Software Engineering I

Dr. Elham Mahmoudzadeh

Isfahan University of Technology

mahmoudzadeh@iut.ac.ir

2021

Chapter 4
Functional Modeling(I)

Steps(I)

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

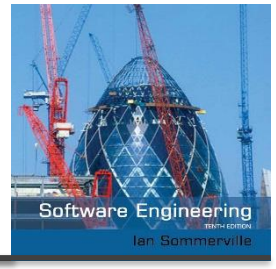
Steps(II)

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

Introduction

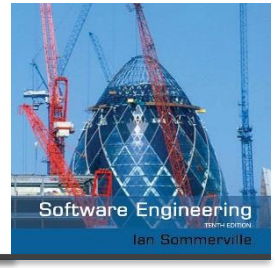
- All object-oriented systems development approaches are use-case driven, architecture-centric, and iterative and incremental.
- *Use case* is a formal way of representing the way a business system interacts with its environment.
- *Use case* is a high-level overview of the business processes in a business information system.
- *Use cases* represent the entire basis for an object-oriented system.
- *Use cases* can document the current system (i.e., as-is system) or the new system being developed (i.e., to-be system).
- *Use cases* also form the foundation for testing and user-interface design .

System modeling



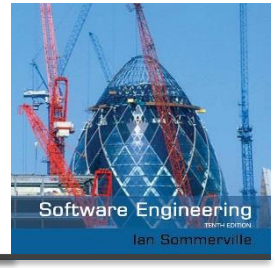
- System modeling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system.
- System modeling has now come to mean representing a system using some kind of graphical notation, which is now almost always based on notations in the Unified Modeling Language (UML).
- System modelling helps the analyst to understand the functionality of the system and models are used to communicate with customers.

System perspectives



- An external perspective, where you model the context or environment of the system.
- A structural perspective, where you model the organization of a system or the structure of the data that is processed by the system.
- A behavioral perspective, where you model the dynamic behavior of the system and how it responds to events.

UML diagram types



- Use case diagrams, which show the interactions between a system and its environment.
- Activity diagrams, which show the activities involved in a process or in data processing .
- Class diagrams, which show the object classes in the system and the associations between these classes.
- Sequence diagrams, which show interactions between actors and the system and between system components.
- State diagrams, which show how the system reacts to internal and external events.

Introduction(Cnt’d)

- From an architecture-centric perspective, use-case modeling supports the creation of an external or functional view of a business process in that it shows how the users view the process rather than the internal mechanisms by which the process and supporting systems operate.

Introduction(Cnt’d)

- *Activity diagrams* are typically used to augment our understanding of the business processes and our use-case model.
- Technically, an activity diagram can be used for any type of process-modeling activity.

Introduction(Cnt’d)

- Activity diagrams and use cases are *logical models*—models that describe the business domain’s activities without suggesting how they are conducted.
- *Logical models* are sometimes referred to as *problem domain models*. Reading a use-case or activity diagram, in principle, should not indicate if an activity is computerized or manual.
- These physical details are defined during design when the logical models are refined into *physical models*. These models provide information that is needed to ultimately build the system.
- By focusing on logical activities first, analysts can focus on how the business should run without being distracted with implementation details.

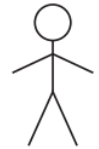
Use-case Diagram

- Employ the use-case diagram to better understand the functionality of the system at a very high level.
- Because a use-case diagram provides a simple, straightforward way of communicating to the users exactly what the system will do, a use-case diagram is drawn when gathering and defining requirements for the system.
- Use-case diagram can encourage the users to provide additional high-level requirements.
- A use-case diagram illustrates in a very simple way the main functions of the system and the different kinds of users that will interact with it.

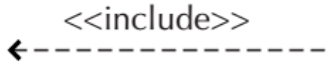
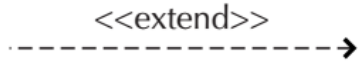
Let’s start

- For identifying use cases, Jacobson et al. (1992) recommend that you ask the following questions:
  - What are the main tasks performed by each actor?
  - Will the actor read or update any information in the system?
  - Will the actor have to inform the system about changes outside the system?
  - Does the actor have to be informed of unexpected changes?

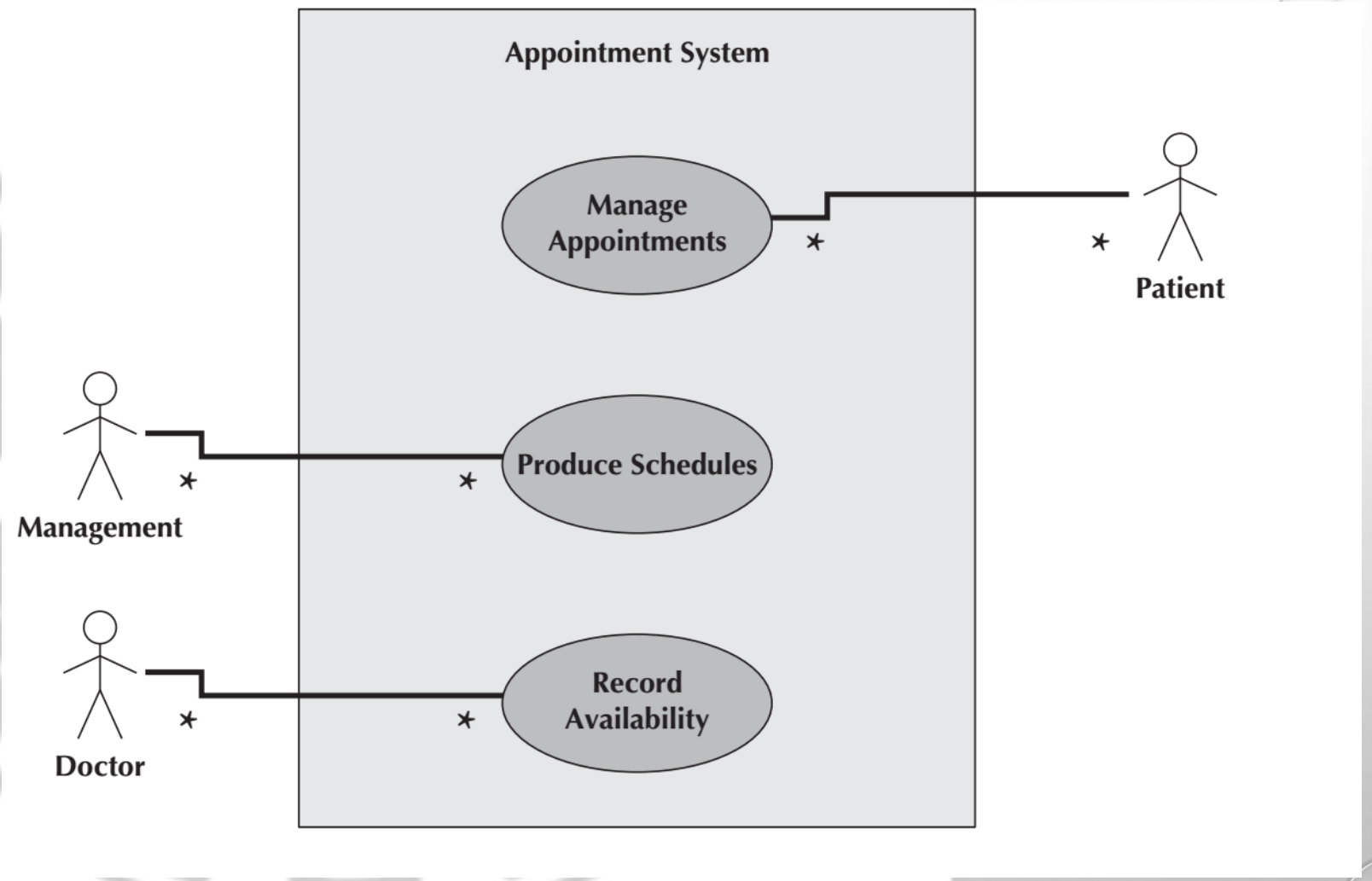
Elements of Use-Case Diagrams(I)

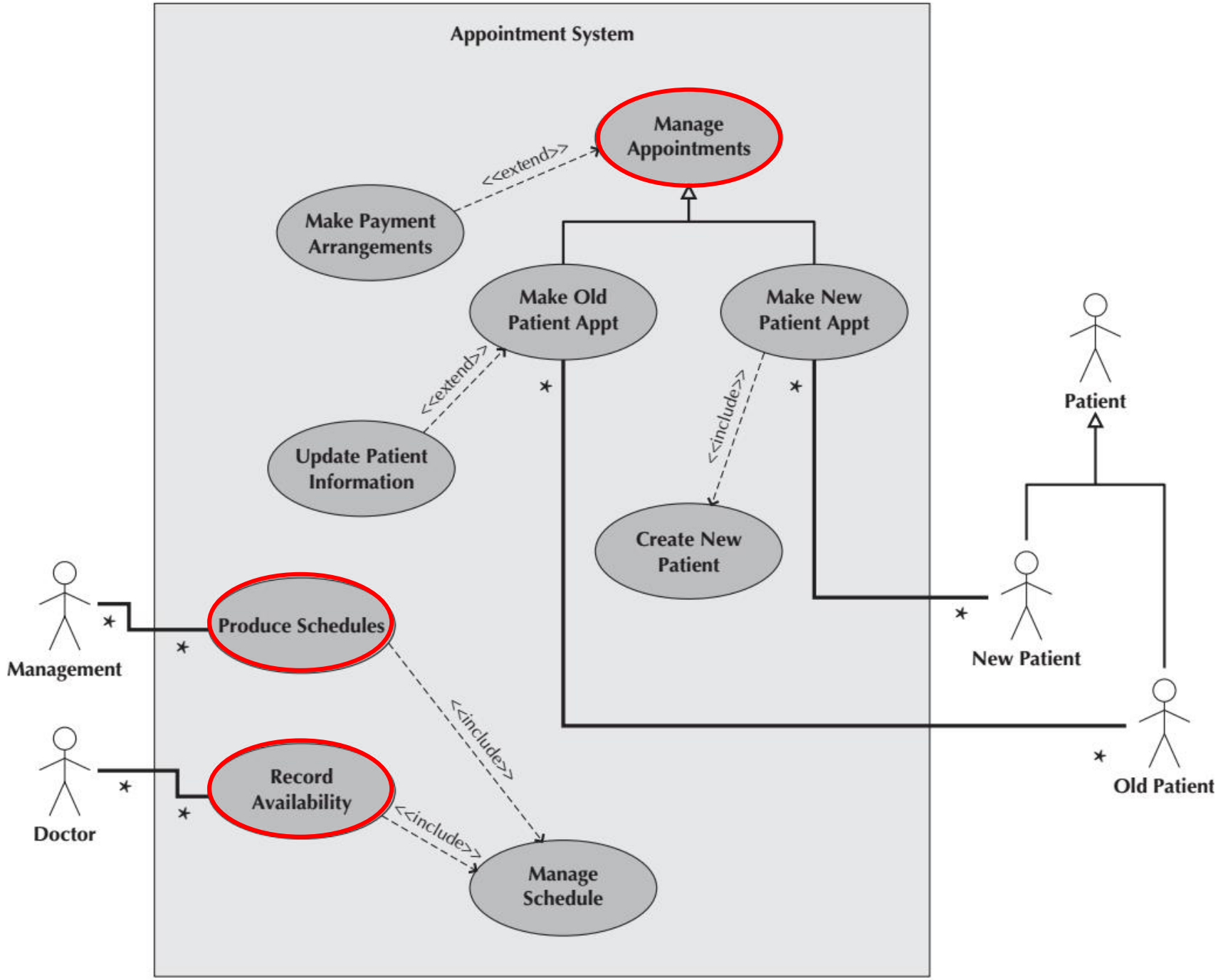
An actor: ■ Is a person or system that derives benefit from and is external to the subject. ■ Is depicted as either a stick figure (default) or, if a nonhuman actor is involved, a rectangle with <<actor>> in it (alternative). ■ Is labeled with its role. ■ Can be associated with other actors using a specialization/superclass association, denoted by an arrow with a hollow arrowhead. ■ Is placed outside the subject boundary.	 Actor/Role <<actor>> Actor/Role
A use case: ■ Represents a major piece of system functionality. ■ Can extend another use case. ■ Can include another use case. ■ Is placed inside the system boundary. ■ Is labeled with a descriptive verb–noun phrase.	Use Case
A subject boundary: ■ Includes the name of the subject inside or on top. ■ Represents the scope of the subject, e.g., a system or an individual business process.	Subject

Elements of Use-Case Diagrams(II)

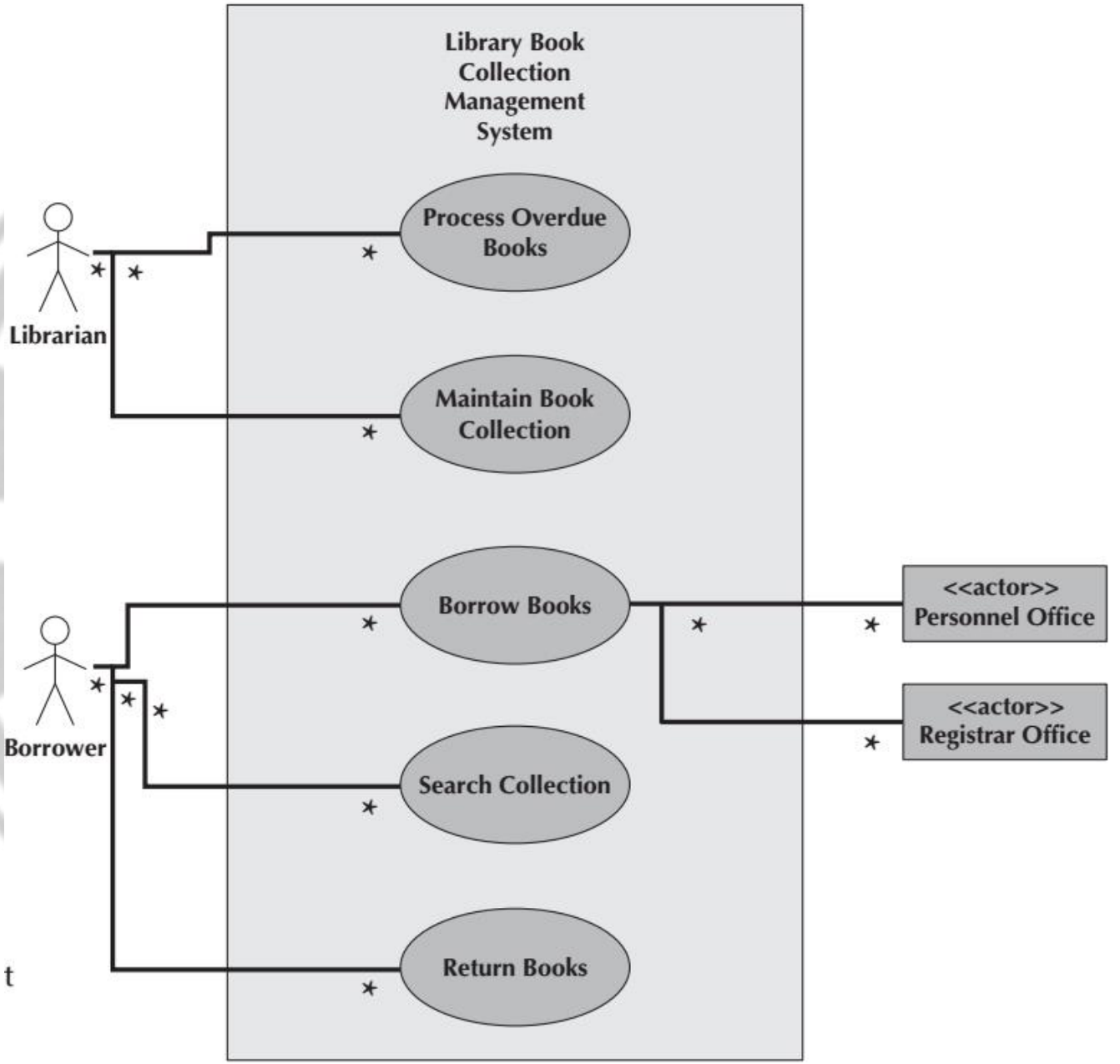
An association relationship: ■ Links an actor with the use case(s) with which it interacts.	
An include relationship: ■ Represents the inclusion of the functionality of one use case within another. ■ Has an arrow drawn from the base use case to the used use case.	
An extend relationship: ■ Represents the extension of the use case to include optional behavior. ■ Has an arrow drawn from the extension use case to the base use case.	
A generalization relationship: ■ Represents a specialized use case to a more generalized one. ■ Has an arrow drawn from the specialized use case to the base use case.	

First Example




Appointment System
Manage Appointments
<<extend>>
Make Payment Arrangements
Make Old Patient Appt
Make New Patient Appt
<<extend>>
<<include>>
Update Patient Information
Create New Patient
Patient
Produce Schedules
Management
New Patient
<<include>>
Doctor
Record Availability
<<include>>
Manage Schedule
Old Patient

Second example



What should you do for your project?

1. Create use-case diagram, level 0.

We will work in the lab.

Reference

- **Dennis, Wixon, Tegarden**, "**System Analysis and Design, An Object Oriented Approach with UML", 5th Edition, 2015.**
- **Valacich, J. S., J. F. George, "Modern systems analysis and design", 8th Edition, 2017.**