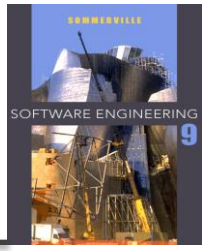


Software Engineering (CSM 31212)

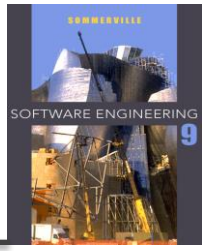
MM. Mohamed Mufassirin

Lecturer in Computer Science

Dept. of Computer Science

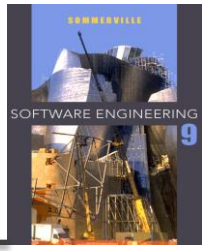
FAS/SEUSL

Topics covered



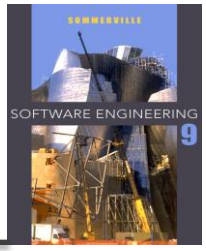
- ✧ Context models
- ✧ Interaction models
- ✧ Structural models
- ✧ Behavioral models
- ✧ Model-driven engineering

System modeling



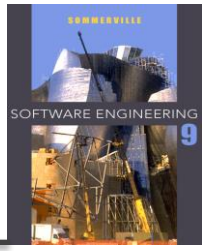
- System modeling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system.
- System modeling has now come to mean representing a system using some kind of graphical notation, which is now almost always based on notations in the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

Existing and planned system models



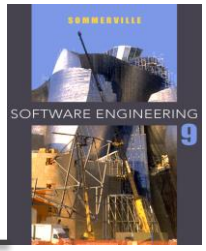
- Models of the existing system are used during requirements engineering. They help clarify what the existing system does and can be used as a basis for discussing its strengths and weaknesses. These then lead to requirements for the new system.
- Models of the new system are used during requirements engineering to help explain the proposed requirements to other system stakeholders. Engineers use these models to discuss design proposals and to document the system for implementation.
- In a model-driven engineering process, it is possible to generate a complete or partial system implementation from the system model.

System perspectives



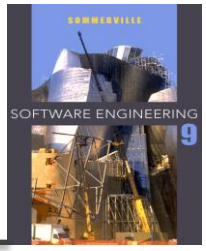
- **An external perspective**, where you model the context or environment of the system.
- **An interaction perspective**, where you model the interactions between a system and its environment, or between the components of a system.
- **A structural perspective**, where you model the organization of 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

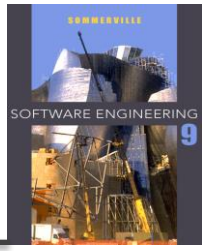


- **Activity diagrams**, which show the activities involved in a process or in data processing .
- **Use case diagrams**, which show the interactions between a system and its environment.
- **Sequence diagrams**, which show interactions between actors and the system and between system components.
- **Class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

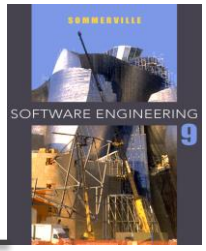
Use of graphical models



- As a means of facilitating discussion about an existing or proposed system
  - Incomplete and incorrect models are OK as their role is to support discussion.
- As a way of documenting an existing system
  - Models should be an accurate representation of the system but need not be complete.
- As a detailed system description that can be used to generate a system implementation
  - Models have to be both correct and complete.

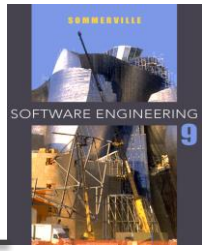

SOMMERVILLE
SOFTWARE ENGINEERING
9

Context models

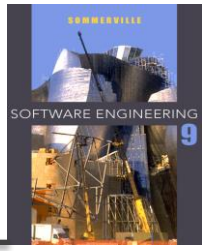


- Context models are used to illustrate the operational context of a system - they show what lies outside the system boundaries.
- Social and organisational concerns may affect the decision on where to position system boundaries.
- Architectural models show the system and its relationship with other systems.

System boundaries



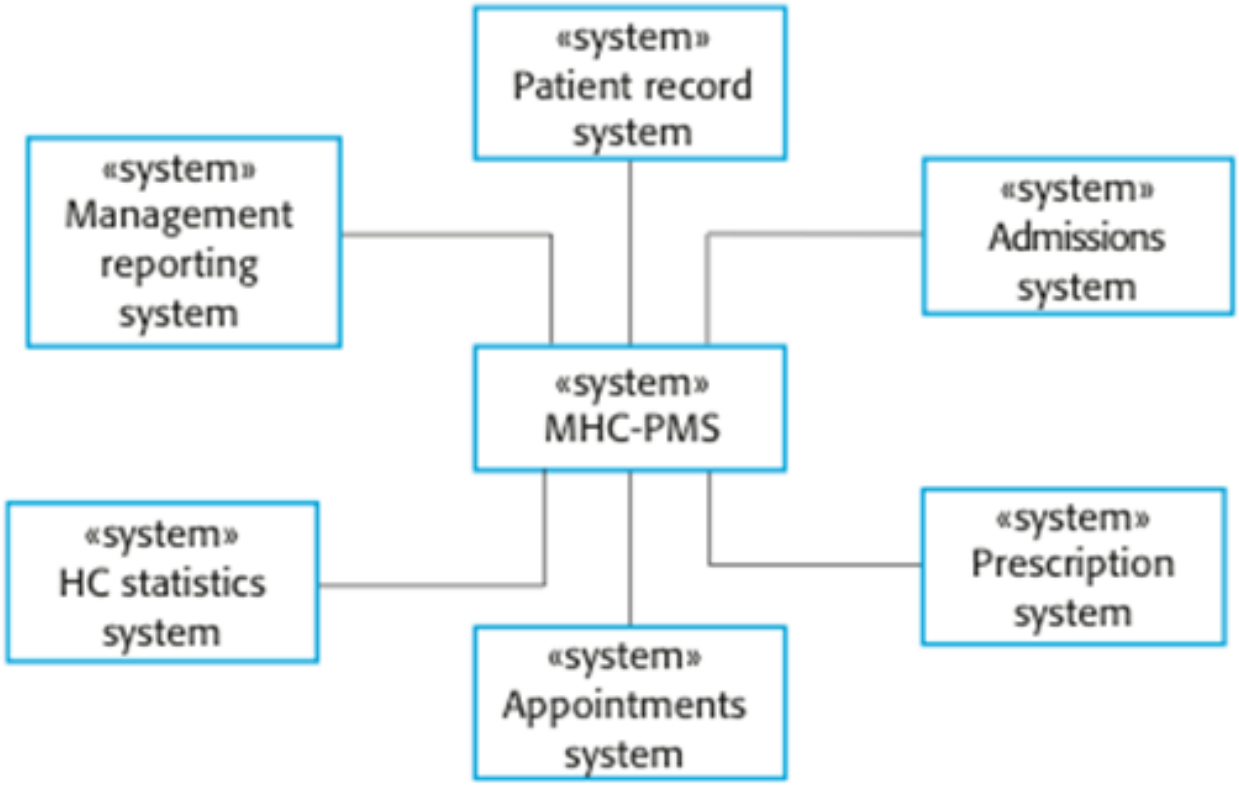
- System boundaries are established to define what is inside and what is outside the system.
  - They show other systems that are used or depend on the system being developed.
- The position of the system boundary has a profound effect on the system requirements.
- Defining a system boundary is a political judgment
  - There may be pressures to develop system boundaries that increase / decrease the influence or workload of different parts of an organization.



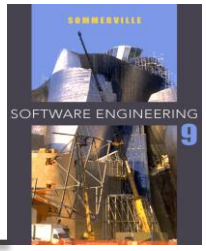
CASE STUDY

- **Mental Health Care - Patient Management System (MHC-PMS)**

The context of the MHC-PMS

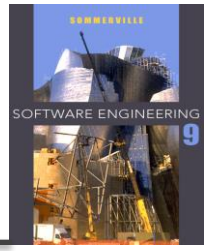


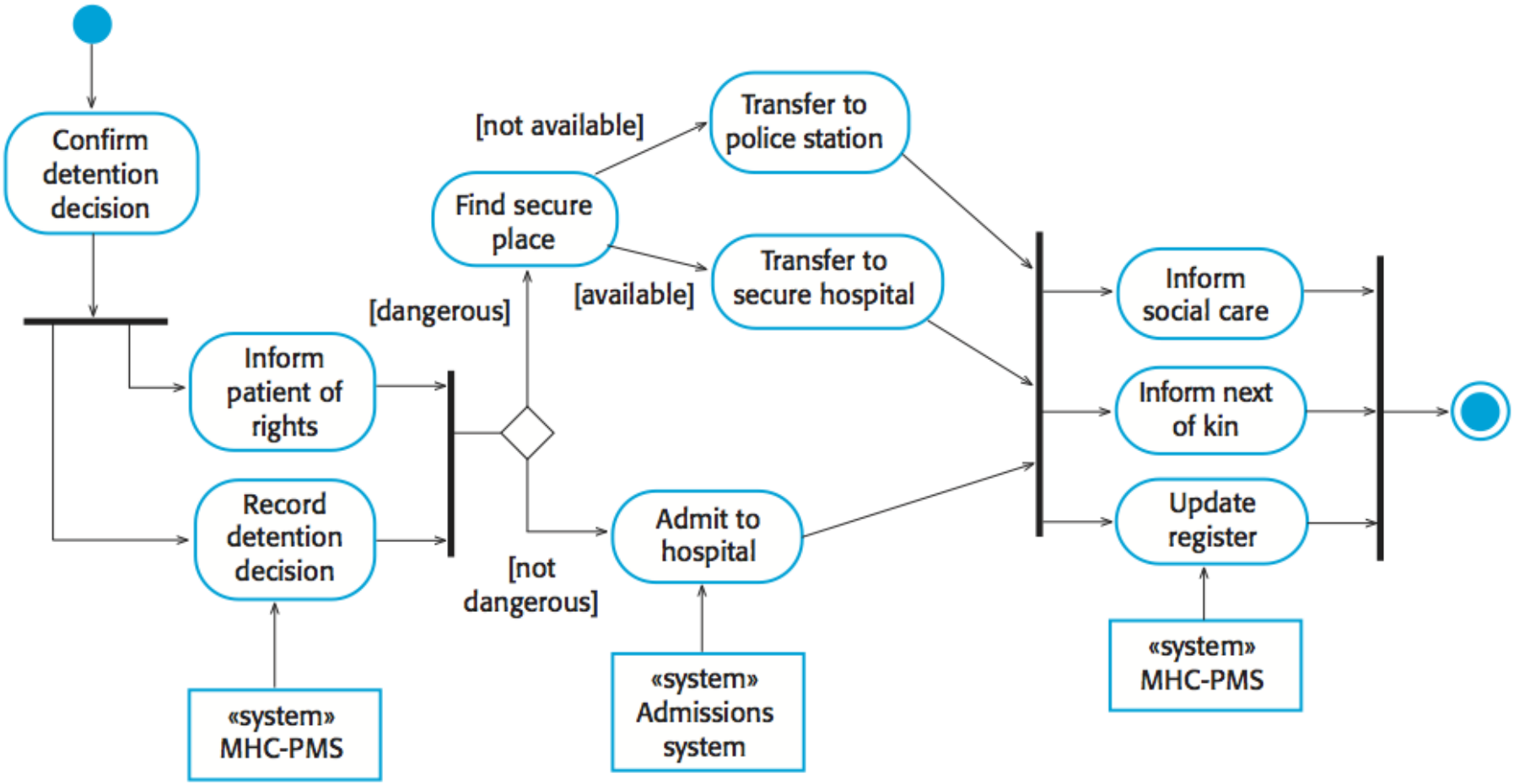
Process Perspective



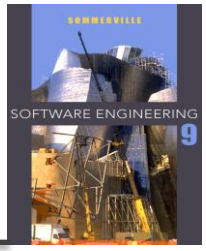
- Context models simply show the other systems in the environment, not how the system being developed is used in that environment.
- Process models reveal how the system being developed is used in broader business processes.
- UML activity diagrams may be used to define business process models.

Process model of involuntary detention



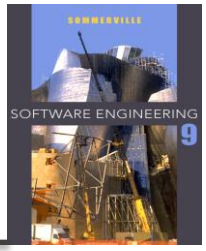


Interaction Models



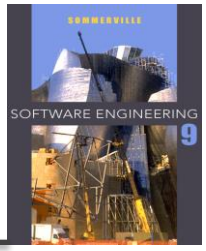
- Modeling user interaction is important as it helps to identify user requirements.
- Modeling system-to-system interaction highlights the communication problems that may arise.
- Modeling component interaction helps us understand if a proposed system structure is likely to deliver the required system performance and dependability.
- Use case diagrams and sequence diagrams may be used for interaction modeling.

Use Case Modeling

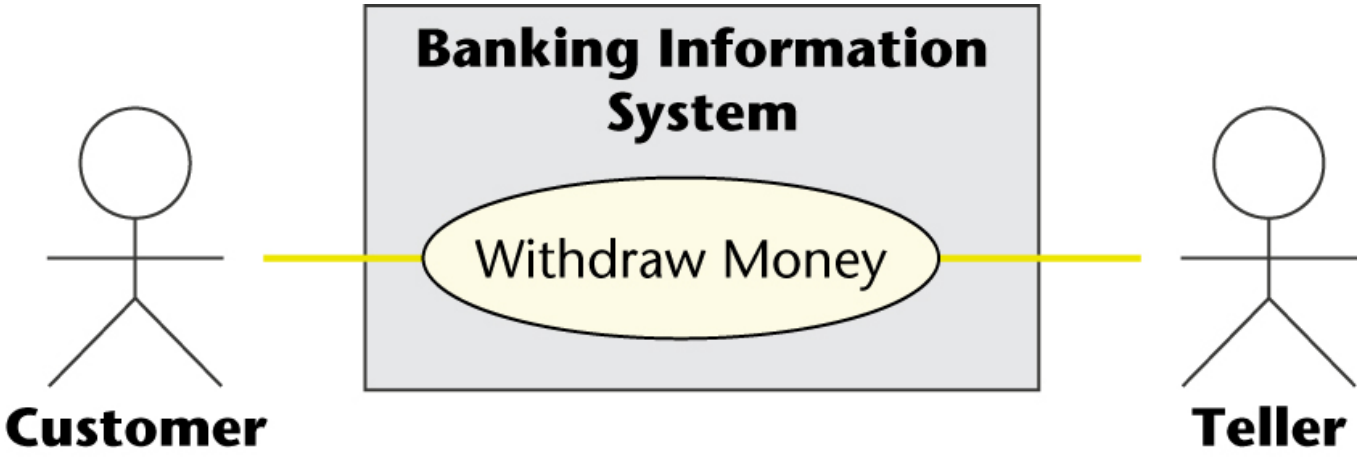


- Use cases were developed originally to support requirements elicitation and now incorporated into the UML.
- Each use case represents a discrete task that involves external interaction with a system.
- Actors in a use case may be people or other systems.
- Represented diagrammatically to provide an overview of the use case and in a more detailed textual form.

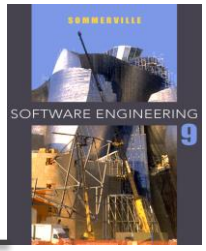
Use Cases



- A use case models an interaction between the information system itself and the users of that information system (*actors*)
- **Example 01:**

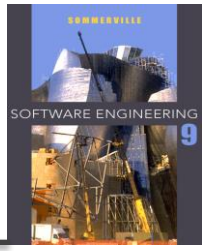


Use Cases



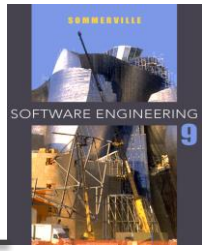
- A use case shows the interaction between
  - The information system and
  - The environment in which the information system operates
- Each use case models one type of interaction
  - There can be just a few use cases for an information system, or there can be hundreds
- The rectangle in the use case represents the information system itself

Actors



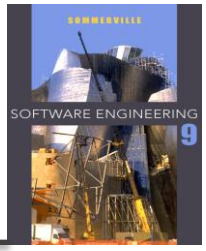
- An actor is a member of the world outside the information system
- It is usually easy to identify an actor
  - An actor is frequently a user of the information system
- In general, an actor plays a role with regard to the information system. This role is
  - As a user; or
  - As an initiator; or
  - As someone who plays a critical part in the use case
- A user of the system can play more than one role

Actors



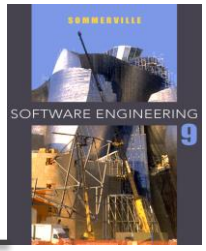
- Example: A customer of the bank can be
  - A Borrower or
  - A Lender
- Conversely, one actor can be a participant in multiple use cases
- Example: A Borrower may be an actor in
  - The Borrow Money use case;
  - The Pay Interest on Loan use case; and
  - The Repay Loan Principal use case
- Also, the actor Borrower may stand for many thousands of bank customers

Actors

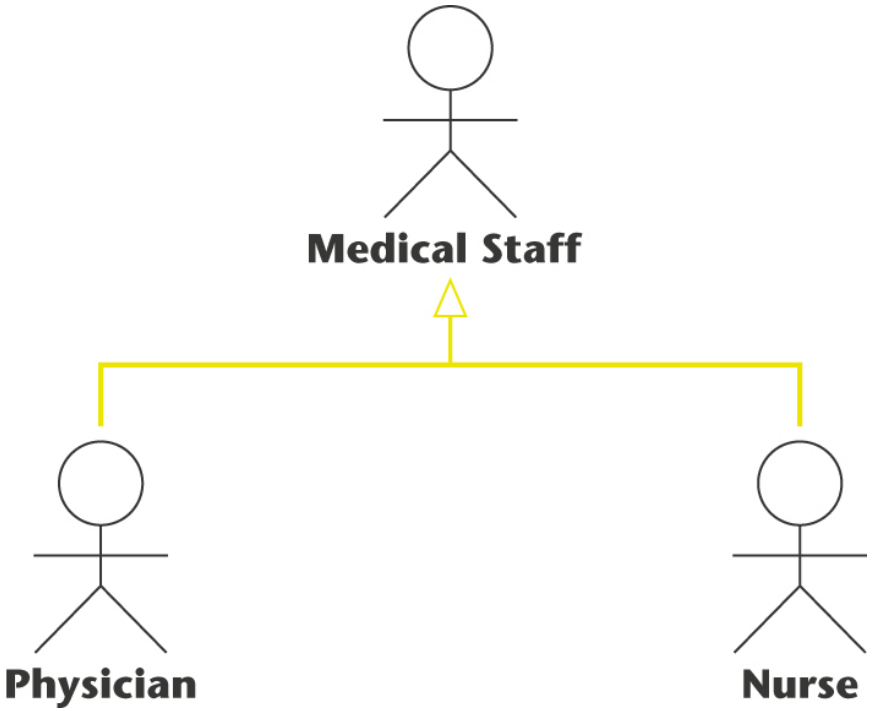


- An actor needs not be a human being
- Example: The Cardholder Clothing Company information system has to interact with the credit card company information system
  - The credit card company information system is an actor from the viewpoint of the Cardholder Clothing Company information system
  - The Cardholder Clothing Company information system is an actor from the viewpoint of the credit card company information system
- A potential problem when identifying actors
  - Overlapping actors
- Example: Hospital Information System
  - One use case has actor Nurse
  - A different use case has actor Medical Staff
  - Better:
    - Actors: Physician and Nurse

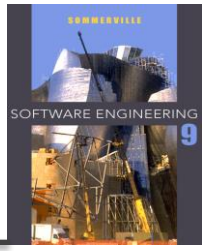
Actors



- Alternatively:
  - Actor Medical Staff with two specializations: Physician and Nurse

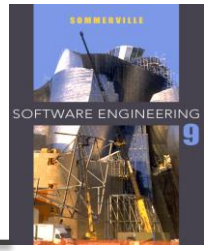


Case Study - Osbert Oglesby



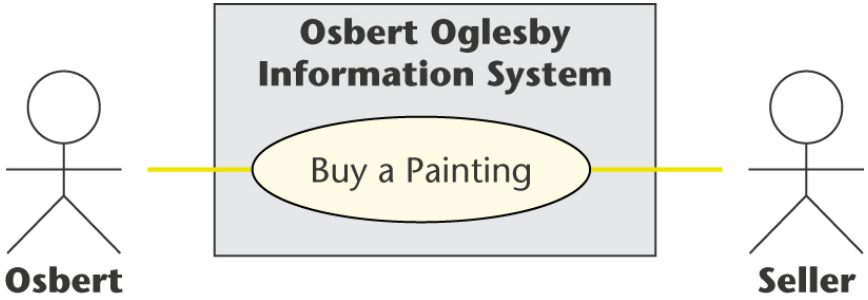
- Osbert wants an information system, running on his laptop computer, that will
  - Determine the maximum price he should pay for a painting
  - Detect new trends in the art market as soon as possible
- To do this, the information system needs to keep a record of all purchases and all sales
- Currently, Osbert produces reports of annual sales and purchases by hand
  - At only a small additional cost, the information system can also print these two reports on demand
- Osbert *wants* an information system that can
  - Compute the highest price he should pay for a painting; and
  - Detect new art trends
- Osbert *needs* an information system that can also
  - Provide reports on purchases and sales
- It is vital to determine the client's needs up front, and not after the information system has been delivered

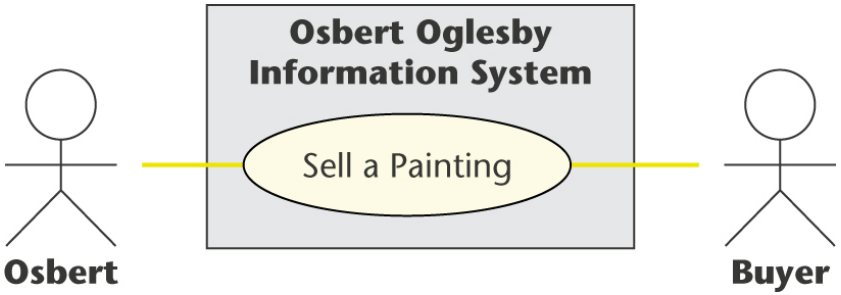
Case Study - Osbert Oglesby

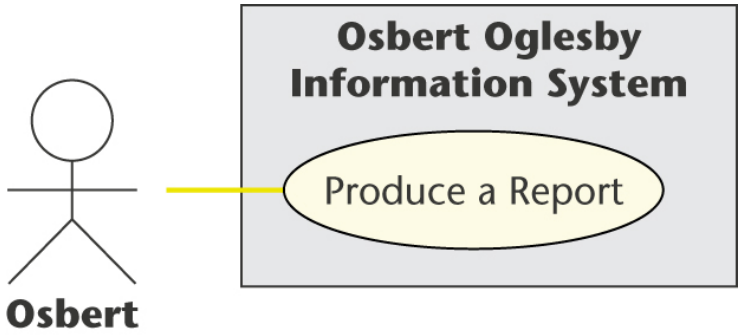


✧ Osbert has three business activities:

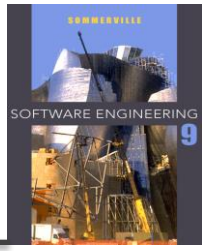
- He buys paintings
- He sells paintings
- He produces reports



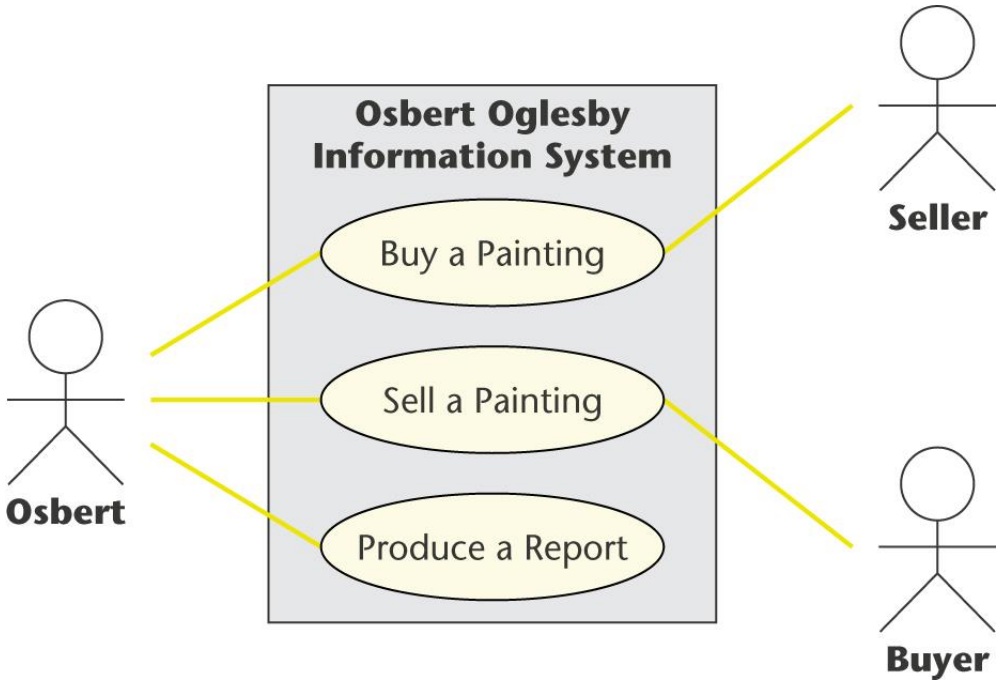




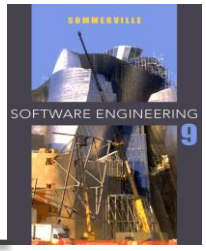
Case Study - Osbert Oglesby



✧ For conciseness, all three use cases are combined into a *use-case diagram*

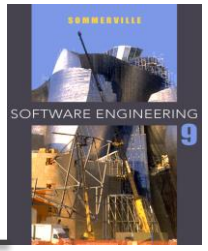


Case Study - Osbert Oglesby



- The only person who uses the current (manual) information system is Osbert
  - Osbert is therefore an actor in all three use cases
- The customer may initiate the Buy a Painting or the Sell a Painting use case
- The customer plays a critical part in both use cases by providing data entered into the information system by Osbert
  - The customer is therefore an actor in both these use cases
- Next, the use cases have to be annotated

Case Study - Osbert Oglesby

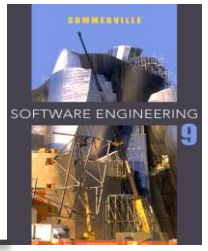


✧ Here are the initial use-case descriptions

Buy a Painting use case

Brief Description
The `Buy a Painting` use case enables Osbert Oglesby to buy a painting.
Step-by-Step Description
1. Osbert inputs details of the painting he is considering buying. 2. The information system responds with the maximum purchase price he should offer. 3. If the seller accepts Osbert's offer to buy the painting, Osbert enters further details. *Note:* Details of the algorithm for determining the maximum price will be obtained later.

Sell a Painting use case



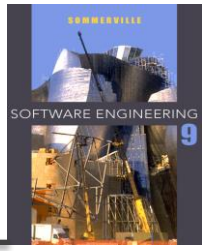
Brief Description

The `Sell a Painting` use case enables Osbert Oglesby to sell a painting.

Step-by-Step Description

1. Osbert inputs details of the painting he has sold.

Produce a Report use case



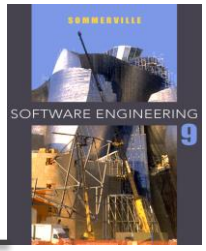
Brief Description

The `Produce a Report` use case enables Osbert Oglesby to obtain information on paintings he has bought or sold in the past year or to detect new trends in the art market.

Step-by-Step Description

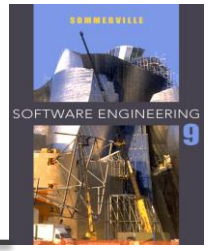
1. Osbert requests a report of the type that he needs. The report is printed.

Some Relationship Between Use Cases…



- **Extend** is used when a use case adds steps to another first class use case.
  - For example, imagine "Withdraw Cash" is a use case of an ATM machine. "Assess Fee" would extend Withdraw Cash and describe the *conditional* "extension point" that is instantiated when the ATM user doesn't bank at the ATM's owning institution. Notice that the basic "Withdraw Cash" use case stands on its own, without the extension.
- **Include** is used to extract use case fragments that are *duplicated* in multiple use cases. The included use case cannot stand alone and the original use case is not complete without the included one. This should be used sparingly an only in cases where the duplication is significant and exists by design (rather than by coincidence).
  - For example, the flow of events that occurs at the beginning of every ATM use case (when the user puts in their ATM card, enters their PIN, and is shown the main menu) would be a good candidate for an **include**.

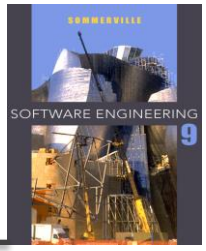
Transfer-data use case



✧ **Example 02:**

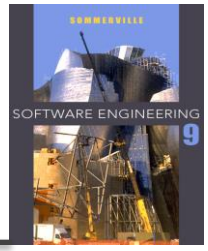
✧ A use case in the MHC-PMS

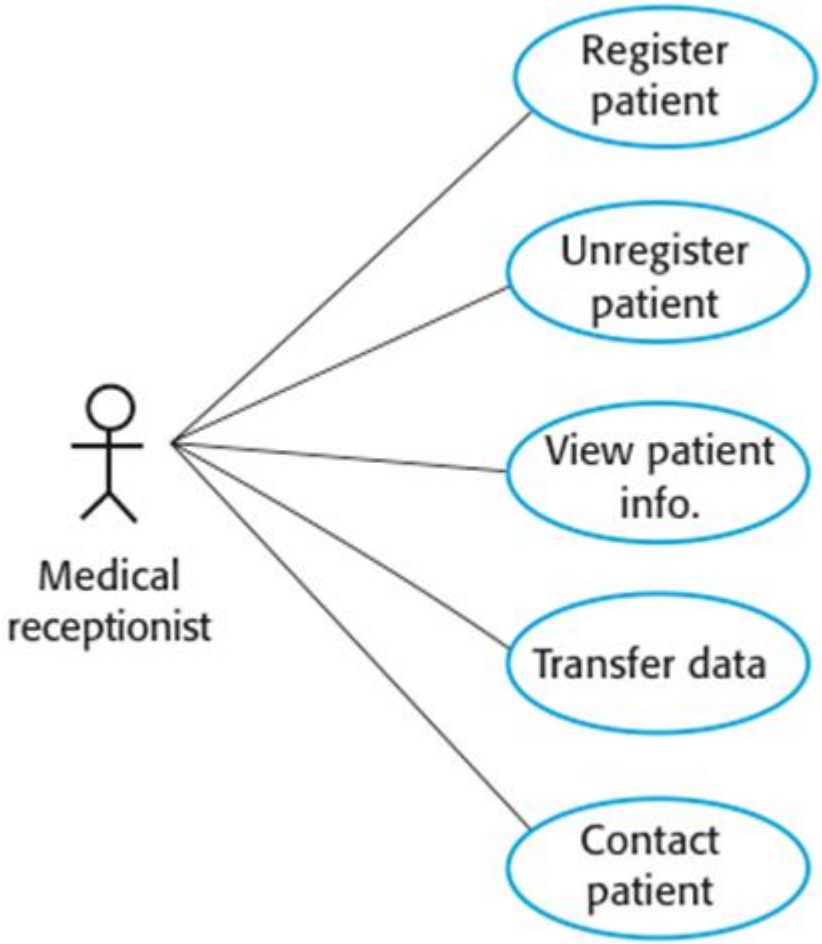
Tabular description of the ‘Transfer data’ use-case

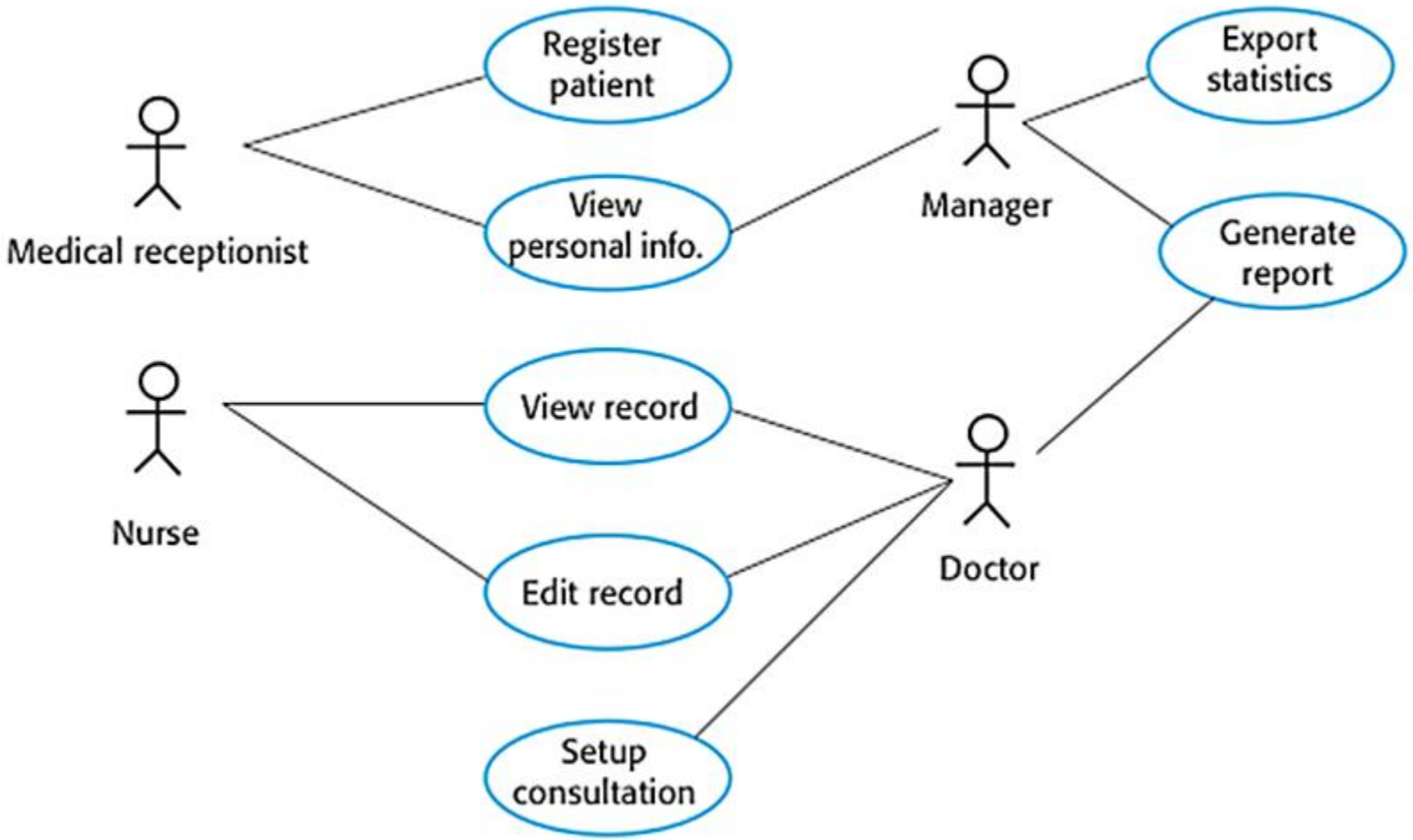


MHC-PMS: Transfer data	
Actors	Medical receptionist, patient records system (PRS)
Description	A receptionist may transfer data from the MHC-PMS to a general patient record database that is maintained by a health authority. The information transferred may either be updated personal information (address, phone number, etc.) or a summary of the patient’s diagnosis and treatment.
Data	Patient’s personal information, treatment summary
Stimulus	User command issued by medical receptionist
Response	Confirmation that PRS has been updated
Comments	The receptionist must have appropriate security permissions to access the patient information and the PRS.

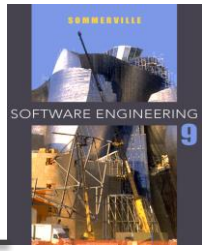
Use cases in the MHC-PMS involving the role ‘Medical Receptionist’






Register patient
Export statistics
Medical receptionist
View personal info.
Manager
Generate report
View record
Nurse
Doctor
Edit record
Setup consultation

Exercise 1 – Use Case



Customers bring their cars to the garage for servicing and repairing. The attendant must check the car in, record details about the owner and the car, along with any specific customer requests. The workshop manager inspects each car and creates a job specification for it. He then schedules the job and assigns a mechanic to complete the specified tasks. When the job is finished the mechanic completes a report detailing the time spent, work done and materials used. Then the work the workshop manager re-inspects the car and report created by the mechanic. The attendant creates an invoice for the customers based on the report, when they come to collect their car uses this information.

Draw a Use Case diagram to represent this scenario.

Exercise 2 – Use Case

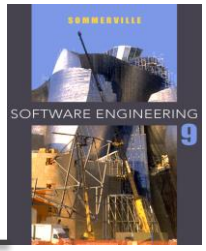
The following is a description of the business process of organisation of conferences with regards to the submitting, reviewing and accepting papers.

The author completes an online form that requests the user to input author name, correspondence address, email and, title of paper. The system validates this data and, if correct, asks the author to submit the paper. The author then browses to find the correct paper on their system and submits it. Once received and stored, the system returns to the author a reference number for the paper. Authors may submit as many papers as they like to be considered for acceptance to the conference up until the deadline date for submissions. Papers are allocated to referees for assessment. They review each paper and submit to the system their decision. Once the programme organiser has agreed the decisions authors are informed by email. Accepted papers are then schedule to be delivered at a conference. This involves allocating a date, time and place for the presentation of the paper.

i. Analyse the above text and draw a use case diagram using standard UML notations. Clearly indicate if you made any assumption.

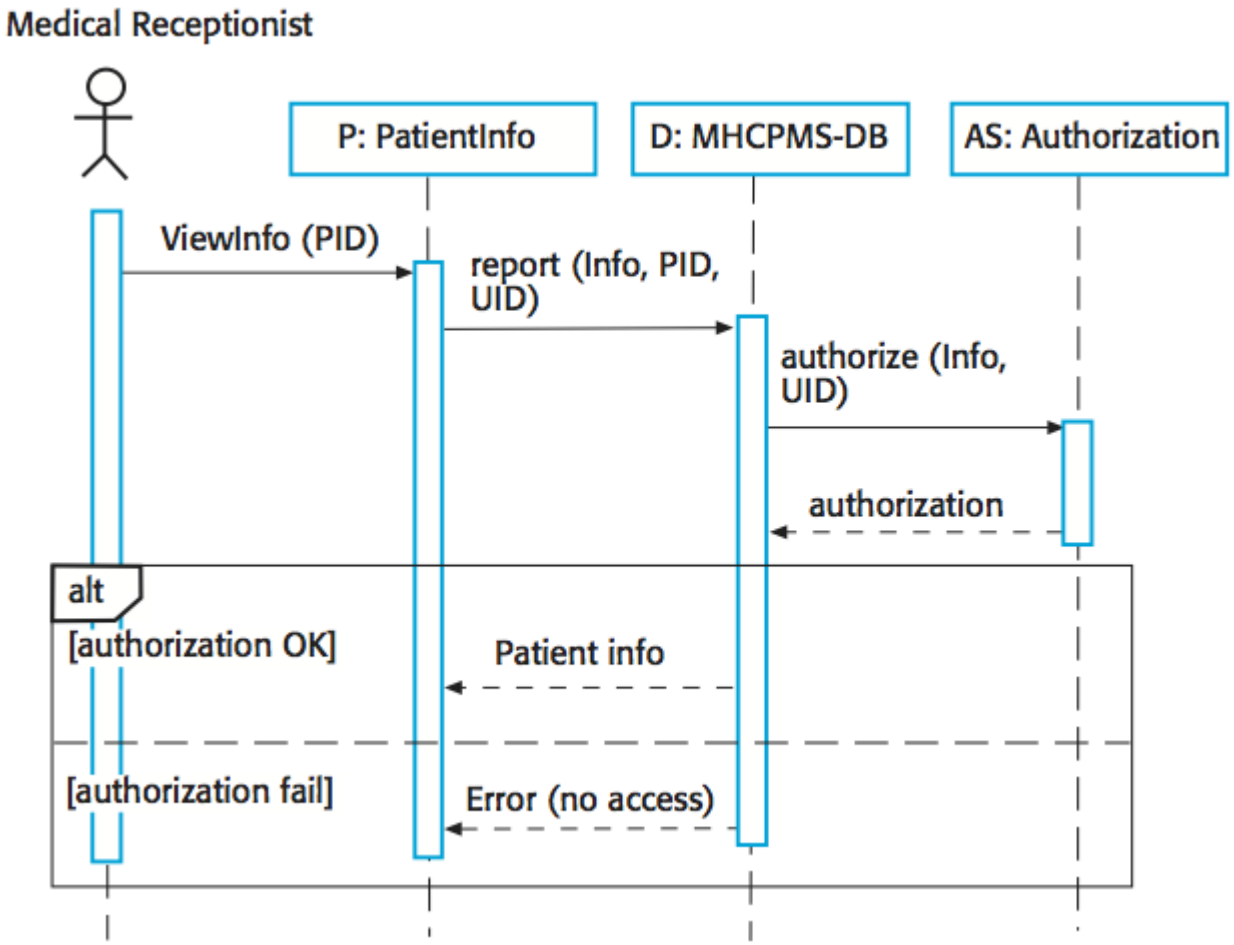
ii. Write use-case descriptions for any two use cases in your diagram.

Sequence diagrams

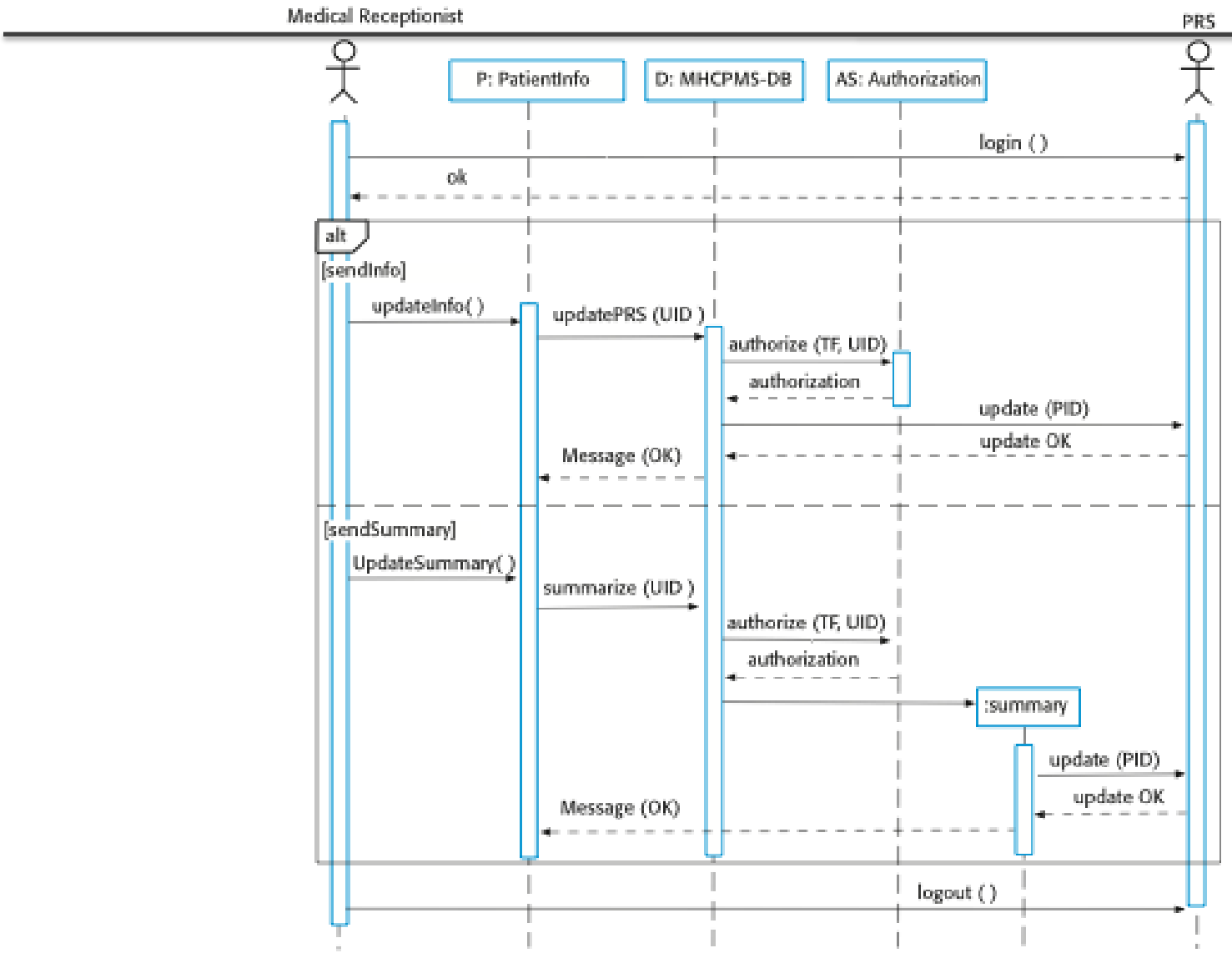


- Sequence diagrams are part of the UML and are used to model the interactions between the actors and the objects within a system.
- A sequence diagram shows the sequence of interactions that take place during a particular use case or use case instance.
- The objects and actors involved are listed along the top of the diagram, with a dotted line drawn vertically from these.
- Interactions between objects are indicated by annotated arrows.

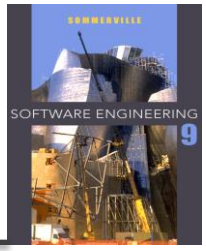
Sequence diagram for View patient information



Sequence diagram for Transfer Data

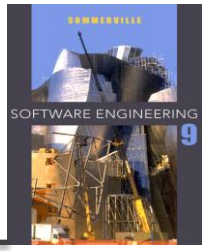


Structural models



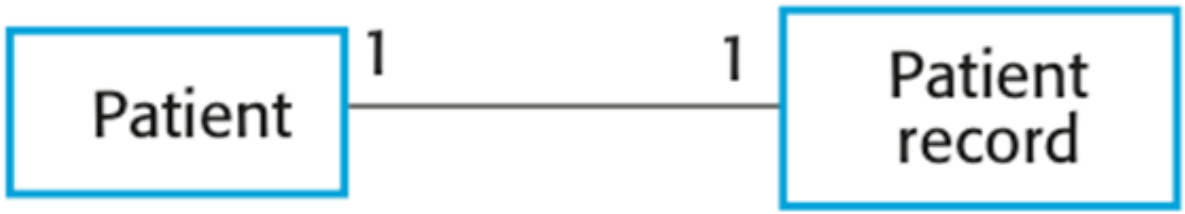
- Structural models of software display the organization of a system in terms of the components that make up that system and their relationships.
- Structural models may be static models, which show the structure of the system design, or dynamic models, which show the organization of the system when it is executing.
- You create structural models of a system when you are discussing and designing the system architecture.

Class diagrams

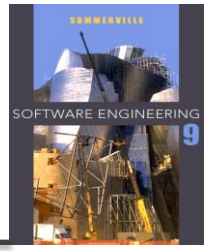


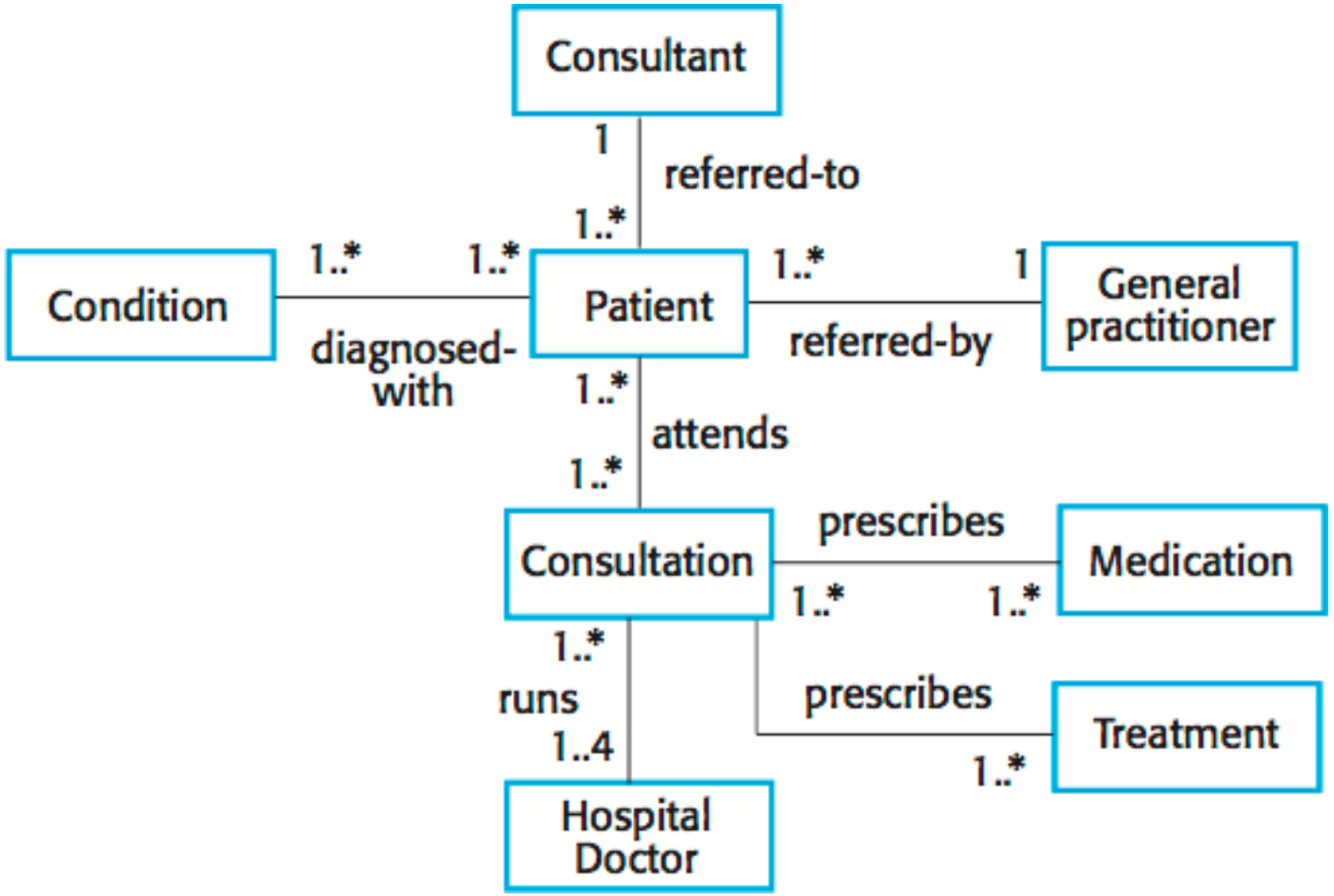
- Class diagrams are used when developing an object-oriented system model to show the classes in a system and the associations between these classes.
- An object class can be thought of as a general definition of one kind of system object.
- An association is a link between classes that indicates that there is some relationship between these classes.
- When you are developing models during the early stages of the software engineering process, objects represent something in the real world, such as a patient, a prescription, doctor, etc.

UML classes and association

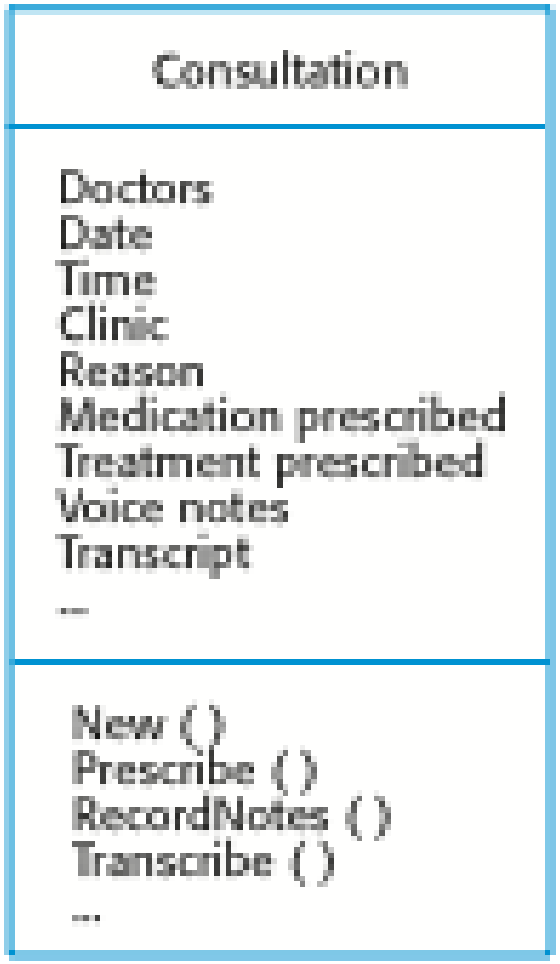


Classes and associations in the MHC-PMS

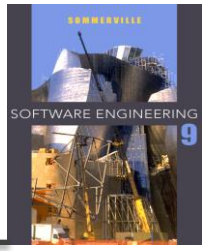




The Consultation class

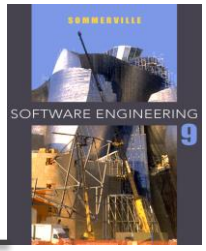


Key points



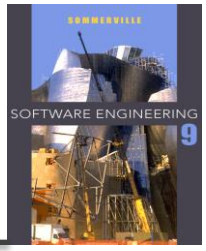
- A model is an abstract view of a system that ignores system details. Complementary system models can be developed to show the system's context, interactions, structure and behavior.
- Context models show how a system that is being modeled is positioned in an environment with other systems and processes.
- Use case diagrams and sequence diagrams are used to describe the interactions between users and systems in the system being designed. Use cases describe interactions between a system and external actors; sequence diagrams add more information to these by showing interactions between system objects.
- Structural models show the organization and architecture of a system. Class diagrams are used to define the static structure of classes in a system and their associations.

Generalization



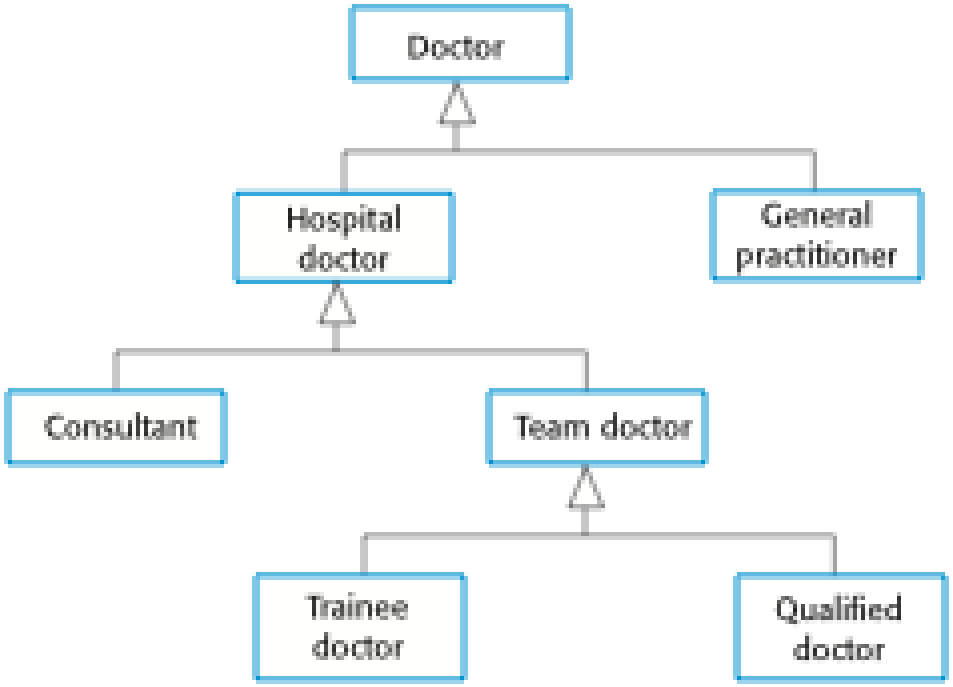
- Generalization is an everyday technique that we use to manage complexity.
- Rather than learn the detailed characteristics of every entity that we experience, we place these entities in more general classes (animals, cars, houses, etc.) and learn the characteristics of these classes.
- This allows us to infer that different members of these classes have some common characteristics e.g. squirrels and rats are rodents.

Generalization

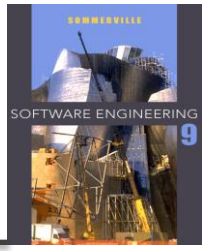


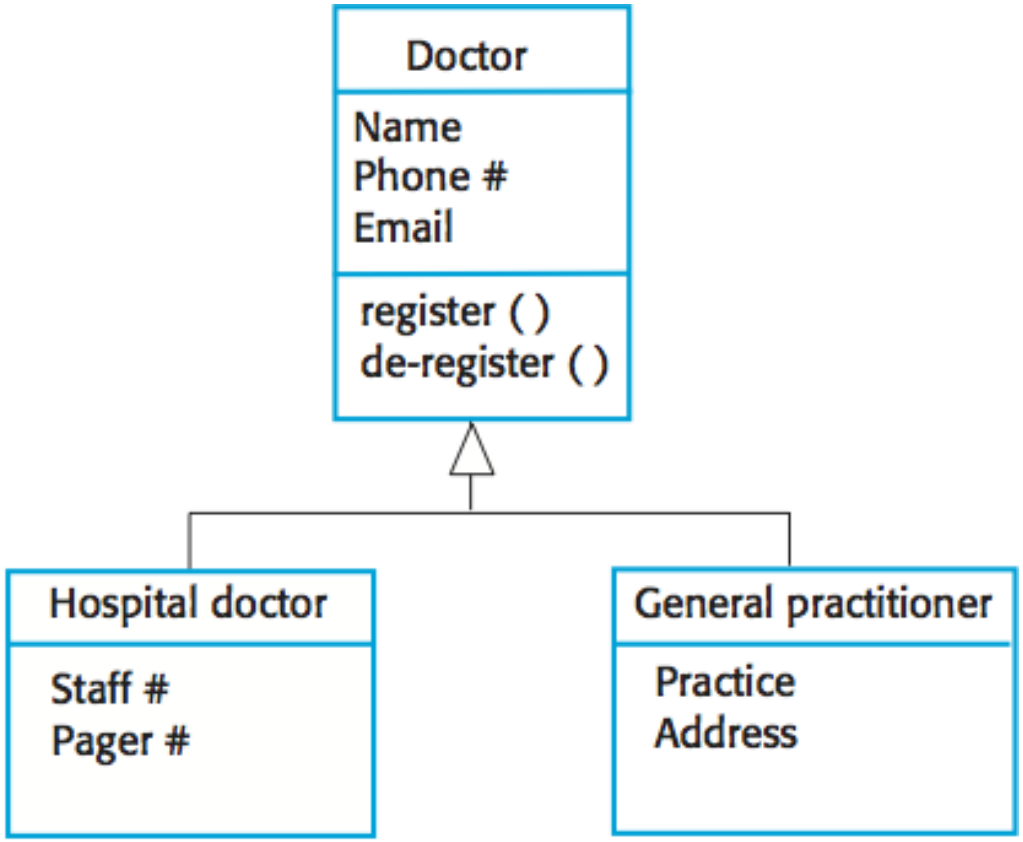
- In modeling systems, it is often useful to examine the classes in a system to see if there is scope for generalization. If changes are proposed, then you do not have to look at all classes in the system to see if they are affected by the change.
- In object-oriented languages, such as Java, generalization is implemented using the class inheritance mechanisms built into the language.
- In a generalization, the attributes and operations associated with higher-level classes are also associated with the lower-level classes.
- The lower-level classes are subclasses inherit the attributes and operations from their superclasses. These lower-level classes then add more specific attributes and operations.

A generalization hierarchy

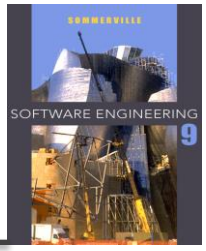


A generalization hierarchy with added detail



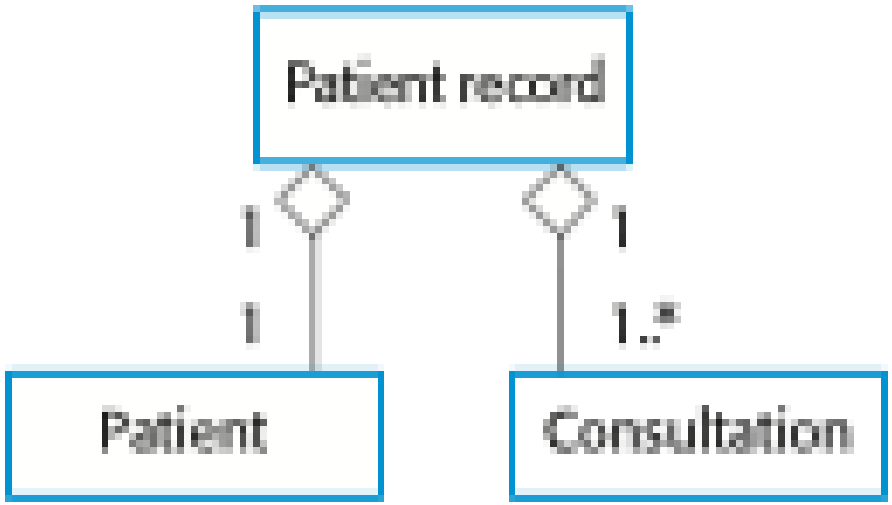


Object class aggregation models

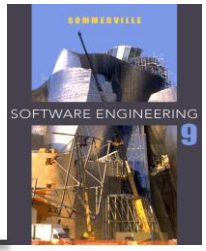


- An aggregation model shows how classes that are collections are composed of other classes.
- Aggregation models are similar to the part-of relationship in semantic data models.

The aggregation association

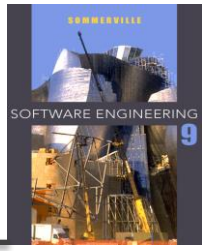


Behavioral models

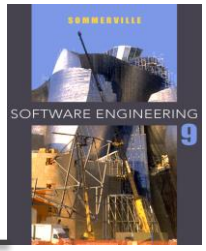


- Behavioral models are models of the dynamic behavior of a system as it is executing. They show what happens or what is supposed to happen when a system responds to a stimulus from its environment.
- You can think of these stimuli as being of two types:
  - Data Some data arrives that has to be processed by the system.
  - Events Some event happens that triggers system processing. Events may have associated data, although this is not always the case.

Data-driven modeling



- Many business systems are data-processing systems that are primarily driven by data. They are controlled by the data input to the system, with relatively little external event processing.
- Data-driven models show the sequence of actions involved in processing input data and generating an associated output.
- They are particularly useful during the analysis of requirements as they can be used to show end-to-end processing in a system.



Questions?