

# Engenharia de Software (14341, 16230, 15386)

## System Modeling #2

(adapted from *Software Engineering: International Version (10th Edition)*, Ian Sommerville, Pearson, 2015)

# Topics covered

---

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

---

# Structural models

# Structural models

---

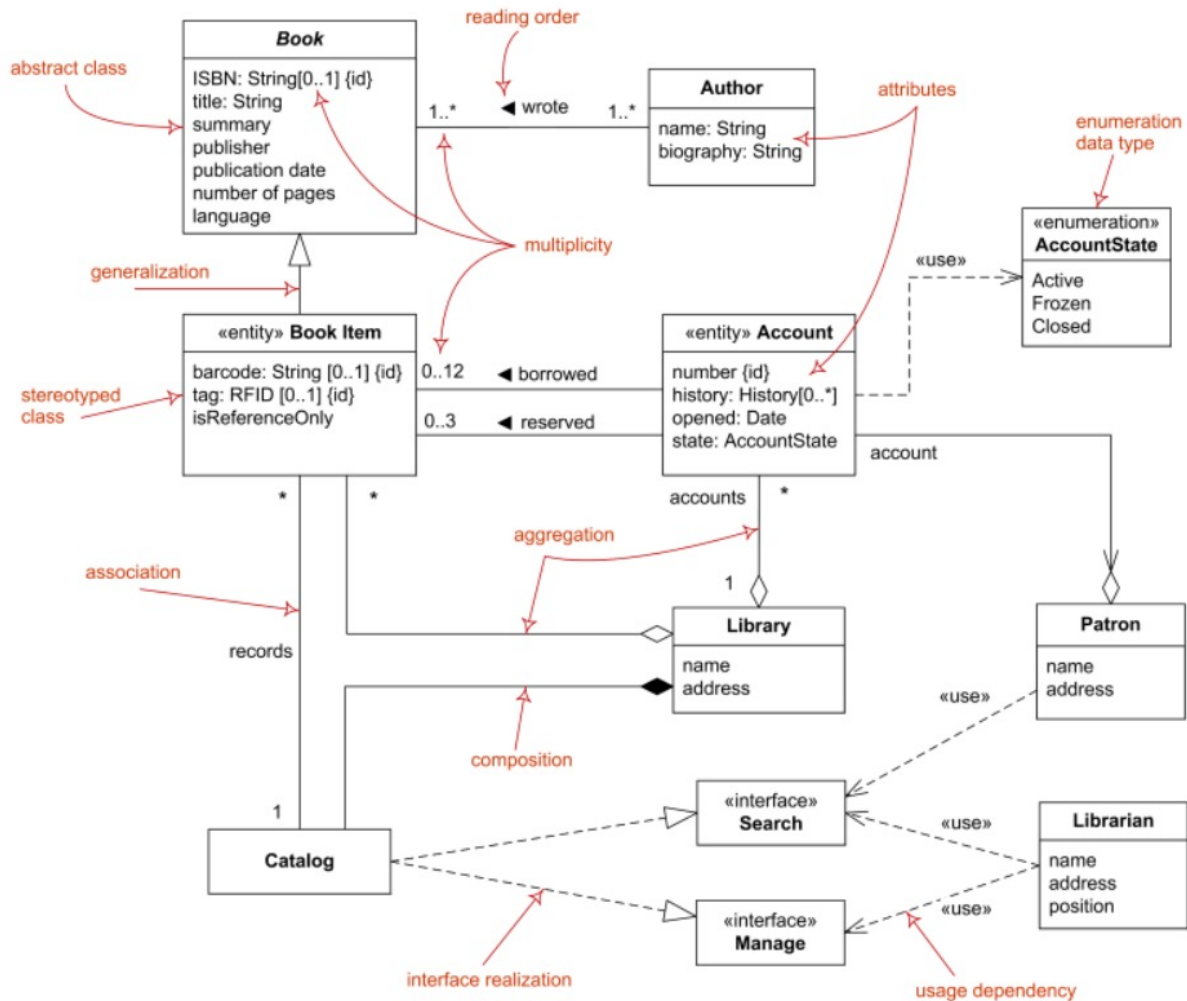
- ✧ Structural models of software display the organization of a system in terms of the components that make up that system and their relationships.
- ✧ Structural models may be static models, which show the structure of the system design, or dynamic models, which show the organization of the system when 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.

# Class diagrams



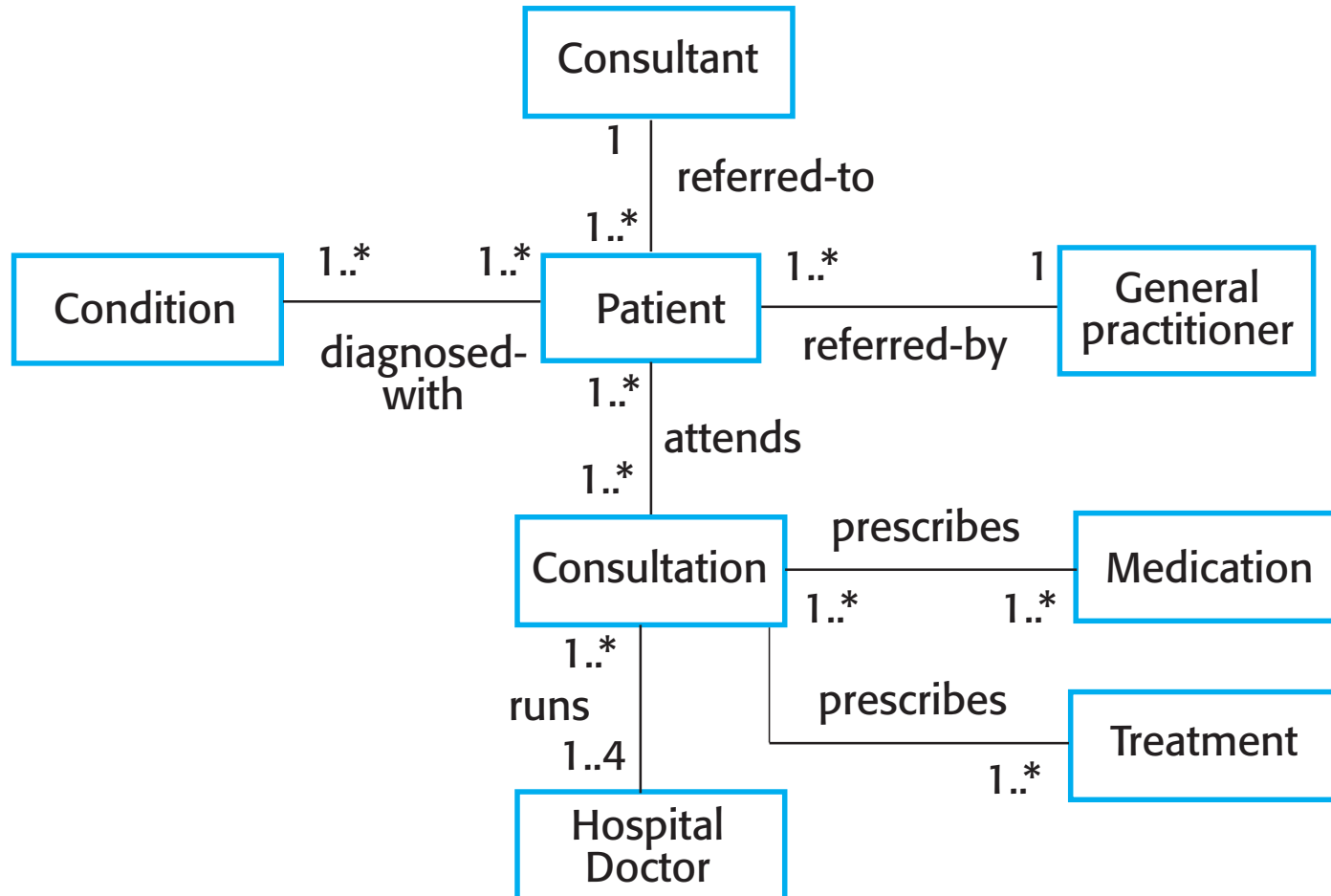
# UML classes and association

---



# Classes and associations in the MHC-PMS

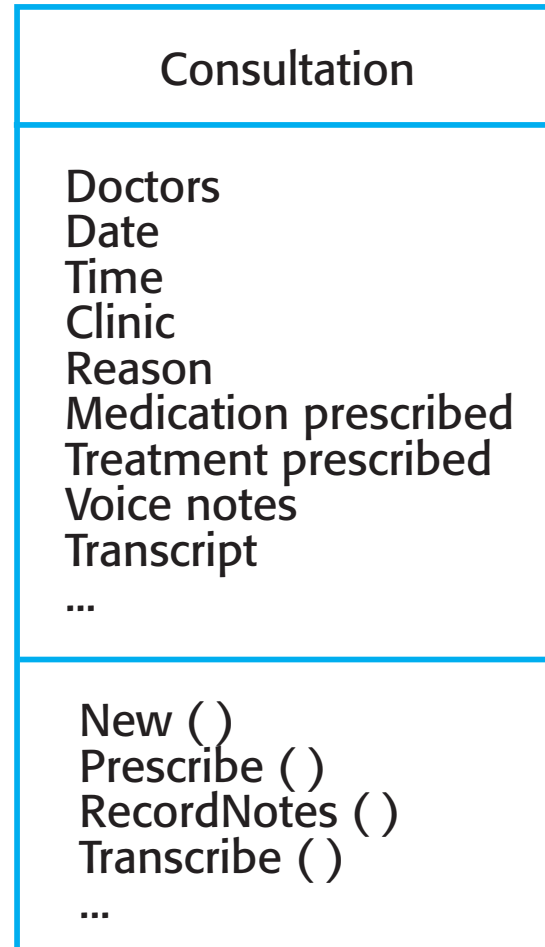
---





# The Consultation class

---



# 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.

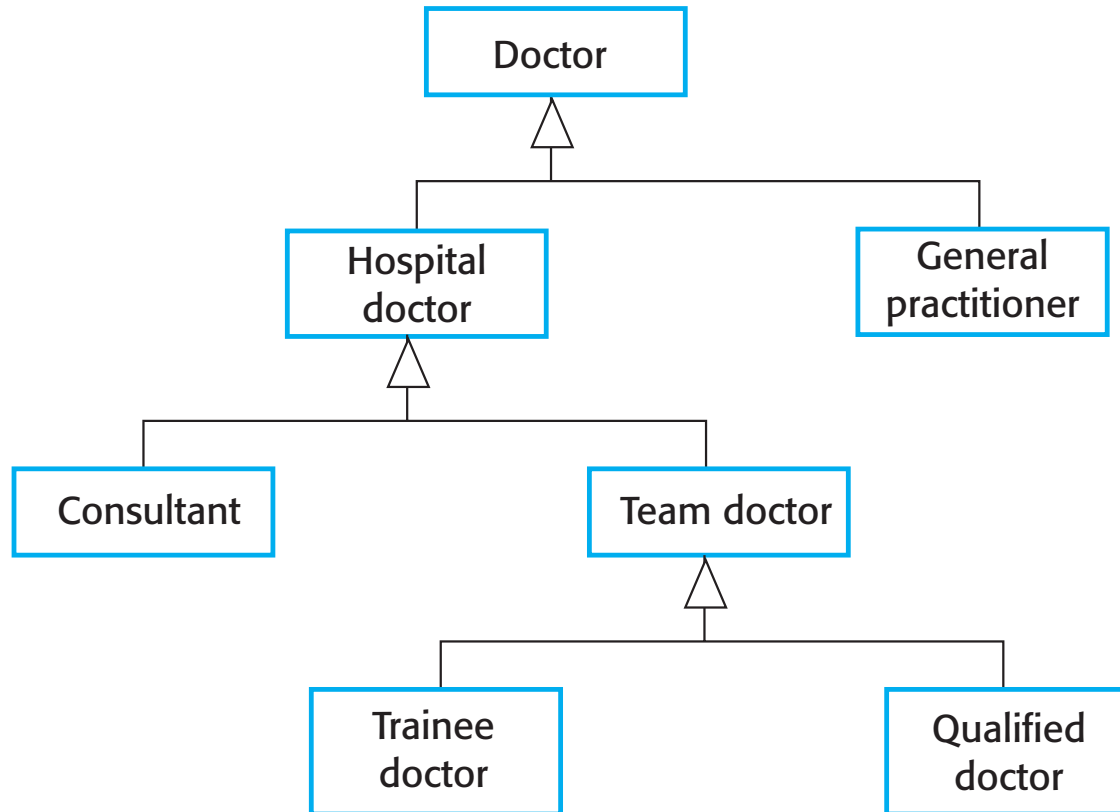
# Generalization

---

- ✧ 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.

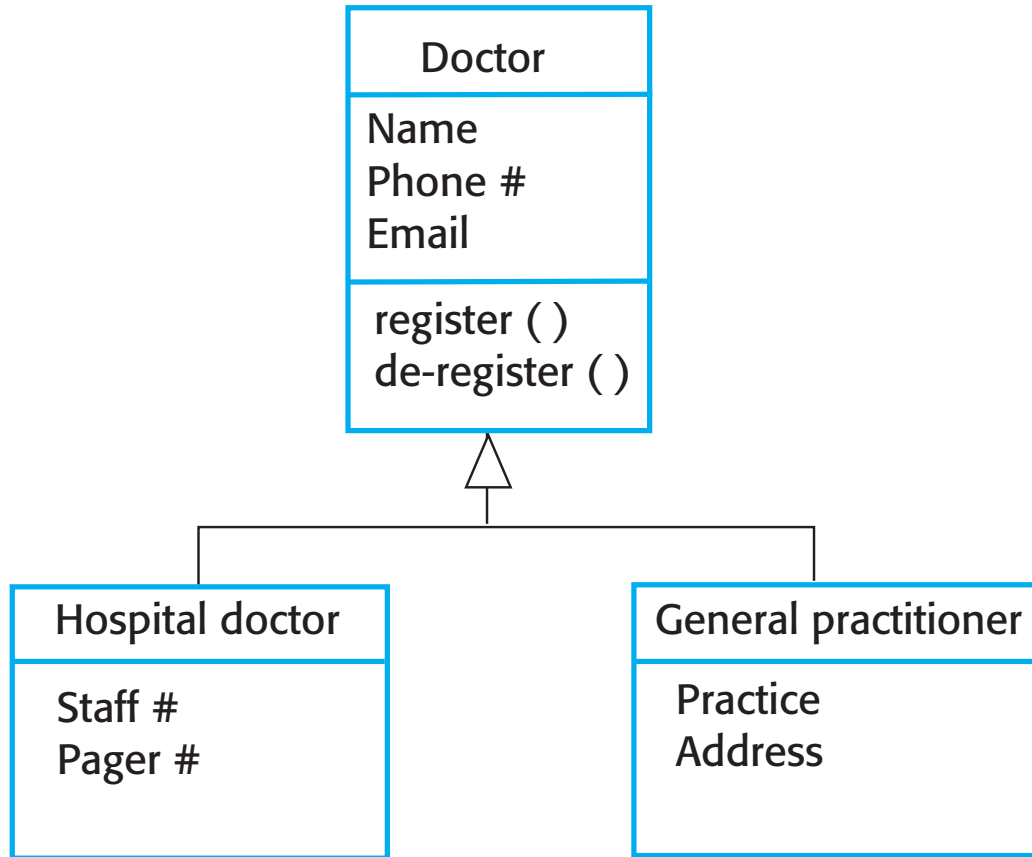
# A generalization hierarchy

---



# A generalization hierarchy with added detail

---



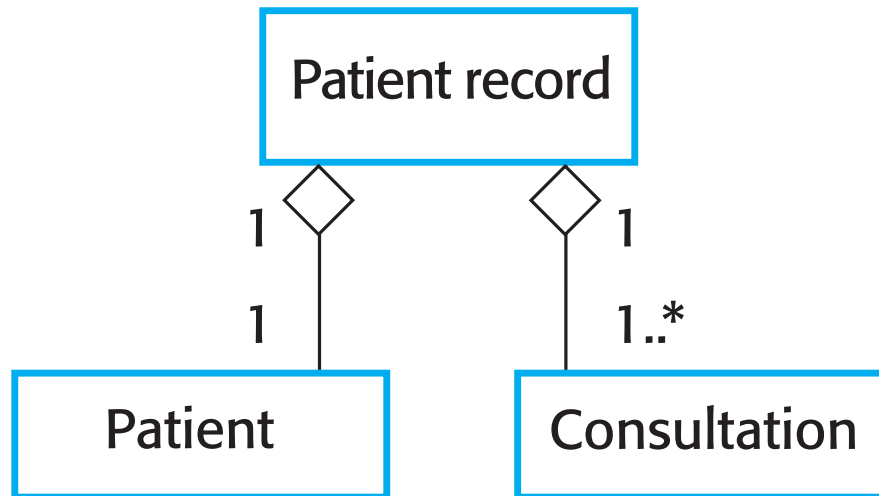
## Object class aggregation models

---

- ✧ 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

---

- ✧ 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.

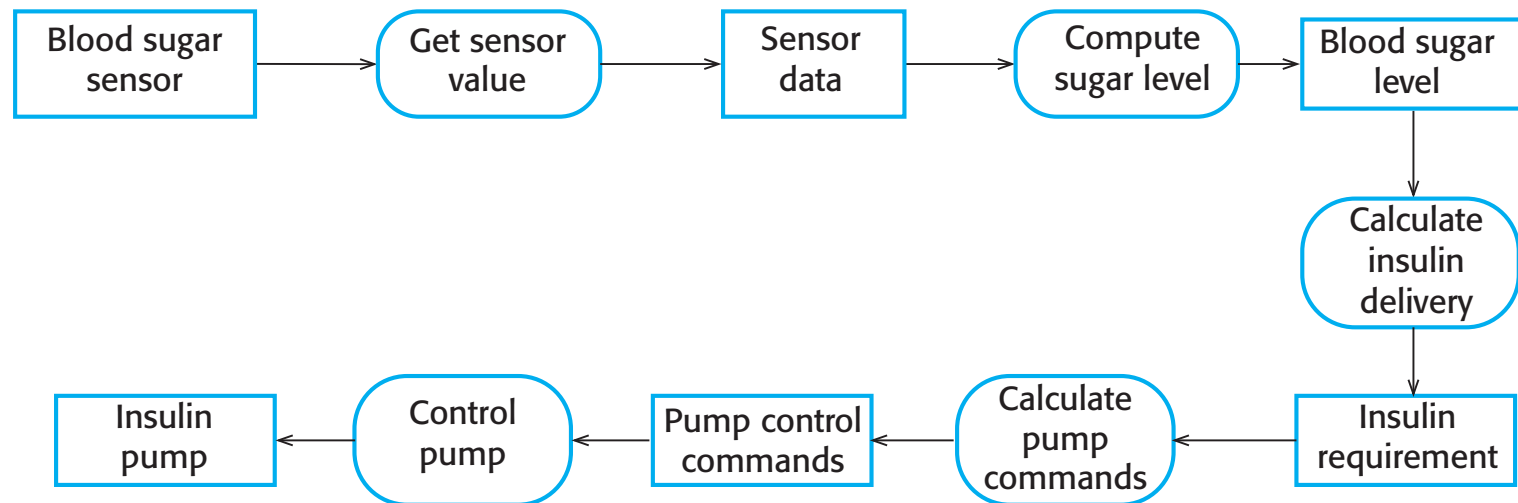
# Data-driven modeling

---

- ✧ 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.

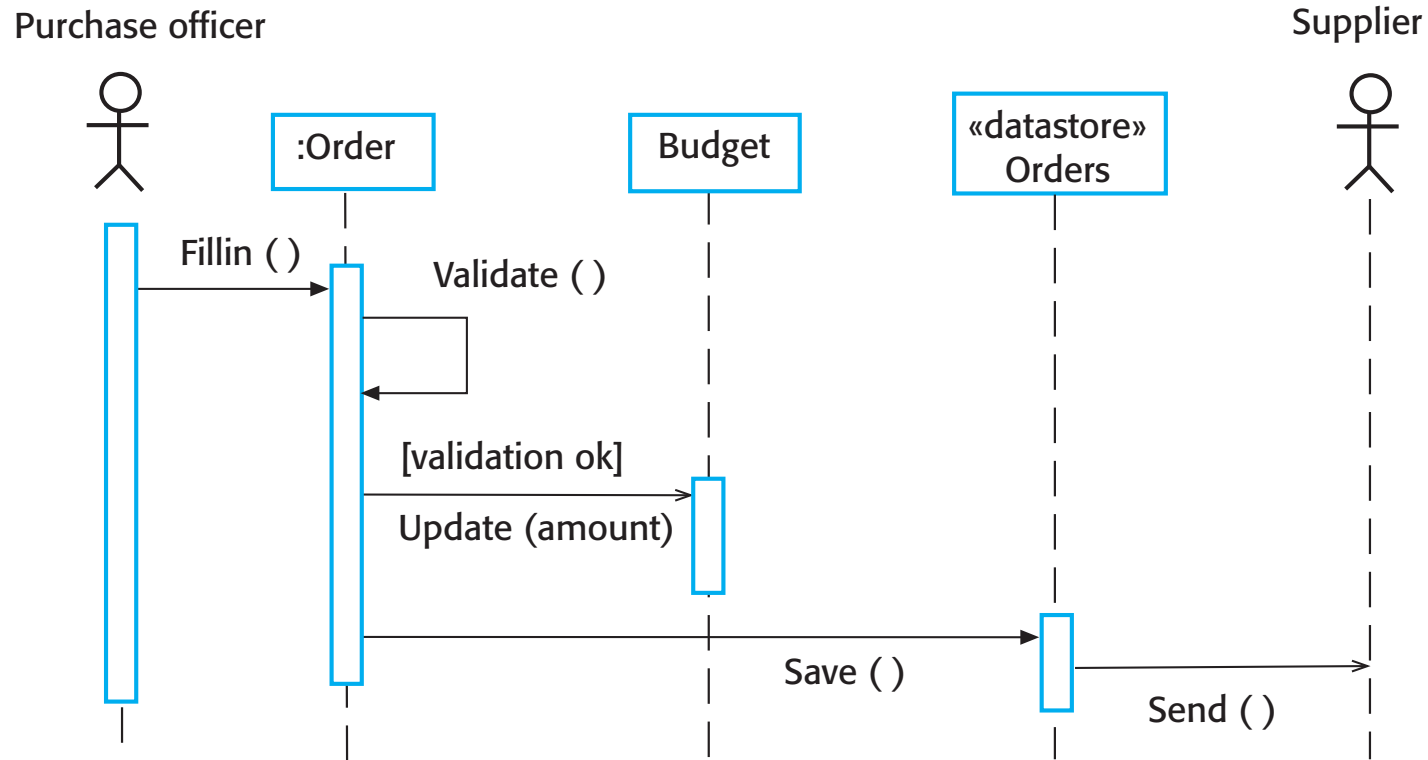
# An activity model of the insulin pump's operation

---



- ✧ Processing steps (rounded rectangles) representing activities
- ✧ Data flowing between these steps (rectangles), represented as objects

# Order processing



- ✧ A sequence diagram can be used to model interaction, in which messages are only sent from left to right

# Event-driven modeling

---

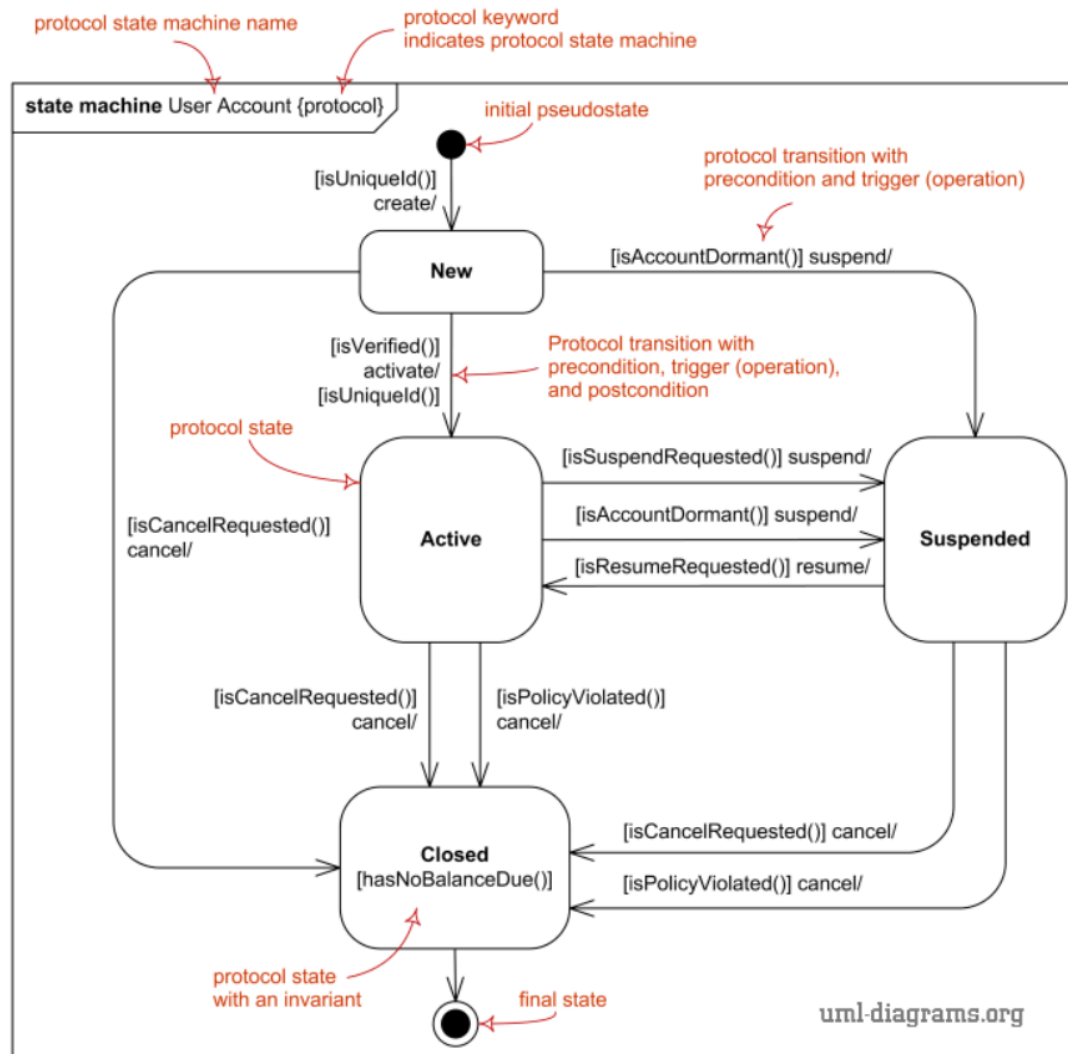
- ✧ Real-time systems are often event-driven, with minimal data processing.
- ✧ Event-driven modeling shows how a system responds to external and internal events.
- ✧ It is based on the assumption that a system has a finite number of states and that events (stimuli) may cause a transition from one state 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.
- ✧ State charts are an integral part of the UML and are used to represent state machine models.

# State diagrams



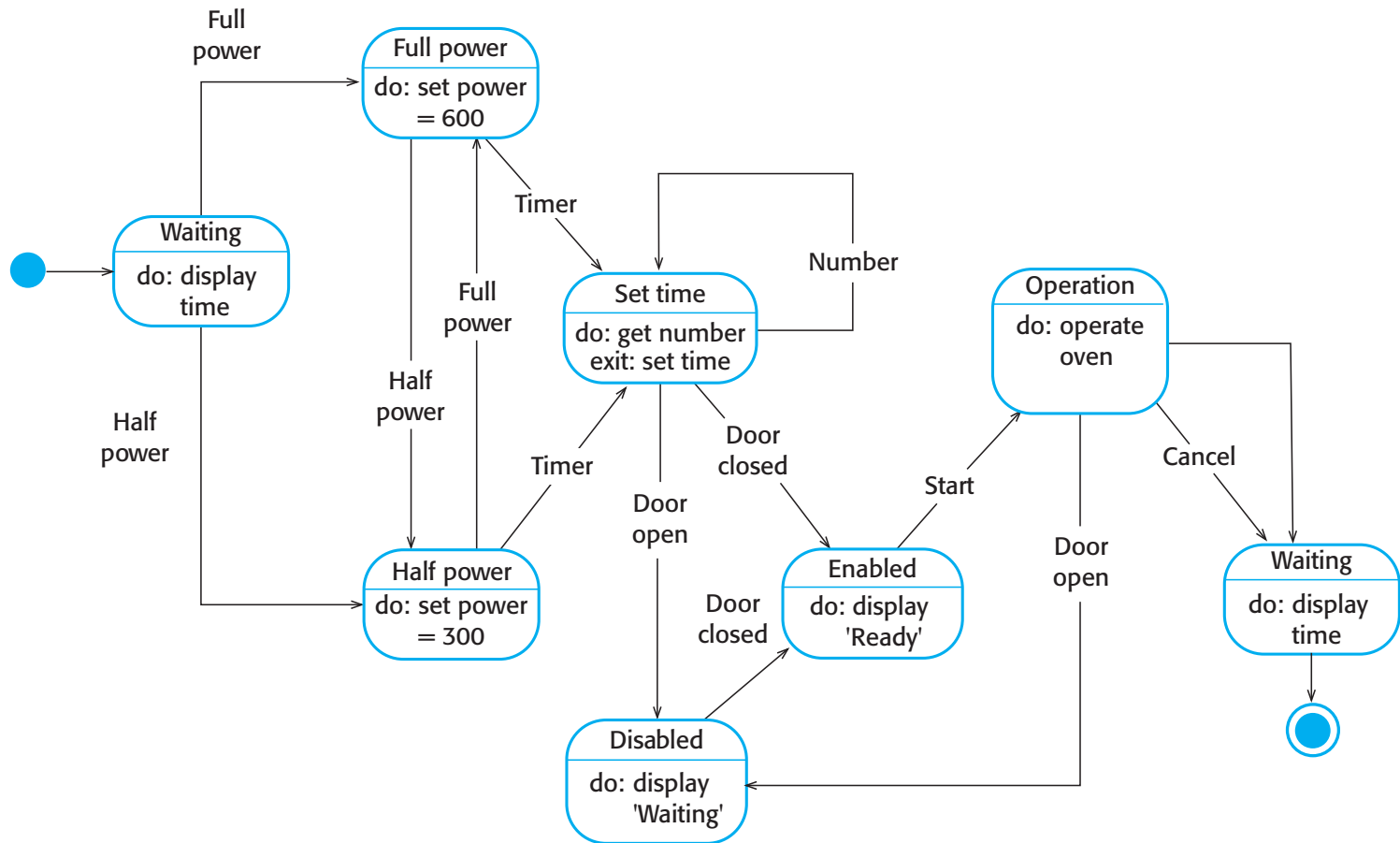
# State diagrams

---

- ✧ State: models a situation during which some (usually implicit) invariant condition holds. The invariant may represent a static situation such as an object waiting for some external event to occur. However, it can also model dynamic conditions such as the process of performing some behavior (i.e., the model element under consideration enters the state when the behavior commences and leaves it as soon as the behavior is completed).
- ✧ Transition: takes the state machine from one state configuration to another, representing the complete response of the state machine to an occurrence of an event of a particular type.



# 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.

## Practice: state diagram

---

- ✧ Determine the state diagram of the login process in the iLearn system.



# Key points

---

- ✧ 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.
- ✧ Behavioral models are used to describe the dynamic behavior of an executing system. This behavior can be modeled from the perspective of the data processed by the system, or by the events that stimulate responses from a system.
- ✧ Activity diagrams may be used to model the processing of data, where each activity represents one process step.
- ✧ State diagrams are used to model a system's behavior in response to internal or external events.

