FACULDADE ENGENHARIA

Departamento de Informática

Engenharia de Software (14341, 16230, 15386)

System Modeling #1

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

Topics covered

✧ Interaction models

✧ Context 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 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 modeling helps the analyst to understand the functionality of the system and models are used to communicate with customers.

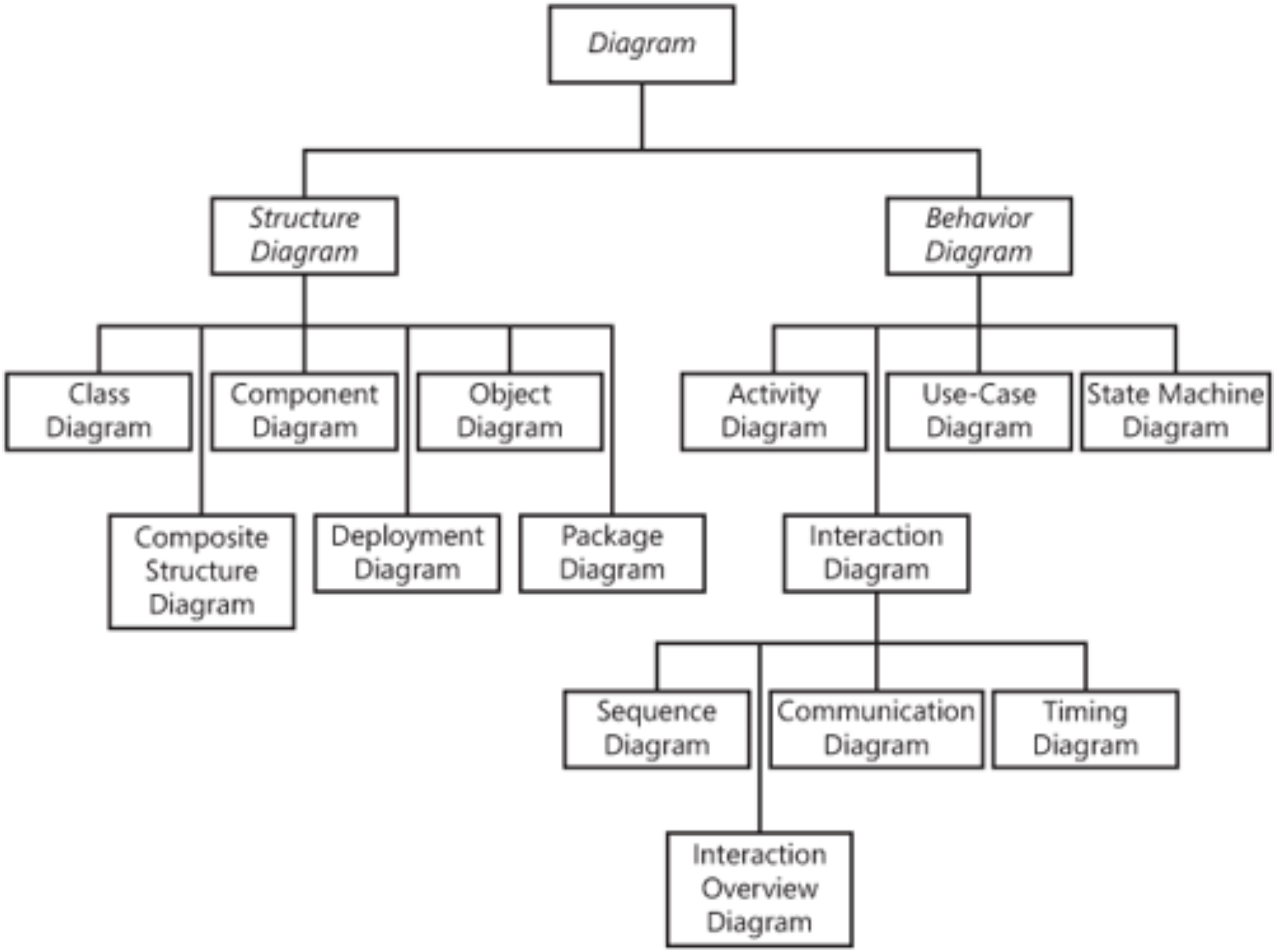
Existing and planned system models

- 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.

Hierarchy of UML diagrams



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.

Interaction models

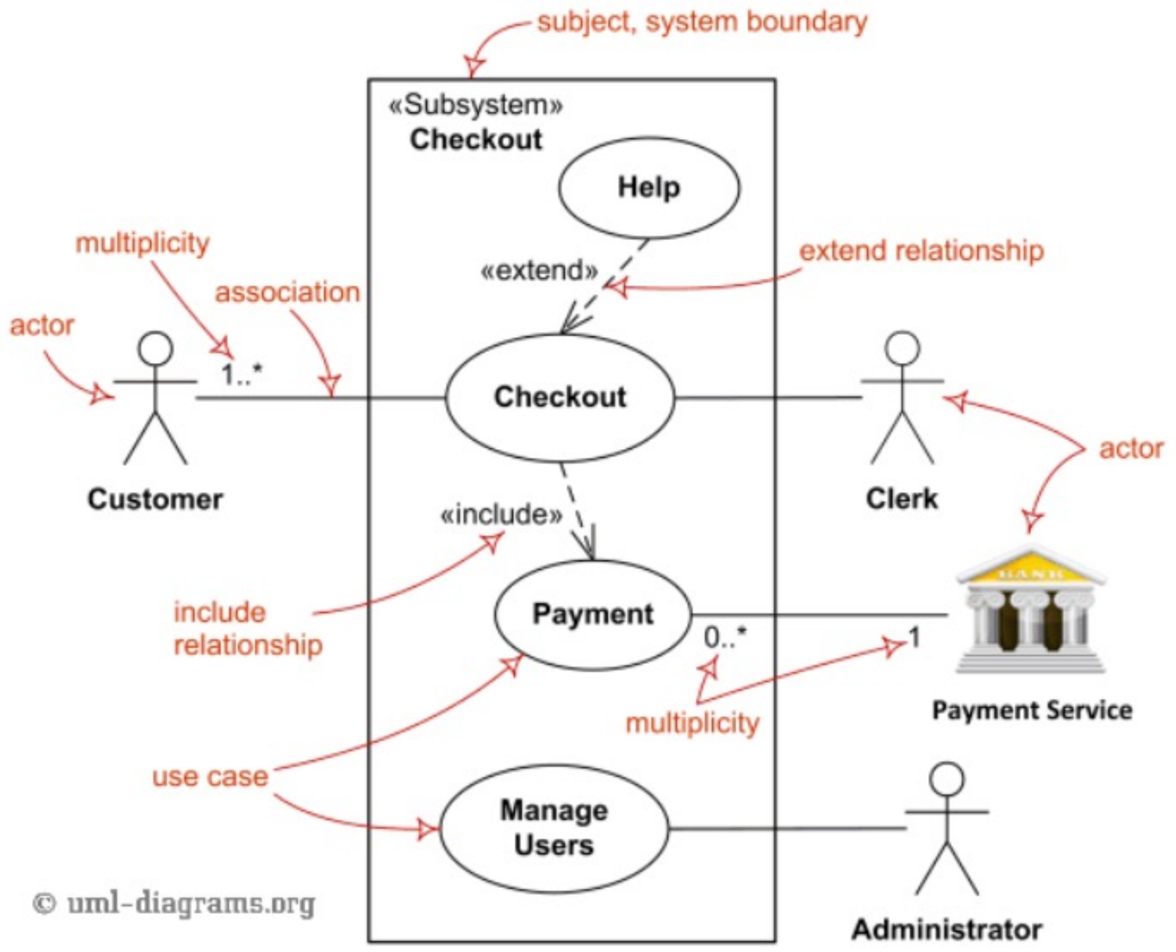
Interaction models

- ✧ 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 case diagrams

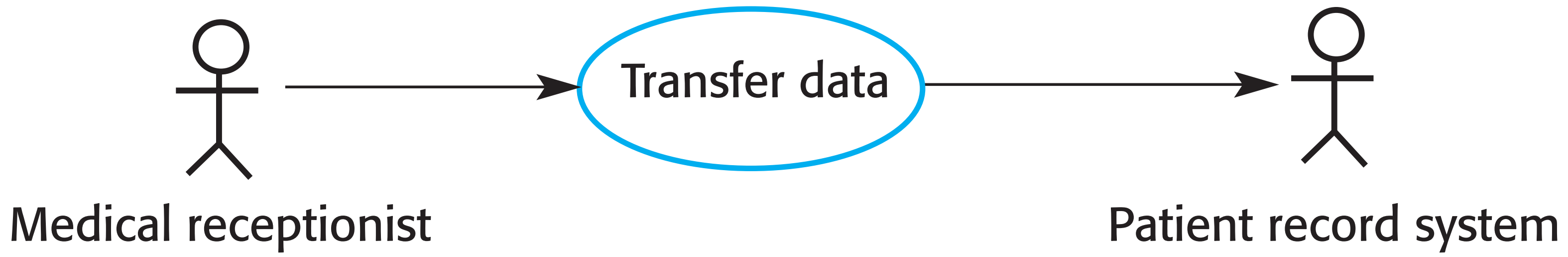


Use case diagrams

- ✧ <u>Actor</u>: is an entity outside of a process which trigger information exchange with the process.
- ✧ <u>Use case</u>: is an event taking place within a process and it is often triggered by an actor.
- ✧ <u>Relationship</u>: information flow between an actor and a use case or between two use cases. An <u>*extend relationship*</u> exists between two similar <u>use cases</u> where the second one has some extra activities, that is, the activities of the first use case are extended in the second one. On the contrary, an <u>*include relationship*</u> is a generalization denoting the inclusion of the behavior described by another use case.

Transfer-data use case

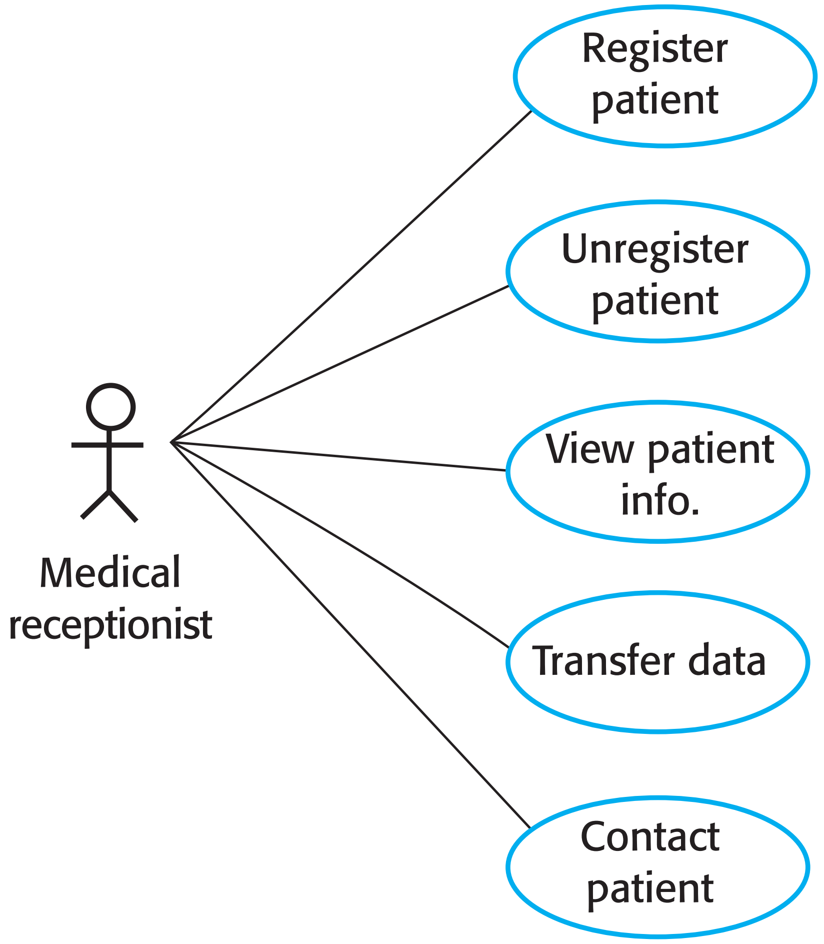
✧ A use case in the Mentcare system



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 Mentcare 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'



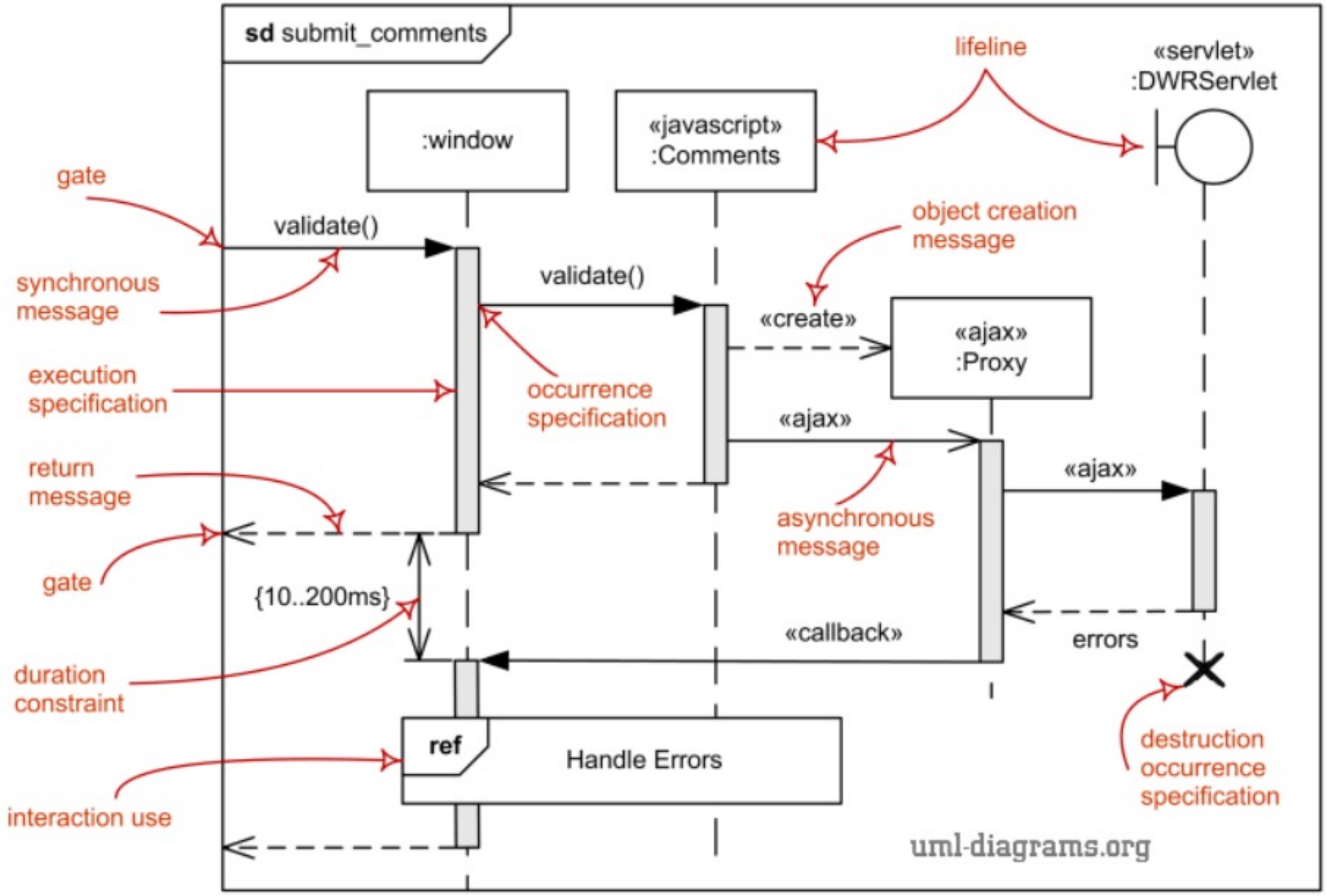
Practice: use case diagram

✧ Determine the <u>use case diagram</u> of the login process in the iLearn system.

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 diagrams



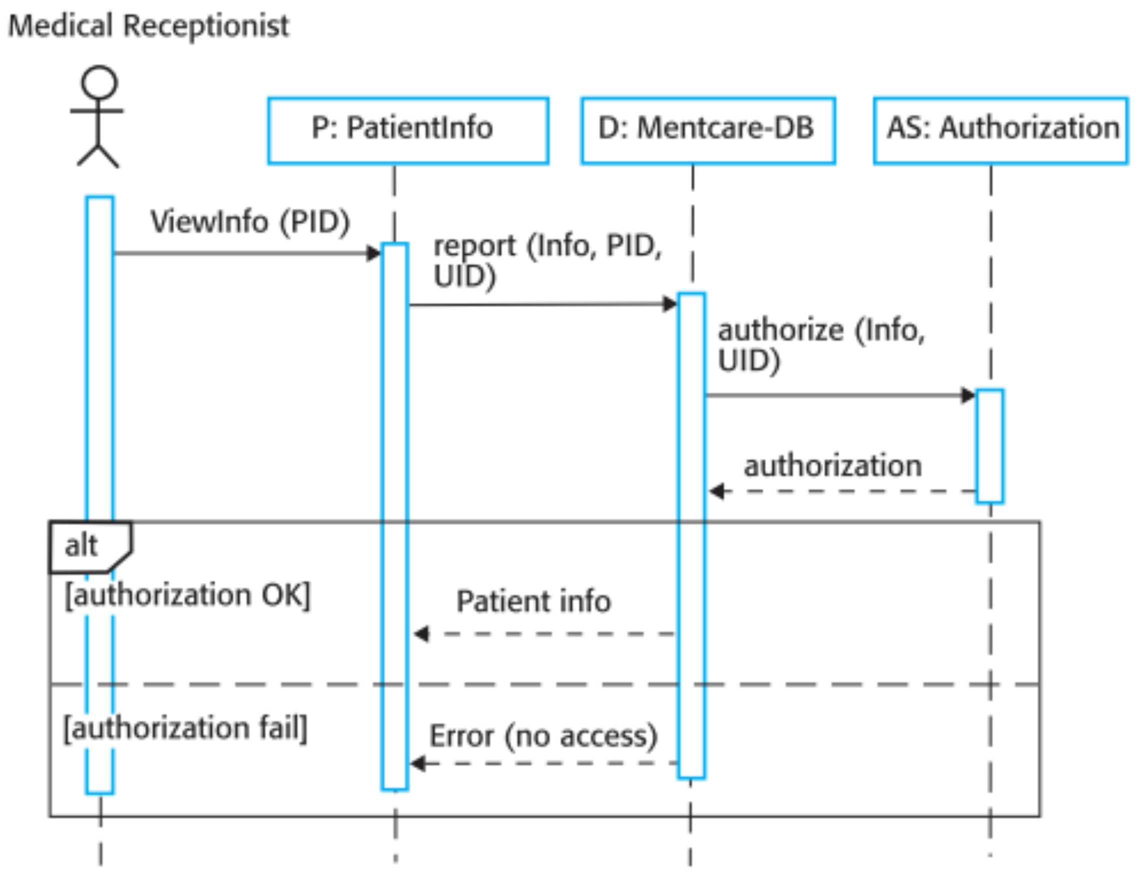
Sequence diagrams

- ✧ Arrow: dashed arrow back indicates return. Standard arrow (forward or back) indicates flat flow control.
- ✧ Asynchronous message: invocation of operation of target lifetime. The sender does not pass the control to the receiver. The sender and the receiver carry on their work concurrently.
- ✧ Execution specification (or activation): means that an object is running its code or it is in the stack waiting for another objects' method. May also represents self-calls and callbacks.
- ✧ Lifetime: represents an individual participant in the interaction, namely, entity object models information. It holds information and some operations that naturally related to the information.

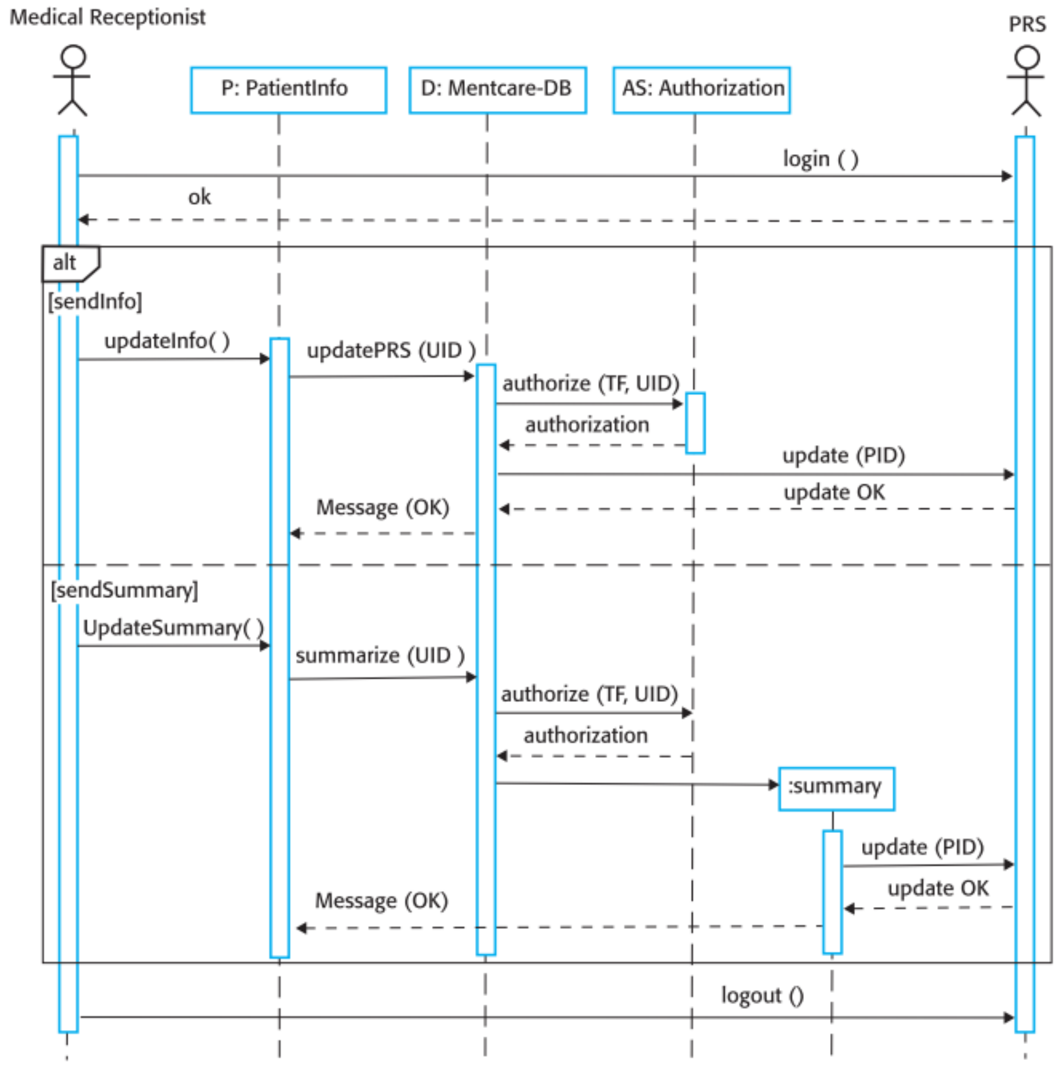
Sequence diagrams

- ✧ Message (method call): horizontal arrow to other object. Write message name and arguments above arrow.
- ✧ Object: name syntax: <objectname>:<classname>.
  - > *objectname* not defined: *anonymous object*
  - > *classname* not defined: object of unknown class
- ✧ Return message: pass of information back to the caller of a correspondent former message.
- ✧ Synchronous message: invocation of operation of target lifetime. The sender passes the control to the receiver and cannot do anything until the receiver sends the control back.

Sequence diagram for View patient information



Sequence diagram for Transfer Data



Context models

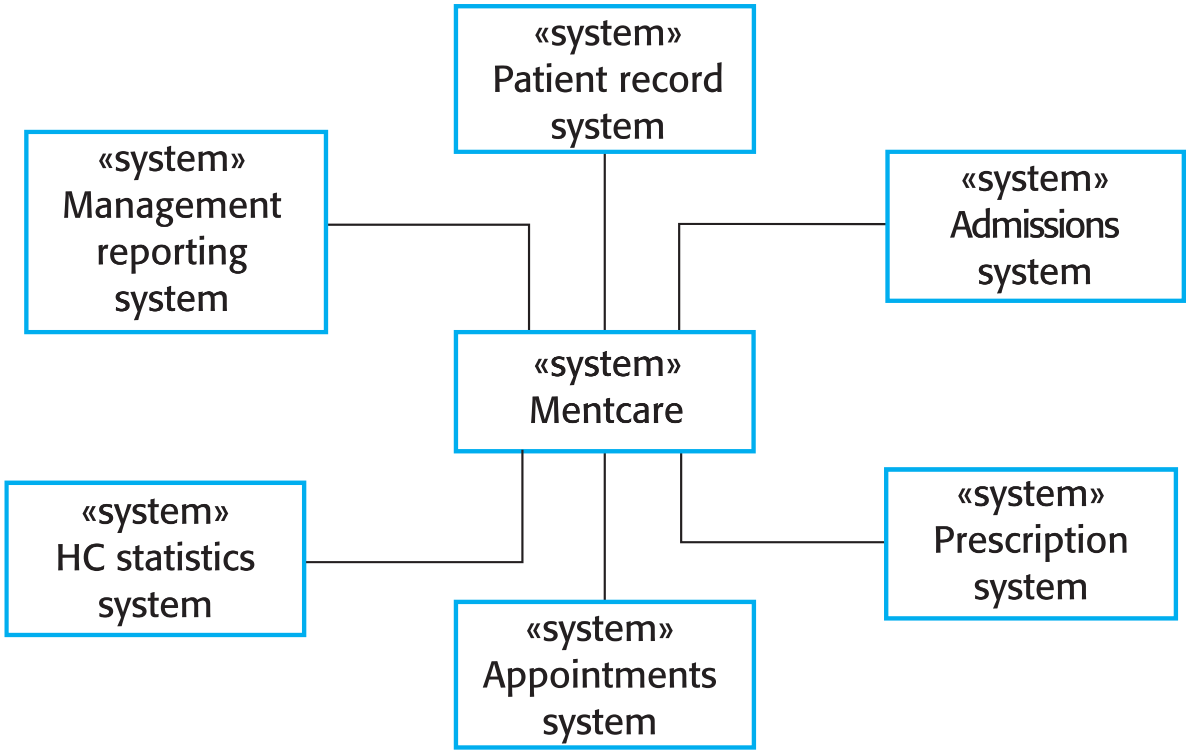
Context models

- ✧ 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.

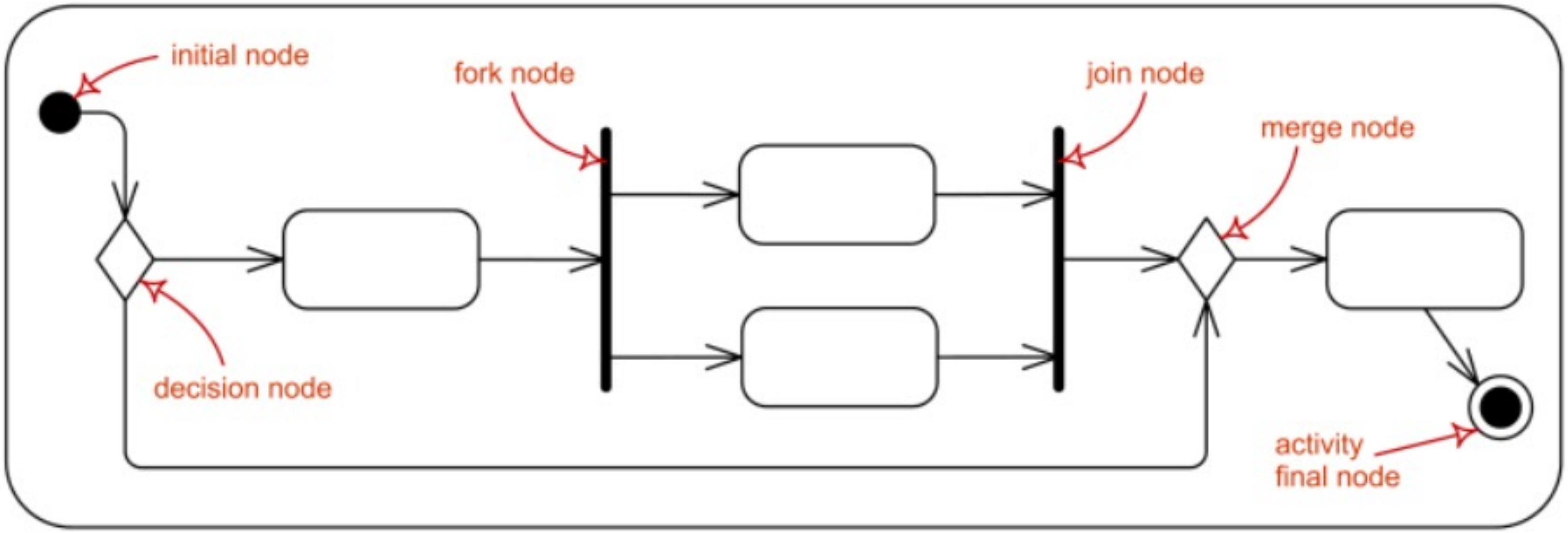
The context of the Mentcare system


«system» Patient record system
«system» Management reporting system
«system» Admissions system
«system» Mentcare
«system» HC statistics system
«system» Prescription system
«system» Appointments 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.

Activity diagrams



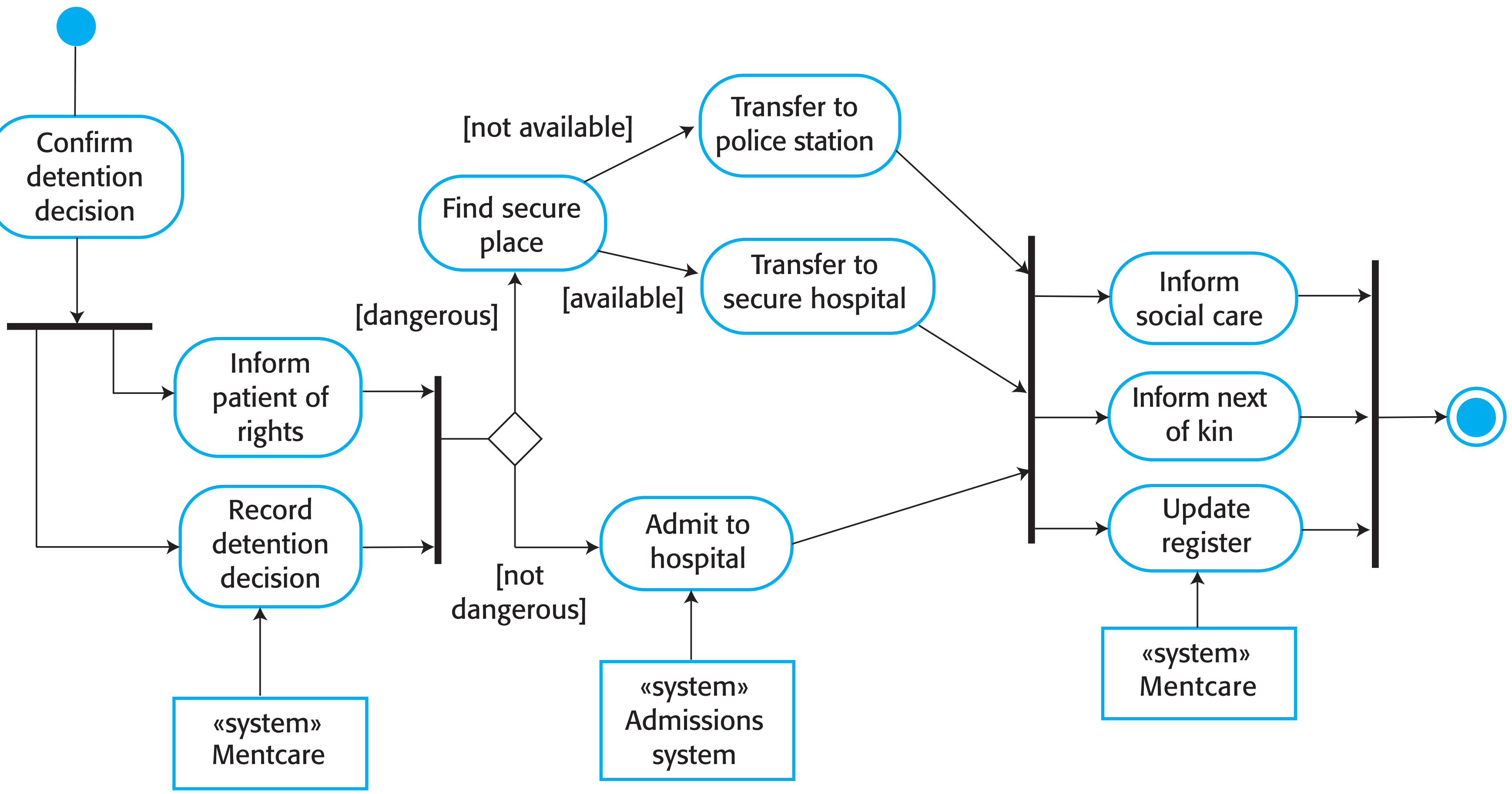
Activity diagrams

- ✧ Action: represents a single step within an activity that is not further decomposed within the activity.
- ✧ Decision node: accepts tokens on an incoming edge and presents them to multiple outgoing edges. Which of the edges is actually traversed depends on the evaluation of the guards on the outgoing edges.
- ✧ Final node: an activity may have more than one activity final node; the first one reached stops all flows in the activity.
- ✧ Initial node: is a control node at which flow starts when the activity is invoked. An activity may have more than one initial node.

Activity diagrams

- ✧ Fork node: one incoming transition, and multiple outgoing parallel transitions and/or object flows.
- ✧ Join node: multiple incoming transitions and/or object flows; one outgoing transitions. The outgoing continuation does not happen until all the inputs arrive from all flows.
- ✧ Merge node: is a control node that brings together multiple alternate flows. It is not used to synchronize concurrent flows but to accept one among several alternate flows. A merge node has multiple incoming edges and a single outgoing edge.
- ✧ Object node: is an abstract activity node that helps to define the object flow in an activity. In addition, indicates that an instance of a classifier might be available at a particular point in the activity.

Process model of involuntary detention



Practice: activity diagram

✧ Determine the activity diagram of the login process in the iLearn system.

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 behaviour.

- ✧ 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.

HOW
WHY
WHERE
HOW
WHEN
WHERE
WHO
WHY
WHERE
WHO
WHEN
WHAT
WHAT
WHO
WHY
WHEN
HOW
WHAT
WHAT
WHEN
WHO
WHERE
WHY
HOW