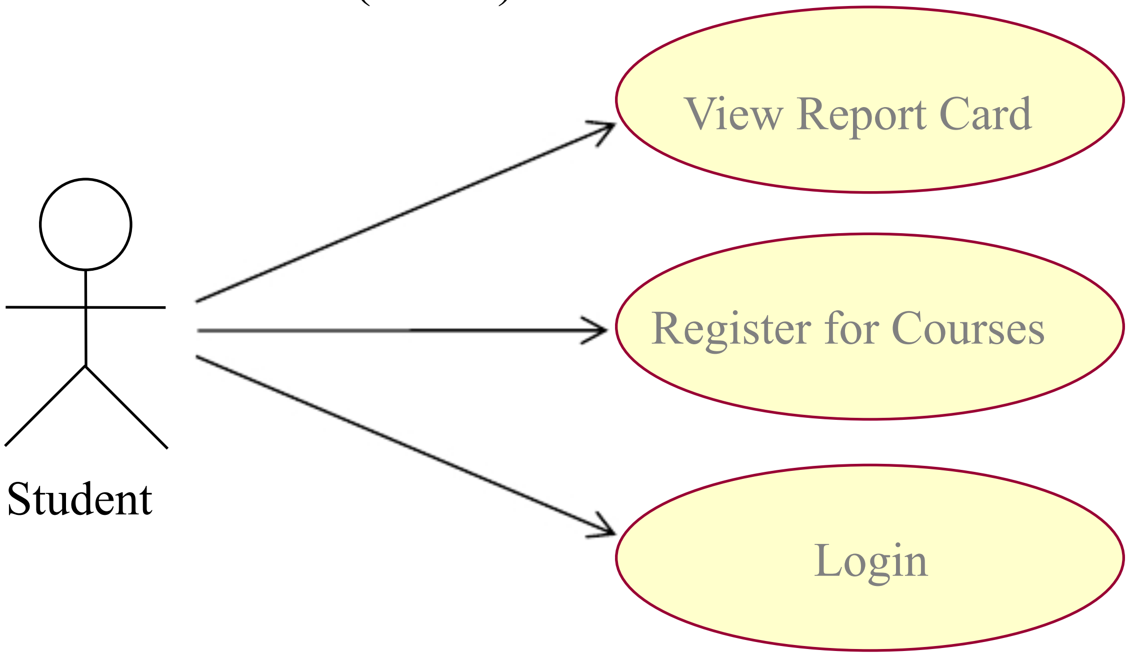
Requirements Artifacts



Use-Case Model

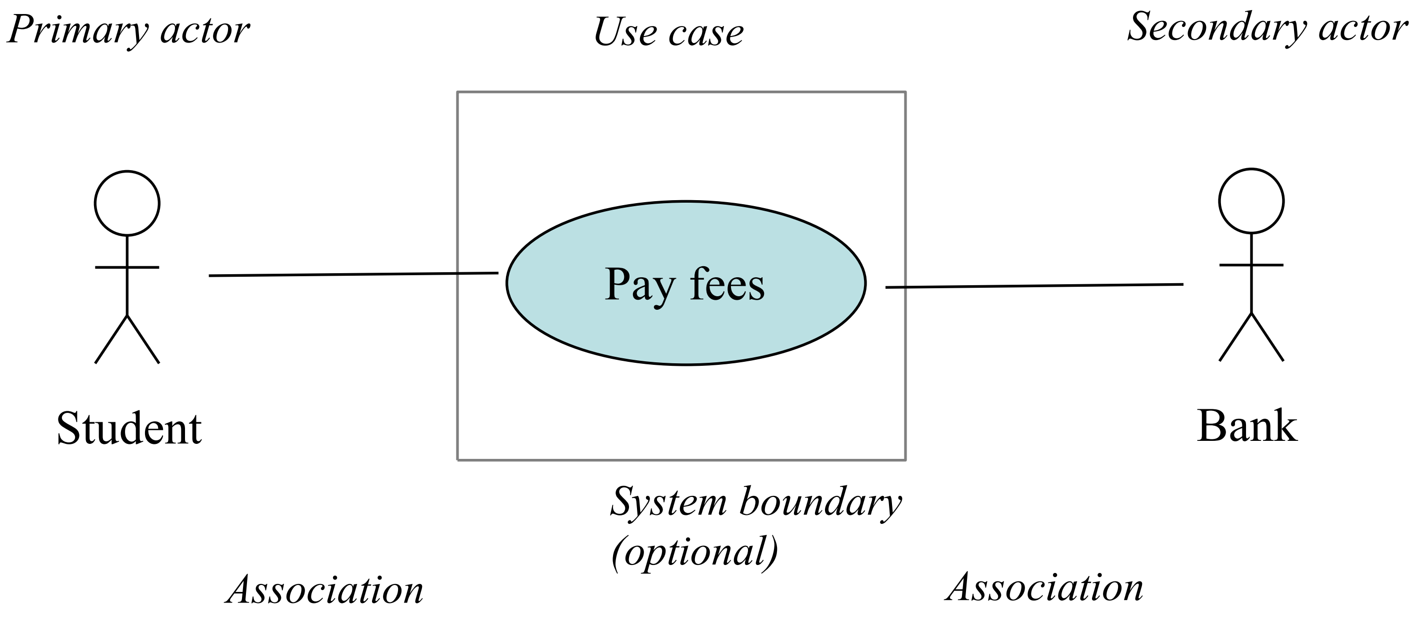
Use Case Model

- A model that describes a system's functional requirements in terms of use cases
- A model of the system's intended functionality (use cases) and its environment (actors)



Use case diagram

- A UML use case diagram provides an overview of use cases and actors pertinent to your business domain.

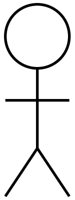


Use case diagram

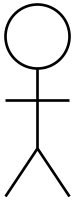
- A **use case** describes a sequence of actions that provide measurable value to an actor.
  - Horizontal ellipse, name of use case inside or below
- An **actor** is a person, organization or external system that plays a role in one or more interactions with your system.
  - Stick figure
- An **association** between a use case and an actor indicates that the actor is involved in the interaction described by the use case.
  - Solid line connecting actor to use case
  - May include an arrow head on one end of the line to indicate the direction of the initial invocation of the relationship and/or the primary actor. *Does not indicate information flow*.
- System **boundary box** (optional)
  - A rectangle around your use cases to explicitly indicate the scope of your system
  - Usually redundant as the set of defined use cases implicitly defines system scope

- **Actor**
  - An external entity that interacts with the system
    - **Uses** one or more functionalities

      ➔ Principal actor
    - **Participates** in the realization of the functionalities
      - ➔ Secondary actor



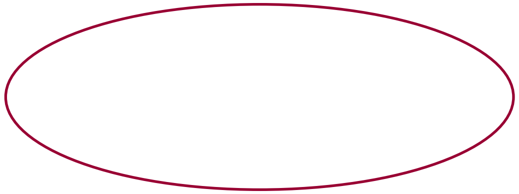
Identifying actors



- To help find actors in your system you should ask yourselves the following questions:
  - Who is the main customer of the system?
  - Who obtains information from this system?
  - Who provides information to the system?
  - Who installs the system?
  - Who operates the system?
  - Who shuts down the system?
  - What other objects intract with the system?
  - Who will supply, use, or remove information from the system?
  - Where does the system get information?
- The system itself can also be an actor
  - For tasks that happen regularly or at a preset time
  - Indicated with an no actor, or an actor named "system"

Use Case Models: Concepts

- A Use Case
  - One functionality of the system
  - A set of actions
- Links and Relationships
  - Inheritance between actors



Use Case Diagrams

How to identify use case ?

- What are the functionalities proposed by the system?
  - Each functionality is represented as a use case.
- What are the interactions Actor-System?

  For each identified actor

  - Identify the functionalities used by this actor.

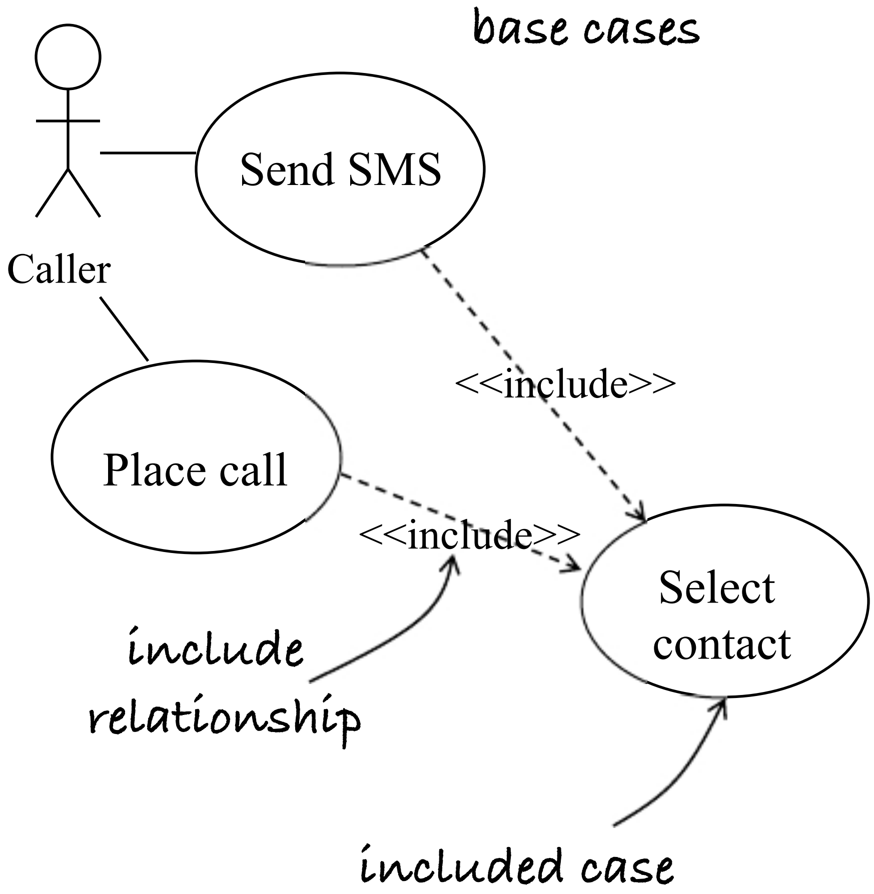
include relationship

UC01: Send SMS

Objective : Send a text message to another GSM user

Basic course

1. Caller selects "send SMS" function
2. Caller types in text message
3. Caller selectes contact
   include UC 04 Select contact
4. Caller presses "send"
5. GSM phone asks Network to deliver the message (text, GSM number)
6. Network signals successful delivery.
7. GSM phone displays "SMS delivered".



extend relationship

<u>UC01 : Send SMS</u>

Objective : Send a text message to another GSM user

<u>Basic course</u>

1. Caller selects "send SMS" function

2. System displays message entry field.

2. Caller types in text message
   (extended by UC10: Load saved message)

3. …

<u>UC10 : Load saved message</u>

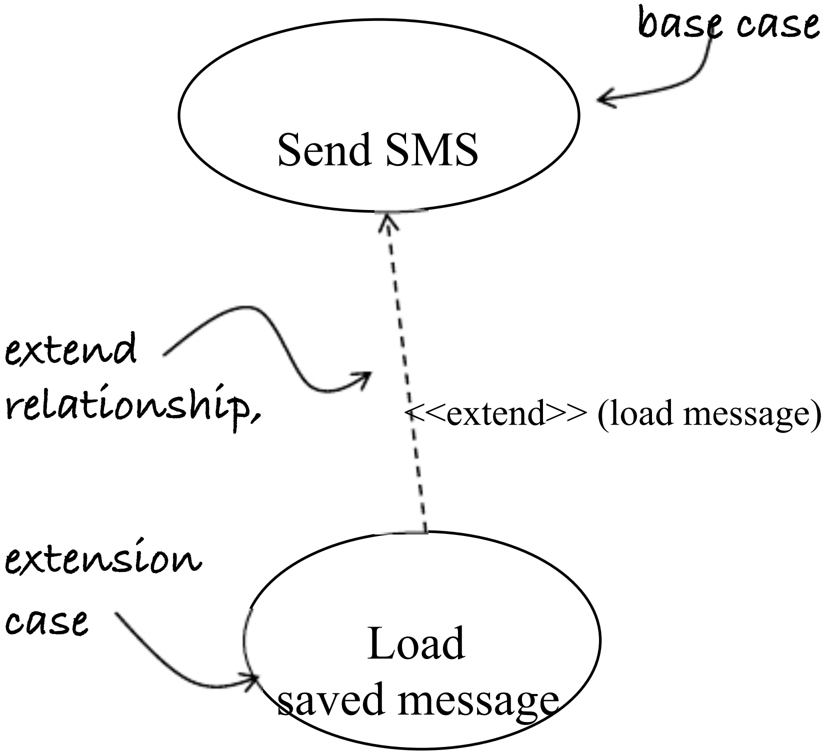
<u>Basic course</u>

1. Caller selects »Insert saved message"

2. System displays a list of saved messages

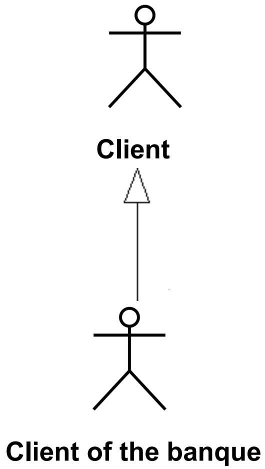
3. Caller selects a message

4. System enters saved message in message entry field

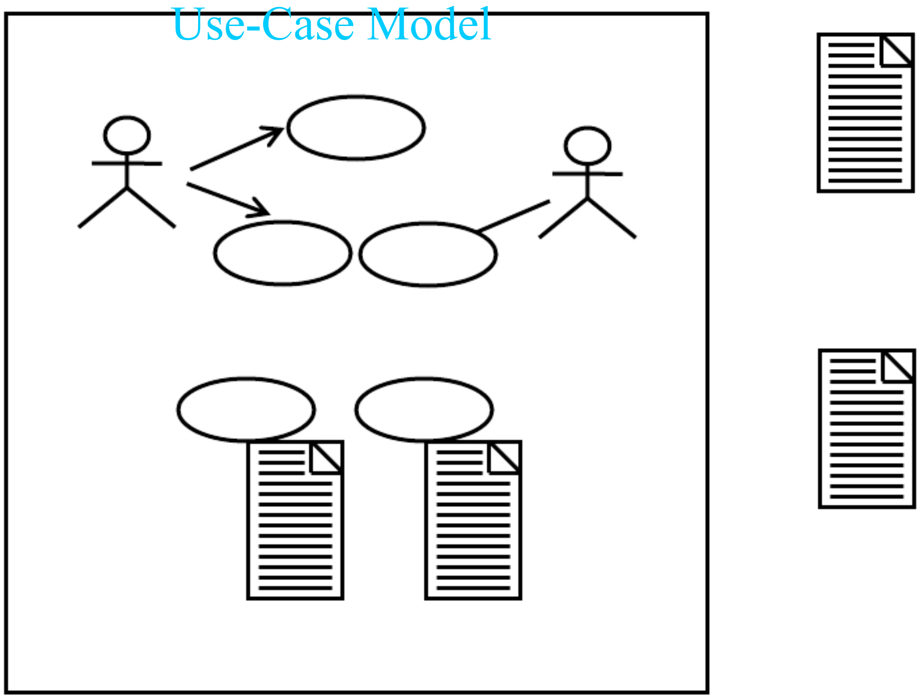


generalisation relationship

- Used in to specify inheritance between actors.

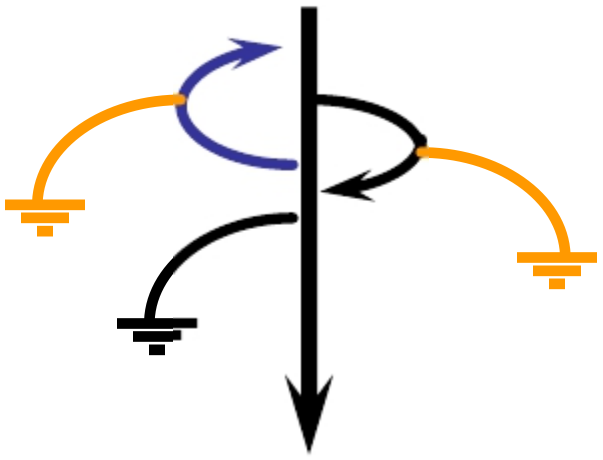


Requirements Artifacts



Use-Case Description

- For each use case we can add a detailed description to specify different flows



Use case example

Name: Enroll in seminar

Identifier: UC03

Description: Enroll an existing student in a seminar for which she is eligible. In case of a schedule conflict or insufficient prerequisites, she will not be allowed to enroll.

Preconditions: The student is registered at the university

Postconditions: The student is enrolled in the seminar or the system is unchanged.

Basic course of action:

1. The UC begins when a student indicates they want to enroll in a seminar.
2. The student identifies him or herself. Include UC04 « Connect to the system »
3. The system verifies that the student has not yet reached the number of seminars he has paid for.
4. The system checks the seminars that the student has already followed, displays the list of seminars for which he satisfies the pre requisites.
5. The student selects the seminar in which he wants to enroll.
6. The system displays the sudent's current timetable superimposed with the new seminar.
7. The system asks if the student still wants to enroll.
8. The student confirms.
9. The system updates the student's timetable with the new seminar and publishes it.
10. The system asks if the student wants a printed copy of his new timetable.
11. The student indicates yes.
12. The system prints the new timetable for the semester.
13. The student takes the timetable
14. The UC ends when the student takes his timetable.

Use case example (continued!)

Name: Enroll in seminar

Identifier: UC03

Description: Enroll an existing student in a seminar for which she is eligible. In case of a schedule conflict or insufficient prerequisites, she will not be allowed to enroll.

Preconditions: The student is registered at the university

Postconditions: The student is enrlled in the seminar or the system is unchanged.

Alternate course A: The student has already enrolled in all the seminars he has paid for.

A4. The system informs the student that he cannot enroll in any more seminars unless he pays more fees.

A5. The system invites the student to return after paying new fees.

A6. This UC ends.

Alternate course B: The student does not have sufficient pre requisites for any seminar

B4. The system informs the student that he does not have sufficient pre requisites to follow any more seminars.

B5. The UC ends.

Alternate course C: The student does not like the new timetable and decides not to enroll

C8. The student cancels the enrolment process.

C9. The UC ends.