SOFTWARE ENGINEERING

CO3001

CHAPTER 5 – SYSTEM MODELING

Adapted from https://iansommerville.com/software-engineering-book/slides/

WEEK 5,6

TOPICS COVERED

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

SYSTEM MODELING

- the process of developing abstract models of a system
  - each model presenting a different view or perspective
- means representing a system using some kind of graphical notation
  - almost always based on notations in the Unified Modeling Language (UML).
- helps the analyst to
  - understand the functionality of the system
  - use models to communicate with customers.

EXISTING AND PLANNED SYSTEM MODELS

- ✓ Models of the existing system
  - 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
  - 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
  - models the context or environment of the system.
- ✓ An interaction perspective
  - models the interactions between a system and its environment, or between the components of a system.
- ✓ A structural perspective
  - models the organization of a system or the structure of the data that is processed by the system.
- ✓ A behavioral perspective
  - models the dynamic behavior of the system and how it responds to events.

SYSTEM PERSPECTIVES - EXAMPLE

http://www.youtube.com/watch?v=24Zxr2KHW6s

Biogas system

http://www.youtube.com/watch?v=FxIs2GVsqgY

Coffee maker

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
  - may be incomplete
- ✓ As a way of documenting an existing system
  - should be an accurate representation of the system
- ✓ As a detailed system description that can be used to generate a system implementation
  - Models have to be both correct and complete.

EXTERNAL PERSPECTIVES

CONTEXT MODELS

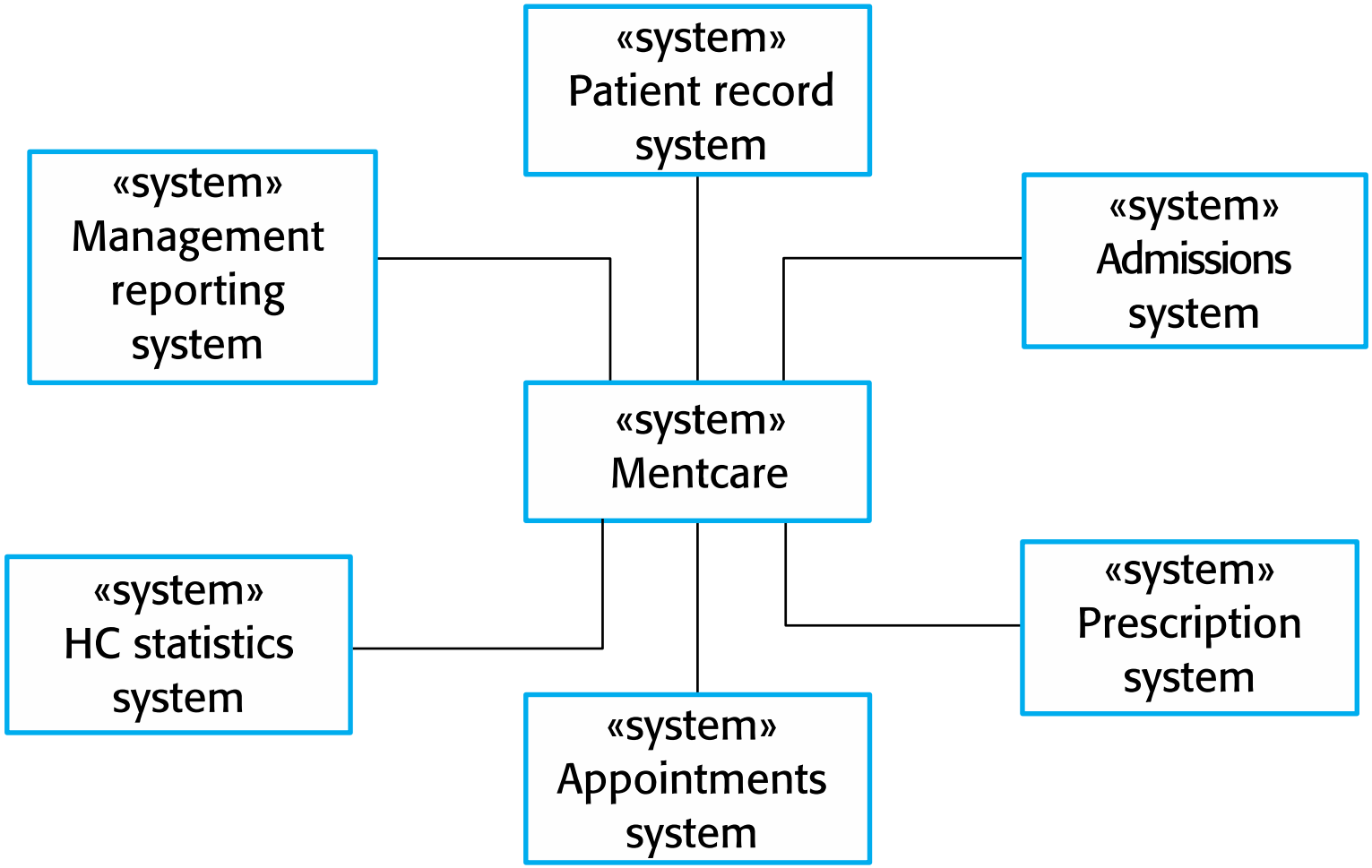
- ✓ To illustrate the operational context of a system – the boundaries
  - they show what lies outside the system boundaries.

- ✓ 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 CONTEXT OF THE MENTCARE SYSTEM



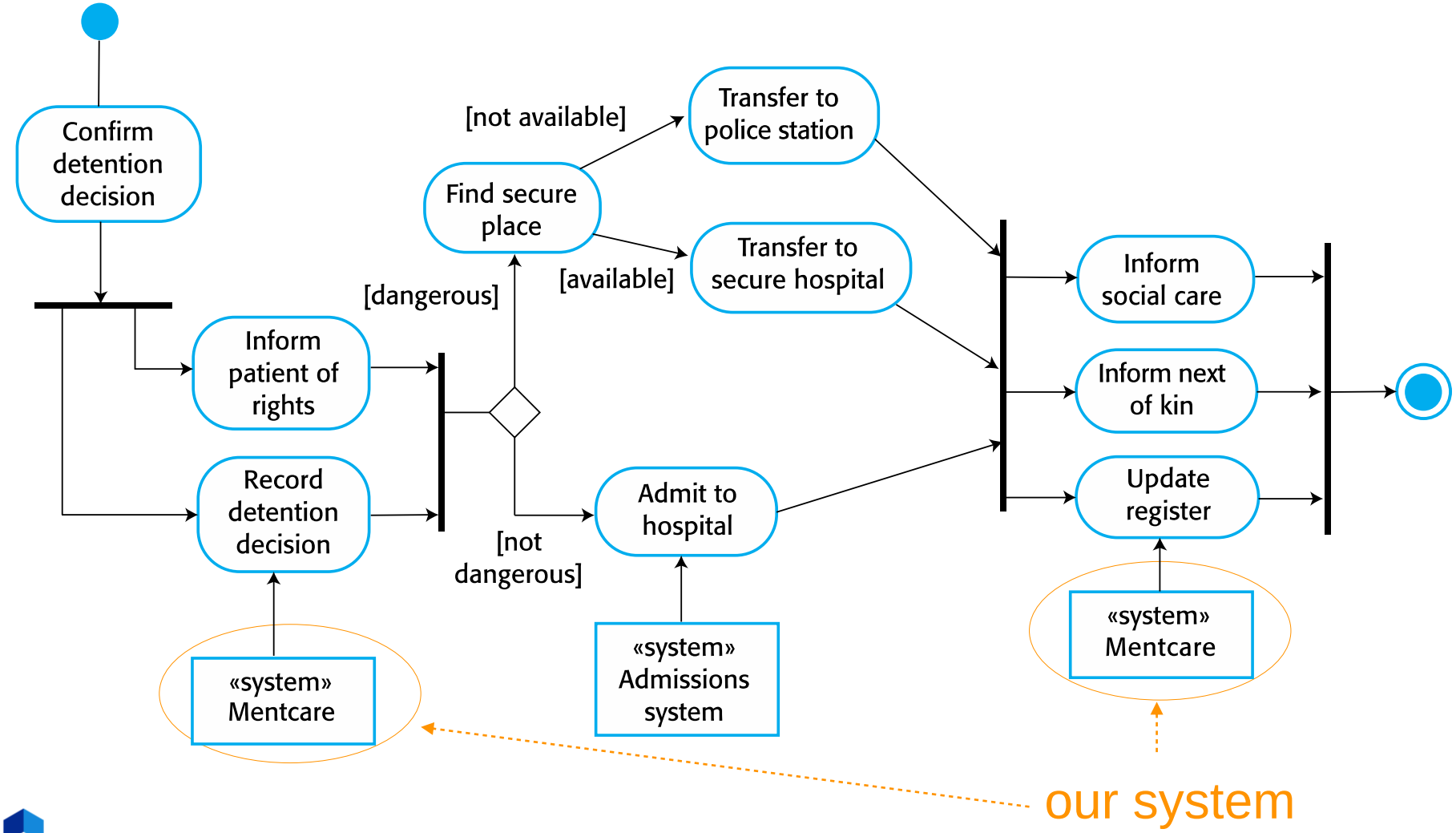
PROCESS PERSPECTIVE

- 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 PERSPECTIVES

INTERACTION MODELS

- ✓ Modeling user interaction
  - helps to identify user requirements.
- ✓ Modeling system-to-system interaction
  - highlights the communication problems that may arise.
- ✓ Modeling component interaction
  - to 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.

http://tynerblain.com/blog/2007/01/22/how-to-write-good-use-case-names/

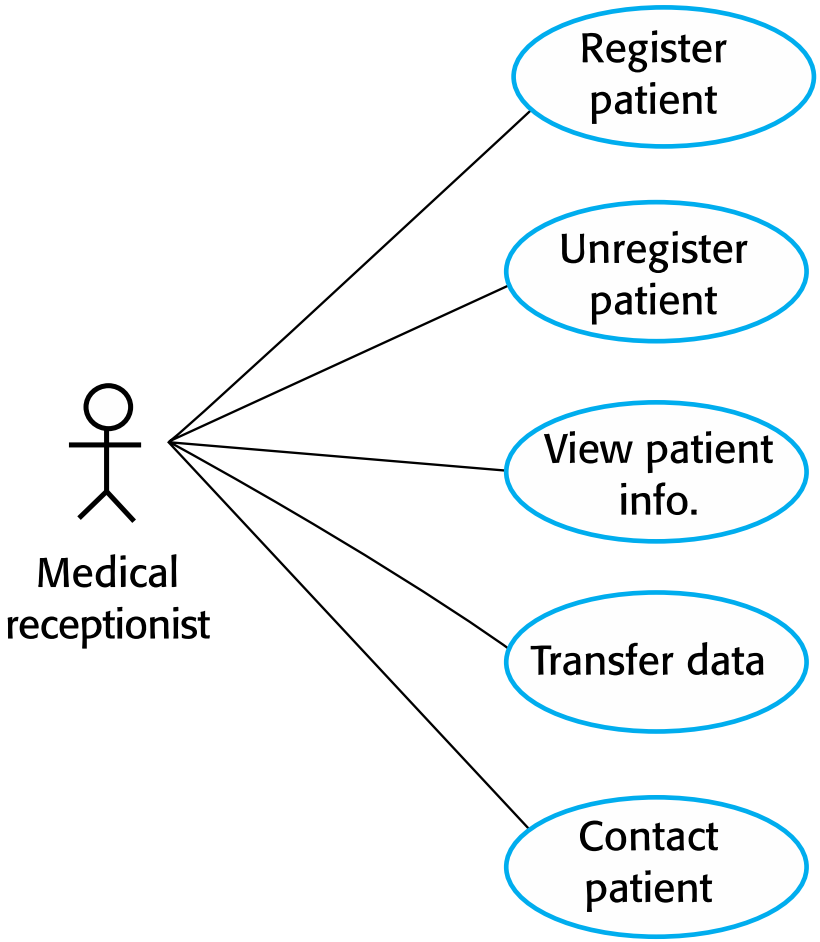
TRANSFER-DATA USE CASE

- ✓ A use case in the Mentcare system

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 Mentcase system 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 MENTCARE SYSTEM INVOLVING THE ROLE 'MEDICAL RECEPTIONIST'



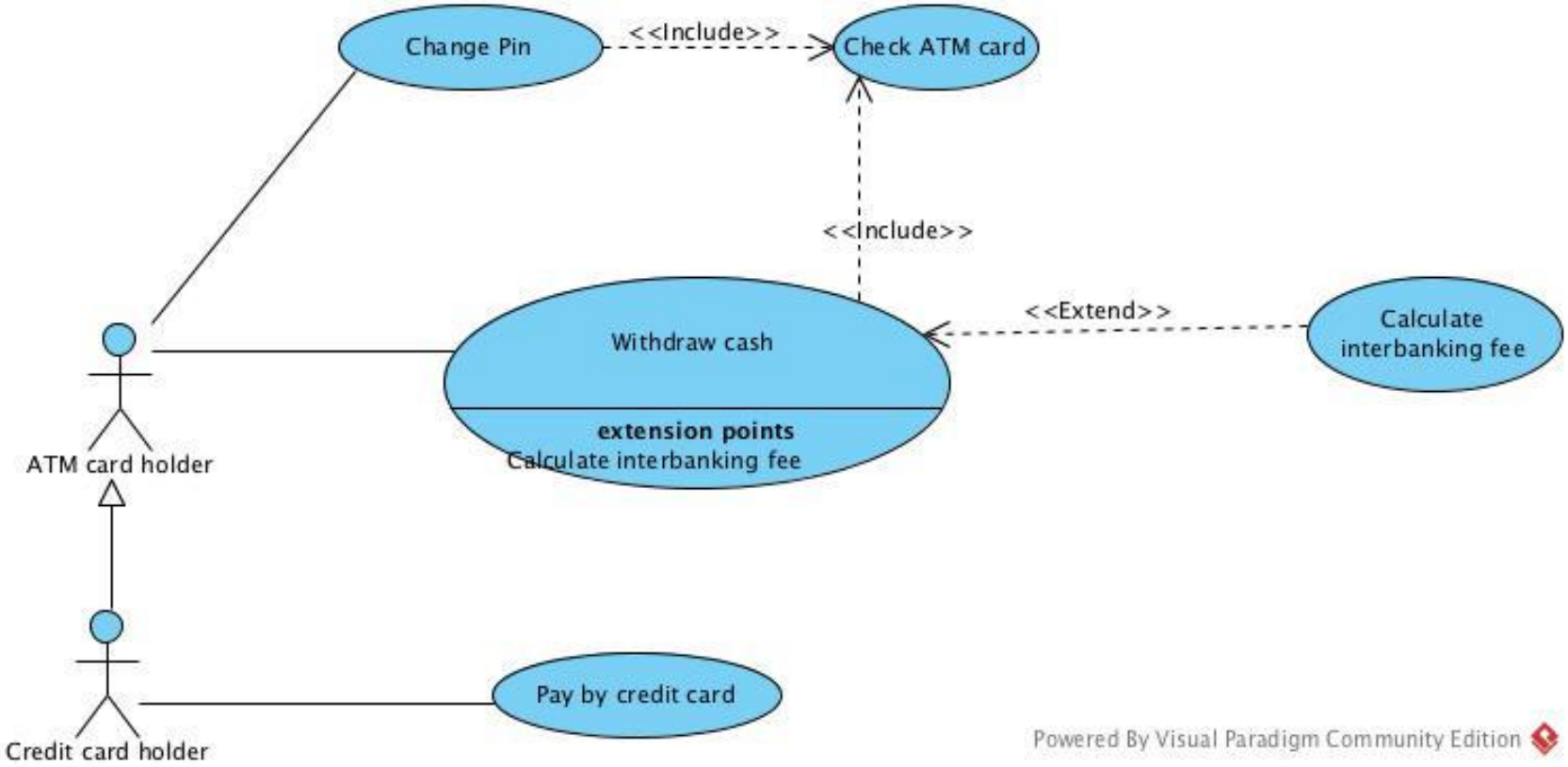
USE-CASE SCENARIO TABULAR TEMPLATE

Use Case ID:			
Use Case Name:			
Created By:		Last Updated By:	
Date Created:		Date Last Updated:	
Actors:			
Description:			
Trigger:			
Preconditions:			
Postconditions:			
Normal Flow:	1. … 2. …		
Alternative Flows:	Alternative 1: … Alternative x: …		
Exceptions:	Exception 1: … Exception x: …		
Notes and Issues:			

A USE-CASE EXAMPLE

Use-case name	**Search club events**
Actor	A member of the public (MP)
Description	The MP is searching for club events on a particular date.
Preconditions	The MP is at the university home page.
Normal Flow	1. MP selects "Search Events" on MP home page 2. System presents a page with choice of dates for the current month 3. MP selects a date from among the choices 4. System presents a page with events for that date, giving time and club name 5. MP selects an event 6. System presents a page with details of that event, including location, description and cost
Exceptions	*Exception 1: at step 4* 4a. If there are no events for the selected date, System presents a page saying that there are no events for the selected date
Alternative Flows	*Alternative 1: at step 3* 3a. MP selects a different month 3b. System presents a page with choice of dates for the month *Continue step 3 in the normal flow*

MORE USE-CASE ANNOTATION



use-case extended point: when/where to extend
actor generalization: similar to class generalization

A - - «includes» - -> B: start at A, may do B, end at A
A <- - «extend» - - B: start at A, may do B (at an extended point) and (may) end at B

SEQUENCE DIAGRAMS

- Sequence diagrams are part of the UML

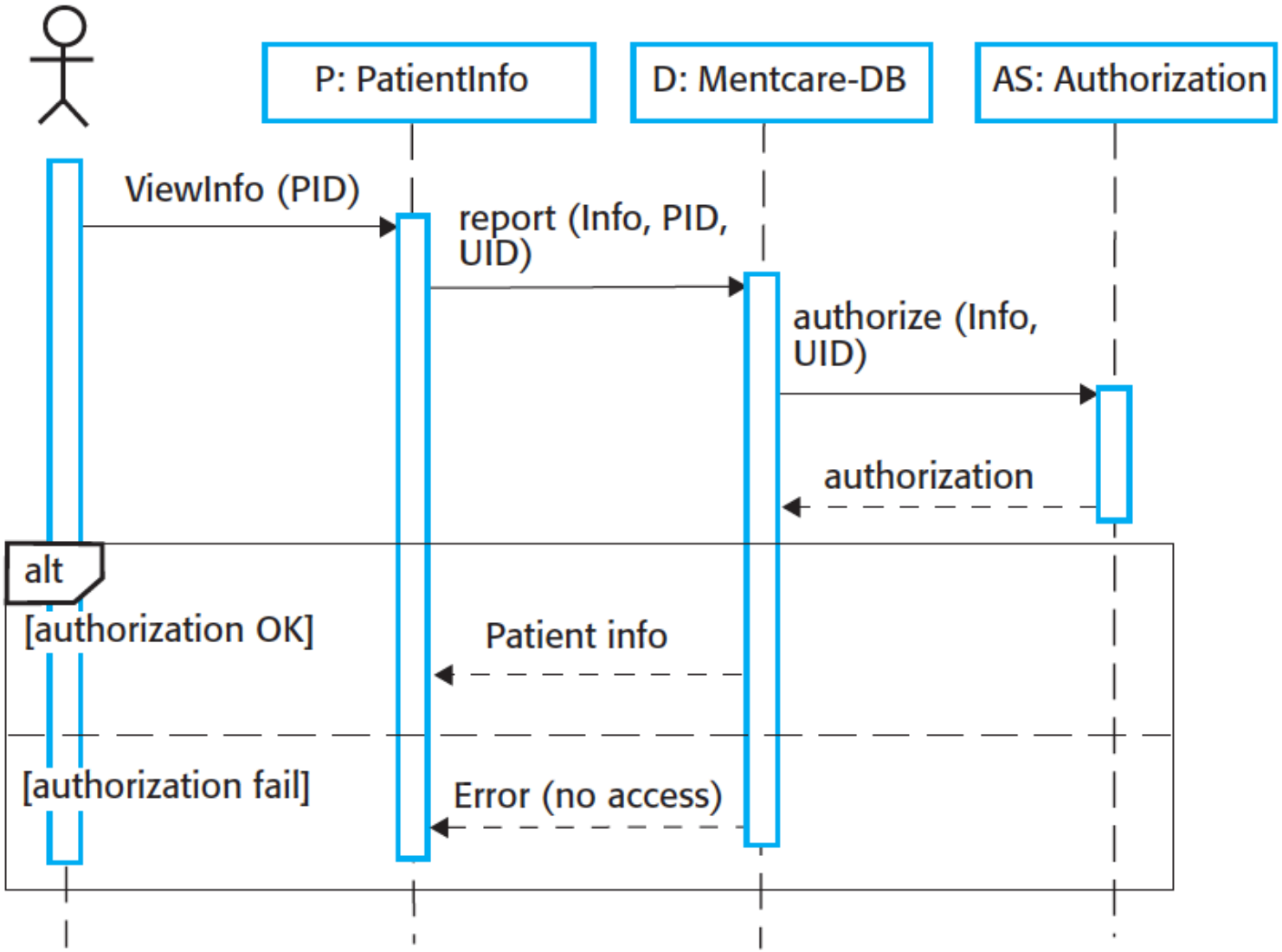
✓ used to model the interactions between the actors and the objects within a system.

✓ 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.

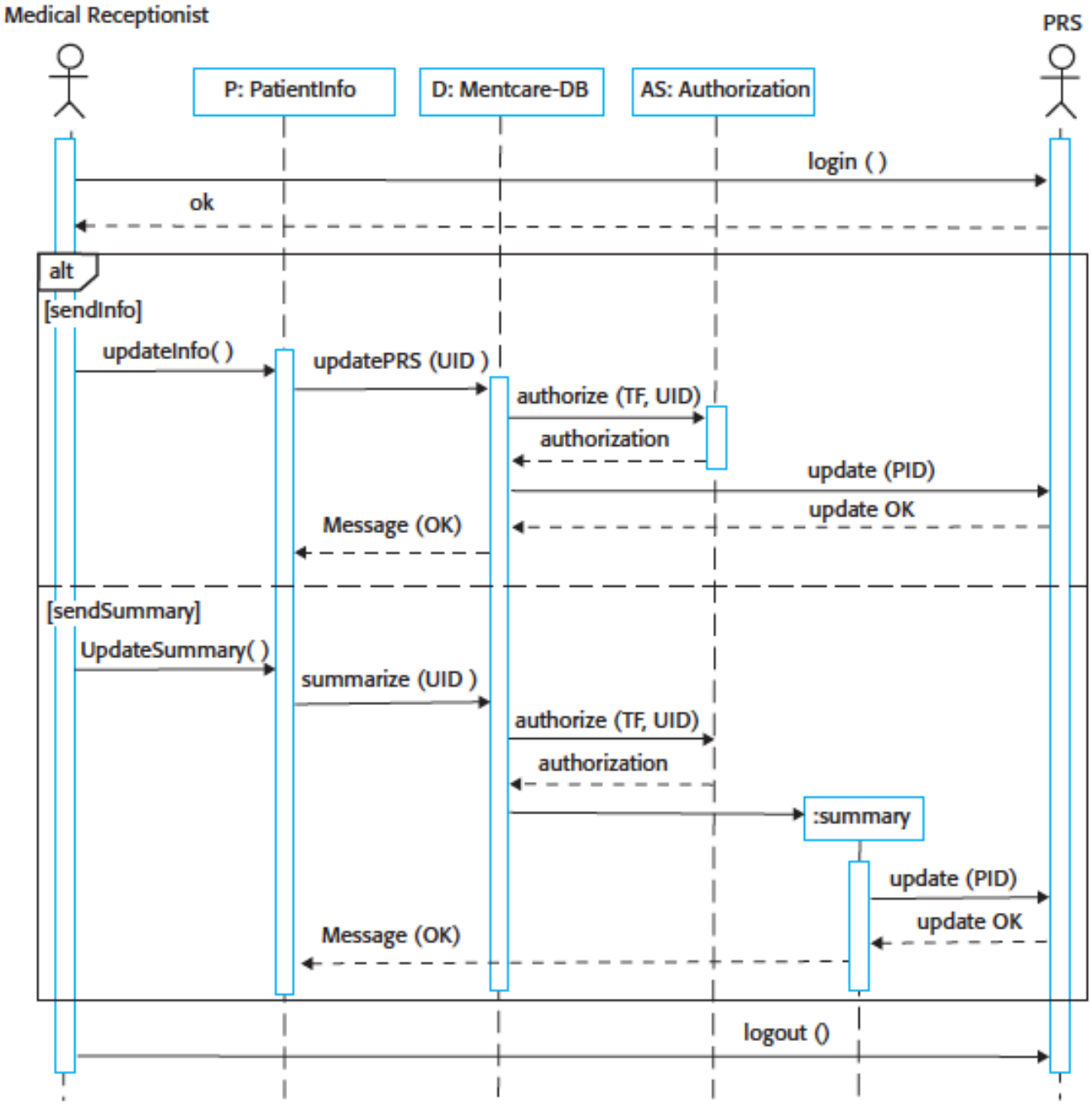
http://creately.com/blog/diagrams/sequence-diagram-tutorial/

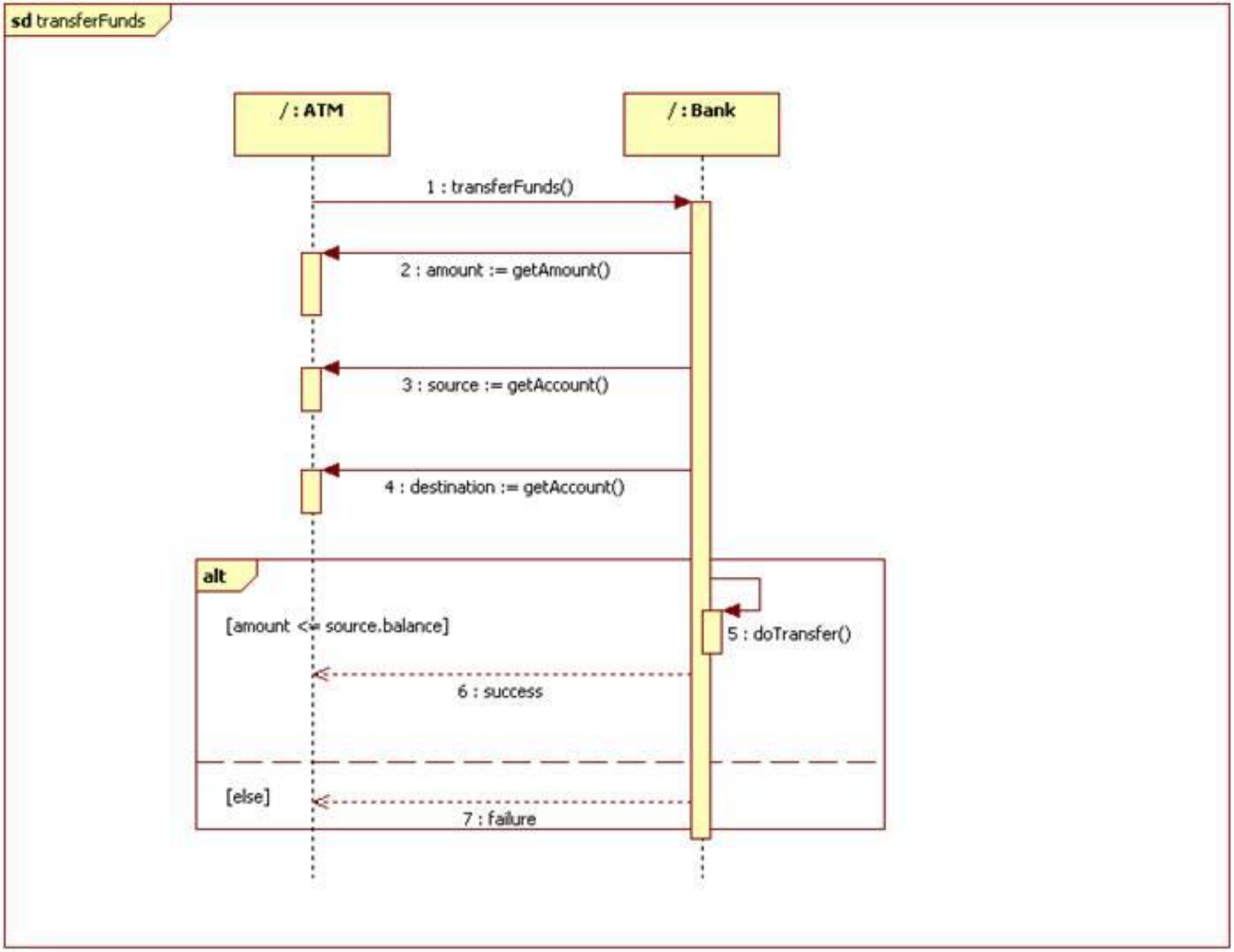
SEQUENCE DIAGRAM FOR VIEW PATIENT INFORMATION

Medical Receptionist



SEQUENCE DIAGRAM FOR TRANSFER DATA




sd transferFunds
/ : ATM
/ : Bank
1 : transferFunds()
2 : amount := getAmount()
3 : source := getAccount()
4 : destination := getAccount()
alt
[amount <= source.balance]
5 : doTransfer()
6 : success
[else]
7 : failure

IN-CLASS ACTIVITY

- ✓ Use-case and sequence diagrams for some examples

- ✓ Note:
  - 1. If in seq. diagram A ---- method1(xxx) ----> B, then “method1” is of A or B?

STRUCTURAL PERSPECTIVES

STRUCTURAL MODELS

- ✓ 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: show the structure of the system design,
  - or dynamic models: show the organization of the system when it is executing.

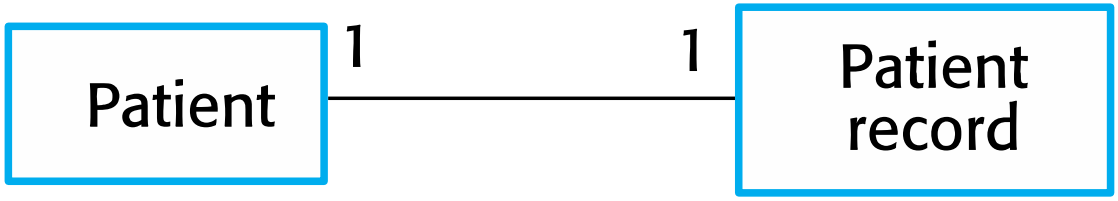
Create structural models of a system when discussing and designing the system architecture.

CLASS DIAGRAMS

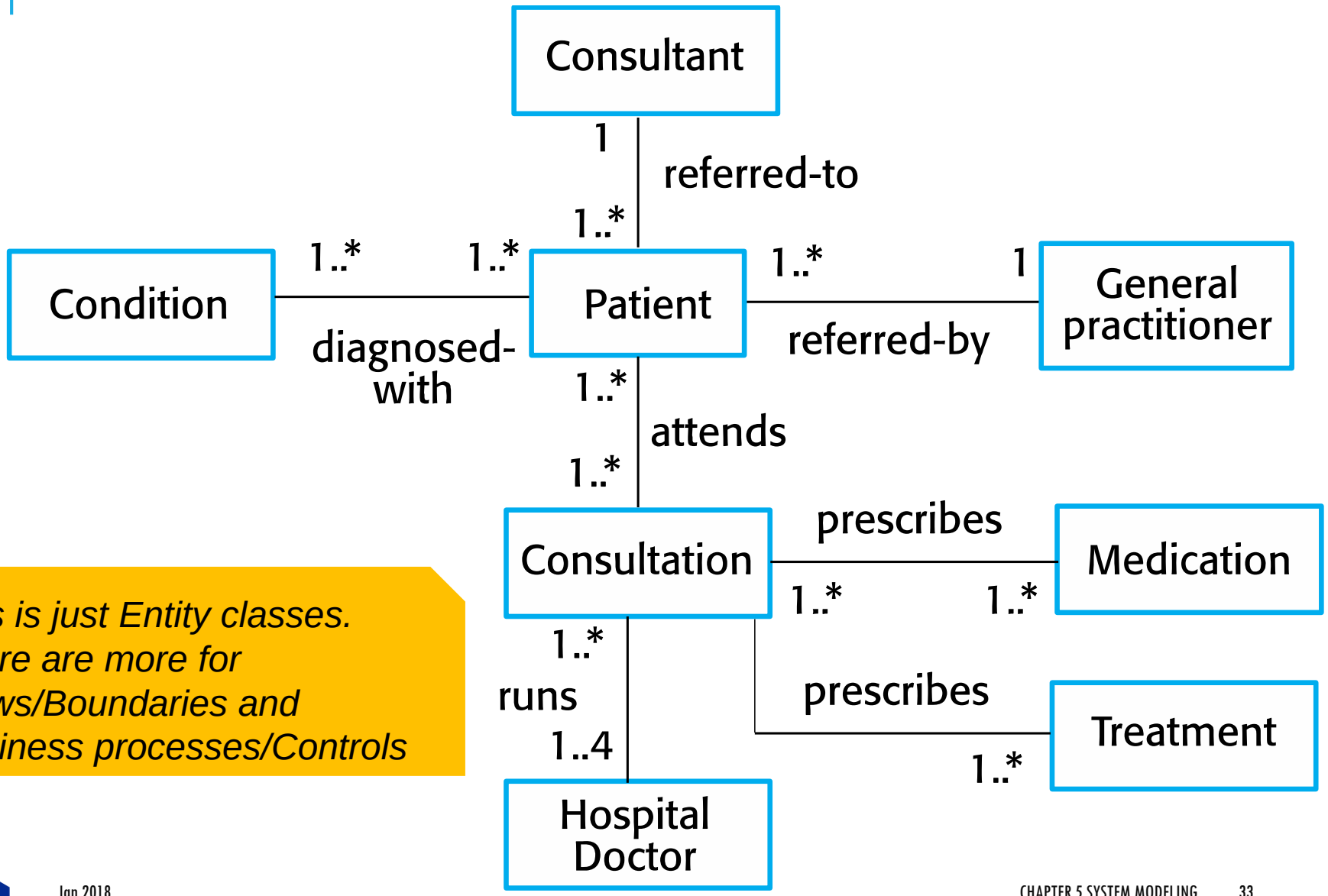
- ✓ Used when developing an object-oriented system model to show the classes in a system and the associations between these classes.

  - 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.

UML CLASSES AND ASSOCIATION

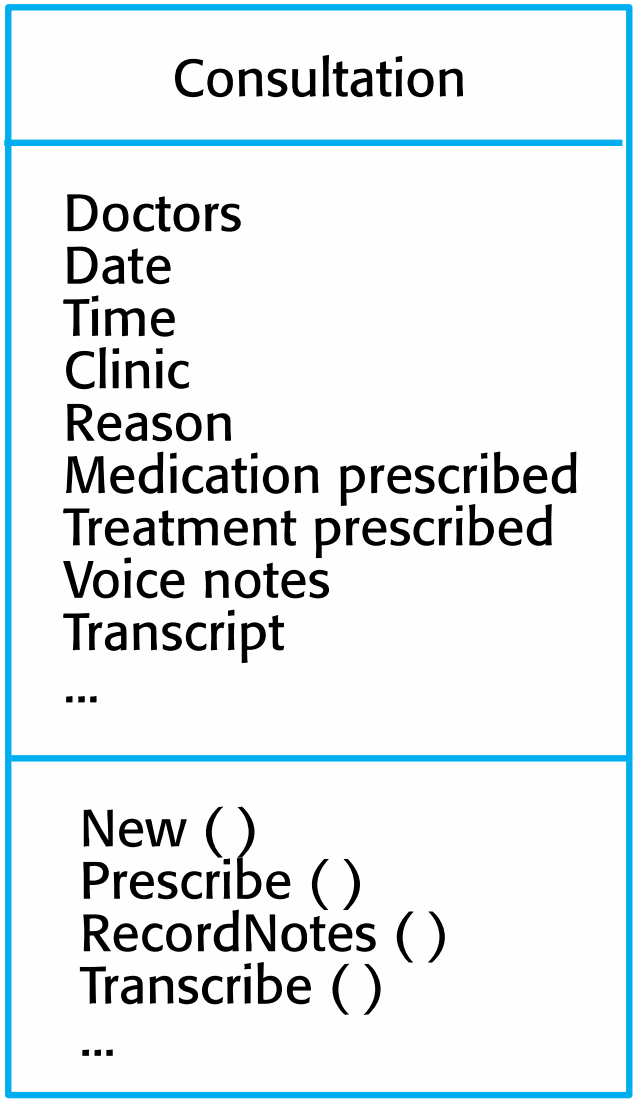


CLASSES AND ASSOCIATIONS IN THE MENTCARE



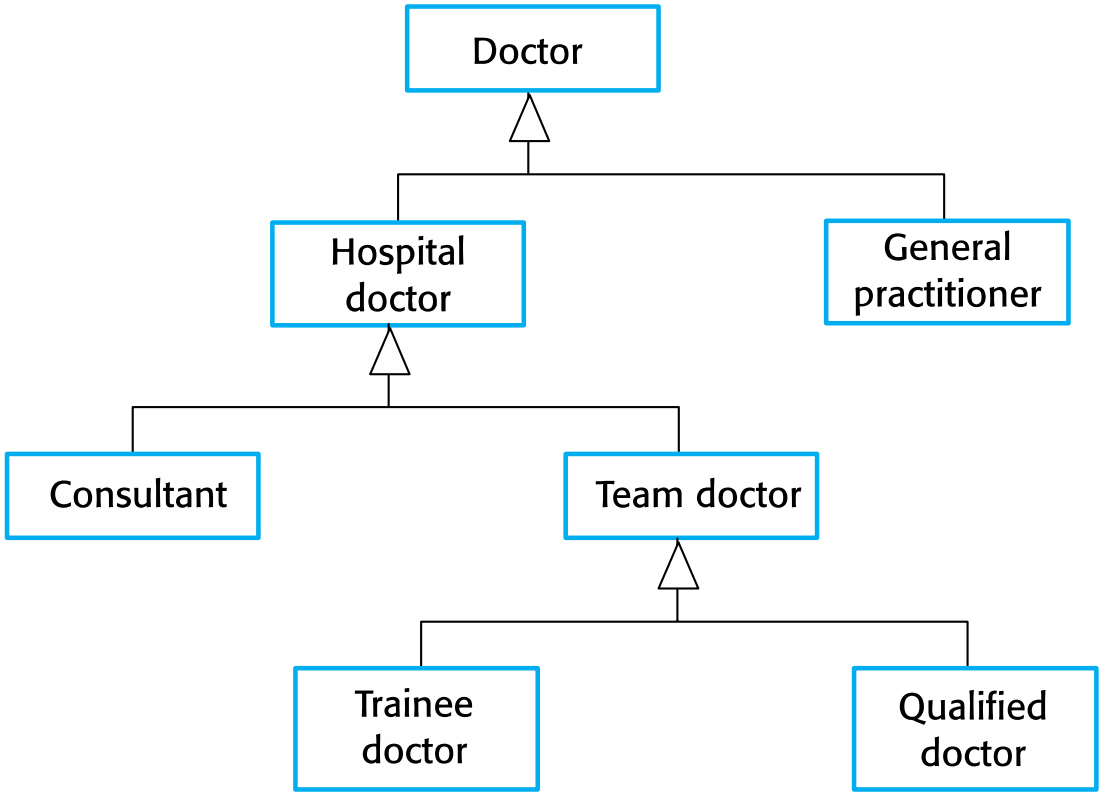
This is just Entity classes. There are more for Views/Boundaries and Business processes/Controls

THE CONSULTATION CLASS

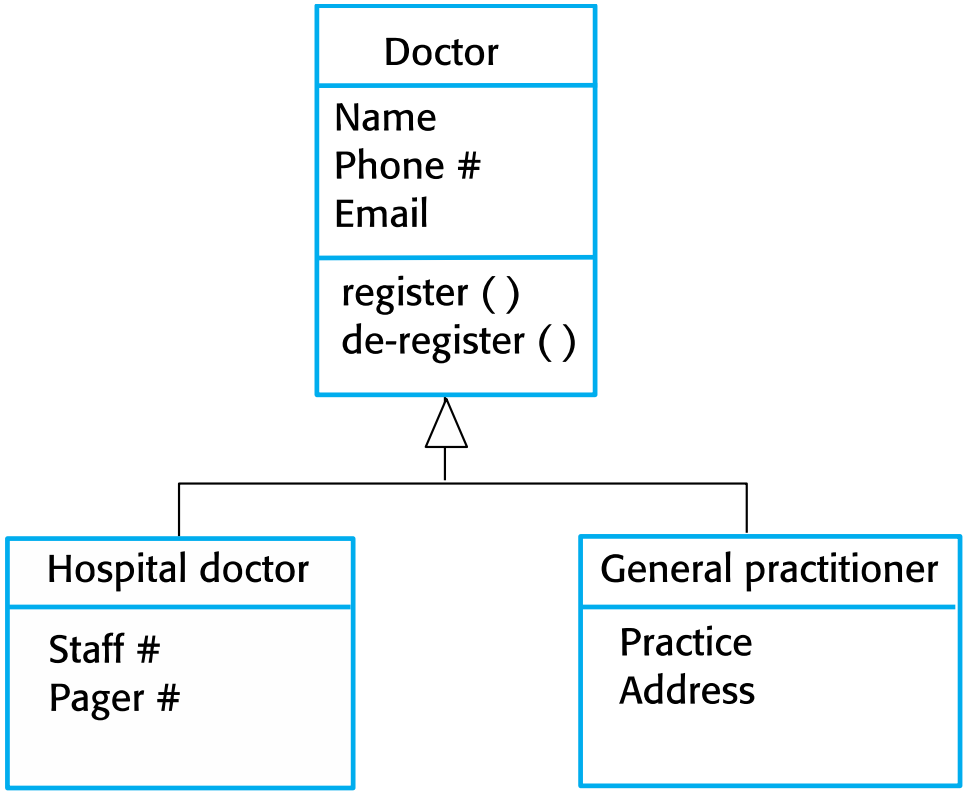


GENERALIZATION

- Rather than learn the detailed characteristics of every entity, place these entities in more general classes (animals, cars, houses, etc.) and learn the characteristics of these classes

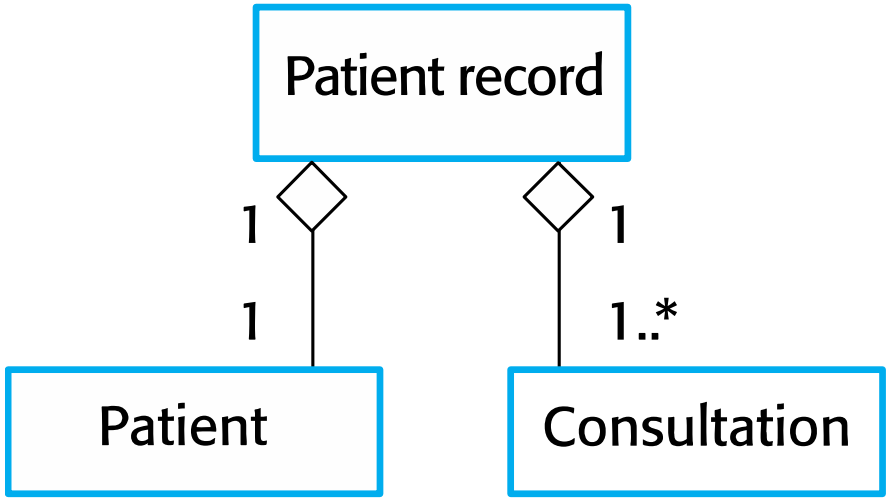


A GENERALIZATION HIERARCHY WITH ADDED DETAIL



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.



AGGREGATION VS COMPOSITION RELATIONSHIP

- ✓ Aggregation: specifies a whole/part relationship between the aggregate (whole) and component part (the component may survive the aggregate object)
- ✓ Composition: composite object takes ownership of the component(s)

DATABASE DIAGRAMS VS CLASS DIAGRAMS

- ✓ Entity/Relation/Table vs. class
- ✓ Entity/Relation/Table relationship vs class relationship

- ✓ When and why we need
  - Only database
  - Only classes
  - Both

BEHAVIORAL PERSPECTIVES

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.

- Stimuli:
  - **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.

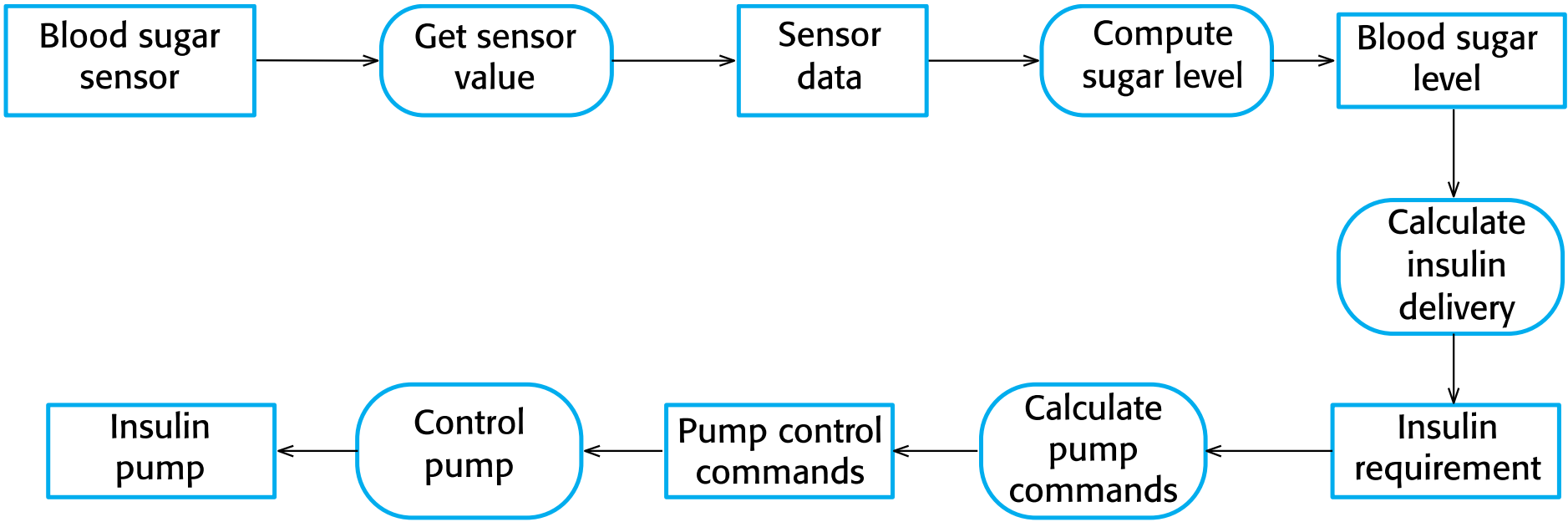
DATA-DRIVEN MODELING

- 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.

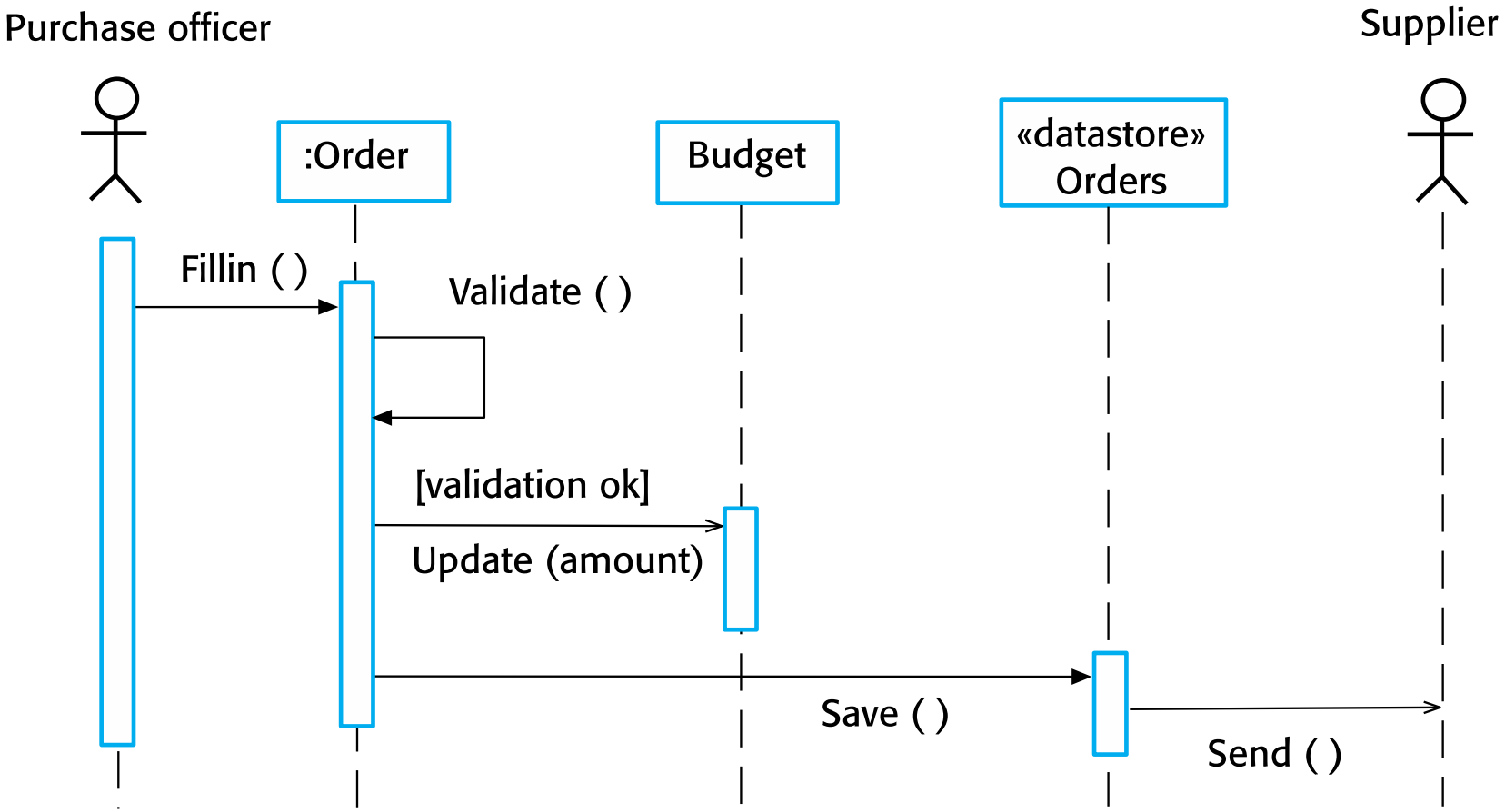
- ✓ Data-driven models show the sequence of actions involved in processing input data and generating an associated output.

- ✓ Data-Flow-Diagrams (DFD) ?
  - Not UML

AN ACTIVITY MODEL OF THE INSULIN PUMP'S OPERATION



ORDER PROCESSING – AN ALTERNATIVE TO REPRESENT BEHAVIORS



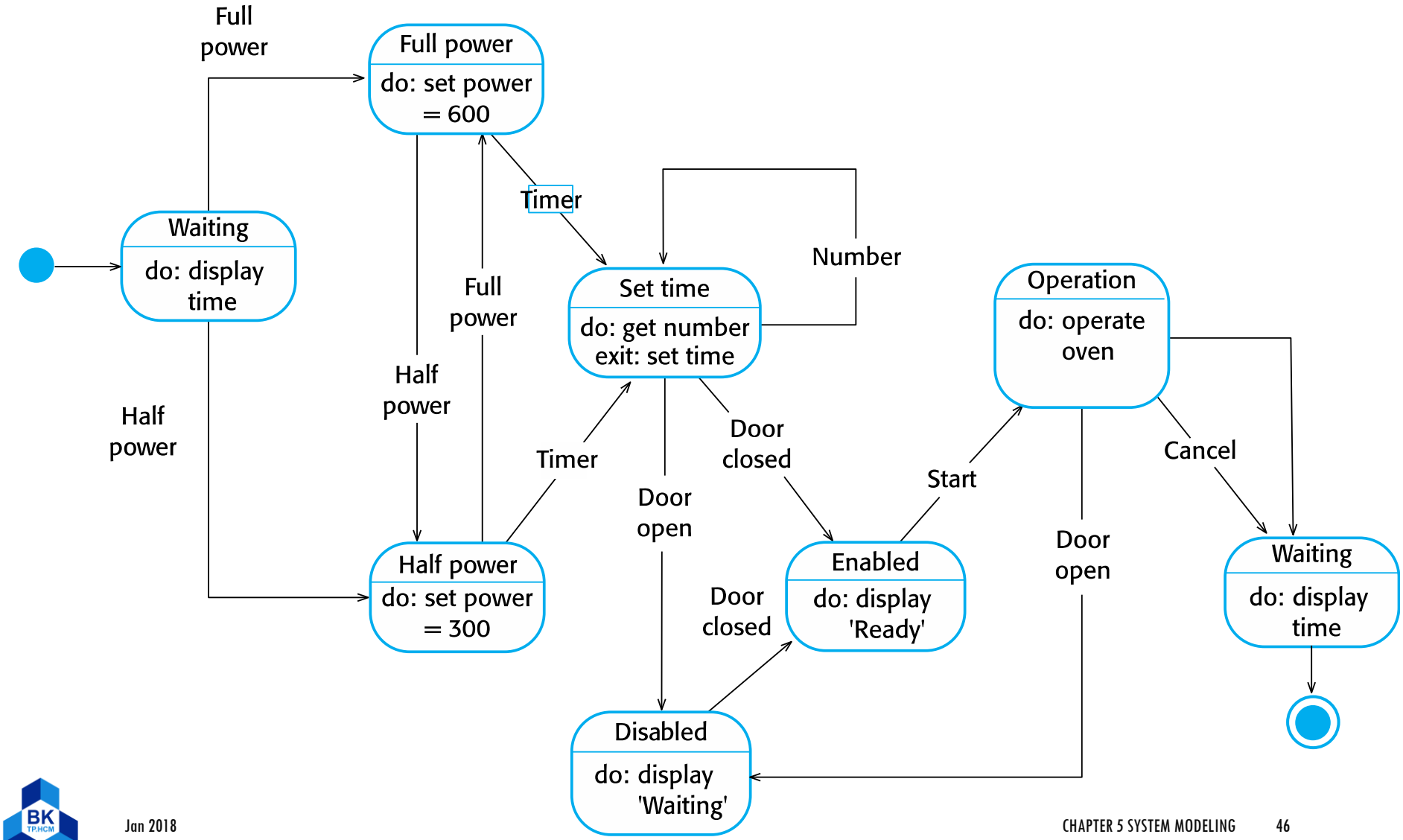
EVENT-DRIVEN MODELING

- 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.

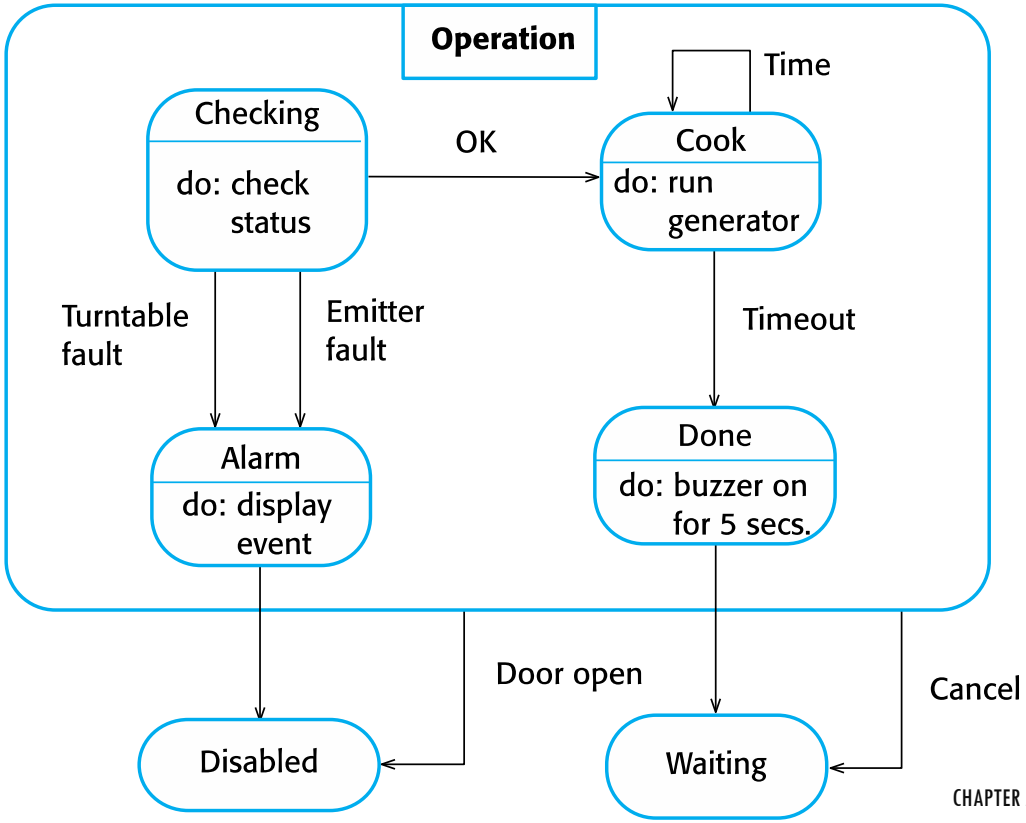
- It is based on the assumption that a system has a finite number of states and that events (stimuli) may cause a transition from one state to another.

STATE DIAGRAM OF A MICROWAVE OVEN



MICROWAVE OVEN OPERATION

- Superstate encapsulates a number of separate states.
  - looks like a single state on a high-level model
  - expanded to show more detail on a separate diagram.



STATES AND STIMULI FOR THE MICROWAVE OVEN

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 (CONT.)

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.

MODEL-DRIVEN ENGINEERING

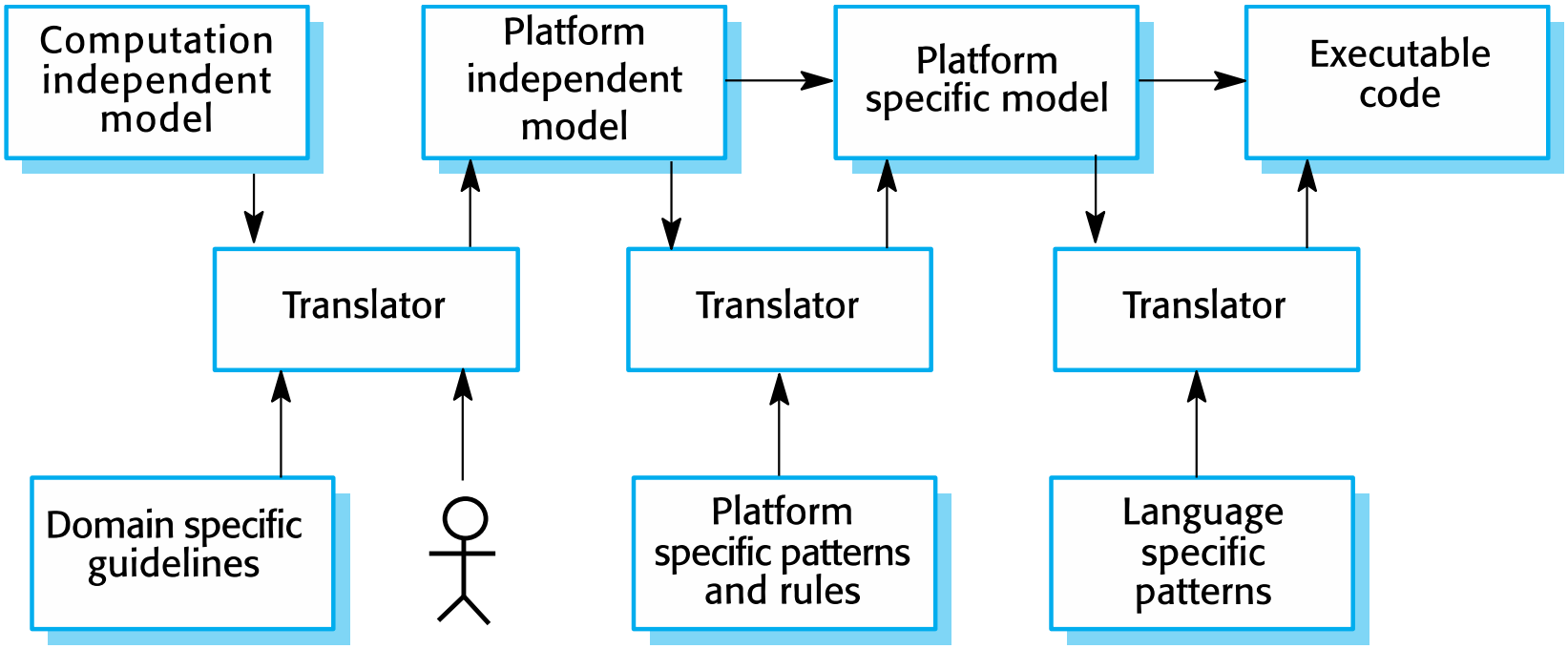
Self-study

MODEL-DRIVEN ENGINEERING

- Model-driven engineering (MDE) is an approach to software development where models rather than programs are the principal outputs of the development process.
  - The programs are then generated automatically from the models.
- Pros
  - Allows systems to be considered at higher levels of abstraction
  - Generating code automatically means that it is cheaper to adapt systems to new platforms.
- Cons
  - Models for abstraction and not necessarily right for implementation.
  - Savings from generating code may be outweighed by the costs of developing translators for new platforms.

MODEL DRIVEN ARCHITECTURE (MDA)

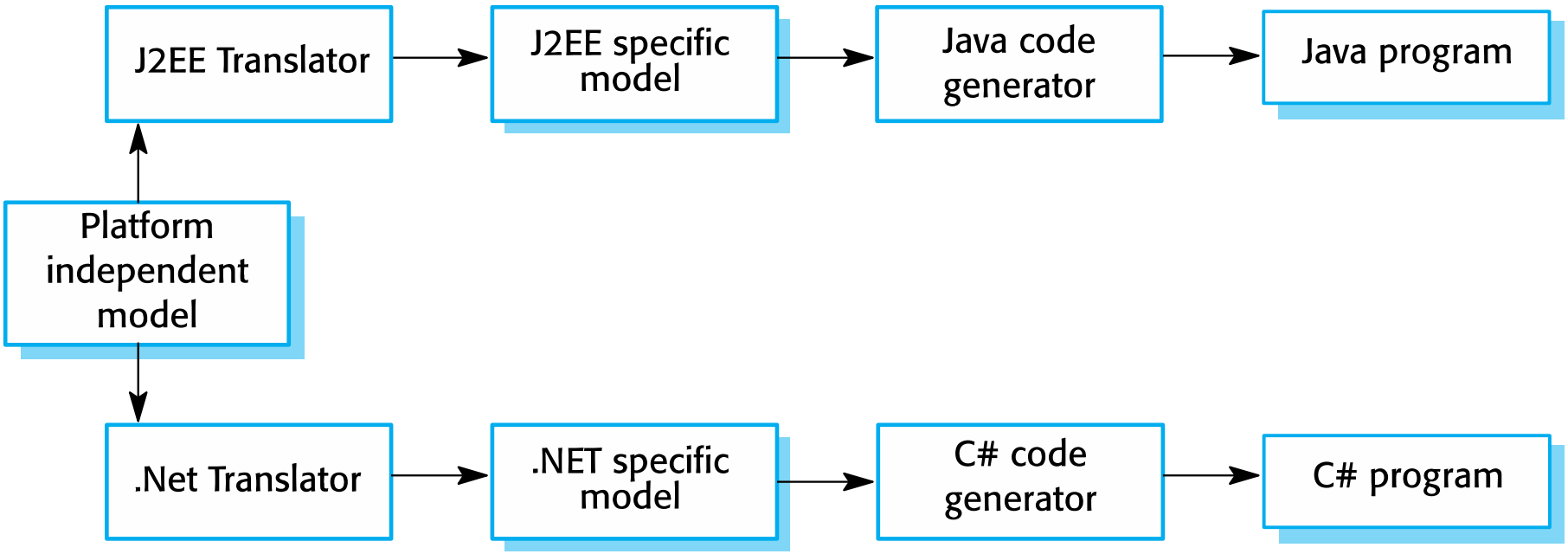
- ✓ The precursor of more general model-driven engineering
- ✓ A model-focused approach



TYPES OF MODEL

- A computation independent model (CIM)
  - These model the important domain abstractions used in a system. CIMs are sometimes called domain models.
- A platform independent model (PIM)
  - These model the operation of the system without reference to its implementation. The PIM is usually described using UML models that show the static system structure and how it responds to external and internal events.
- Platform specific models (PSM)
  - These are transformations of the platform-independent model with a separate PSM for each application platform. In principle, there may be layers of PSM, with each layer adding some platform-specific detail.

MULTIPLE PLATFORM-SPECIFIC MODELS



SUMMARY

- ✓ A model is an abstract view of a system
- ✓ Context models show how a system 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
  - 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.

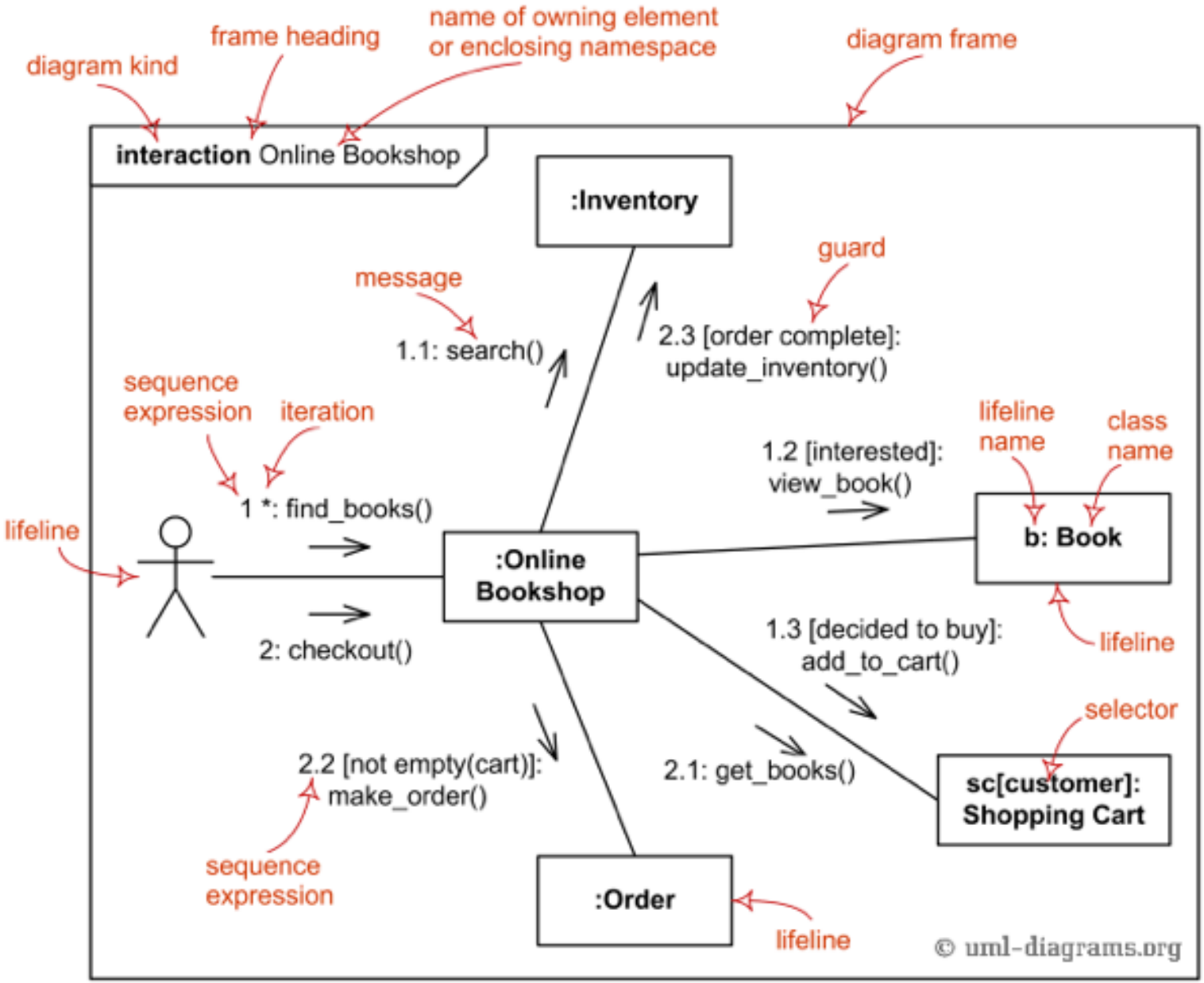
SUMMARY (CONT.)

- ✓ Behavioral models are used to describe the dynamic behavior of an executing system.
  - Can be by 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.

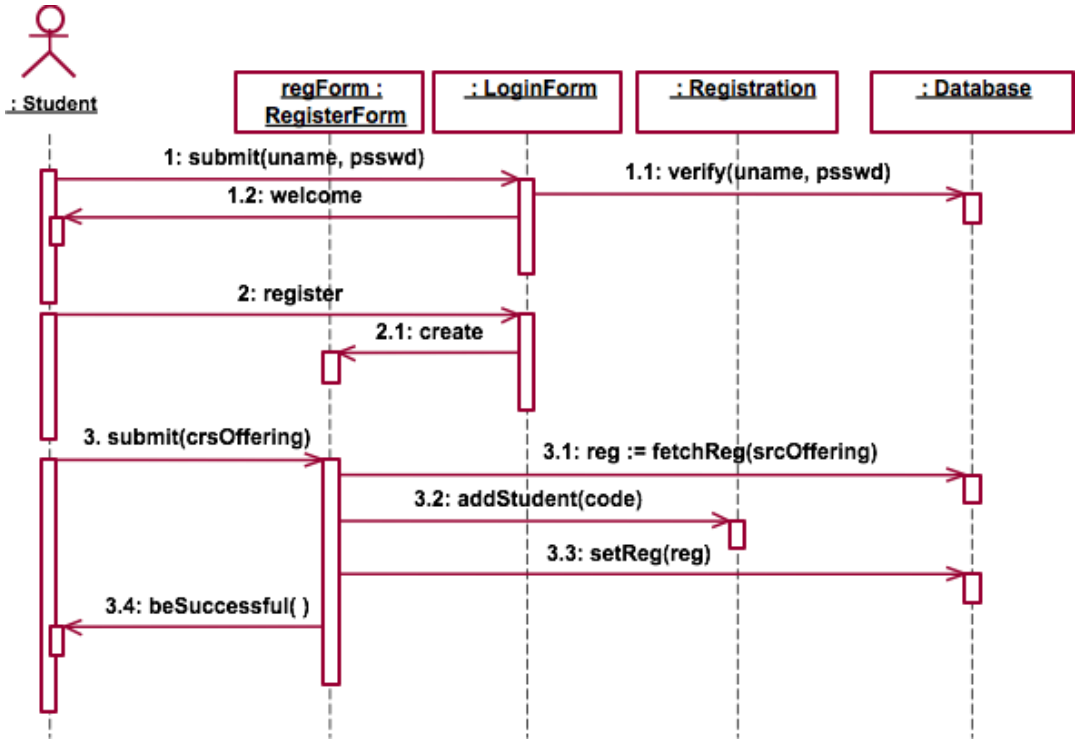
MORE ON UML

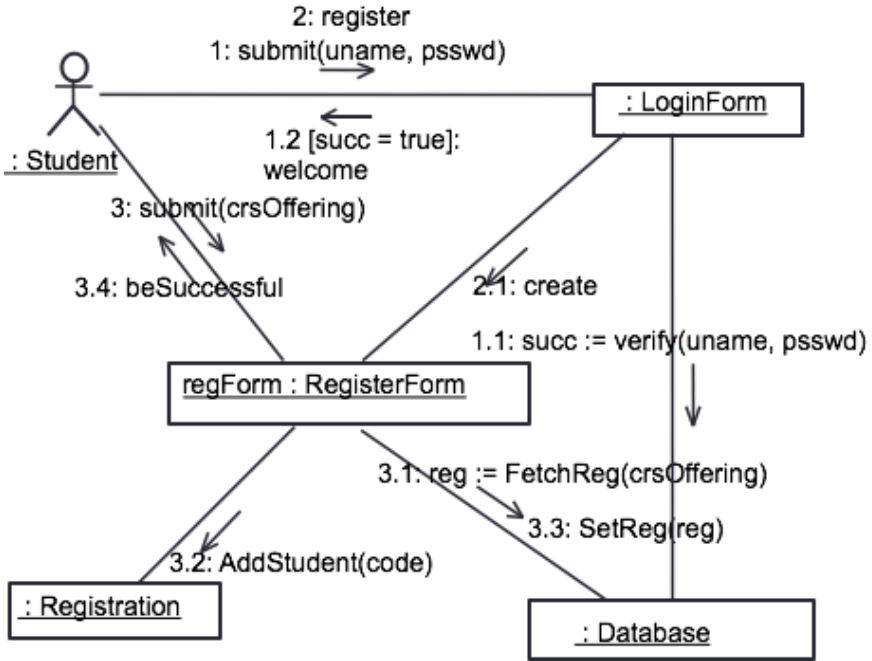
Self-study

COLLABORATION / COMMUNICATION DIAGRAMS

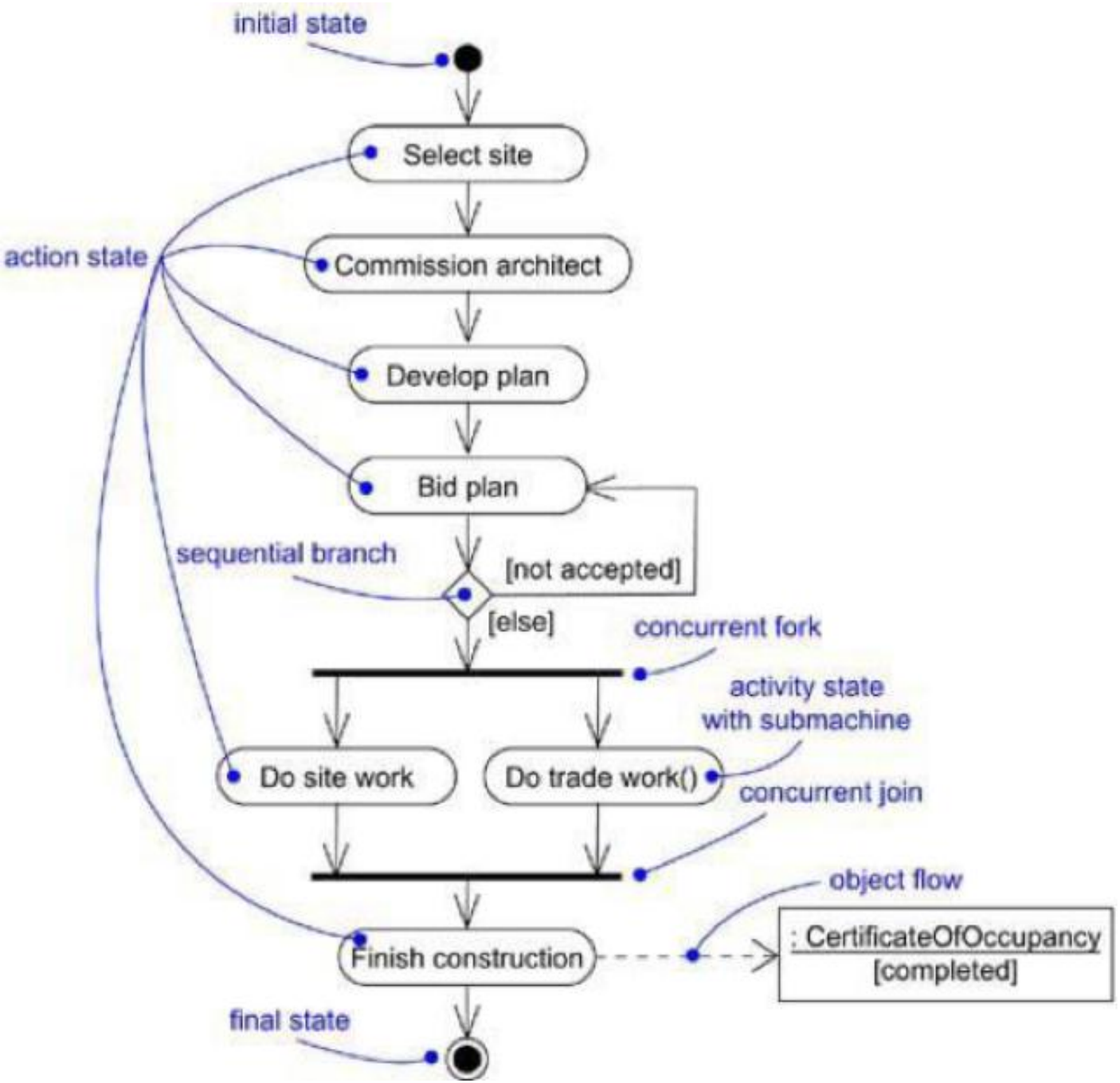


SEQUENCE VS. COLLABORATION DIAGRAMS





ACTIVITY DIAGRAMS



ACTIVITY DIAGRAMS (CONT.)