Inf2: Software Engineering and Professional Practice
Lecture 5: Use Cases, UML Use Case Diagrams

Cristina Adriana Alexandru

School of Informatics
University of Edinburgh

This lecture

- Use cases
  - Notions of a use case, actor, use case scenario (instance)
  - A template for describing use cases
  - Connections and scope of use cases
  - Use case diagrams defined by the Unified Modelling Language
  - Requirements engineering organised by use cases
  - Uses and problems with use cases

Introduction to Use cases

- An important part of any requirements document for a system is a description of the system's behaviour from the viewpoint of its *users*.

  IMPORTANT NOTE Here, a *user* is anything external to the system which interacts with it, e.g. a human user, another system, a hardware device, etc.

- Behaviour can be broken down into units, each triggered by some *user*.

- Use cases are one way of describing these units

What is a use case?

A "task or coherent unit of functionality which the system is required to support", and which has value for at least one *user* (see previous IMPORTANT NOTE).

(*Stevens* Chapter 7).

Named beginning with a verb

E.g. "Log in', 'Buy a Product', 'Complete returns form', 'Log out' can all be use case names.

Actors in use cases

Actors are a kind of *user* (see previous IMPORTANT NOTE) who take an active part in the use case.

An actor can be:

- a human user of the system *in a particular role* E.g., Bank Customer and not Mary.
- an external system, which *in some role* interacts with the system.
- an external device, which *in some role* interacts with the system.

The same human user, external system or device may interact with the system in more than one role, and thus be (partly) represented by more than one actor (e.g., a bank teller may happen also to be a customer of the bank).

Actors in use cases

The primary actor usually is the one (there can only be one!) triggering the use case. E.g. Customer can trigger 'Buy a Product'

Supporting actors may also be involved

Some stakeholders may be neither primary nor supporting actors

Each use case

- has a discrete goal the primary actor wishes to achieve; Short verb phrase used as name of use case.
- includes a description of the sequence of messages exchanged between the system and actors (primary or supporting) in order to achieve the goal.

Exercise 1: Identify all the actors

A university provides a submit system that students can use to submit their coursework, and later retrieve marks and feedback. Before a course begins, the configuration for that course is set up by the EUCLID student record system interacting with our system. Thereafter, any student can submit work onto the system, and the lecturer can retrieve the work submitted so far. Once marking is ready, lecturers and markers can submit marks and feedback on the system, which notifies the students.

Exercise 1: Identify all the actors

A university provides a submit system that students can use to submit their coursework, and later retrieve marks and feedback. Before a course begins, the configuration for that course is set up by the EUCLID student record system interacting with our system. Thereafter, any student can submit work onto the system, and the lecturer can retrieve the work submitted so far. Once marking is ready, lecturers and markers can submit marks and feedback on the system, which notifies the students.

Answer: Student, StudentRecordSystem (role, not name!), Lecturer, Marker

Use case scenarios (instances)

Usually a use case describes the main sequence of steps (i.e. path) necessary to achieve the use case's goal.

However might be alternative paths, including some handling when all does not go to plan and the goal is not achieved.

Each path through use case called a *use-case instance* or *scenario*

- One talks about the main success scenario and alternate success or failure scenarios.

A use case is a set of scenarios tied together by a common user goal.

Warning: Sometimes *scenario* and *use-case* are synonyms (but not in this course!)

Example of use case scenarios (instances)

Goal/ Use case name: Buy a Product

Main Success Scenario (MSS)

1. Customer browses catalogue and selects items to buy
2. Customer goes to check out
3. Customer fills in shipping info
4. System presents full pricing info
5. Customer fills in credit card info
6. System authorises purchase with customer
7. System confirms sale to customer
8. System sends confirmation email to customer

Example of use case scenarios (instances) (cont)

Alternate Scenarios (extensions - variations on MSS)

3a : Customer is regular customer

.1 : System displays current shipping and billing info

.2 : Customer may accept or override these defaults, returns to MSS at step 4, but skips step 5.

6a . System fails to authorize credit card purchase

.1 : Customer may re-enter credit card information or may cancel

Phrase at the start of an extension is an enabling condition for that extension

A template for describing use cases

- Goal – what the primary actor wishes to achieve
- Summary – a one or two sentence description of the use case.
- Stakeholders and each's Interest in the use case
- Primary actor
- Supporting actors
- Trigger – the event that leads to this use case being performed.
- Pre-conditions/Assumptions – what can be assumed to be true when the use case starts
- Guarantees – what the use case ensures at its end
  - Success guarantees
  - Failure guarantees
  - Minimal guarantees
- Main Success Scenario
- Alternate scenarios

Use cases: connections and scope

A use case:

- can have different levels of detail
  - e.g. depending on where it is used in development process
- may refer to other use cases
  - to provide further information on particular steps
- may describe different scopes
  - e.g. a system of systems, a single system or a single component of a system

The Unified Modeling Language

UML is a graphical language for recording aspects of the requirements and design of software systems.

It provides many diagram types; all the diagrams of a system together form a UML model.

Mostly tailored to an OO world-view

Often used just for documentation, but in model-driven development, a UML model may be used e.g. to generate and update code and database schemas automatically.

Many tools available to support UML

Use case diagrams

Are part of the Unified Modeling Language (UML).

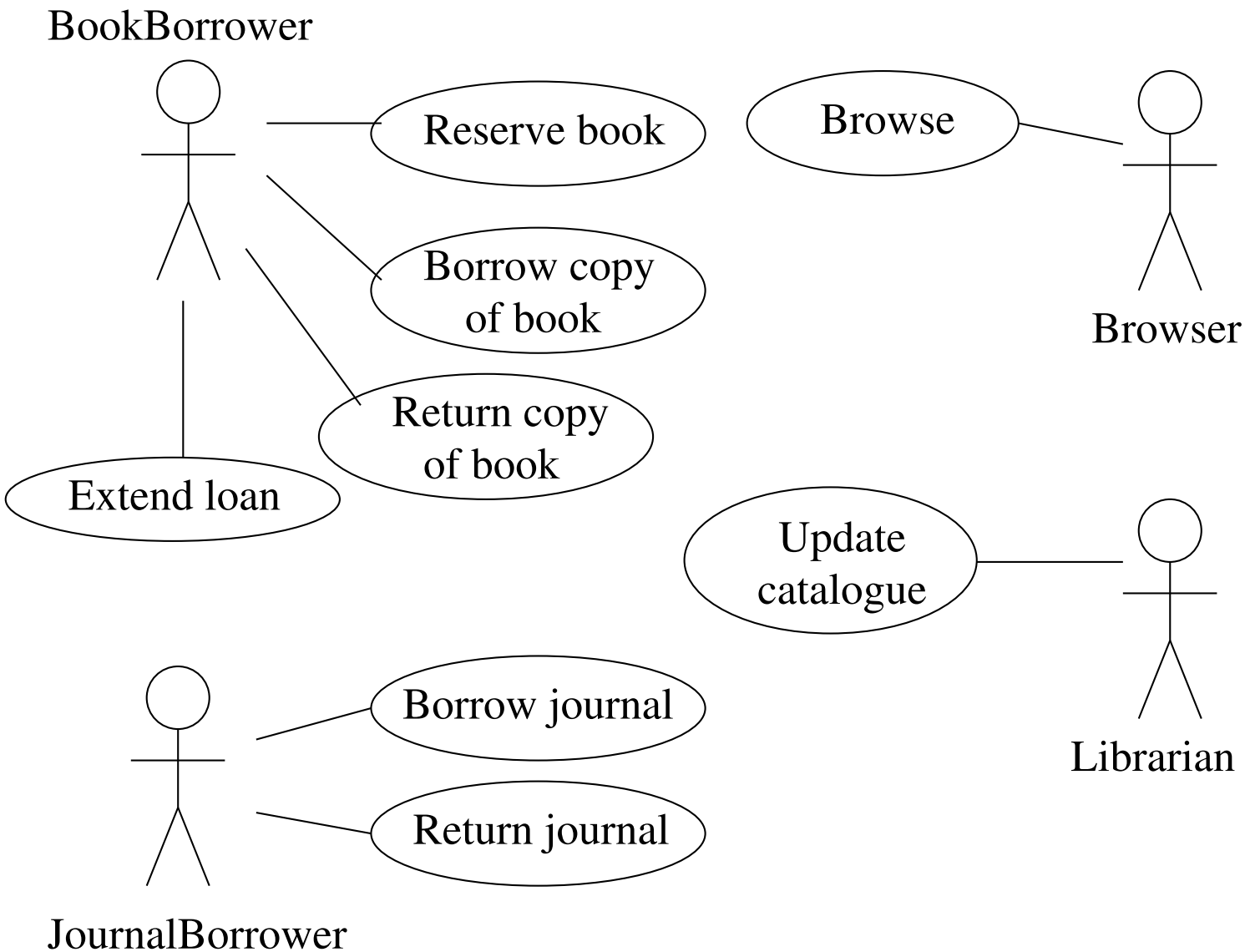
Provide a high level view of all the use cases for a given system.

Are easy to understand in their most basic form, so can be discussed with customers who are not familiar with UML.

Represent:

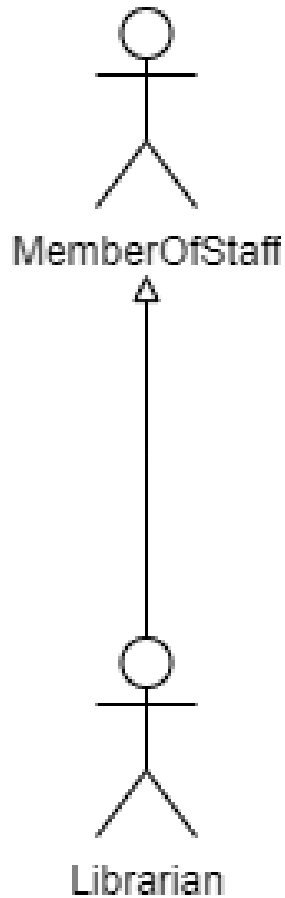
- Actors as stick figures with one-word capital names in sungular
- Use cases as named ovals with capital names starting with a verb
- *Possible* interactions between actors and use cases as the lines connecting them

Use case diagrams: A very simple example



Use case generalization

Used to show an *is-a* relationship: Librarian is-a MemberOfStaff.



Useful for obtaining the one single primary actor for some use cases

Exercise 2: Identify use cases triggered by actors

A university provides a submit system that students can use to submit their coursework, and later retrieve marks and feedback. Before a course begins, the configuration for that course is set up by the EUCLID student record system interacting with our system. Thereafter, any student can submit work onto the system, and the lecturer can retrieve the work submitted so far. Once marking is ready, lecturers and markers can submit marks and feedback on the system, which notifies the students.

Primary Actor	**Use cases**
Student	?
StudentRecordSystem	?
Lecturer	?
Marker	?

Exercise 2: Identify use cases triggered by actors

A university provides a submit system that students can use to submit their coursework, and later retrieve marks and feedback. Before a course begins, the configuration for that course is set up by the EUCLID student record system interacting with our system. Thereafter, any student can submit work onto the system, and the lecturer can retrieve the work submitted so far. Once marking is ready, lecturers and markers can submit marks and feedback on the system, which notifies the students.

Answer:

Primary Actor	**Use cases**
Student	'Submit coursework', 'Retrieve marks and feedback'
StudentRecordSystem	'Set up course configuration'
Lecturer	'Retrieve coursework', 'Submit marks and feedback'
Marker	'Submit marks and feedback'

Exercise 3: Draw use case diagram

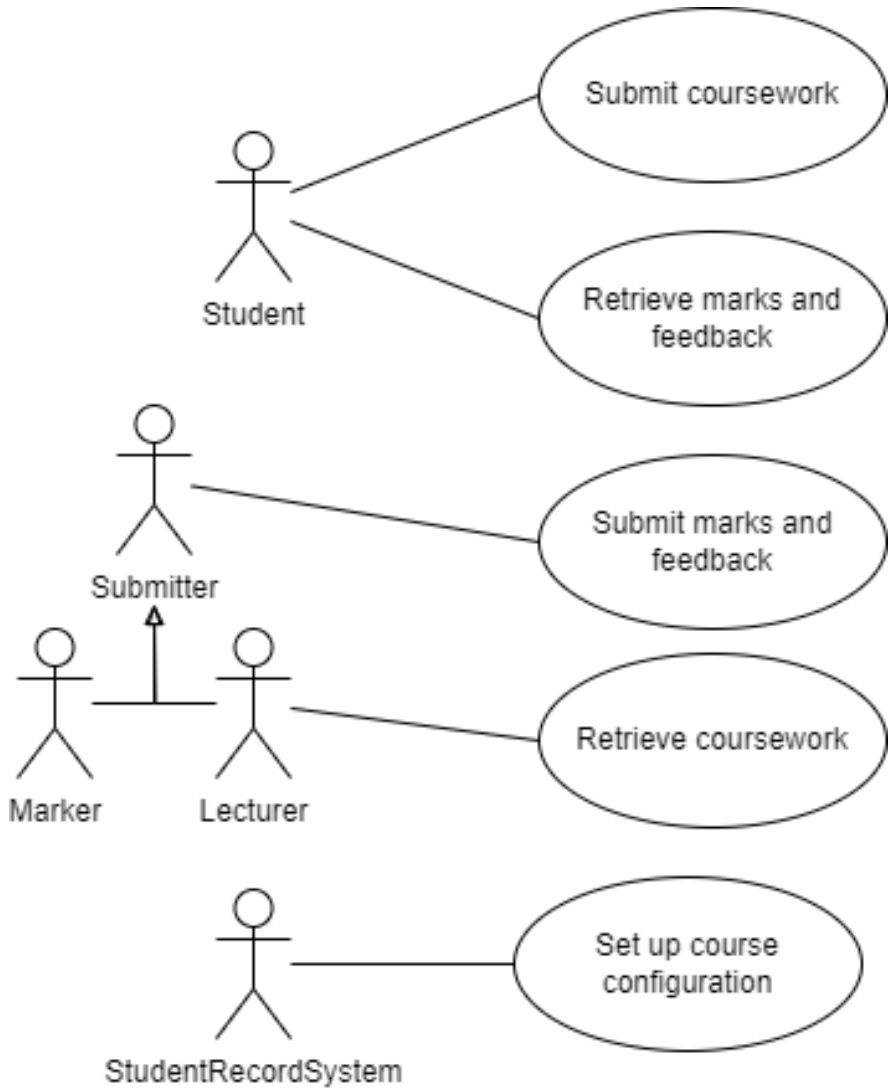
1) Using the derived primary actors and their use cases, draw the use case diagram.

Primary Actor	**Use cases**
Student	'Submit coursework', 'Retrieve marks and feedback'
StudentRecordSystem	'Set up course configuration'
Lecturer	'Retrieve coursework', 'Submit marks and feedback'
Marker	'Submit marks and feedback'

2) Then, also identify any supporting actors and add the possible interactions between them and use cases.

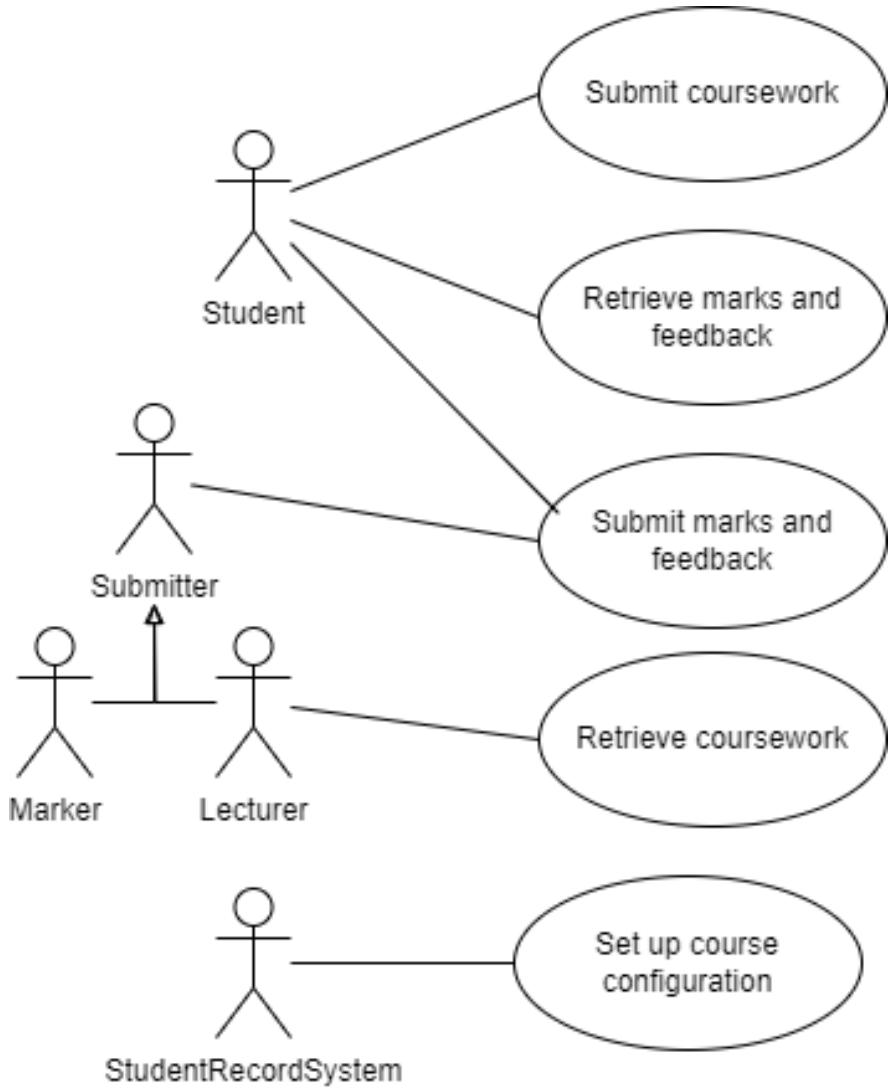
Exercise 3: Draw use case diagram

1) Solution representing only the possible interactions between primary actor - use cases.



Exercise 3: Draw use case diagram

2) Adding the possible interactions between supporting actors - use cases (Student supporting actor in 'Submit marks and feedback')



Requirements engineering organised by use cases

Use cases can help with requirements engineering by providing a structured way to go about it:

1. identify the actors
2. for each actor, find out
   - what they need from the system
   - any other interactions they expect to have with the system
   - which use cases have what priority for them

Good for both requirements specification and iterated requirements elicitation.

Use cases primarily capture functional requirements, but sometimes non-functional requirements are attached to a use case.

Other times, non-functional requirements apply to subsets or all of use-cases.

Uses of use cases in software processes

Driving design

Design validation

- You can walk through how a design realises a use case, checking that the set of classes provides the needed functionality and that the interactions are as expected.

Testing

- Use cases can be a good source of system tests

Possible problems with use cases

- Interactions spelled out may be too detailed, may needlessly constrain design
- May specify supporting actors that are not essential for fulfilling goal of primary actor
  - Does borrowing a book have to involve a librarian?
- Focus on operational nature of system may result in less attention to software architecture and static object structure
- May miss requirements not naturally associated with actors

Reading

Sommerville SE. Use Cases discussed are both in Requirements and System Modeling chapters. Look up Use Cases in index to find the relevant sections.

Stevens, Chapter 7.