Software Engineering I

Dr. Elham Mahmoudzadeh

Isfahan University of Technology

mahmoudzadeh@iut.ac.ir

2021

Chapter 4
Functional Modeling(III)

Steps(I)

1. Preparing proposal
2. Requirements determination
   - User story
3. Abstract Business Process Modelling
4. Analysis
   - Functional Modelling
   - Structural Modelling
   - Behavioral Modelling

Steps(II)

5. Design
   - Optimization
   - Database Management
   - User Interface
   - Physical Architecture

Introduction

- Use-case diagrams provided a bird's-eye view of the basic functionality of the business processes contained in the evolving system.
- Activity diagrams, in a sense, open up the black box of each business process by providing a more-detailed graphical view of the underlying activities that support each business process.
- Use-case descriptions provide a means to more fully document the different aspects of each individual use case.

The use-case descriptions are based on the identified requirements, use-case diagram, and the activity diagram of the business processes.

Use-case descriptions contain all the information needed to document the functionality of the business processes.

Introduction

- Use cases are the primary drivers for all the UML diagramming techniques.
- A use case communicates at a high level what the system needs to do, and all the UML diagramming techniques build on this by presenting the use-case functionality in a different way for a different purpose.
- Use cases are the building blocks by which the system is designed and built.

Introduction

- Use cases capture the typical interaction of the system with the system's users (end users and other systems).
- These interactions represent the external, or functional view of the system from the perspective of the user.
- Each use case describes one and only one function in which users interact with the system.
- Although a use case may contain several paths that a user can take while interacting with the system, each possible execution path through the use case is referred to as a *scenario*.

Introduction

- When creating use-case descriptions, the project team must work closely with the users to fully document the functional requirements.
- Organizing the functional requirements and documenting them in a use-case description are a relatively simple process, but it takes considerable practice to ensure that the descriptions are complete enough to use in structural and behavioral modeling.
- The best place to begin is to review the use-case and activity diagrams.
- It is possible that multiple users will play the same role. Therefore, use cases should be associated with the roles played by the users and not with the users themselves.

Types of Use Cases

- Classify a use case based on the purpose of the use case and the amount of information that the use case contains: overview versus detail and essential versus real.

Overview use case vs. *detail use case*

- *Over view use case* is used to enable the analyst and user to agree on a high-level overview of the requirements.
  - Typically, overview use cases are created very early in the process of understanding the system requirements, and they document only basic information about the use case.
  - These can easily be created immediately after the creation of the use-case diagram.
- A *detail use case* typically documents, as far as possible, all the information needed for the use case.
  - These can be based on the activities and control flows contained in the activity diagrams.

Essential use case vs. *real use case*

- An *essential use case* is one that describes only the minimum essential issues necessary to understand the required functionality.
- A *real use case* goes farther and describes a specific set of steps.
- The primary difference is that *essential use cases* are implementation independent, whereas *real use cases* are detailed descriptions of how to use the system once it is implemented.
- Thus, real use cases tend to be used only in the design, implementation, and testing.

Elements of a Use-Case Description

- A use-case description contains all the information needed to build the structural and behavioral diagrams that follow, but it expresses the information in a less-formal way that is usually simpler for users to understand.
- A use-case description has three basic parts: **overview information**, **relationships**, and the **flow of events**.

overview

Relationships

Flow of events

Use Case Name: Make Old Patient Appt	ID: 2	Importance Level: Low
Primary Actor: Old Patient	Use Case Type: Detail, Essential	

Stakeholders and Interests:
Old Patient – wants to make, change, or cancel an appointment
Doctor – wants to ensure patient's needs are met in a timely manner

Brief Description: This use case describes how we make an appointment as well as changing or canceling an appointment for a previously seen patient.

Trigger: Patient calls and asks for a new appointment or asks to cancel or change an existing appointment

Type: External

Relationships:

Association: Old Patient
Include:
Extend: Update Patient Information
Generalization: Manage Appointments

Normal Flow of Events:

1. The Patient contacts the office regarding an appointment.
2. The Patient provides the Receptionist with his or her name and address.
3. If the Patient's information has changed
   Execute the Update Patient Information use case.
4. If the Patient's payment arrangements has changed
   Execute the Make Payments Arrangements use case.
5. The Receptionist asks Patient if he or she would like to make a new appointment, cancel an existing appointment, or change an existing appointment.
   If the patient wants to make a new appointment,
   the S-1: new appointment subflow is performed.
   If the patient wants to cancel an existing appointment,
   the S-2: cancel appointment subflow is performed.
   If the patient wants to change an existing appointment,
   the S-3: change appointment subflow is performed.
6. The Receptionist provides the results of the transaction to the Patient.

SubFlows:

S-1: New Appointment
1. The Receptionist asks the Patient for possible appointment times.
2. The Receptionist matches the Patient's desired appointment times with available dates and times and schedules the new appointment.

S-2: Cancel Appointment
1. The Receptionist asks the Patient for the old appointment time.
2. The Receptionist finds the current appointment in the appointment file and cancels it.

S-3: Change Appointment
1. The Receptionist performs the S-2: cancel appointment subflow.
2. The Receptionist performs the S-1: new appointment subflow.

Alternate/Exceptional Flows:

S-1, 2a1: The Receptionist proposes some alternative appointment times based on what is available in the appointment schedule.

S-1, 2a2: The Patient chooses one of the proposed times or decides not to make an appointment.

Overview Information

- Identifies the use case and provides basic background information.
- The *use-case name* should be a verb–noun.
- The *use-case ID number* provides a unique way to find every use case and also enables the team to trace design decisions back to a specific requirement.
- The *use-case type* is either overview or detail and essential or real.
- The *primary actor* is usually the trigger of the use case—the person or thing that starts the execution of the use case. The primary purpose of the use case is to meet the goal of the primary actor.
- The *brief description* is typically a single sentence that describes the essence of the use case.

Use Case Name: Make Old Patient Appt	ID: 2	Importance Level: Low
Primary Actor: Old Patient	Use Case Type: Detail, Essential	
Stakeholders and Interests: Old Patient – wants to make, change, or cancel an appointment Doctor – wants to ensure patient's needs are met in a timely manner		
Brief Description: This use case describes how we make an appointment as well as changing or canceling an appointment for a previously seen patient.		
Trigger: Patient calls and asks for a new appointment or asks to cancel or change an existing appointment Type: External		

Overview Information(Cnt'd)

- The *importance level* can be used to prioritize the use cases. The importance level enables the users to explicitly prioritize which business functions are most important and need to be part of the first version of the system and which are less important and can wait until later versions if necessary. The importance level can use a fuzzy scale, such as high, medium, and low. It can also be done more formally using a weighted average of a set of criteria.

Use Case Name: Make Old Patient Appt	ID: 2	Importance Level: Low
Primary Actor: Old Patient	Use Case Type: Detail, Essential	
Stakeholders and Interests: Old Patient – wants to make, change, or cancel an appointment Doctor – wants to ensure patient's needs are met in a timely manner		
Brief Description: This use case describes how we make an appointment as well as changing or canceling an appointment for a previously seen patient.		
Trigger: Patient calls and asks for a new appointment or asks to cancel or change an existing appointment Type: External		

Overview Information(Cnt'd)

- A use case may have multiple *stakeholders* that have an interest in the use case. Each use case lists each of the stakeholders with each one's interest in the use case. The stakeholders' list always includes the primary actor.
- Each use case typically has a *trigger*—the event that causes the use case to begin. A trigger can be an *external trigger*, such as a customer placing an order or the fire alarm ringing, or it can be a *internal trigger*.

Use Case Name: Make Old Patient Appt	ID: 2	Importance Level: Low
Primary Actor: Old Patient	Use Case Type: Detail, Essential	
Stakeholders and Interests: Old Patient – wants to make, change, or cancel an appointment Doctor – wants to ensure patient's needs are met in a timely manner		
Brief Description: This use case describes how we make an appointment as well as changing or canceling an appointment for a previously seen patient.		
Trigger: Patient calls and asks for a new appointment or asks to cancel or change an existing appointment Type: External		

Relationships

- Use-case relationships explain how the use case is related to other use cases and users.
- There are four basic types of *relationships:* **association**, **extend**, **include**, and **generalization**.
- An *association relationship* documents the communication that takes place between the use case and the actors that use the use case.
- An *include relationship* represents the mandatory inclusion of another use case. The include relationship enables *functional decomposition*—the breaking up of a complex use case into several simpler ones. The include relationship also enables parts of use cases to be reused by creating them as separate use cases.

Relationships(Cnt’d)

- An *extend relationship* represents the extension of the functionality of the use case to incorporate optional behavior.
- The *generalization relationship* allows use cases to support *inheritance.*

یک رابطه گسترش نشان دهنده گسترش عملکرد مورد استفاده برای ترکیب رفتار اختیاری است

Flow of Events

- Finally, the individual steps within the business process are described. Three different categories of steps, or *flows of events,* can be documented**: normal flow of events**, **sub flows**, and **exceptional flows**.

Normal flow of events

- Includes only steps that normally are executed in a use case. The steps are listed in the order in which they are performed.
- In some cases, the normal flow of events should be decomposed into a set of *sub flows* to keep the normal flow of events as simple as possible.
- Alternatively, we could replace a sub flow with a separate use case that could be incorporated via the include relationships. If it does, then the specific sub flow(s) should be replaced with a call to the related use case, and the use case should be added to the include relationship list.

Alternative or exceptional flows

- Are ones that do happen but are not considered to be the norm. These must be documented.
- Like the sub flows, the primary purpose of separating out alternate or exceptional flows is to keep the normal flow of events as simple as possible.

When should events be factored out

from the normal flow of events into sub flows?

When should sub flows and/or exceptional flows be factored out

into separate use cases?

The answer

- The primary criteria should be based on the level of complexity that the use case entails. The more difficult it is to understand the use case, the more likely events should be factored out into sub flows, or sub flows and/or alternative or exceptional flows should be factored out into separate use cases that are called by the current use case. This, creates more use cases.
- We are trying to represent, in a manner as complete and concise as possible, our understanding of the business processes that we are investigating so that the client can validate the requirements that we are modeling.

Optional Characteristics

- Other characteristics of use cases can be documented by use-case descriptions.
  - Level of complexity of the use case;
  - The estimated amount of time it takes to execute the use case;
  - The system with which the use case is associated;
  - Specific data flows between the primary actor and the use case;
  - Any specific attribute, constraint, or operation associated with the use case;
  - Any preconditions that must be satisfied for the use case to execute;
  - Any guarantees that can be made based on the execution of the use case.
- There is no standard set of characteristics of a use case that must be captured.

Campus Housing Service Add an Apartment Use-Case Description

Use Case Name: Add Apartment	ID: 1	Importance Level: High
Primary Actor: Apartment Owner	Use Case Type: Detail, Essential	

Stakeholders and Interests:
Apartment Owner—wants to advertise available apartment
Campus Housing Service—provides a service that enables the apartment owners to rent their available apartments

Brief Description: This use case describes how the campus housing service can maintain an up-to-date listing of available apartments.

Trigger: Apartment Owner wants to add an available apartment

Type: External

Relationships:
Association: Apartment Owner
Include:
Extend:
Generalization:

Normal Flow of Events:
1. Capture the location of the apartment.
2. Capture the number of bedrooms in the apartment.
3. Capture the monthly rent of the apartment.
4. Add the apartment to the listing of available apartments.

SubFlows:

Alternate/Exceptional Flows:

Campus Housing Service Delete an Apartment Use-Case Description

Use Case Name: Delete Apartment	ID: 2	Importance Level: High
Primary Actor: Apartment Owner	Use Case Type: Detail, Essential	

Stakeholders and Interests:
Apartment Owner—wants to delist apartment
Campus Housing Service—provides a service that enables the apartment owners to rent their available apartments

Brief Description: This use case describes how the campus housing service can maintain an up-to-date listing of available apartments.

Trigger: Apartment Owner wants to delete an available apartment

Type: External

Relationships:

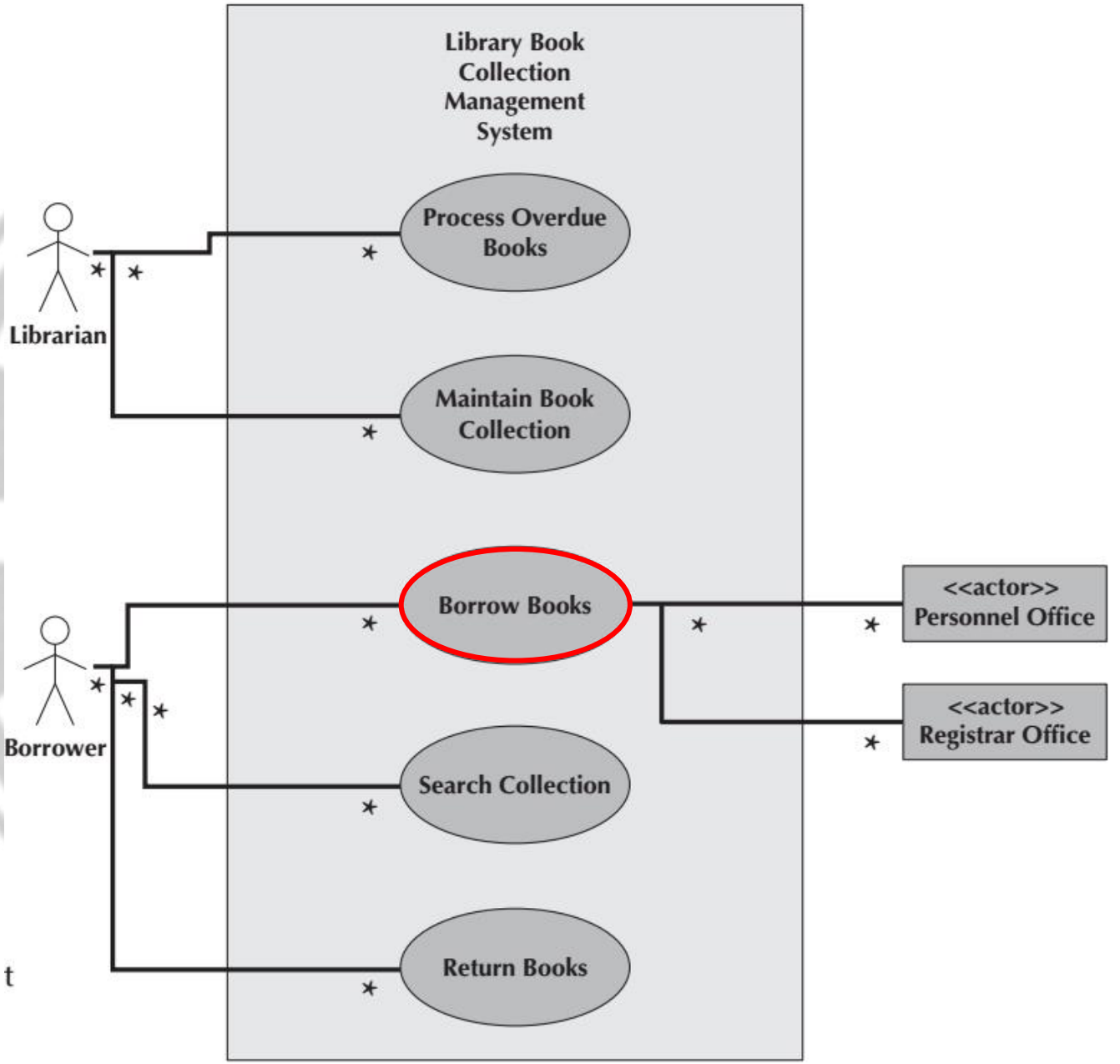
Association: Apartment Owner
Include:
Extend:
Generalization:

Normal Flow of Events:

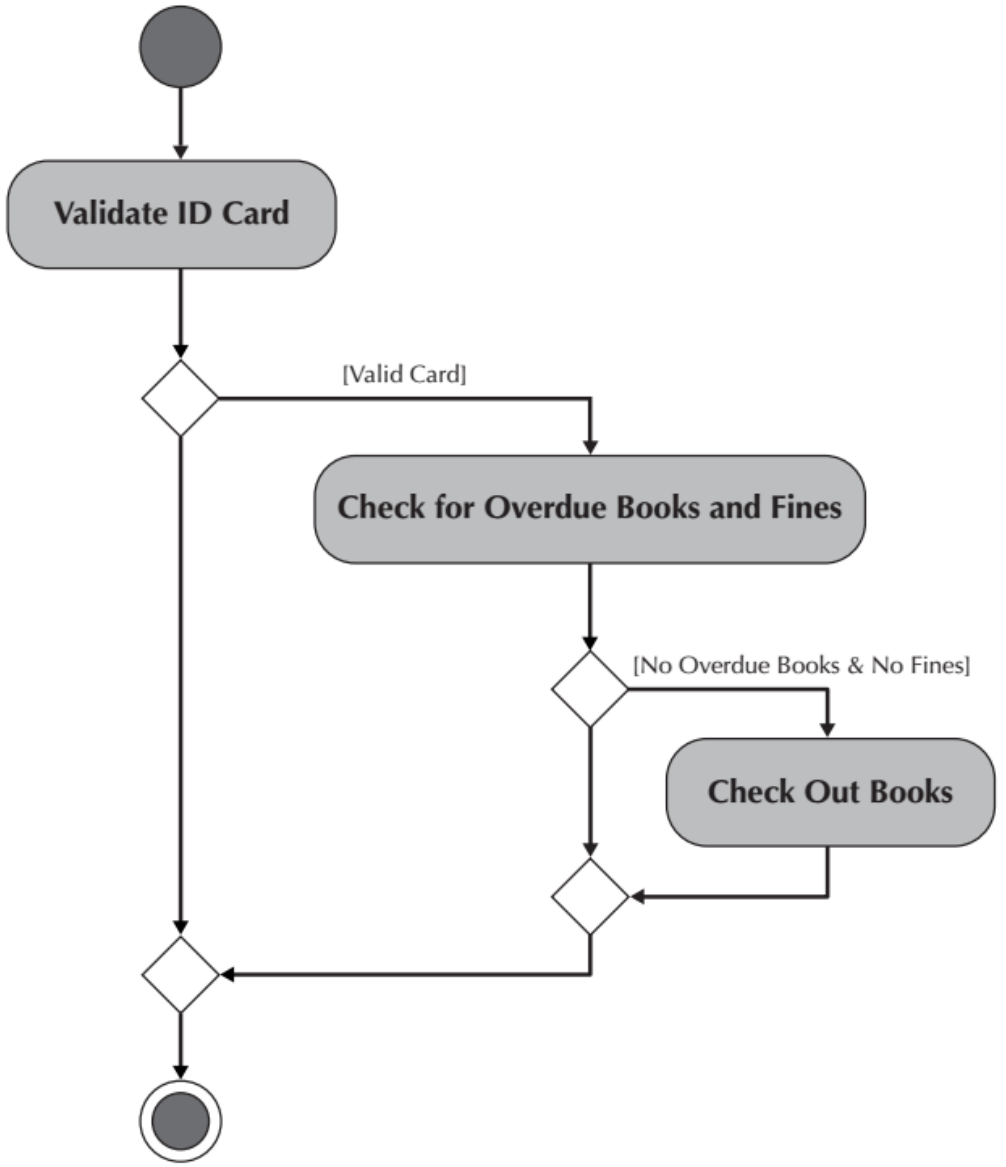
1. Capture the apartment identifier.
2. Delete the apartment from the listing of available apartments.

SubFlows:

Alternate/Exceptional Flows:


Library Book Collection Management System
Process Overdue Books
Librarian
Maintain Book Collection
Borrow Books
<<actor>> Personnel Office
Borrower
<<actor>> Registrar Office
Search Collection
Return Books

t


Validate ID Card
[Valid Card]
Check for Overdue Books and Fines
[No Overdue Books & No Fines]
Check Out Books

Overview Description for the Borrow Books Use Case

Use Case Name: Borrow Books	ID: 2	Importance Level: High
Primary Actor: Borrower	Use Case Type: Detail, Essential	
Stakeholders and Interests: Borrower—wants to check outbooks Librarian—wants to ensure borrower only gets books deserved		
Brief Description: This use case describes how books are checked out of the library.		
Trigger: Borrower brings books to check out desk. Type: External		
Relationships: Association: Borrower, Personnel Office, Registrar's Office Include: Extend: Generalization:		

Flow Descriptions for the Borrow Books Use Case

Normal Flow of Events:

1. The Borrower brings books to the Librarian at the check out desk.
2. The Borrower provides Librarian their ID card.
3. The Librarian checks the validity of the ID Card.
   - If the Borrower is a Student Borrower, Validate ID Card against Registrar's Database.
   - If the Borrower is a Faculty/Staff Borrower, Validate ID Card against Personnel Database.
   - If the Borrower is a Guest Borrower, Validate ID Card against Library's Guest Database.
4. The Librarian checks whether the Borrower has any overdue books and/or fines
5. The Borrower checks out the books

SubFlows:

Alternate/Exceptional Flows:

4a The ID Card is invalid, the book request is rejected.

5a The Borrower either has overdue books, fines, or both, the book request is rejected.

What should you do for your project?

1. Write use case description for each use case.

We will work in the lab.

Reference

- **Dennis, Wixon, Tegarden**, "**System Analysis and Design, An Object Oriented Approach with UML", 5th Edition, 2015.**