

Software Engineering

Prepared by Dr. Yasser
Abdelhamid

IT 382
Software Engineering

Lecture 8



LECTURE OUTLINE

- Context models
- Interaction models
- Structural models
- Behavioral models

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 is a way of **representing** a system using **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.

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.**

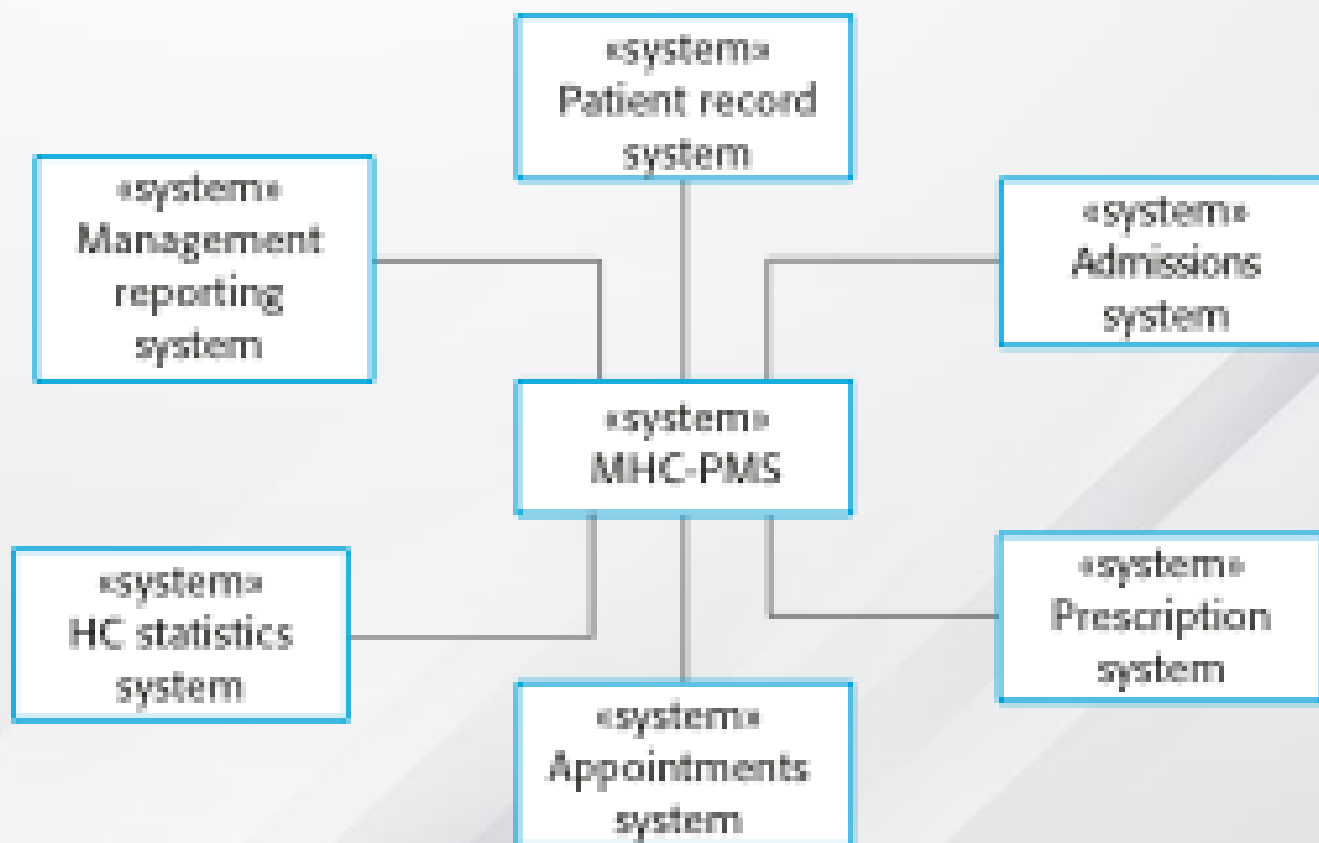
CONTEXT MODELS

- 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 deep effect on the **system requirements**.

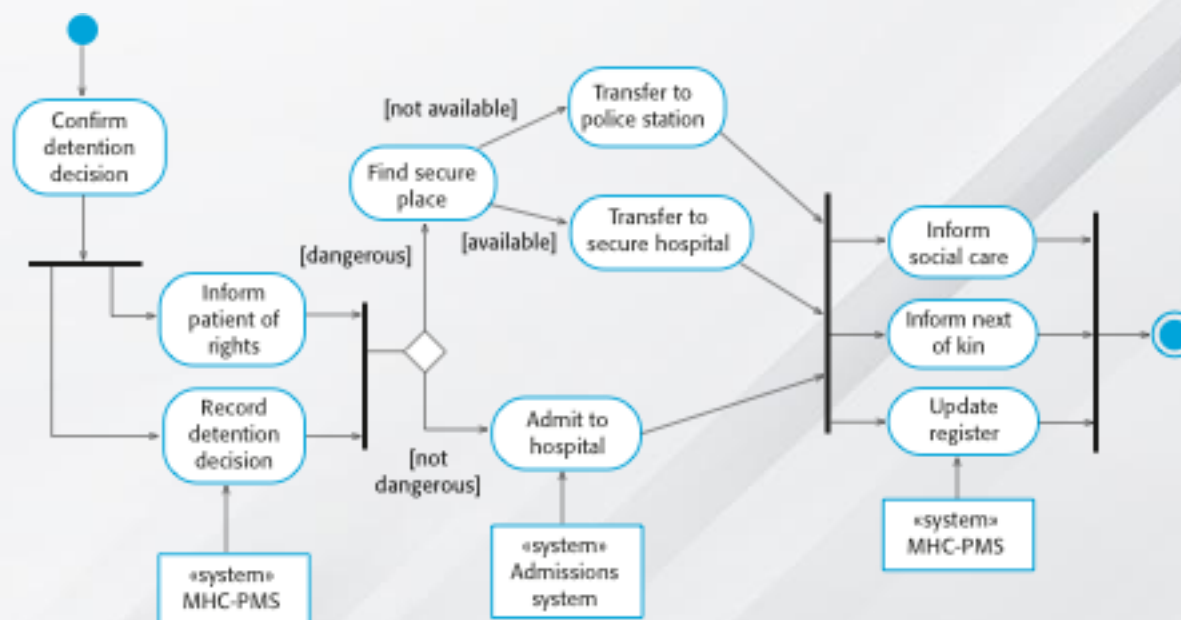
THE CONTEXT OF THE MHC-PMS



PROCESS PERSPECTIVE → USING ACTIVITY MODEL

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

PROCESS MODEL OF INVOLUNTARY DETENTION



INTERACTION MODELS → USE CASE DIAGRAMS AND SEQUENCE DIAGRAMS

- **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 modelling.

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**.

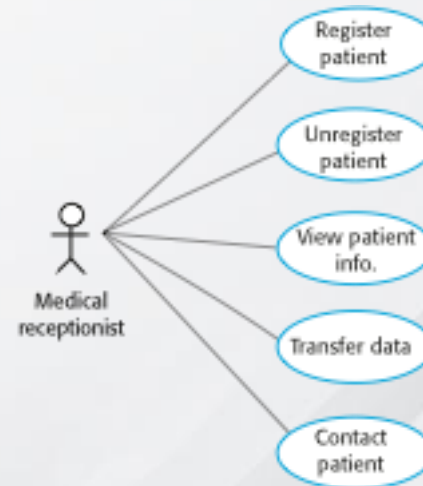
TRANSFER-DATA USE CASE



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.

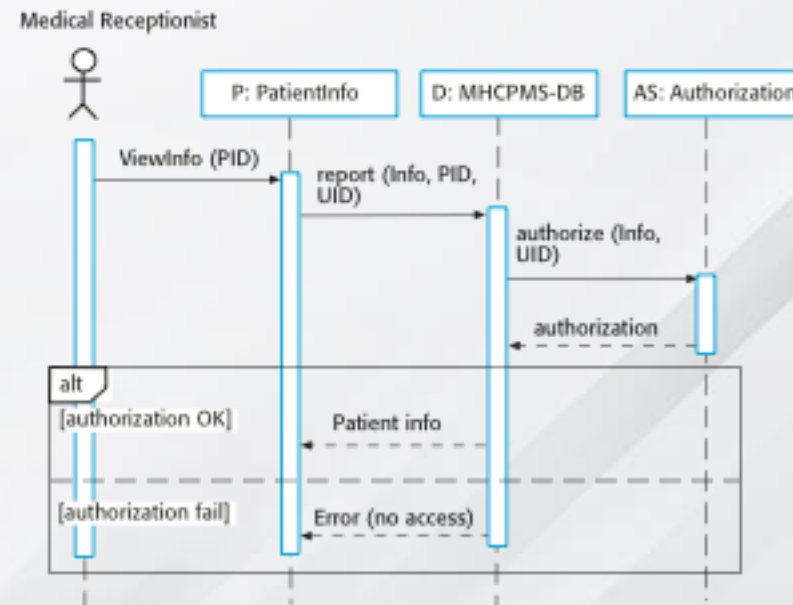
USE CASES IN THE MHC-PMS INVOLVING THE ROLE 'MEDICAL RECEPTIONIST' (COMPOSITE)



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



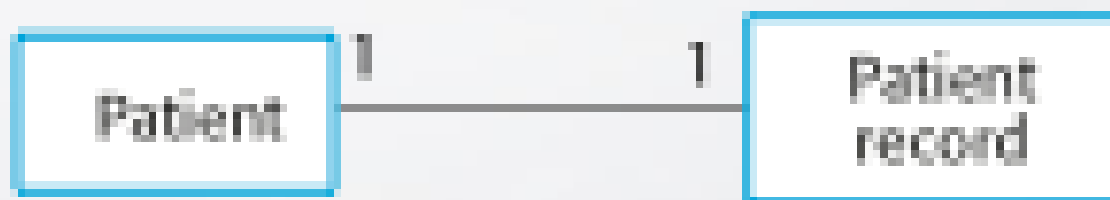
STRUCTURAL MODELS

- 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**.

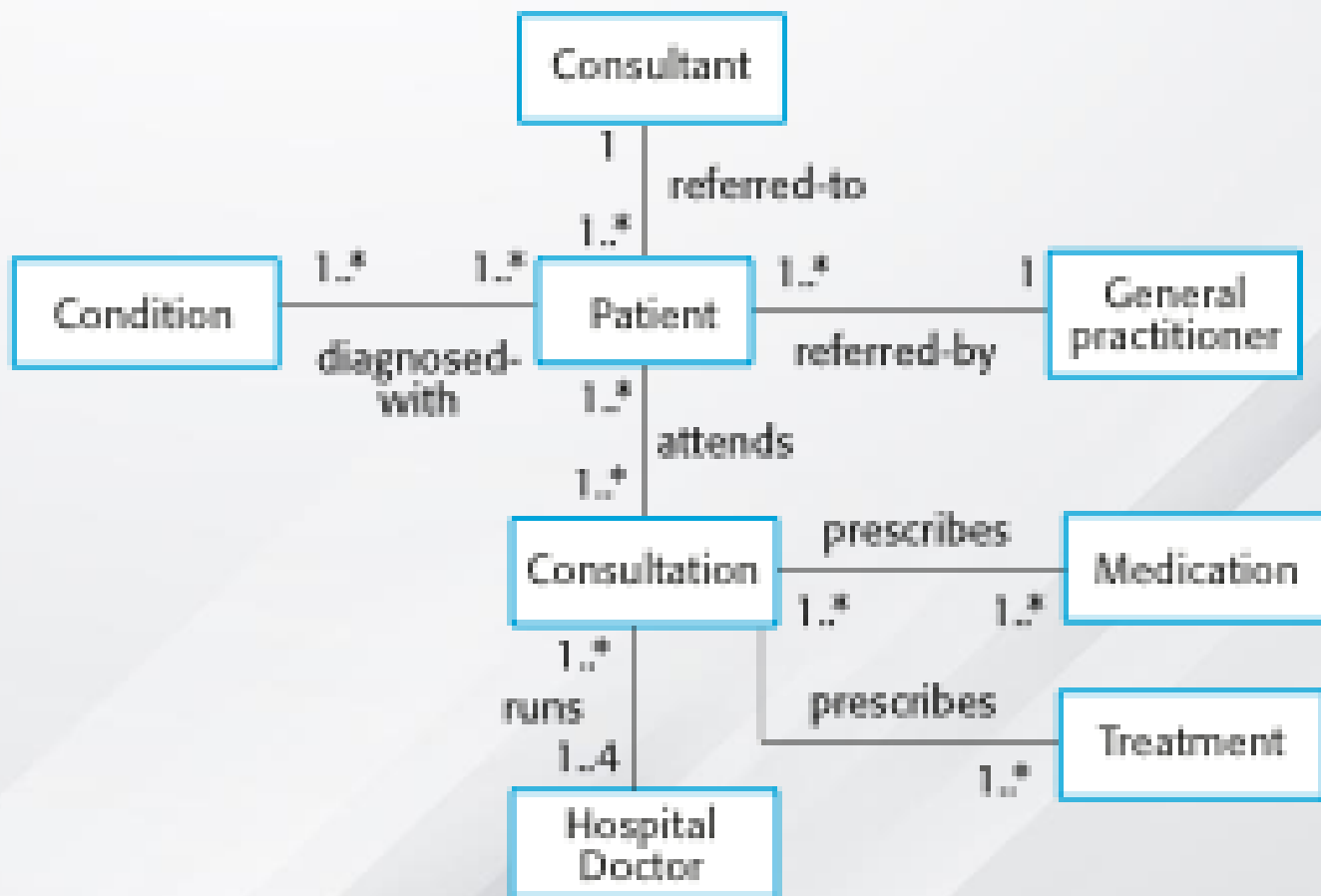
CLASS DIAGRAMS

- 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



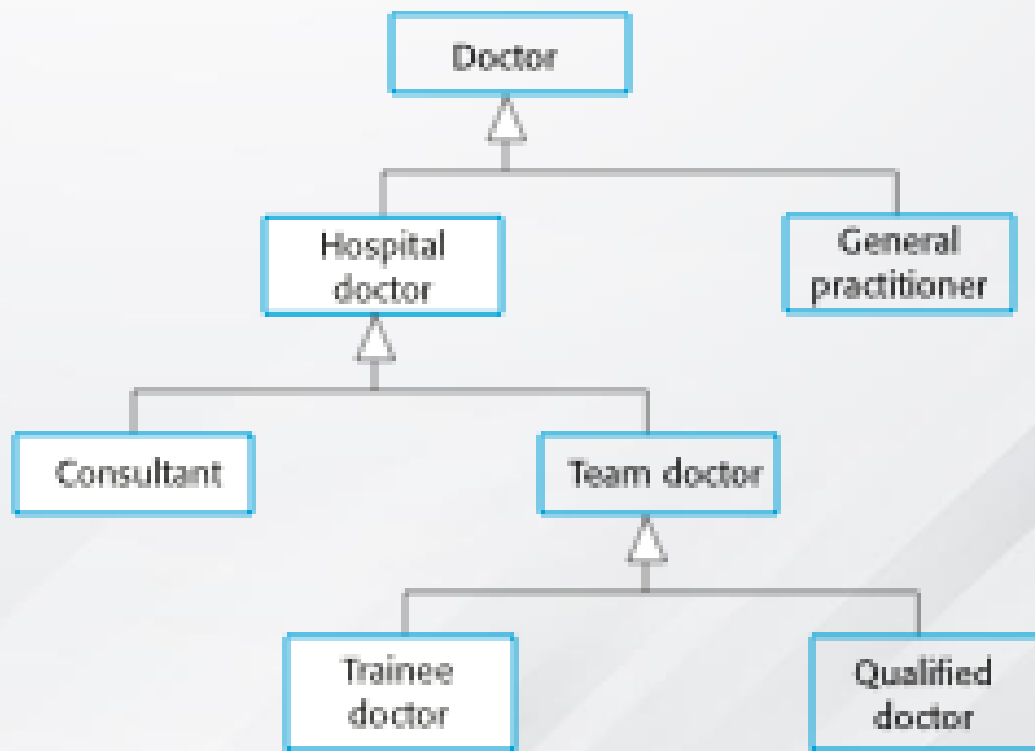
THE CONSULTATION CLASS

Consultation
Doctors Date Time Clinic Reason Medication prescribed Treatment prescribed Voice notes Transcript ...
New () Prescribe () RecordNotes () Transcribe () ...

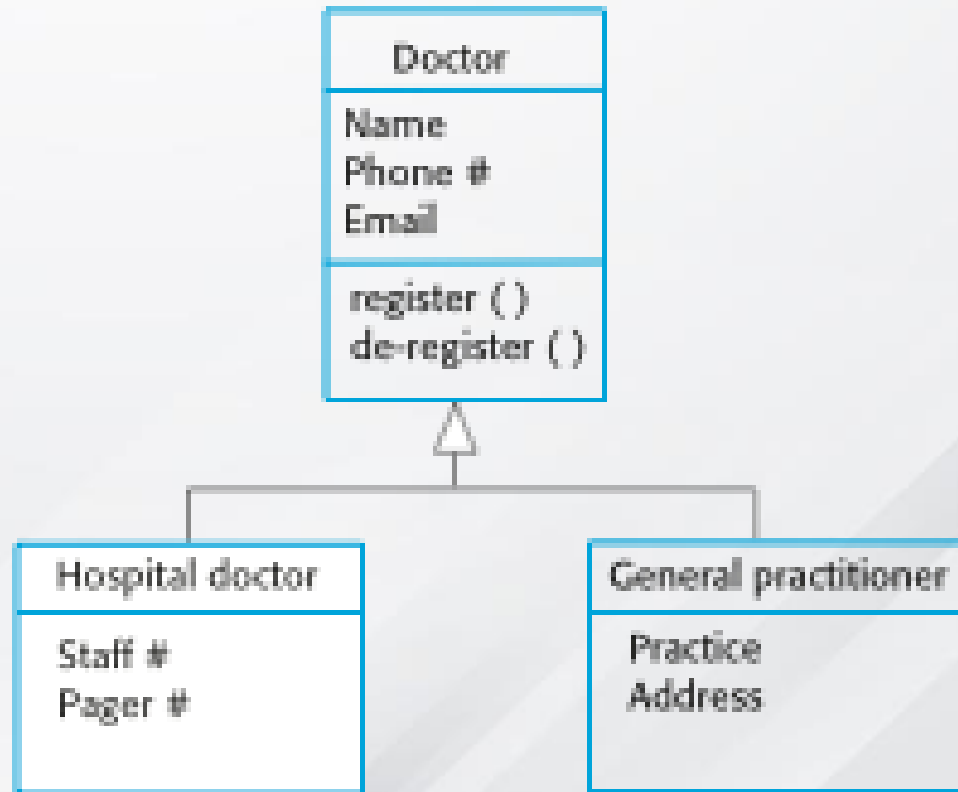
GENERALIZATION

- 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



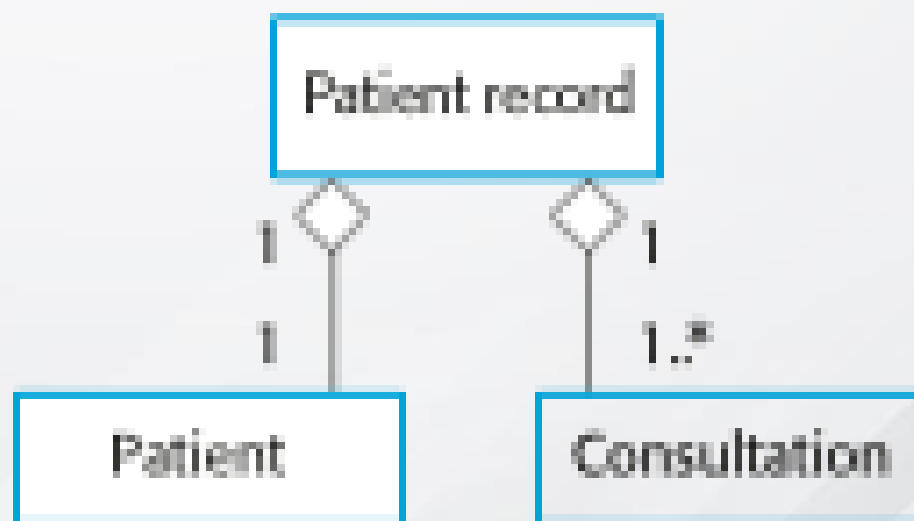
A GENERALIZATION HIERARCHY WITH ADDED DETAIL



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



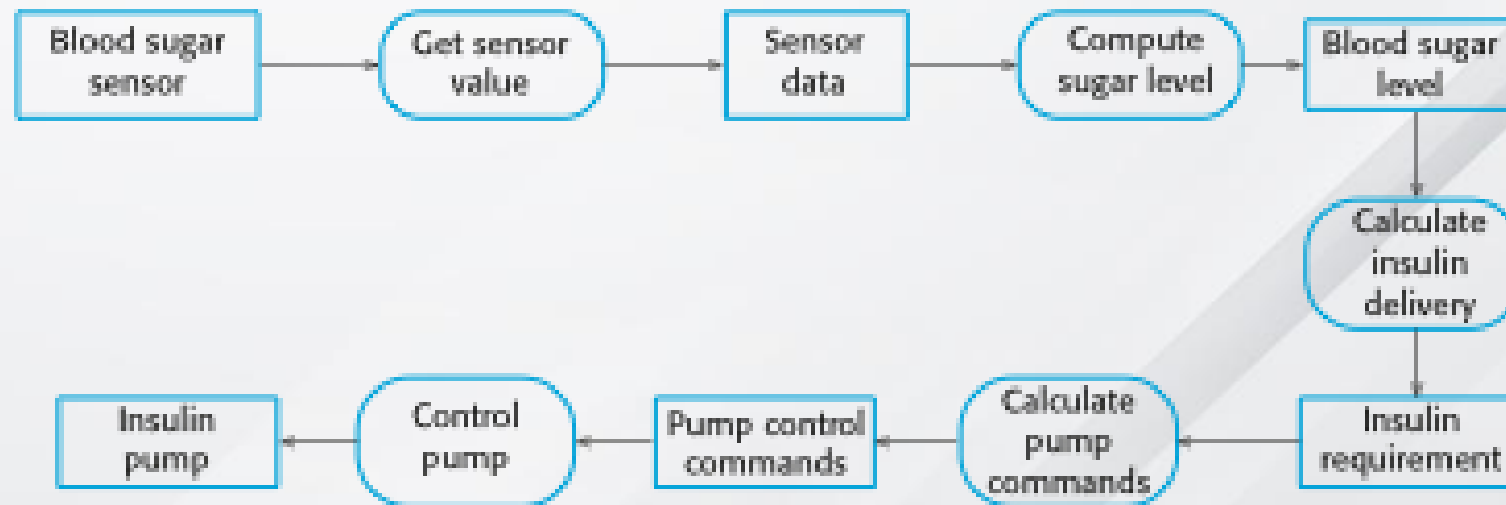
BEHAVIORAL MODELS

- 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.

AN ACTIVITY MODEL OF THE INSULIN PUMP'S OPERATION



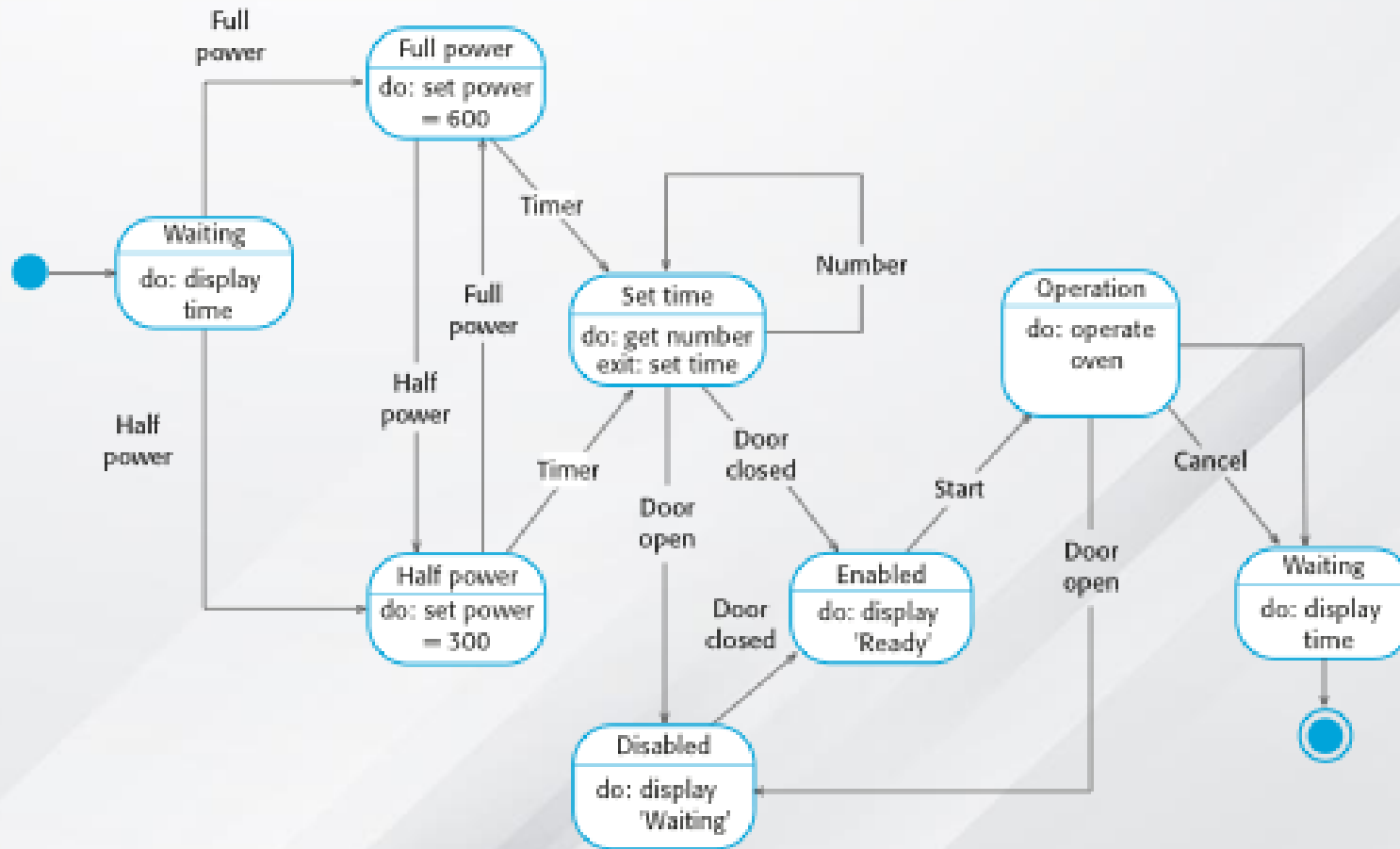
EVENT-DRIVEN MODELING

- Real-time systems are often event-driven, with **minimal data processing**. For example, a landline phone switching system responds to events such as 'receiver off hook' by generating a dial tone.
- Event-driven modeling shows how a system **responds to external and internal events**.
- It is based on the assumption that a system has a **finite number of states** and that events (stimuli) may cause a **transition** from one state to another.

STATE MACHINE MODELS

- These model the behaviour of the system in response to external and internal events.
- They show the system's responses to stimuli so are often used for modelling real-time systems.
- State machine models show system **states as nodes** and **events as arcs** between these nodes. When an event occurs, the system moves from one state to another.
- **Statecharts** are an integral part of the UML and are used to represent state machine models.

STATE DIAGRAM OF A MICROWAVE OVEN



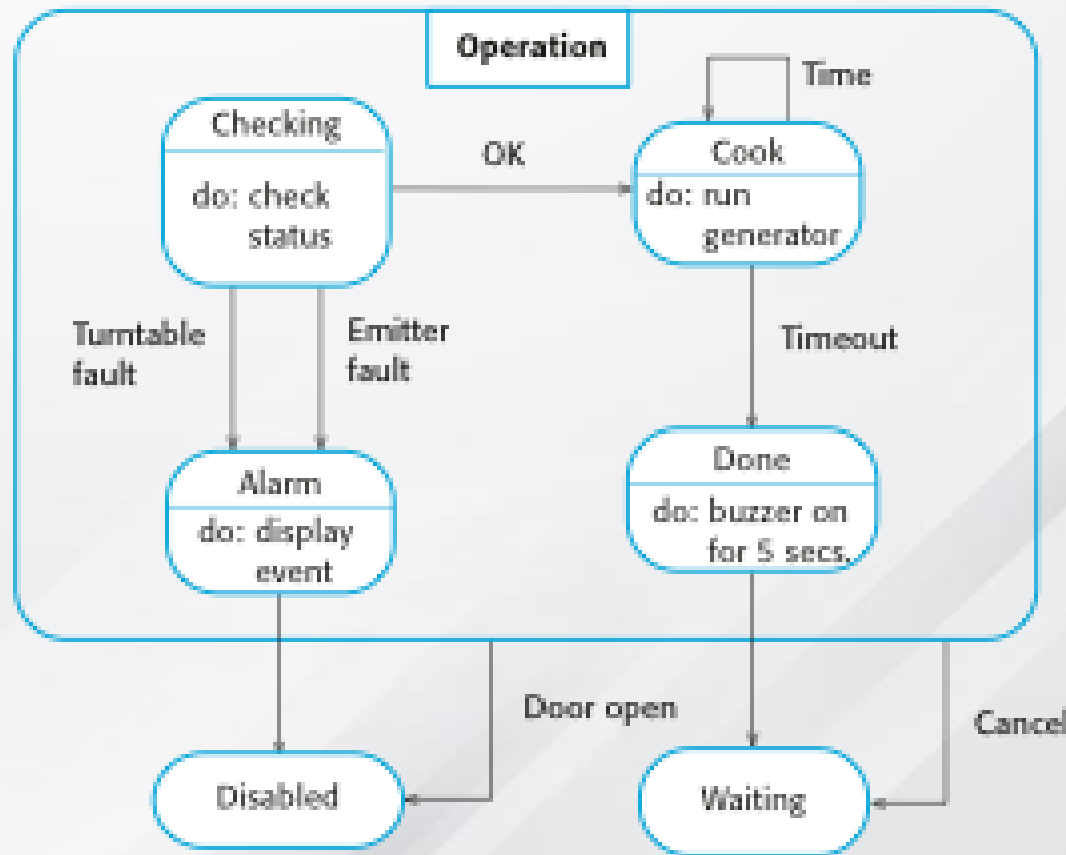
STATES AND STIMULI FOR THE MICROWAVE OVEN (A)

State	Description
Waiting	The oven is waiting for input. The display shows the current time.
Half power	The oven power is set to 300 watts. The display shows 'Half power'.
Full power	The oven power is set to 600 watts. The display shows 'Full power'.
Set time	The cooking time is set to the user's input value. The display shows the cooking time selected and is updated as the time is set.
Disabled	Oven operation is disabled for safety. Interior oven light is on. Display shows 'Not ready'.
Enabled	Oven operation is enabled. Interior oven light is off. Display shows 'Ready to cook'.
Operation	Oven in operation. Interior oven light is on. Display shows the timer countdown. On completion of cooking, the buzzer is sounded for five seconds. Oven light is on. Display shows 'Cooking complete' while buzzer is sounding.

STATES AND STIMULI FOR THE MICROWAVE OVEN (B)

Stimulus	Description
Half power	The user has pressed the half-power button.
Full power	The user has pressed the full-power button.
Timer	The user has pressed one of the timer buttons.
Number	The user has pressed a numeric key.
Door open	The oven door switch is not closed.
Door closed	The oven door switch is closed.
Start	The user has pressed the Start button.
Cancel	The user has pressed the Cancel button.

MICROWAVE OVEN OPERATION





SUMMARY

- understand how graphical models can be used to represent software systems;
- understand why different types of models are required and the fundamental system modeling perspectives of context, interaction, structure, and behavior;
- have been introduced to some of the diagram types in the Unified Modeling Language (UML) and how these diagrams may be used in system modeling;



Thank You

