

4. SYSTEM MODELING

Contents:

- 4.1 Context models
- 4.2 Interaction models
- 4.3 Structural models
- 4.4 Behavioral models
- 4.5 Package Structure

System Modeling

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.

Different perspective for System modeling

1. **An external perspective,**
 - where you model the **context** or **environment** of the system.
2. **An interaction perspective,**
 - where you model the **interactions between a system and its environment**, or between the components of a system.
3. **A structural perspective,**
 - where you model **the organization of a system** or the **structure of the data** that is processed by the system.
4. **A behavioral perspective,**
 - where you model the **dynamic behavior of the system** and how it responds to events.

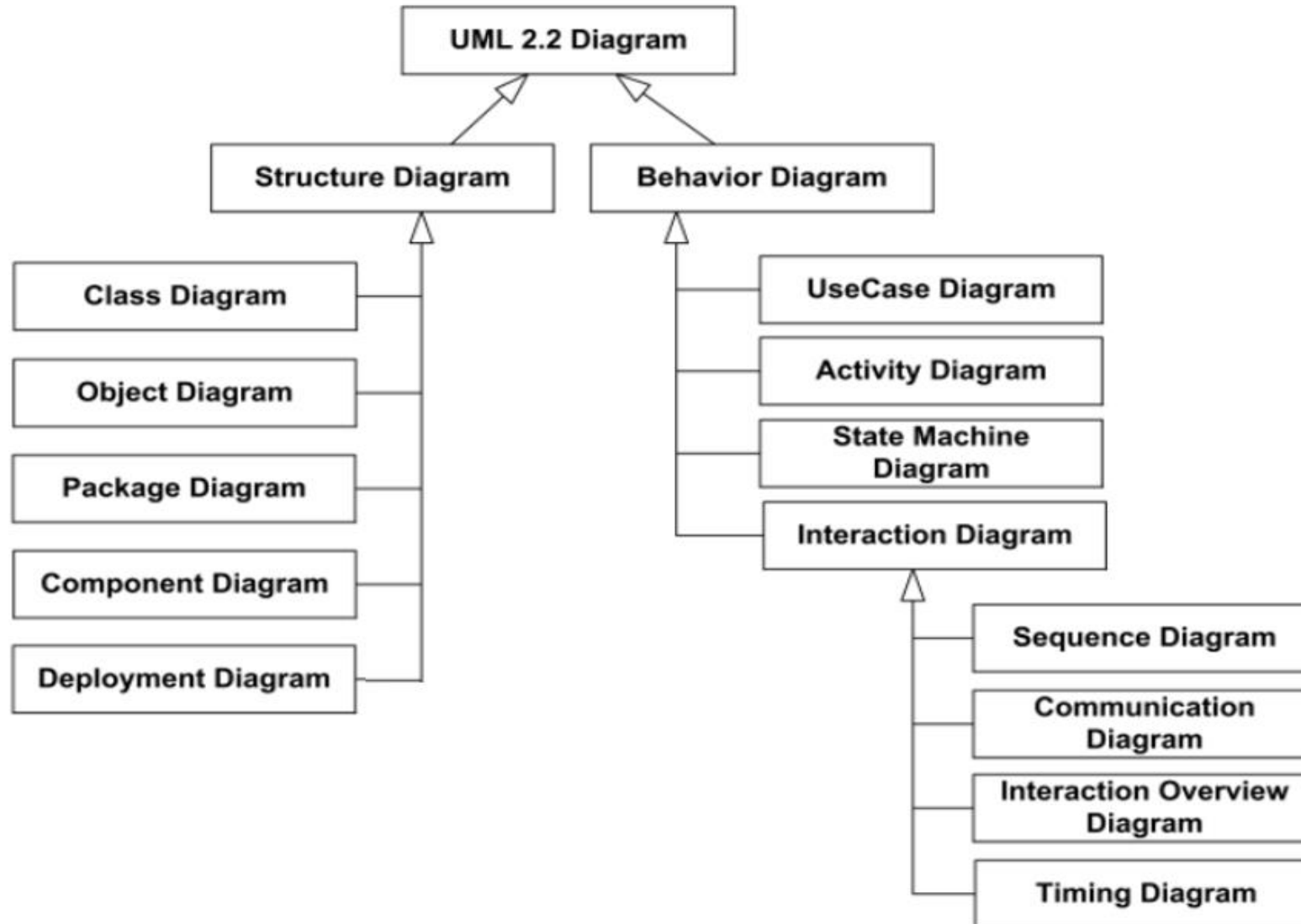
The Unified Modeling Language

- The Unified Modeling Language (UML) is a set of 13 different diagram types that may be **used to model software systems**.
- It emerged from work in the 1990s on object-oriented modeling, where similar object-oriented notations were integrated to create the UML.
- A major revision (UML 2) was finalized in 2004.
- The UML **is universally accepted as the standard approach for developing models of software systems**.
- “The UML **is a graphical language** for specifying, visualizing, constructing and documenting the **artifacts** of the software system”.
- Variants, such as SysML, have been proposed for more general system modeling.

Use of UML

- **Specifying** – provides the means to model the system precisely, unambiguously and completely.
- **Visualizing** – UML provides graphical notation which articulates and unambiguously communicates the overall view of the system.
- **Constructing** - UML provides the 'design' dimension to the models built. These are language independent and can be implemented in any programming language.
- **Documenting** – every software project involves a lot of documentation from the inception phase to the deliverables.

Classification of UML Diagrams



a) Structural Diagrams

- Structural diagrams are used to represent a static view of a system.
- It represents a part of a system that makes up the structure of a system.
- A structural diagram shows various objects within the system.
- Following are the various structural diagrams in UML:
 - Class diagram
 - Object diagram
 - Package diagram
 - Component diagram
 - Deployment diagram

b) Behavioral Diagrams

- Any real-world system can be represented in either a static form or a dynamic form. A system is said to be complete if it is expressed in both the static and dynamic ways.
- The behavioral diagram **represents the functioning of a system.**
- UML diagrams that **deals with the moving or dynamic parts of the system** are called behavioral diagrams.
- Following are the various behavioral diagrams in UML:
 - Activity diagram
 - Use case diagram
 - State machine diagram
 - Interaction diagram

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.

4.1 Context models

4.1 Context models

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

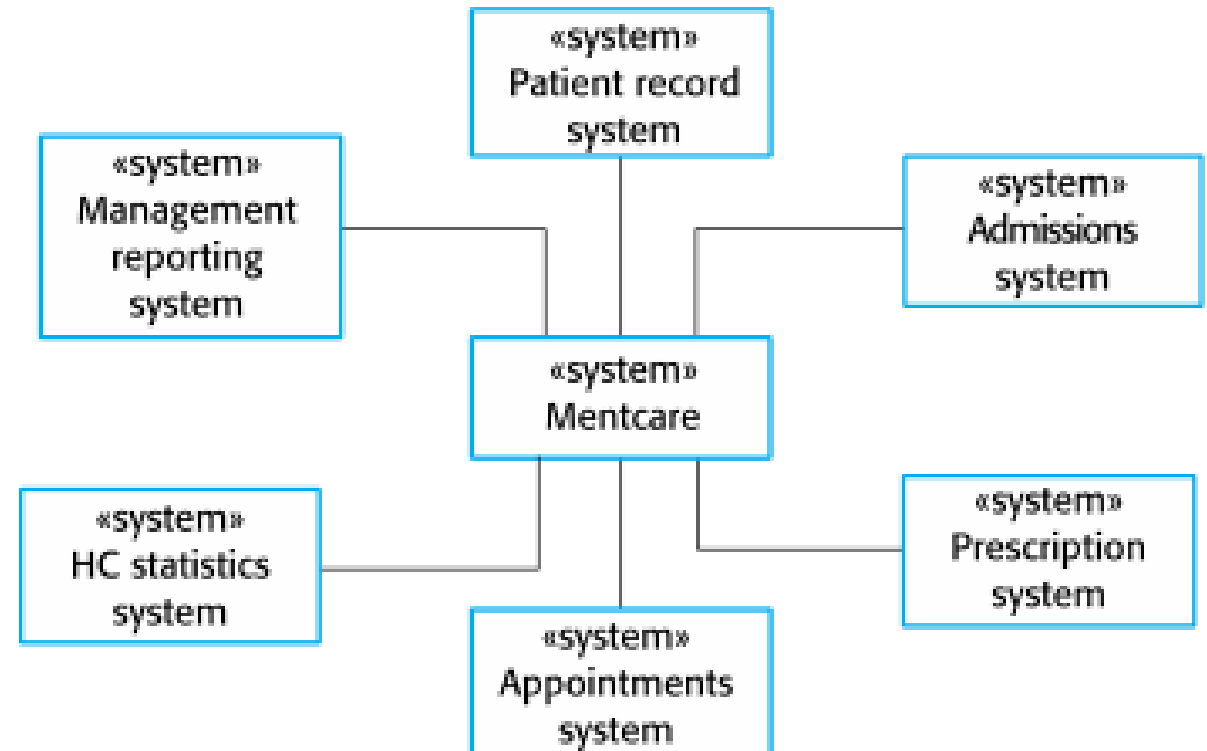
i) Context Diagram

- A system context diagram (SCD) in engineering is **a diagram that defines the boundary between the system, or part of a system, and its environment**, showing the entities that interact with it.
- This diagram is **a high level view of a system**.
- System context diagrams show a system, as a whole and its inputs and outputs from/to external factors.
- System Context Diagrams ... **represent all external entities that may interact with a system** ... Such a diagram pictures the system at the center, with no details of its interior structure, surrounded by all its interacting systems, environments and activities.
- The objective of the system context diagram is to focus attention on external factors and events that should be considered in developing a complete set of systems requirements and constraints.
- System context diagrams **are used early in a project** to get agreement on the scope under investigation.
- Context diagrams are typically **included in a requirements document**.
- These diagrams must be read by all project stakeholders and thus should be written in plain language, so the stakeholders can understand items within the document.

- The context diagram is a simple model that defines the boundaries and interfaces of the proposed systems with the external world.
- It identifies the entities outside the proposed system that interact with the system.
- **When to use context diagram?**
 - A context diagram is helpful in a variety of situations. Here are some of the common scenarios:
 1. Implementing or updating software
 2. Reviewing a business process
 3. Tackling resource management

The context of the Mentcare system

- Mental Health Care – Patient Management Systems
- 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.

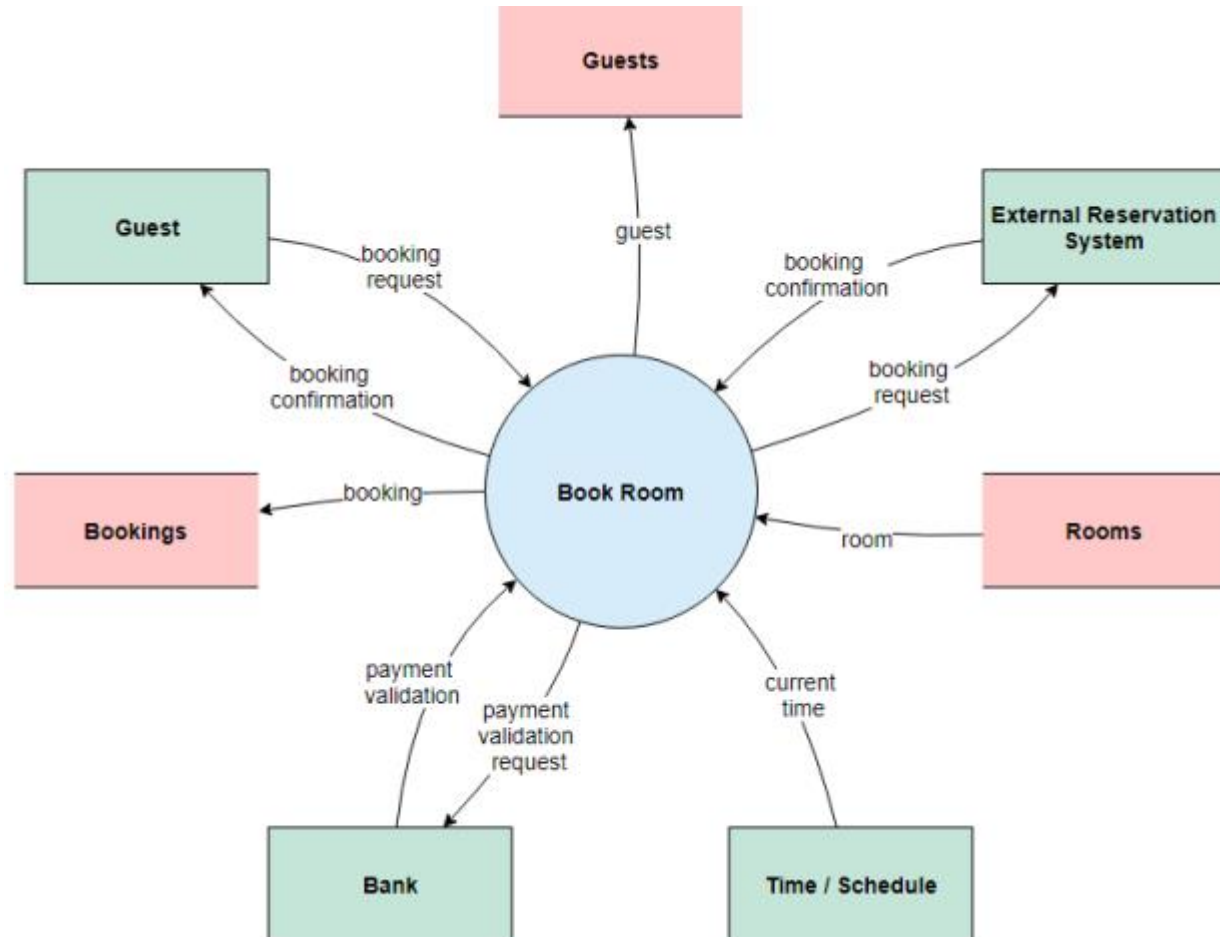


MENTCARE: A PATIENT INFORMATION SYSTEM FOR MENTAL HEALTH CARE

- A patient information system to support mental health care is a medical information system that maintains information about patients suffering from mental health problems and the treatments that they have received. • Most mental health patients do not require dedicated hospital treatment but need to attend specialist clinics regularly where they can meet a doctor who has detailed knowledge of their problems. • To make it easier for patients to attend, these clinics are not just run in hospitals. They may also be held in local medical practices or community centres.
- Mentcare is an information system that is intended for use in clinics. • It makes use of a centralized database of patient information but has also been designed to run on a PC, so that it may be accessed and used from sites that do not have secure network connectivity. • When the local systems have secure network access, they use patient information in the database but they can download and use local copies of patient records when they are disconnected.

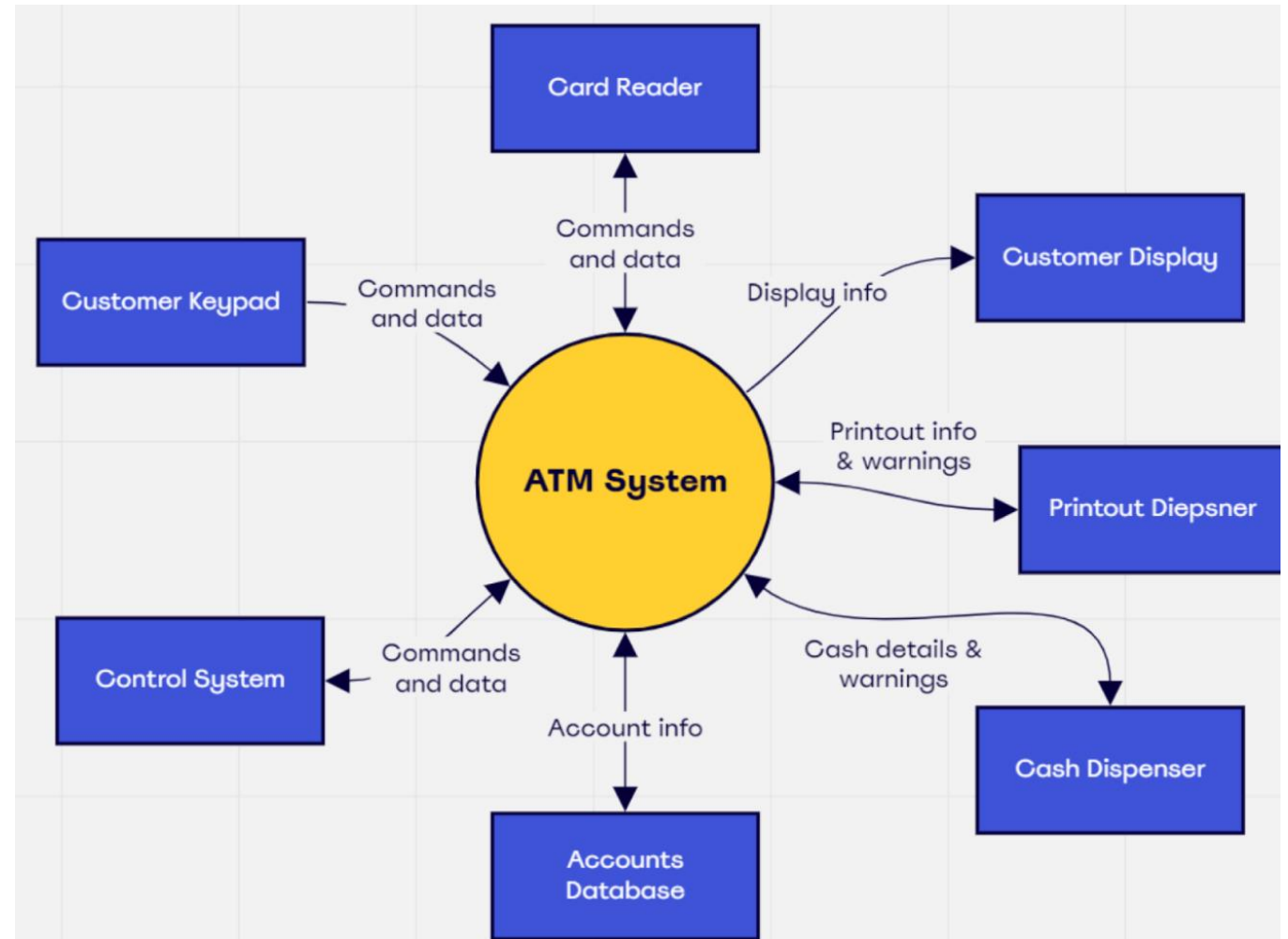
Context Diagram: Online Reservation System

- Context diagram for Online Reservation System.
- Main system in the center circle.
- Using squares, rectangles, or any other shapes you want, you can list external entities that interact with the system.
- arrows are used to represent the flow of data between the system and each external element.



Context Diagram : ATM

- ATM context diagram
- outlines how your banking ATM process will work.
- The diagram shows all the external units that interact with the ATM system and how data flows between them.



4.2 Interaction models

4.2 Interaction Modeling

- 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.
- Interaction models describe the relationship between a user and the system, outlining the flow and sequence of actions and responses.
- Interaction model uses:
 - **Use case diagrams**
 - **sequence diagrams**
 - **Activity diagram**
 - **Collaborative diagram**

i) Use case modeling

- Use cases were developed originally to support requirements elicitation and now incorporated into the UML.
- A use case can be taken as a simple description of **what a user expects from a system** in that interaction.
- 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 in the Mentcare system involving the role 'Medical Receptionist'**

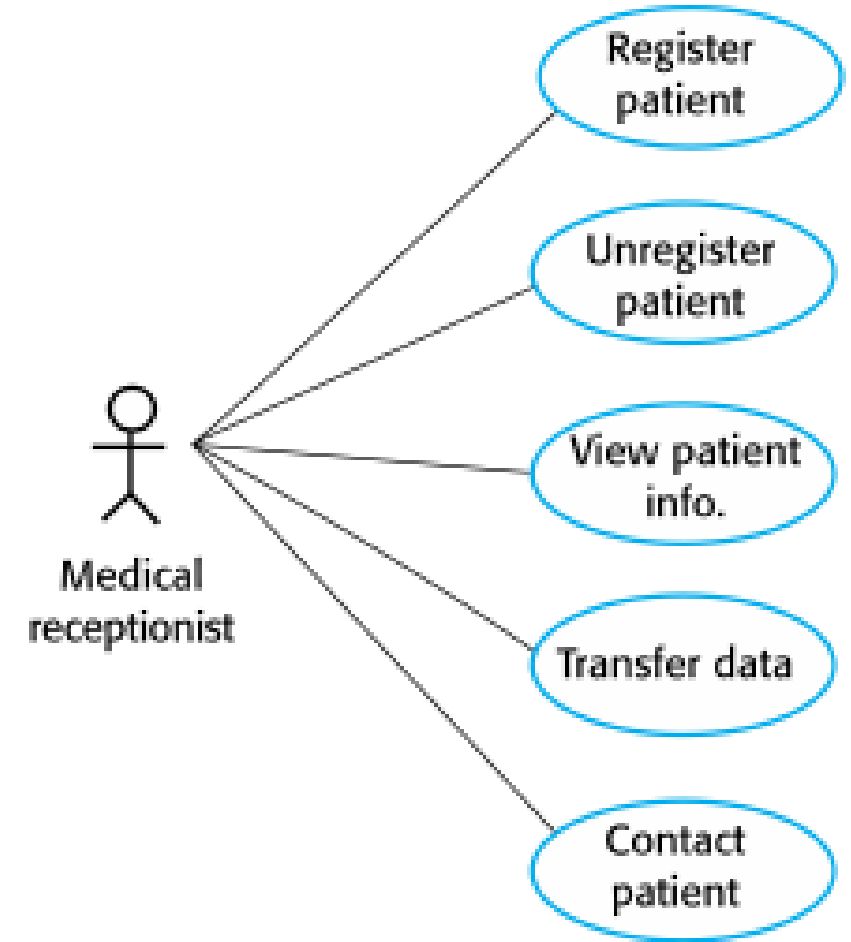


Figure : Use cases involving the role "Medical receptionist"

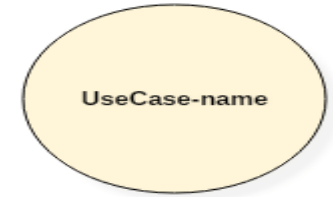
Use-case Diagram

- **Use Case Diagram** captures the system's functionality and requirements by using actors and use cases.
- Use Cases model the services, tasks, function that a system needs to perform.
- Use cases represent high-level functionalities and how a user will handle the system.
- It models how an external entity interacts with the system to make it work.
- Use case diagrams are responsible for visualizing the external things that interact with the part of the system.
- It models the system from the end-users point of views. So, the actor should be identified first and their roles are also defined.
- **Purposes of Use-case diagrams:**
 - To gather requirements of a system
 - To present an outside view of a system
 - Identify external and internal factor influencing the system.
 - Show the interaction among requirements and actors

Notations in Use-case diagrams:

- **Use-case:**

- Use cases are used to represent high-level functionalities and how the user will handle the system.
- It is denoted by an oval shape with the name of a use case written inside the oval shape



- **Actor:**

- The actor is an entity that interacts with the system. A user is the best example of an actor.
- An actor initiates the use case from outside the scope of a use case.
- It can be any element that can trigger an interaction with the use case.
- One actor can be associated with multiple use cases in the system.

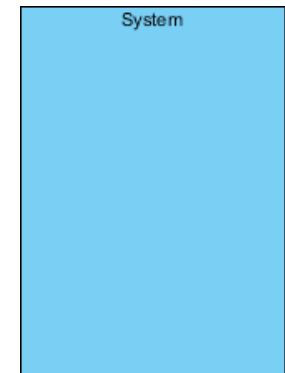


- **Communication Link**

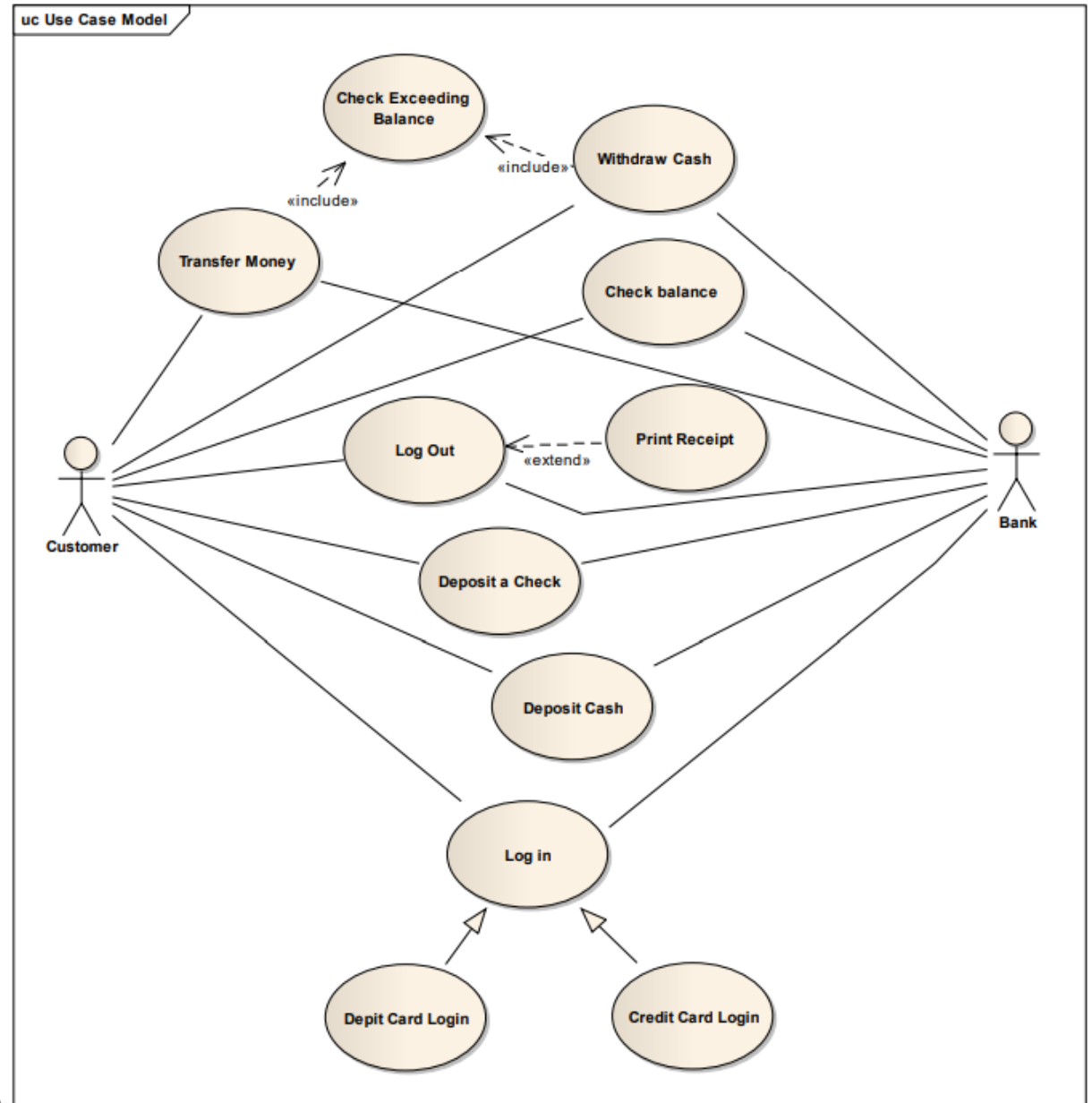
- The participation of an actor in a use case is shown by connecting an actor to a use case by a solid link.

- **Boundary of system**

- The system boundary is potentially the entire system as defined in the requirements document.
- For large and complex systems, each module may be the system boundary.



Examples: Use-case diagrams



Examples: Use-case diagrams

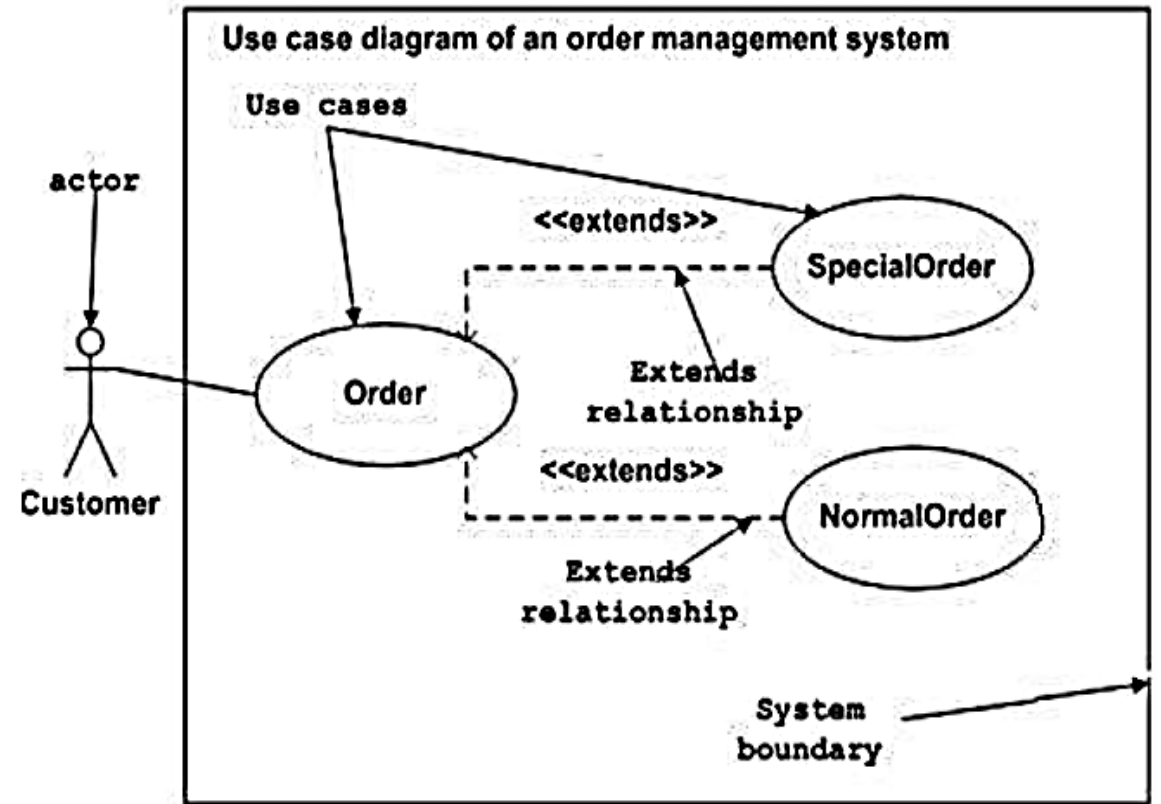
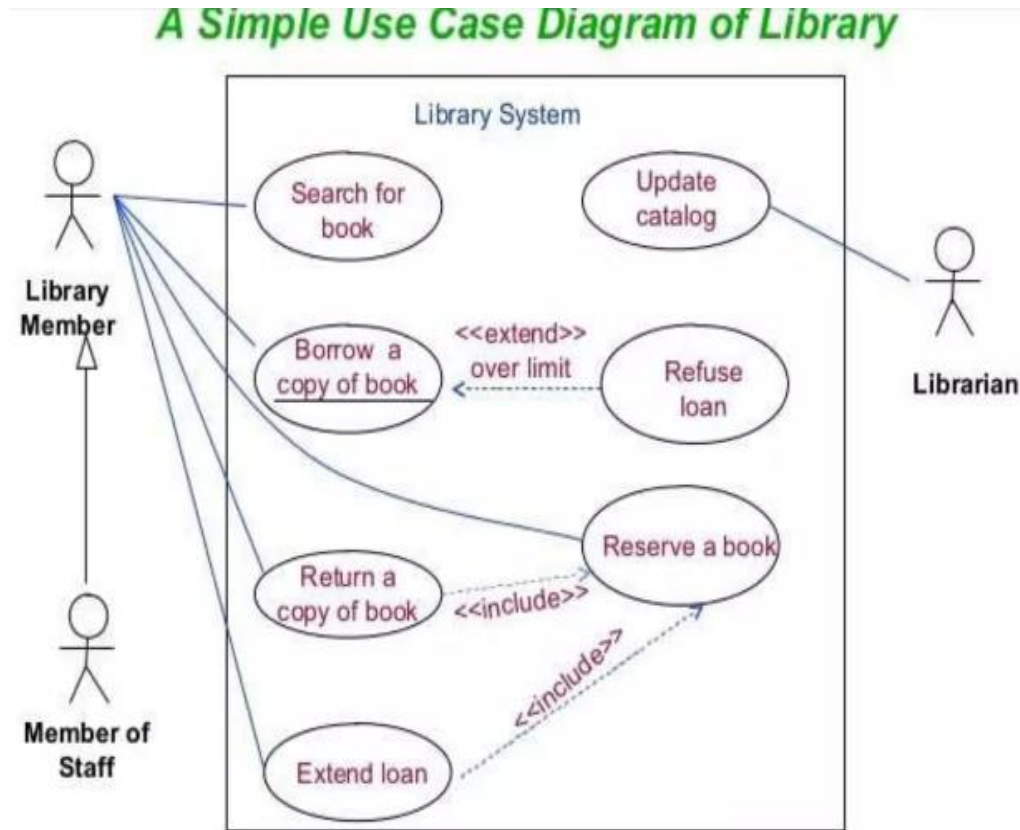
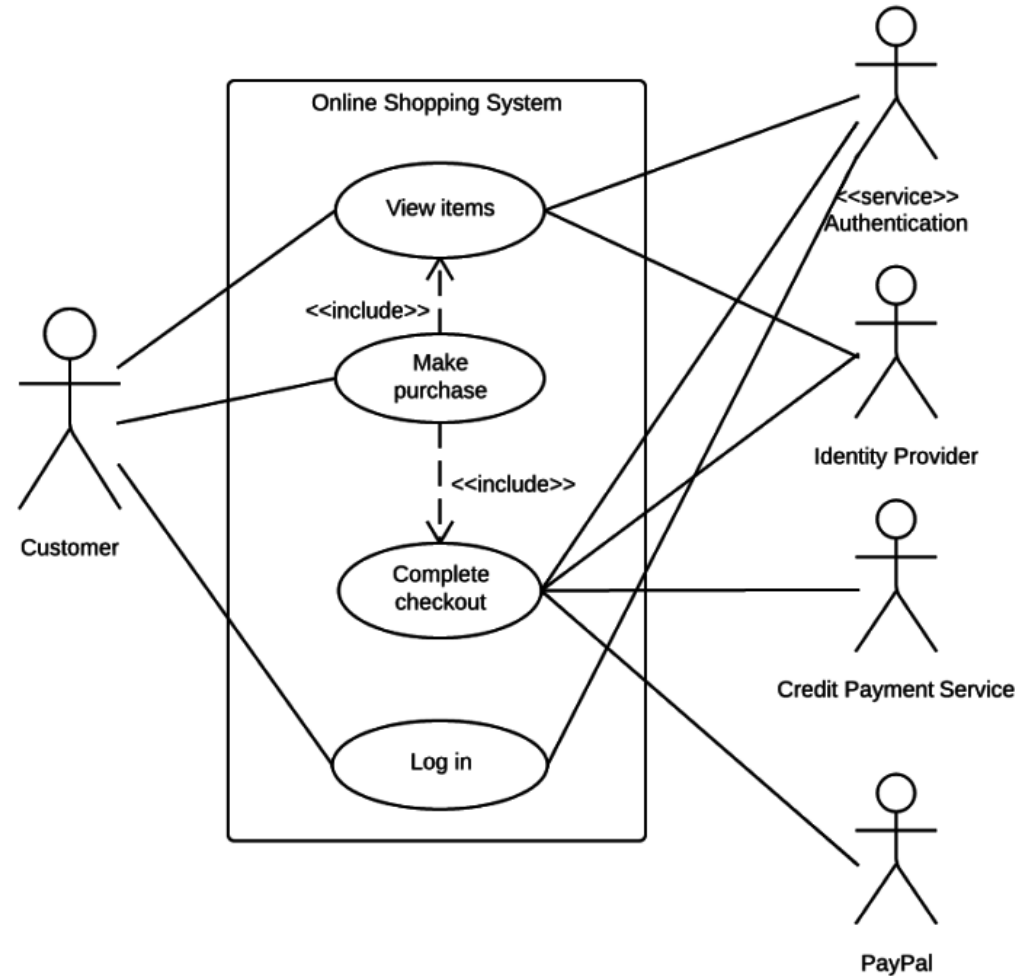
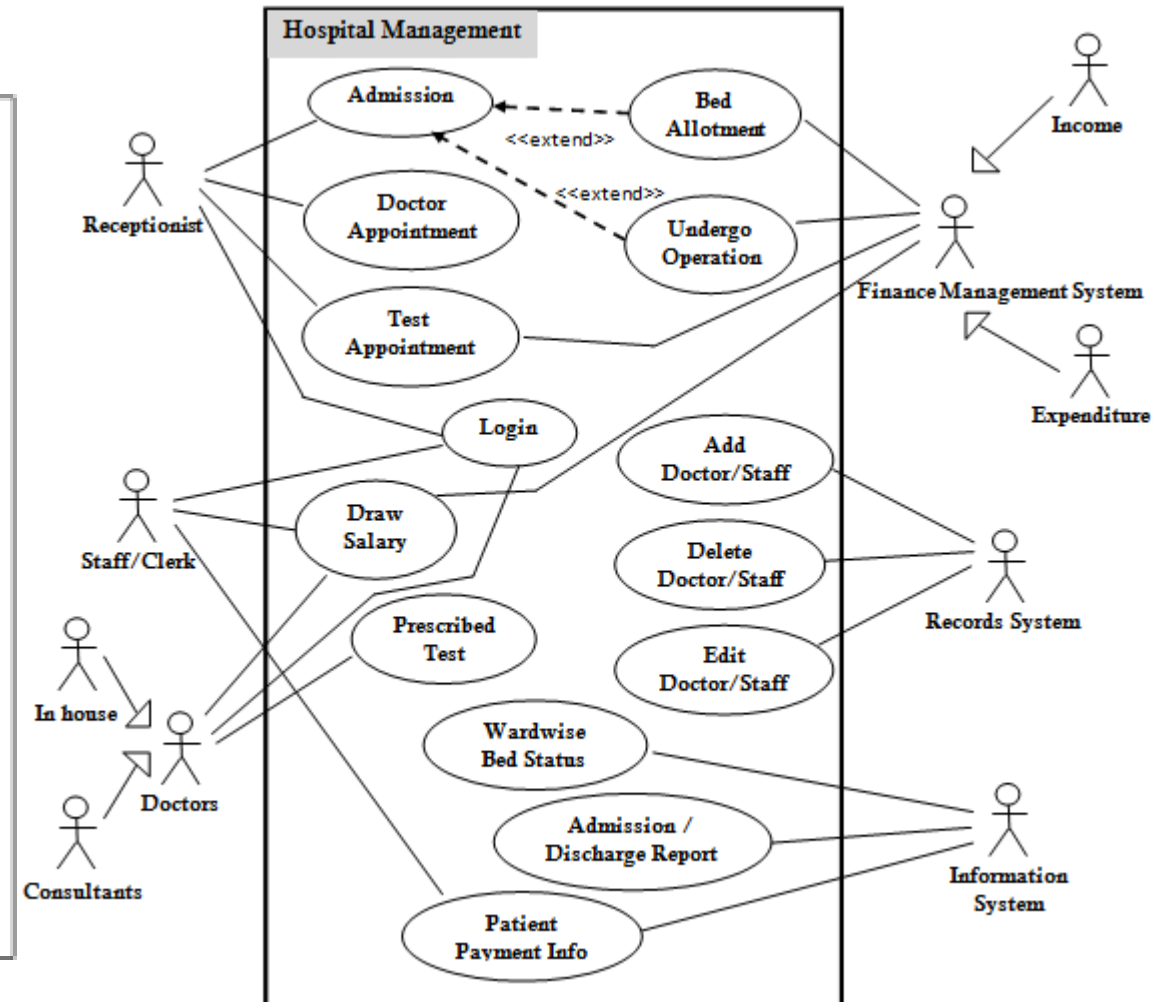
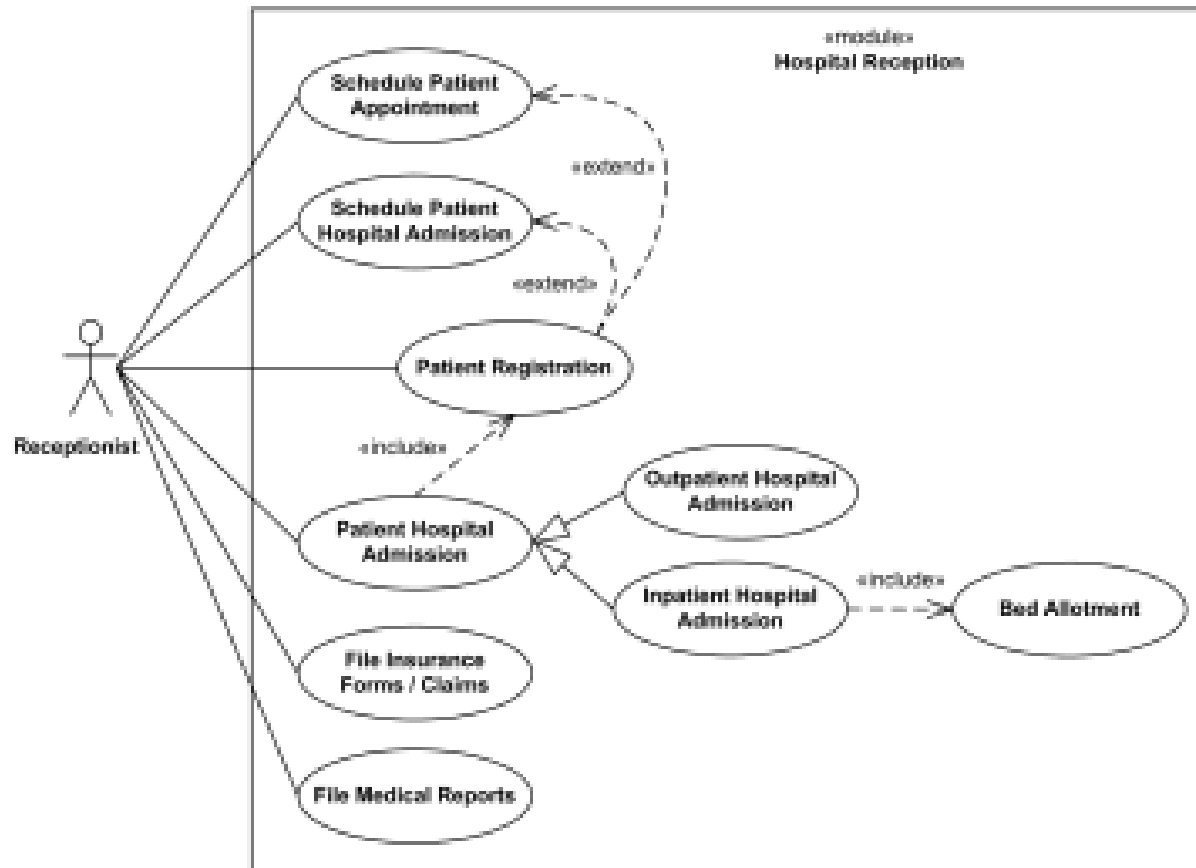


Figure: Sample Use Case diagram

Order Management System: Use-case diagrams





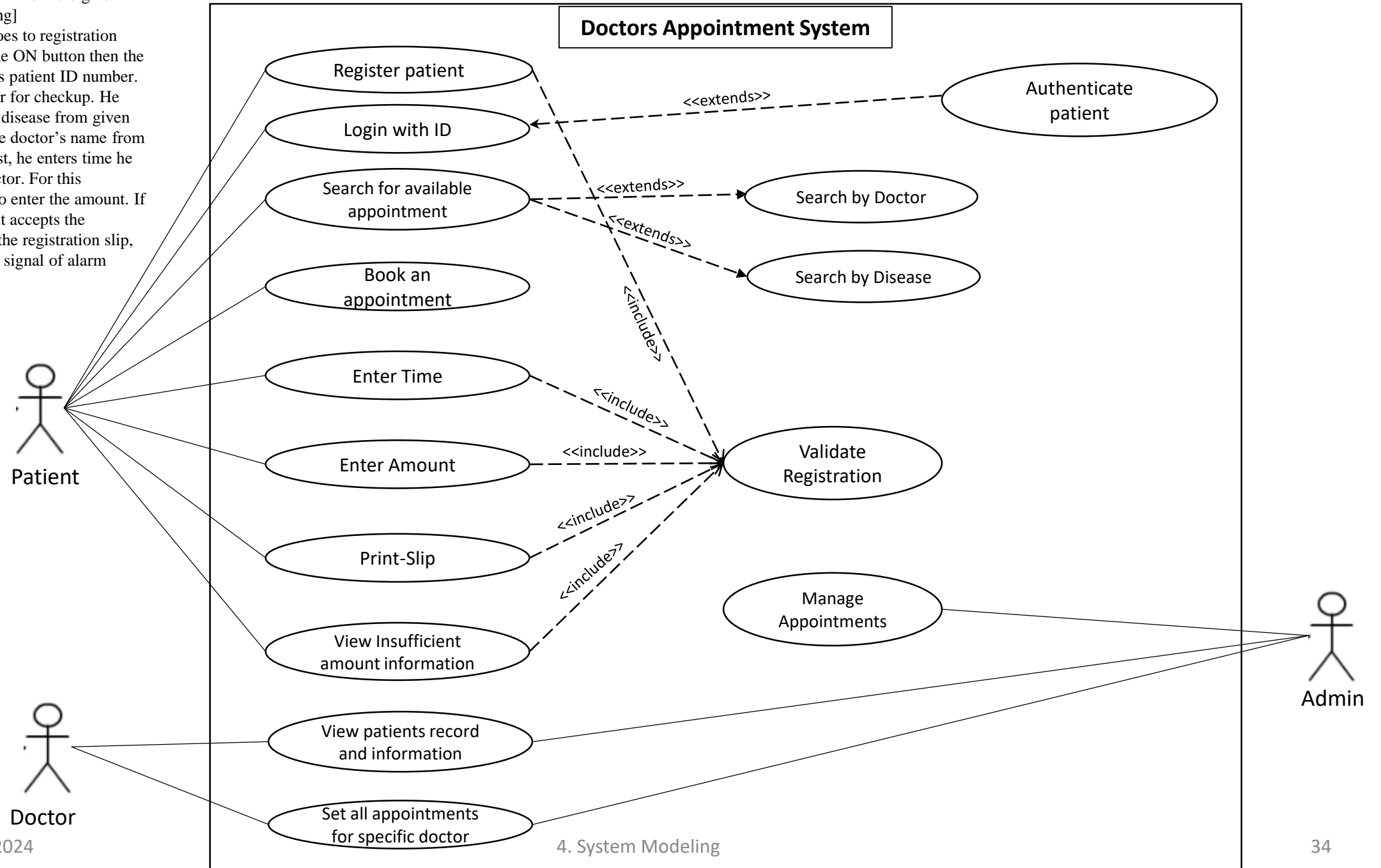
Assignment:

- Draw a use case diagram from the given case study.

In hospital, a patient goes to registration machine. He presses the ON button then the screen opens. He enters patient ID number. He books for the doctor for checkup. He checks the category of disease from given list, then he chooses the doctor's name from given doctor's name list, he enters time he wants to meet with doctor. For this registration, he needs to enter the amount. If the amount digit is ok it accepts the registration and prints the registration slip, otherwise it will give a signal of alarm "insufficient amount".

Draw a use case diagram from the given case study. [2019 Spring]

In hospital, a patient goes to registration machine. He presses the ON button then the screen opens. He enters patient ID number. He books for the doctor for checkup. He checks the category of disease from given list, then he chooses the doctor's name from given doctor's name list, he enters time he wants to meet with doctor. For this registration, he needs to enter the amount. If the amount digit is ok it accepts the registration and prints the registration slip, otherwise it will give a signal of alarm "insufficient amount".



ii) Sequence diagrams

- Sequence diagrams 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.
- A *sequence diagram* is an interaction diagram that **emphasizes the time ordering of messages.**
- It shows a set of objects and the messages sent and received by those objects.
- Graphically, a sequence diagram is a table that shows **objects arranged along the X axis** and **messages, ordered in increasing time, along the Y axis.**

Medical Receptionist

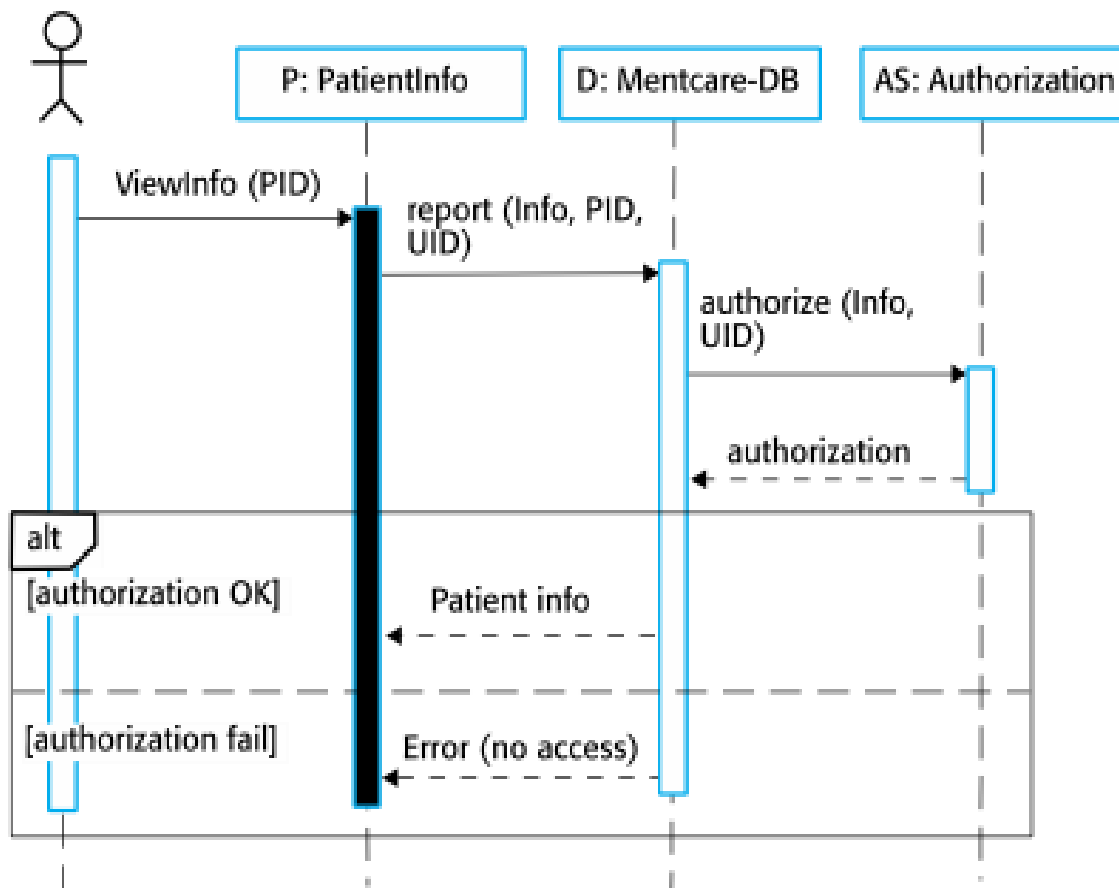
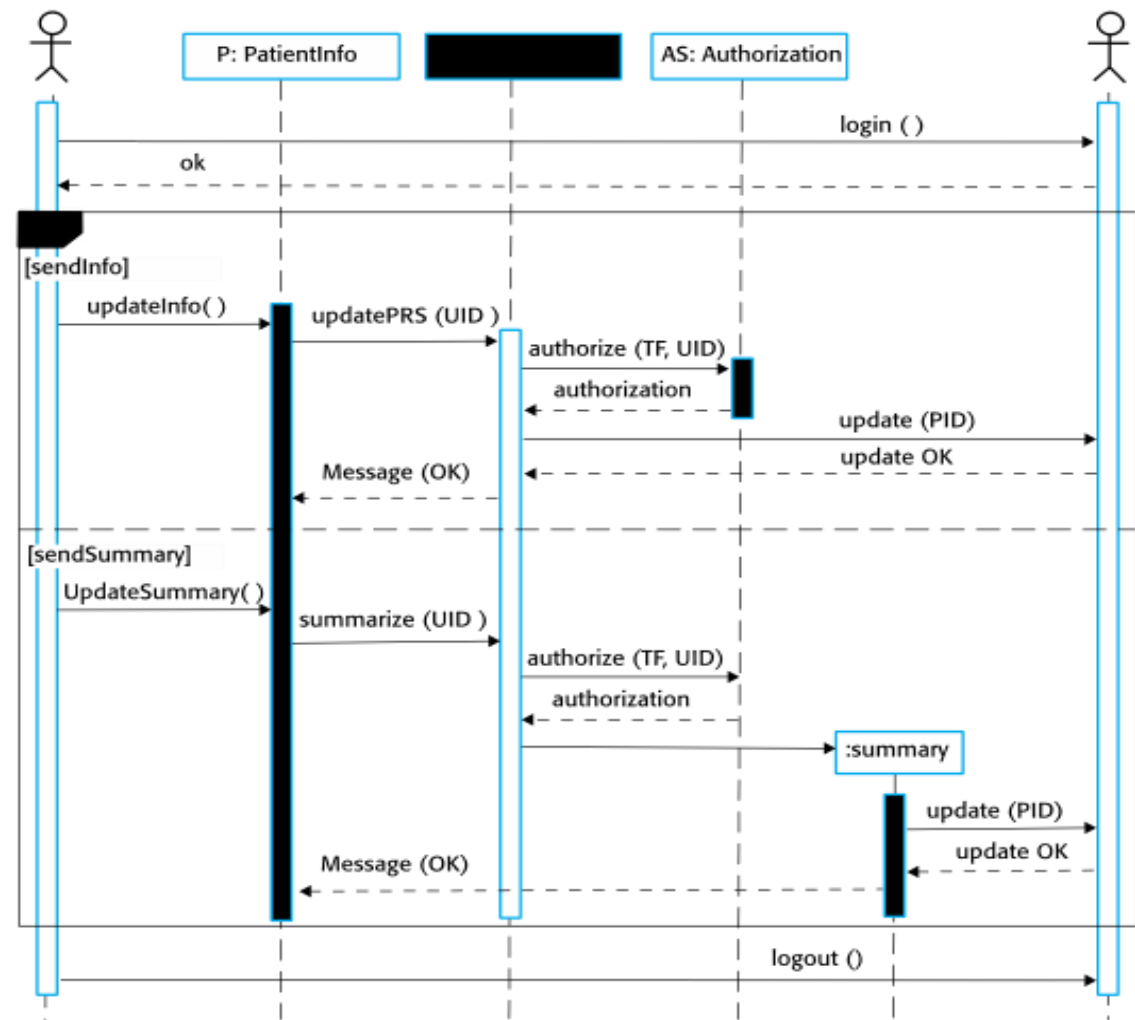


Figure: Sequence diagram for View patient information

Medical Receptionist

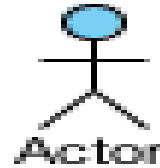


Sequence diagram for Transfer Data

Notations in Sequence Diagram :

1. Actor:

- that interact with the system.
- represent roles played by human users, external hardware, or other subjects.



2. Lifeline

- A lifeline represents an individual participant in the Interaction.

3. Activation

- A thin rectangle on a lifeline) represents the period during which an element is performing an operation.

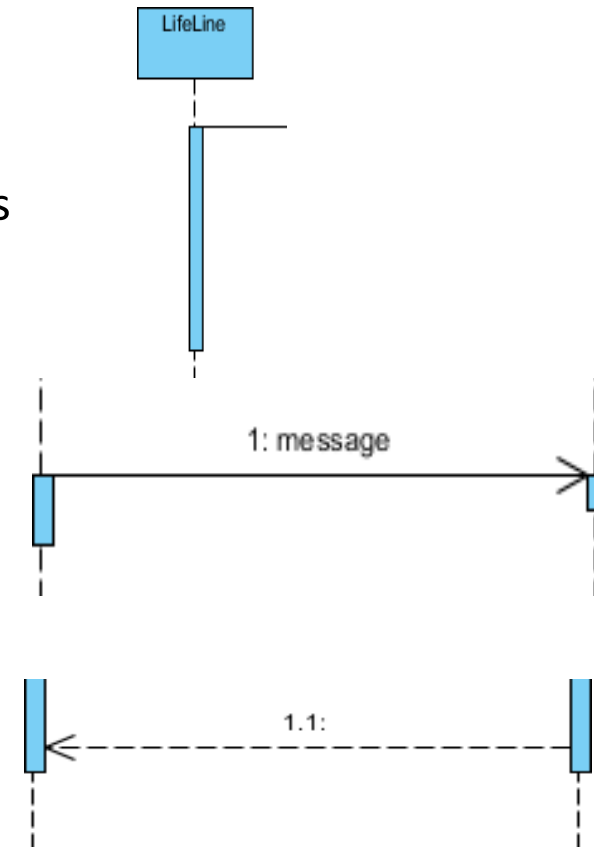
4. Message

- A message defines a particular communication between Lifelines of an Interaction.
- Call message is a kind of message that represents an invocation of operation of target lifeline.

5. Return Message

Return message is a kind of message that represents the pass of information back to the caller of a corresponded former message

- Represented with dotted line.



6. Self Message

Self message is a kind of message that represents the invoking of message of the same lifeline..

7. Recursive Message

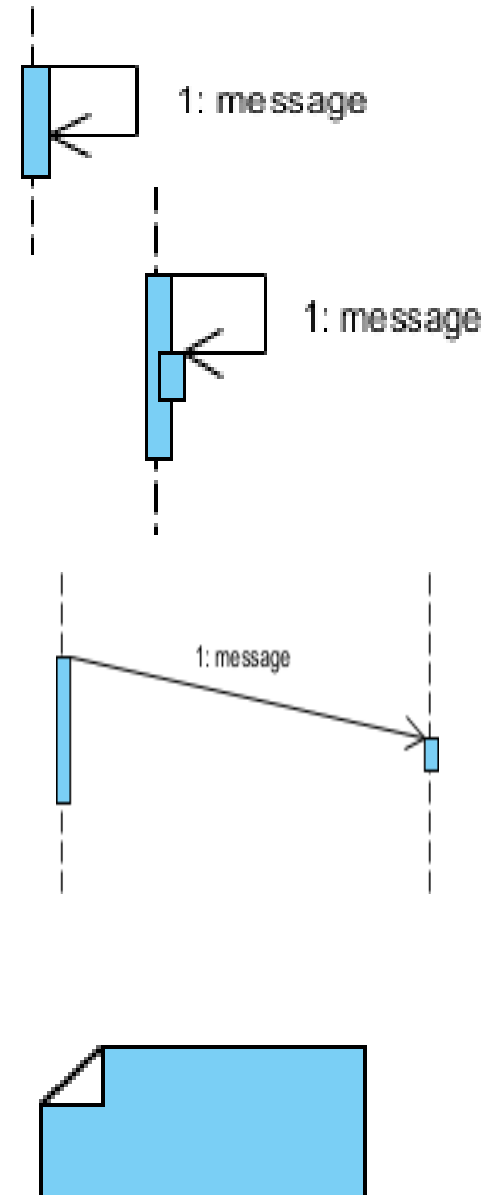
A self message sent for recursive purpose is called a recursive message. In other words, it can be said that the recursive message is a special case of the self message as it represents the recursive calls.

8. Duration Message

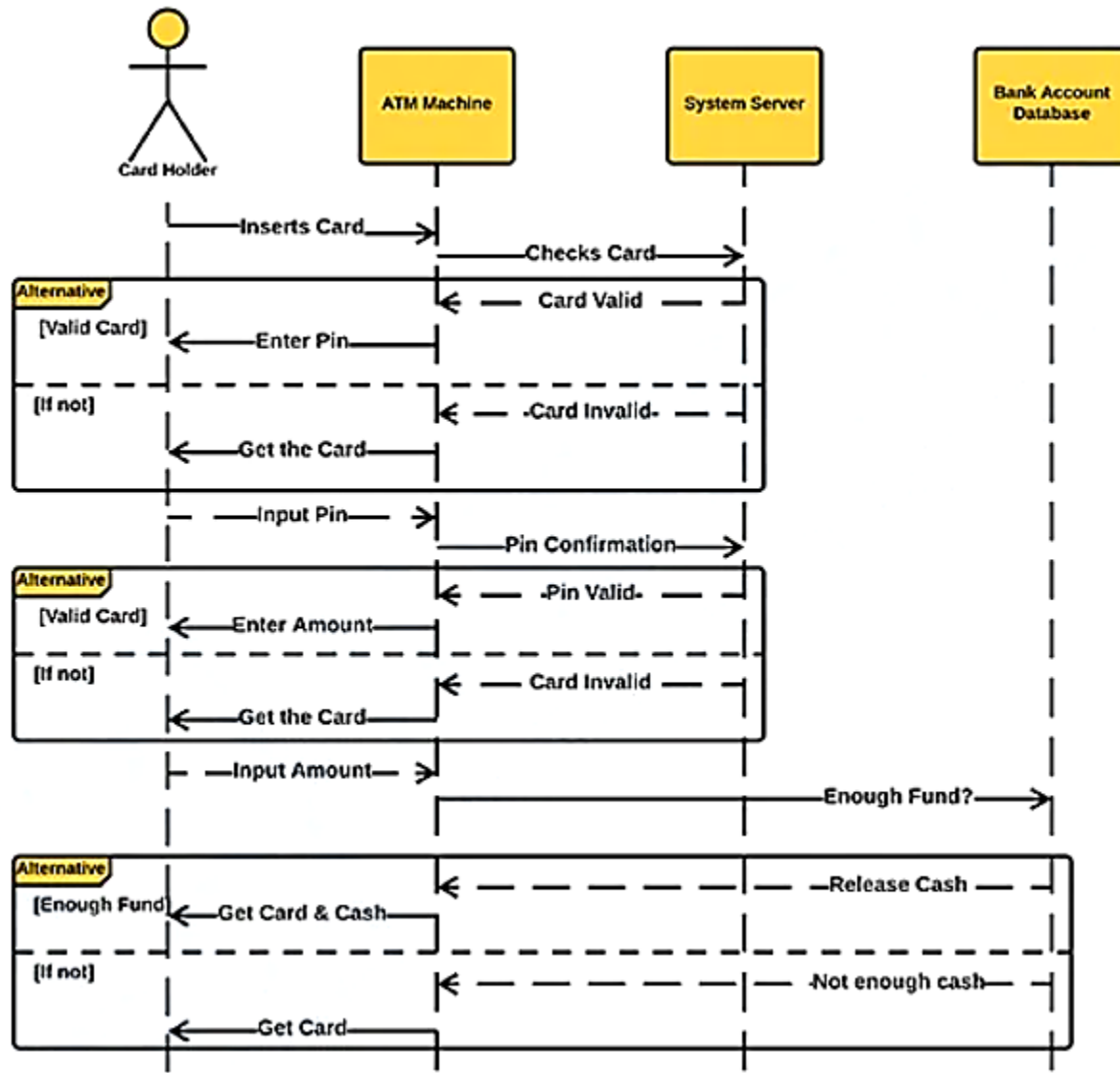
Duration message shows the distance between two time instants for a message invocation.

9. Note

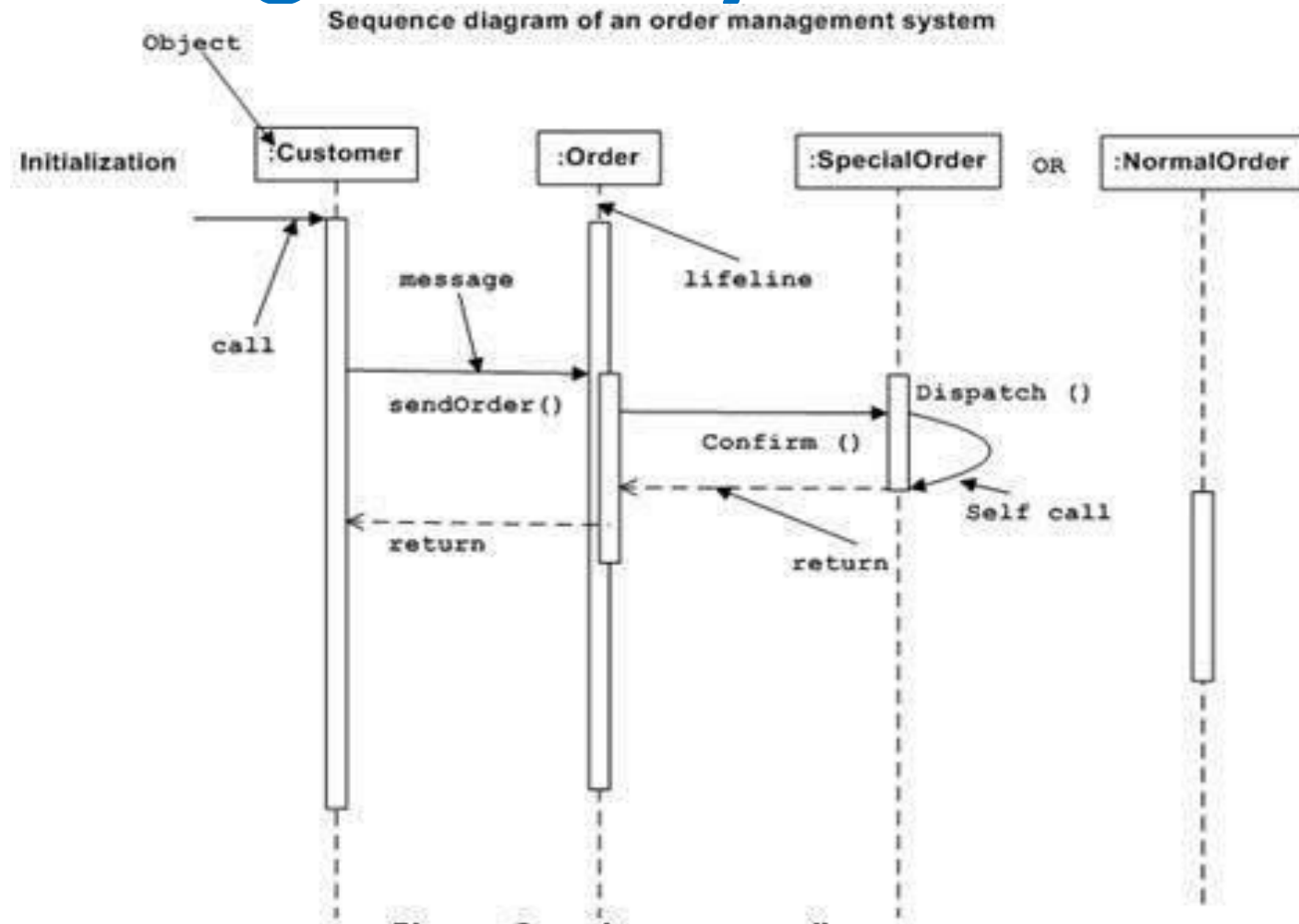
A note (comment) gives the ability to attach various remarks to elements. A comment carries no semantic force, but may contain information that is useful to a model.



ATM System Sequence Diagram



Order Management System : Sequence Diagram



4.3 Structural models

4.3 Structural models

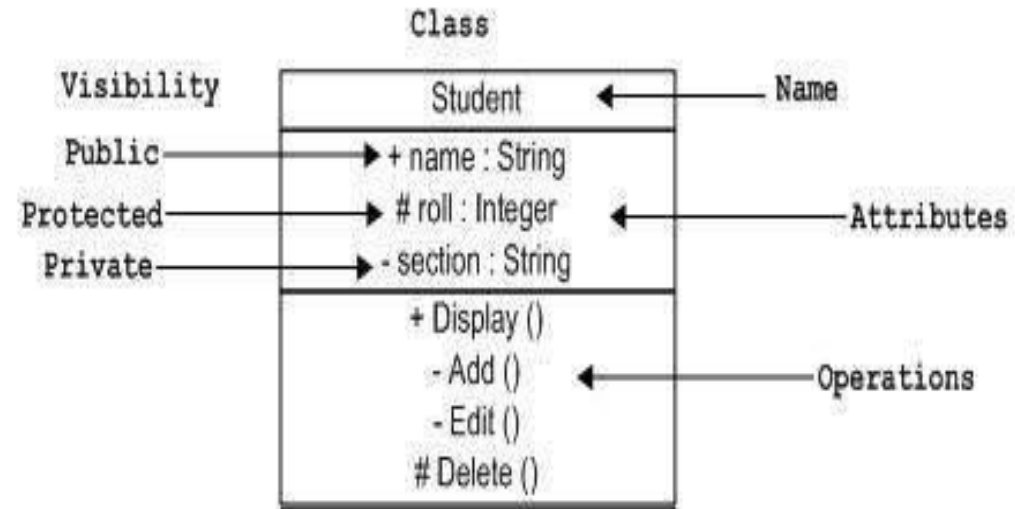
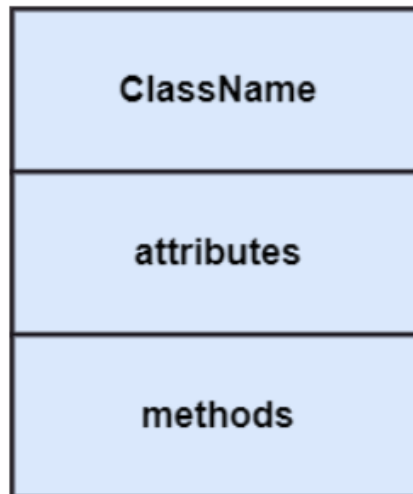
- Structural models of software **display the organization of a system in terms of the components** that make up that system and their relationships.
- Structural models may be:
 - **static models**, which show the structure of the system design, or
 - **dynamic models**, which show the organization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

i) 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.
- 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 describe the static structure of a system, or how it is structured rather than how it behaves.
- These diagrams contain the following elements:
 1. **Classes** , which represent entities with common characteristics or features. These features include attributes, operations, and associations.
 2. **Associations** , which represent relationships that relate two or more other classes where the relationships have common characteristics or features. These features include attributes and operations.

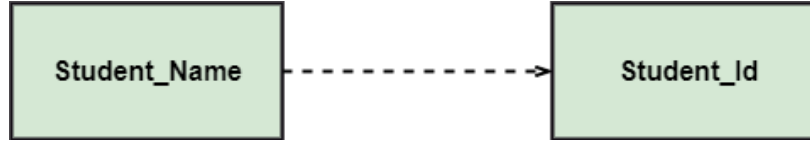
Class Diagram Components:

- **Upper Section:** The upper section encompasses the name of the class.
- **Middle section:** Middle section constitutes the attributes. The attributes are written along with its visibility factors, which are public (+), private (-), protected (#), and package (~).
- **Lower Section:** The lower section contain methods or operations. The methods are represented in the form of a list, where each method is writ



Relationships:

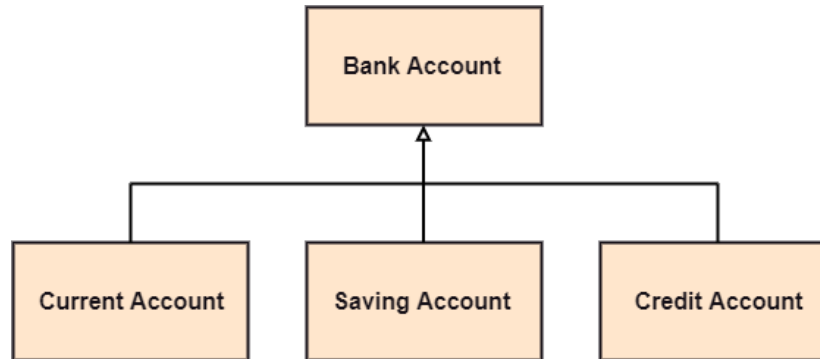
Dependency:



Association:



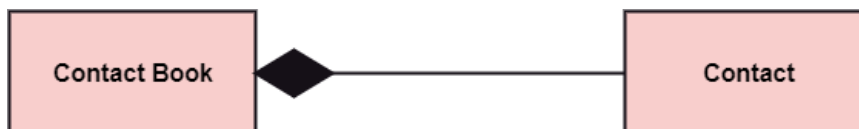
Generalization:



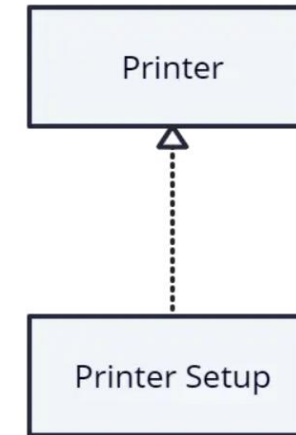
Aggregation:



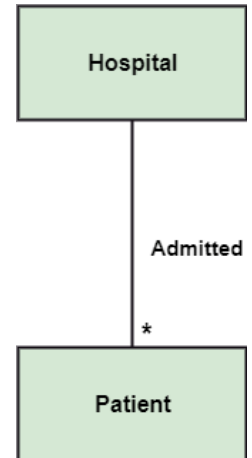
Composition:



Realization:



Multiplicity:



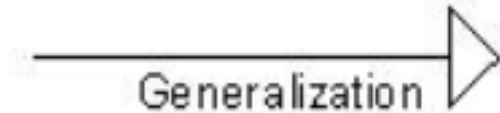
Relationships:

- **Dependency** : Eg: Student_Name is dependent on Student_Id
- **Association** : describes connection between two or more objects
- **Generalization** : Relationship between parent class and child class
- **Aggregation** : subset of association. Eg. Company encompasses a number of employees
- **Composition**: It represents a whole-part relationship.
- **Realization**: denotes the implementation of the functionality defined in one class by another class
- **Multiplicity**: Multiple patients are admitted to one hospital

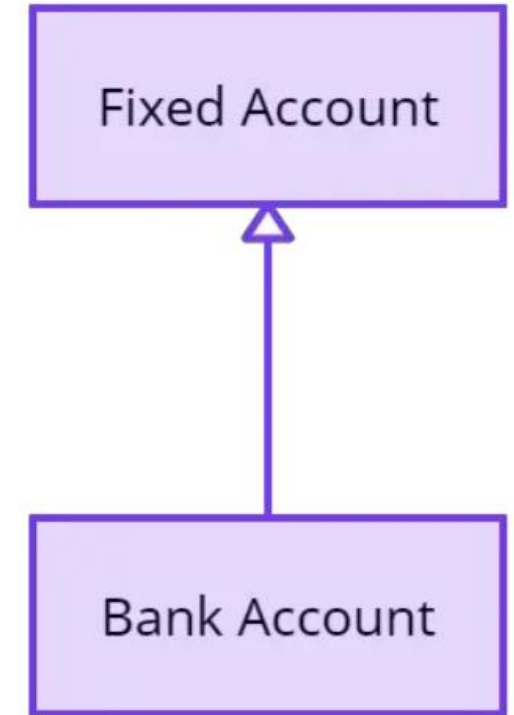
Reference: <https://www.javatpoint.com/uml-class-diagram>

Generalization

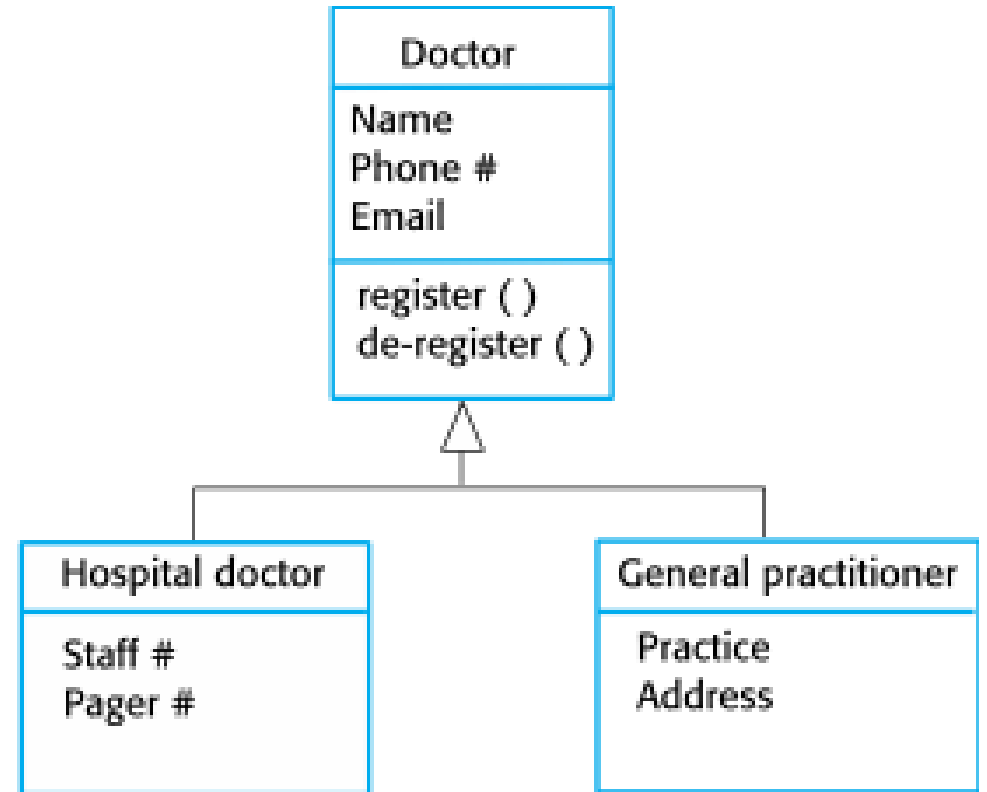
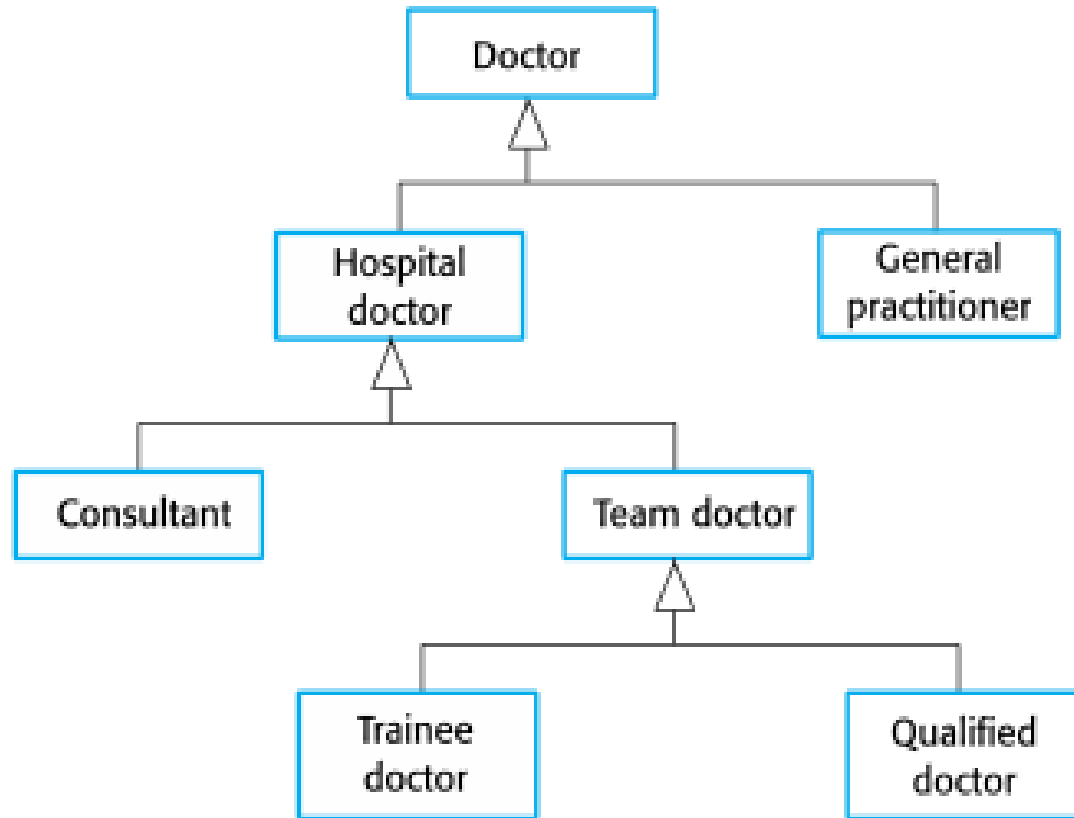
- Generalization is a **relationship between a general thing** (a **parent** or superclass) and a more specific kind of that thing (a **child** or subclass).
- The notation used for generalization is **a line segment with an empty block triangular arrowhead**.
- The **arrowhead points toward the generalized class** or use case or package.



- For example, an animal (generalization) and a cat (specialization) are related by the generalization-specialization relationship.



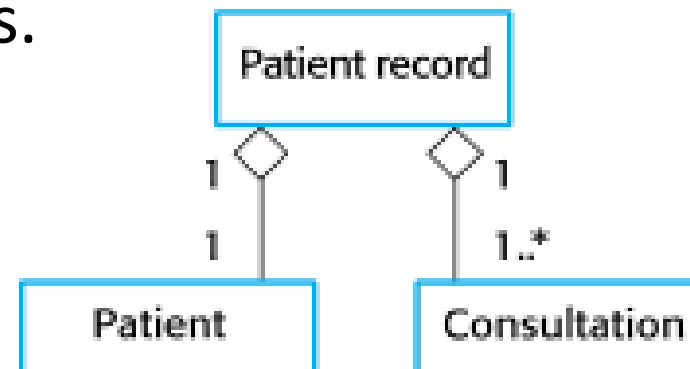
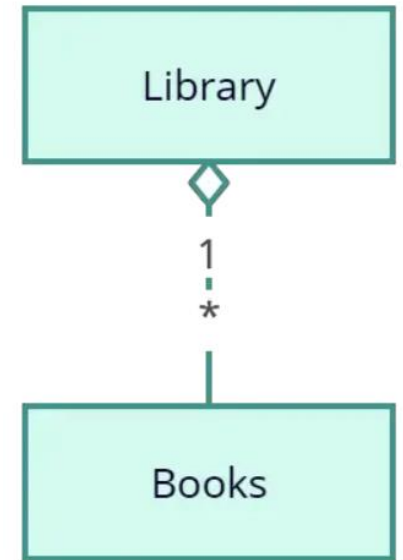
A generalization hierarchy



A generalization hierarchy with added detail

Aggregation

- An aggregation model shows how classes that are collections are composed of other classes.
- It refers to the formation of a particular class as a result of one class being aggregated or built as a collection.
- To show aggregation in a diagram, draw **a line from the parent class to the child class with a diamond shape near the parent class.**
- For example, the class “library” is made up of one or more books, among other materials.



Samples:

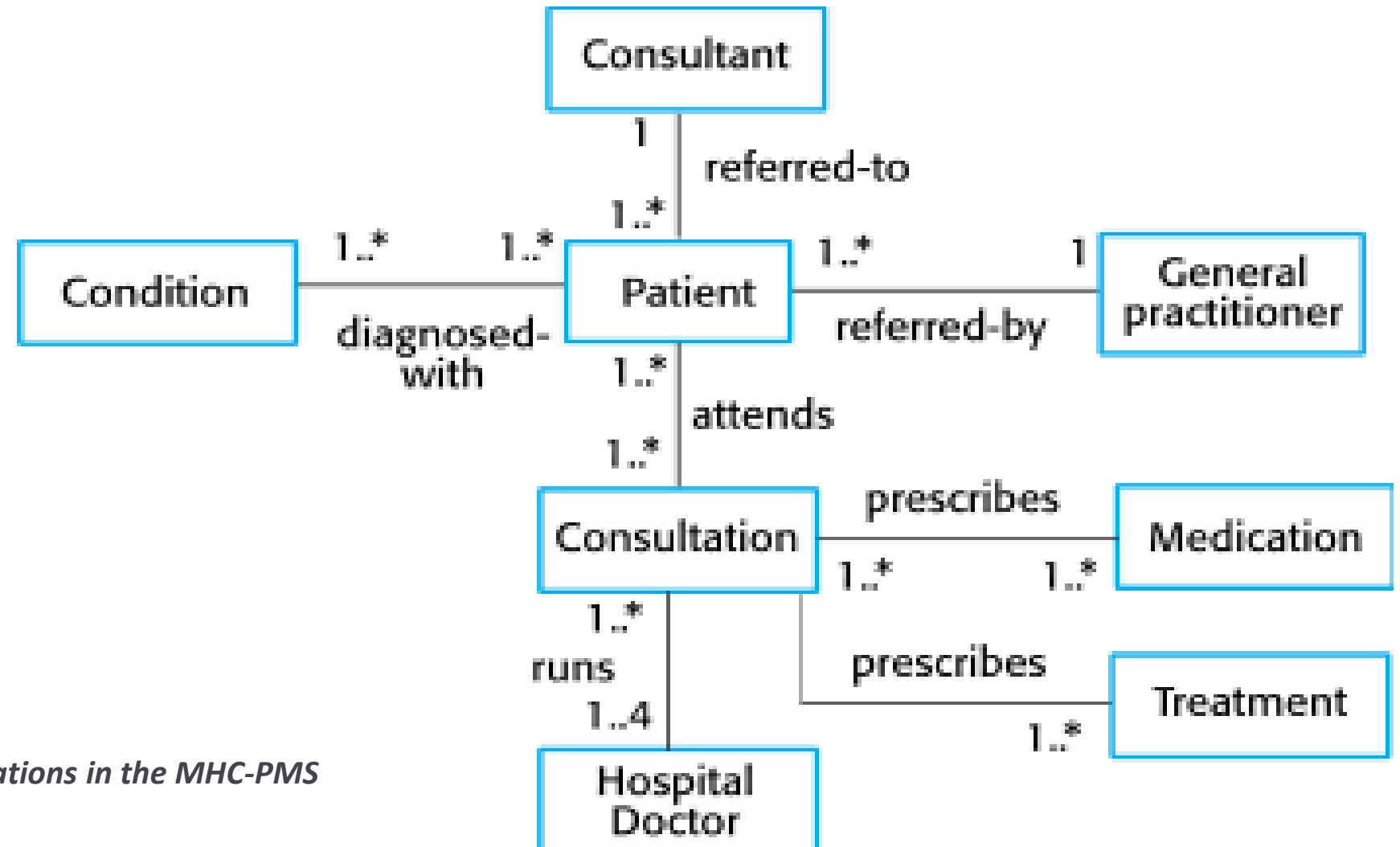
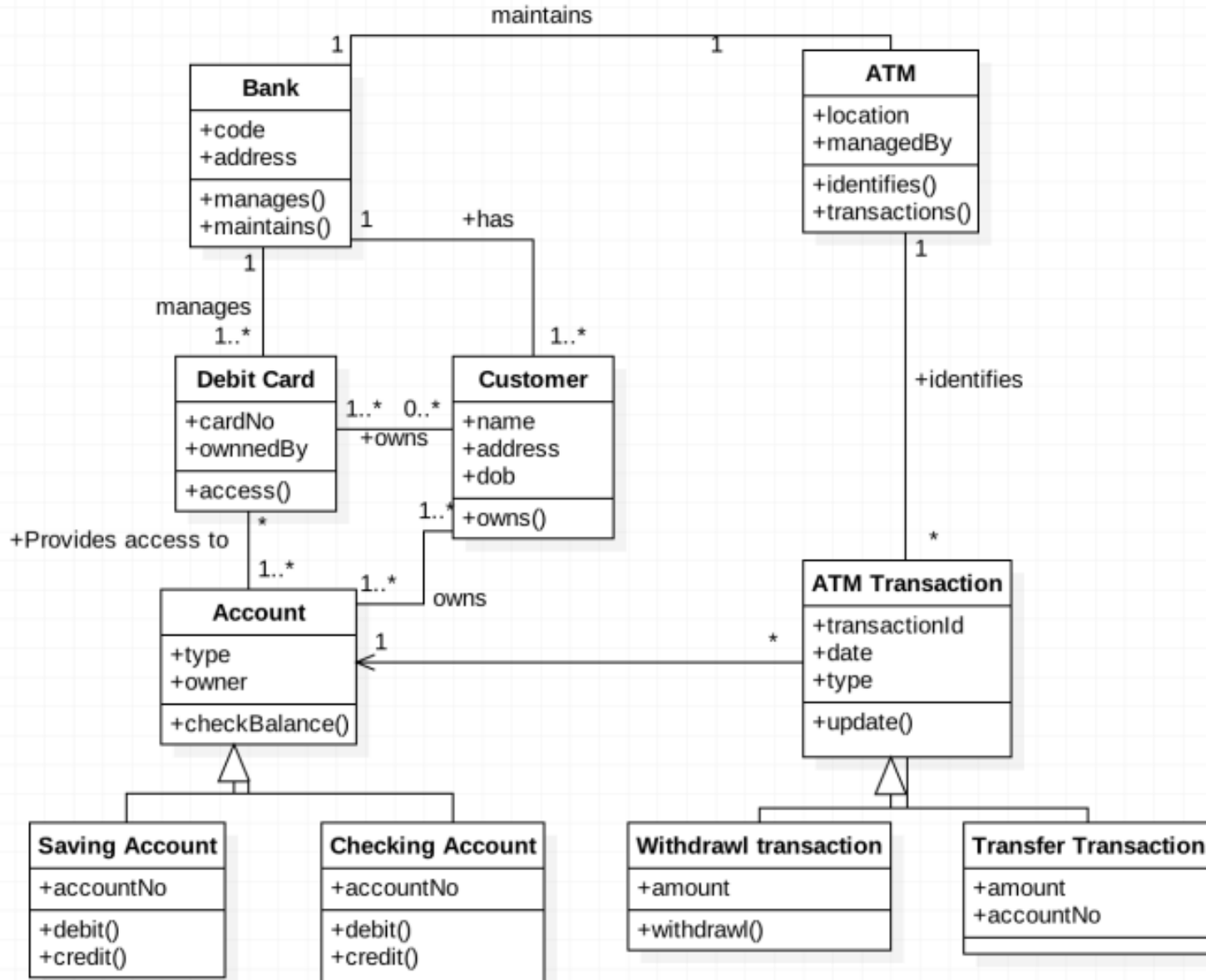


Figure: Classes and associations in the MHC-PMS

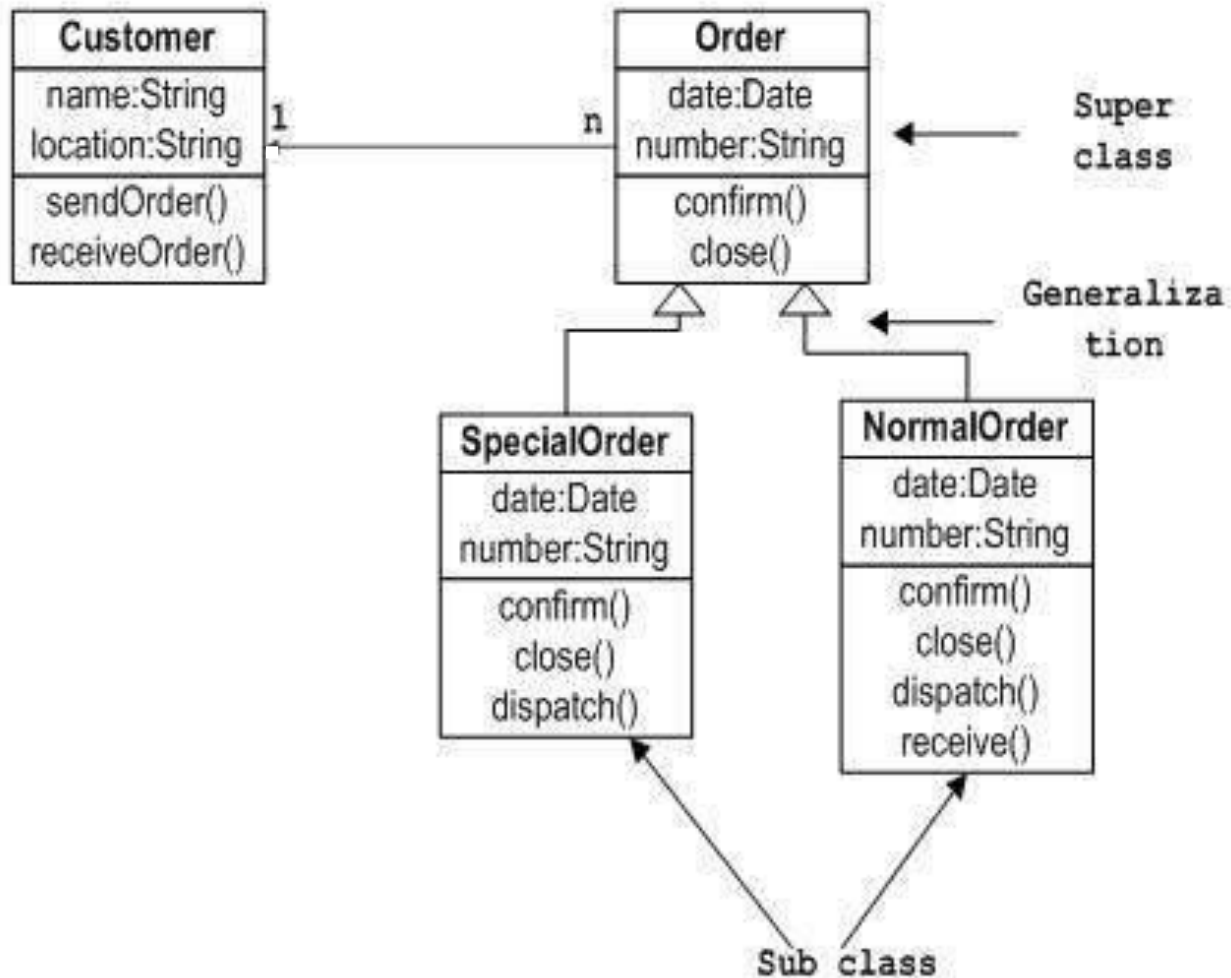


ATM system

Source:

<https://www.davuniversity.org/images/files/study-material/UML-ATM.pdf>

Sample Class Diagram



Order Management System

Source:

https://www.tutorialspoint.com/uml_class_diagram.htm

4.4 Behavioral models

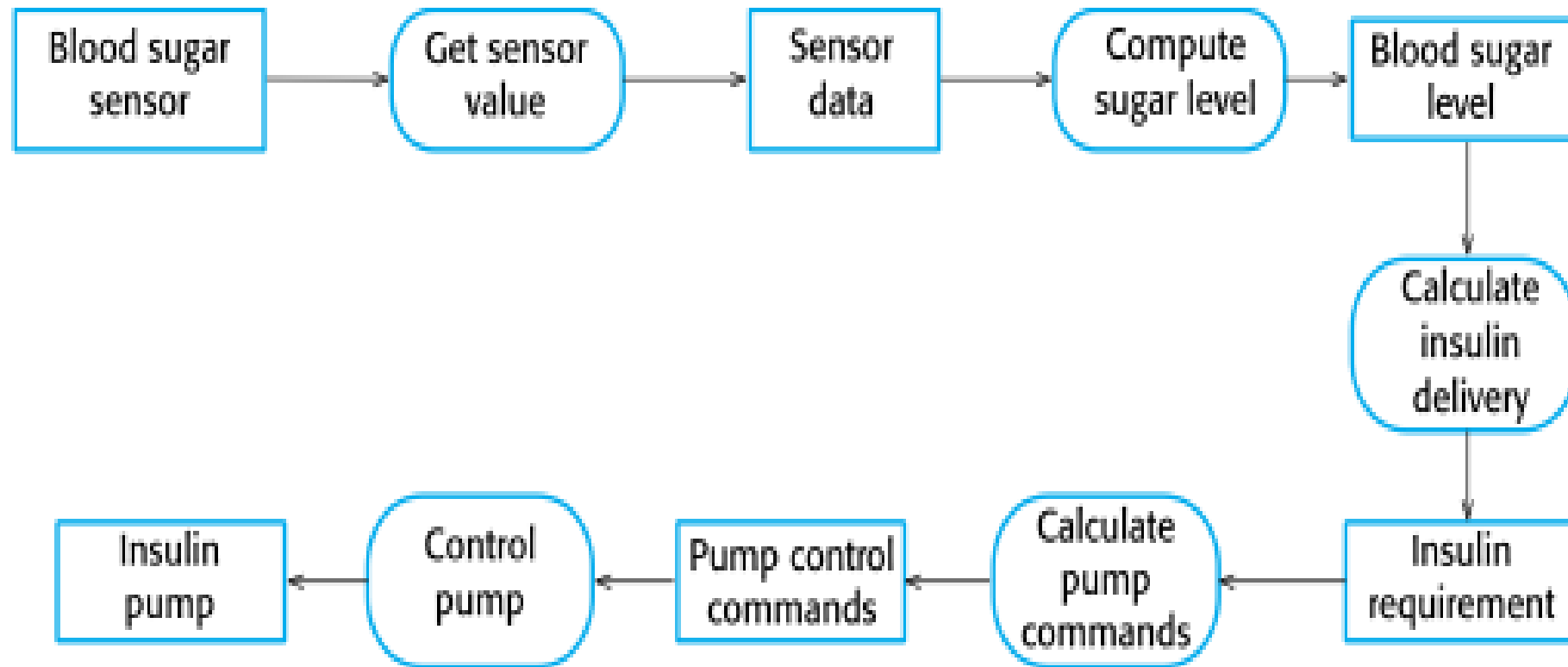
4.4 Behavioral models

- Behavioral model describe the overall behavior of the system.
- Behavioral models are models of the dynamic behavior of a system as it is executing.
- Behavioral modeling use: State Diagram and Activity diagram
- 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.

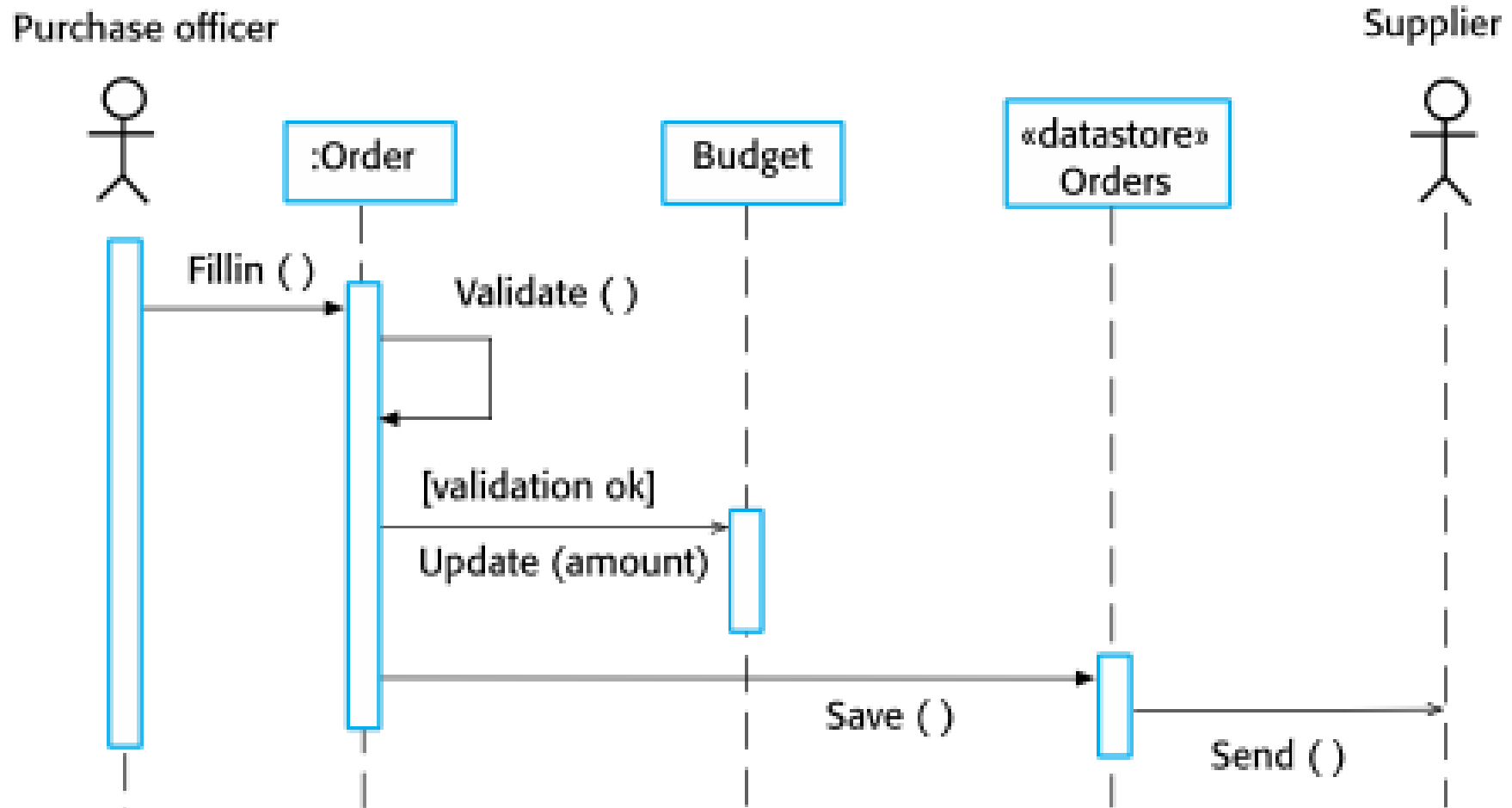
Data-driven modeling

- Data-driven models show the sequence of actions involved in processing input data and generating the associated output.
- This is very useful during the analysis stage since they show end-to-end processing in a system which means that they show the entire action sequence of how input data become output data.
- In other words, it shows the response of the system to particular input.
- In UML, activity and sequence diagrams can be used to describe such data flows.
- 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



Order processing



Event-driven modeling

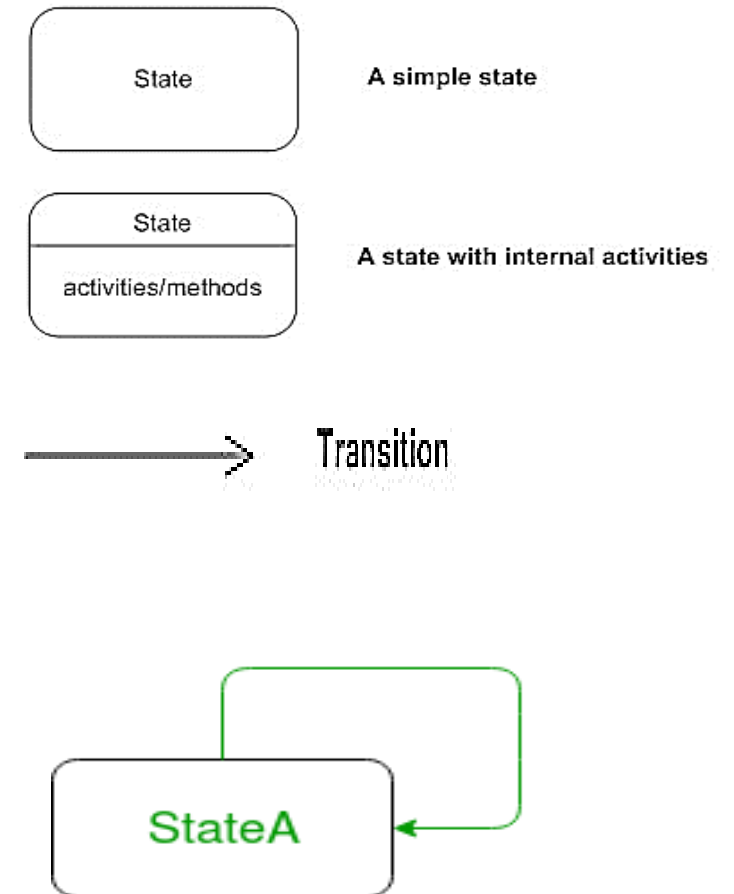
- Event-driven modeling shows how a system responds to external and internal events (stimuli).
- It is based on the assumption that a system has a finite number of states and that an event (stimulus) may cause a transition from one state to another.
- The UML supports event-based modeling using ***state machine diagrams***

a) State Diagrams:

- State transition diagrams provide a way to model the various states in which an object can exist.
- While the class diagram show a static picture of the classes and their relationships, state transition diagrams model the dynamic behavior of a system in response to external events (stimuli).
- State transition diagrams consist of the following:
 1. **States** , which show the possible situations in which an object can find itself
 2. **Transitions** , which show the different events which cause a change in the state of an object.

Notations used in State Chart Diagram

1. **States:** States, illustrated by using a rectangle with rounded corners, represent situations during the life of an object.
2. **Transition:** A solid arrow represents the path between different states of an object.
3. **Self transition:** There might be scenarios when the state of the object does not change upon the occurrence of an event. We use self transitions to represent such cases
4. **Initial State:** A filled circle followed by an arrow represents the object's initial state.
5. **Final State:** An arrow pointing to a filled circle nested inside another circle represents the object's final state.



Initial state



Final state

State Diagrams of ATM session

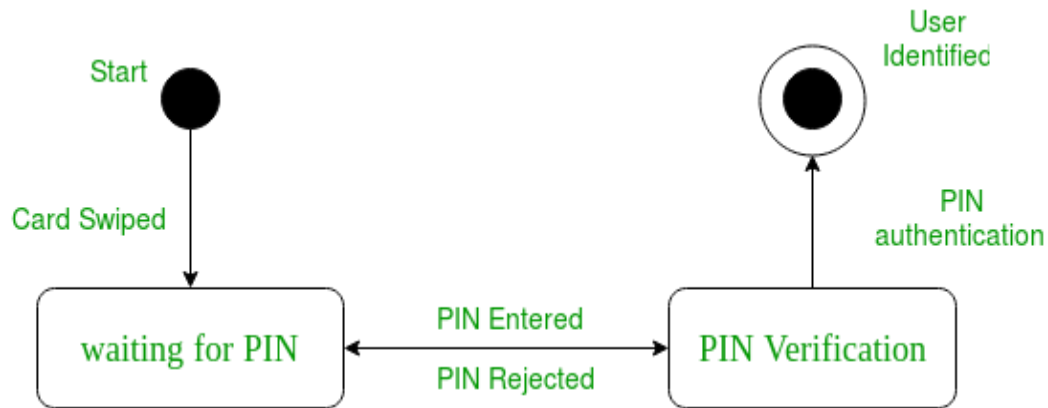
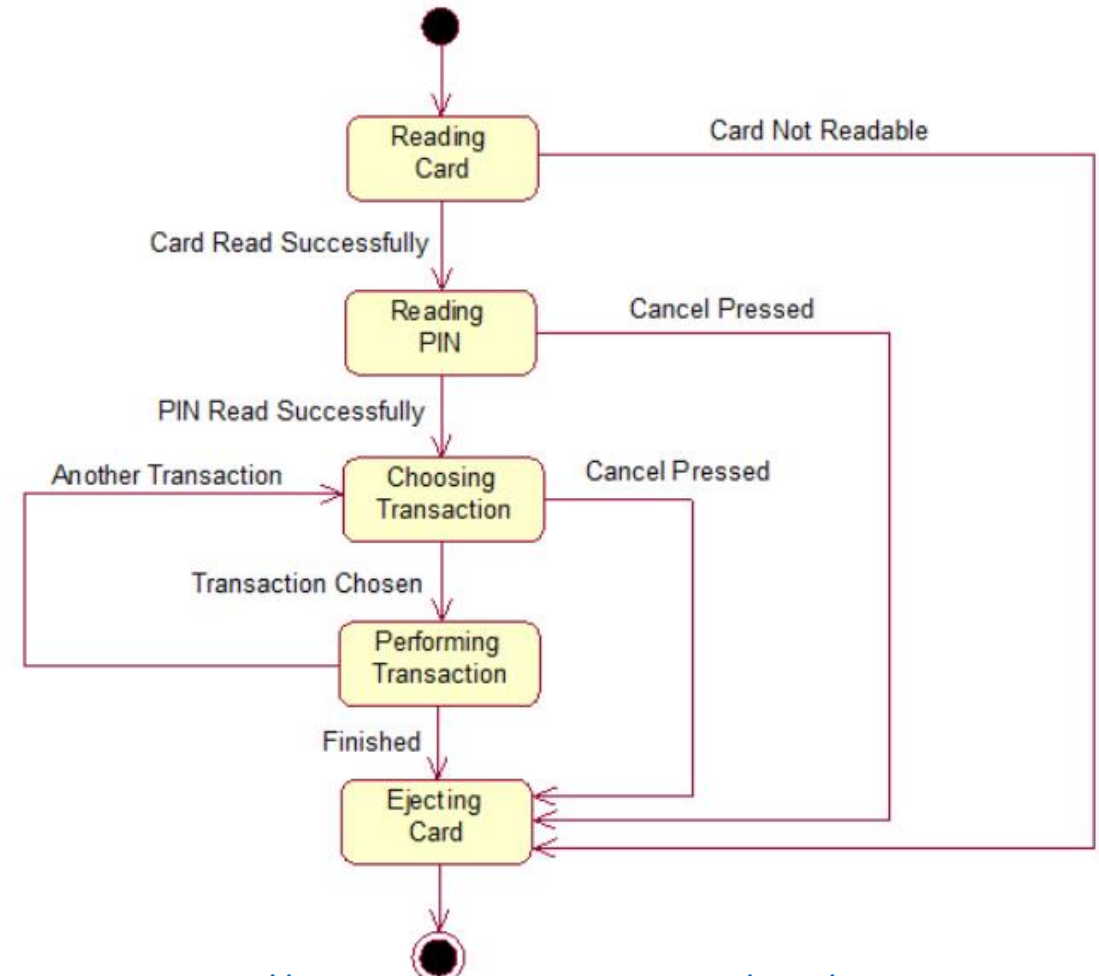


Figure – a state diagram for user verification

Source: <https://www.geeksforgeeks.org/unified-modeling-language-uml-state-diagrams/#:~:text=A%20state%20diagram%20is%20used,machines%20and%20State%2Dchart%2Diagrams.>



Source; https://www.startertutorials.com/uml/uml-diagrams-atm-application.html#Class_diagram

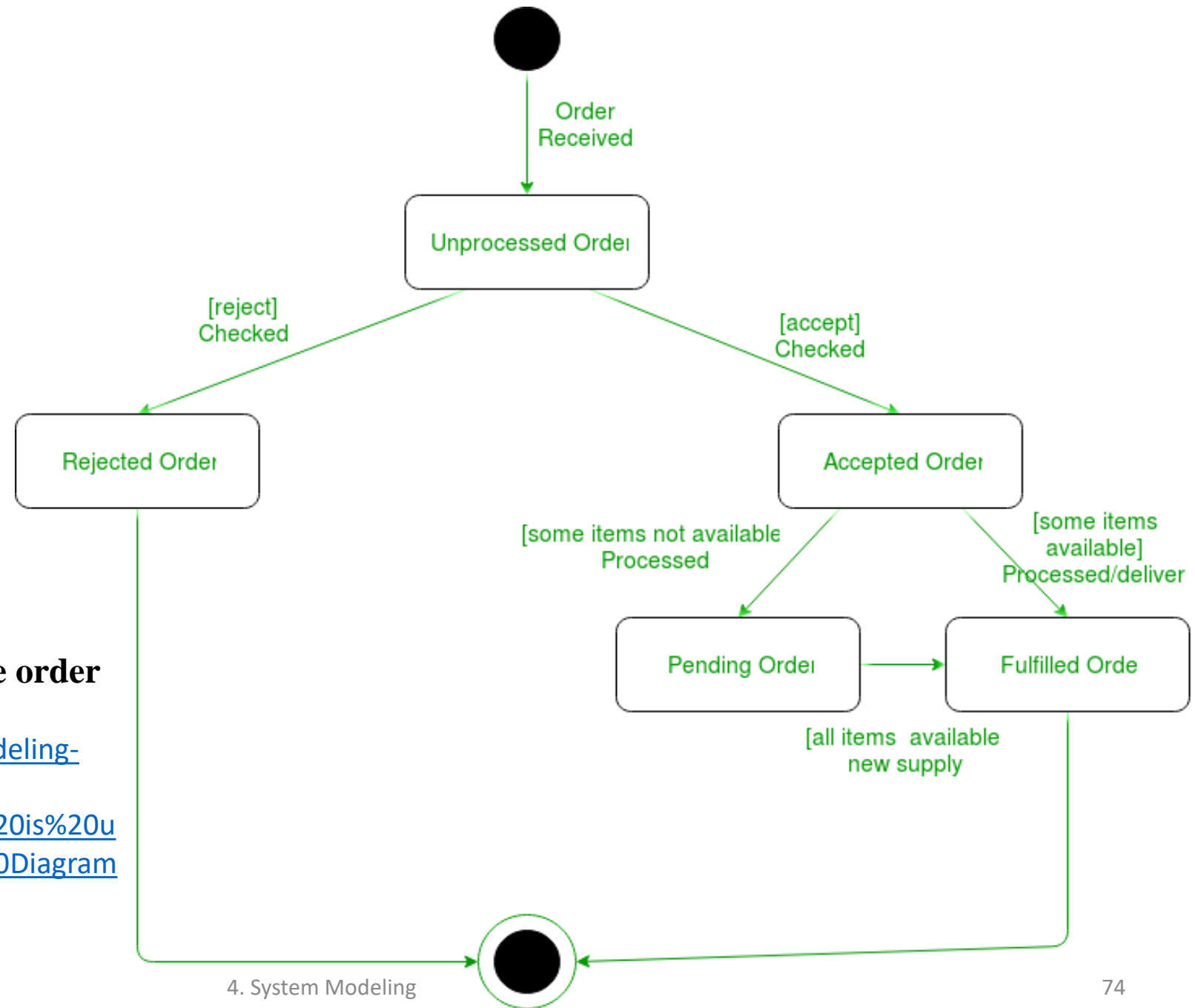


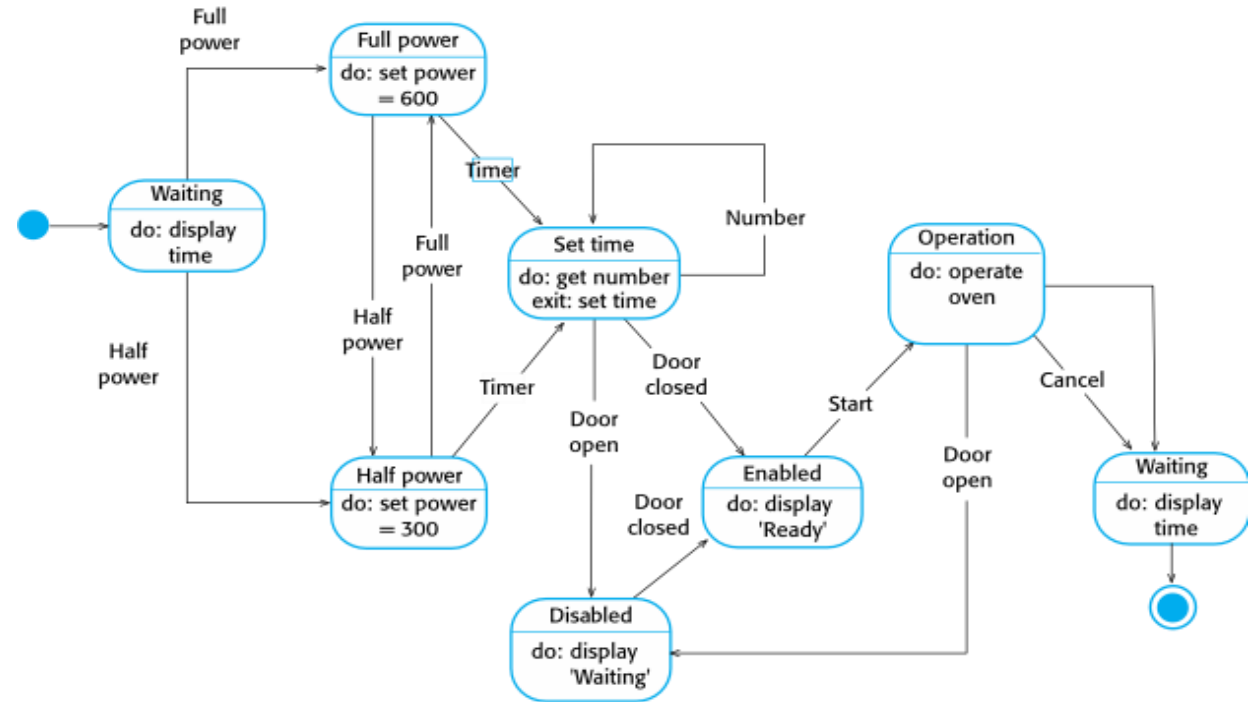
Figure: A State Diagram for an online order

Source:

<https://www.geeksforgeeks.org/unified-modeling-language-uml-state-diagrams/#:~:text=A%20state%20diagram%20is%20used,machines%20and%20State%2Dchart%20Diagram>

State machine models

- These model the behavior 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 diagram of a microwave oven

b) Activity Diagram

- Activity diagrams describe the activities of a class.
- They are similar to state transition diagrams and use similar conventions, but activity diagrams describe the behavior/states of a class in response to internal processing rather than external events.
- They contain the following elements:
 1. **Swimlanes** , which delegate specific actions to objects within an overall activity
 2. **Action States** , which represent uninterruptible actions of entities, or steps in the execution of an algorithm
 3. **Action Flows** , which represent relationships between the different action states on an entity
 4. **Object Flows** , which represent utilization of objects by action states, or influence of action states on objects.

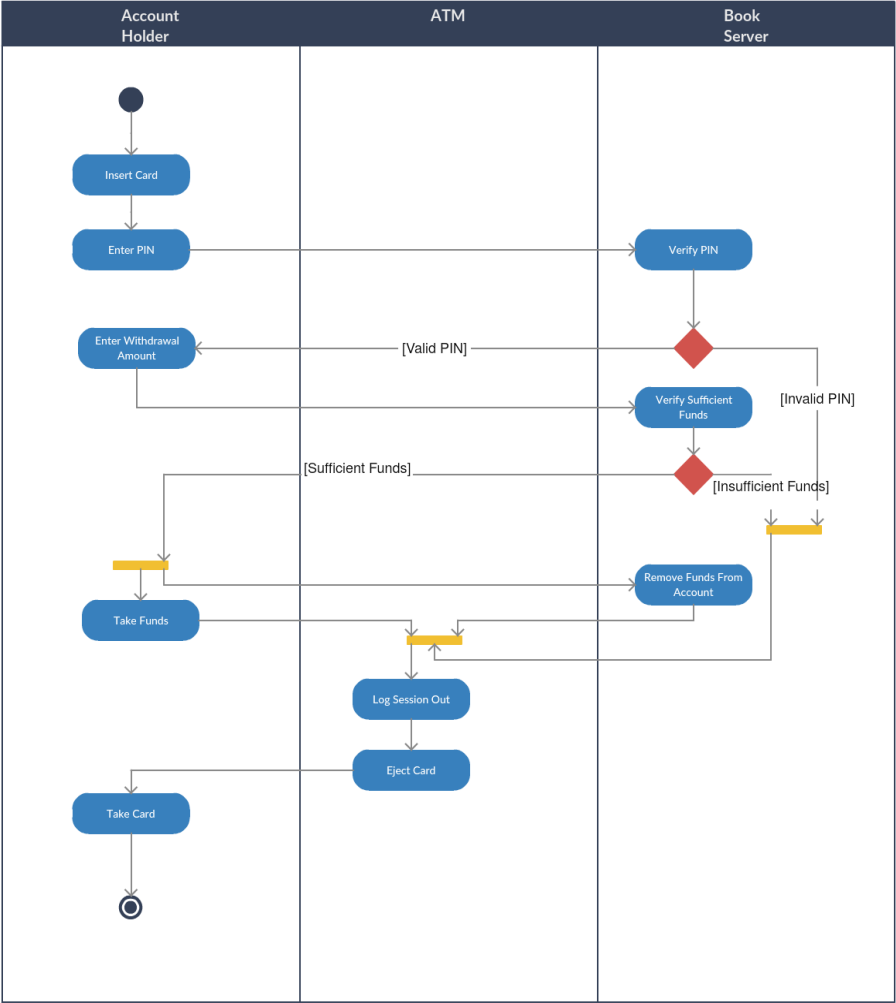
Symbols used in Activity diagram

Symbol	Name	Use
	Start/ Initial Node	Used to represent the starting point or the initial state of an activity
	Activity / Action State	Used to represent the activities of the process
	Control Flow / Edge	Used to represent the flow of control from one action to the other
	Activity Final Node	Used to mark the end of all control flows within the activity
	Decision Node	Used to represent a conditional branch point with one input and multiple outputs

	Merge Node	Used to represent the merging of flows. It has several inputs, but one output. 
	Fork	Used to represent a flow that may branch into two or more parallel flows
	Merge	Used to represent two inputs that merge into one output
	Note/ Comment	Used to add relevant comments to elements

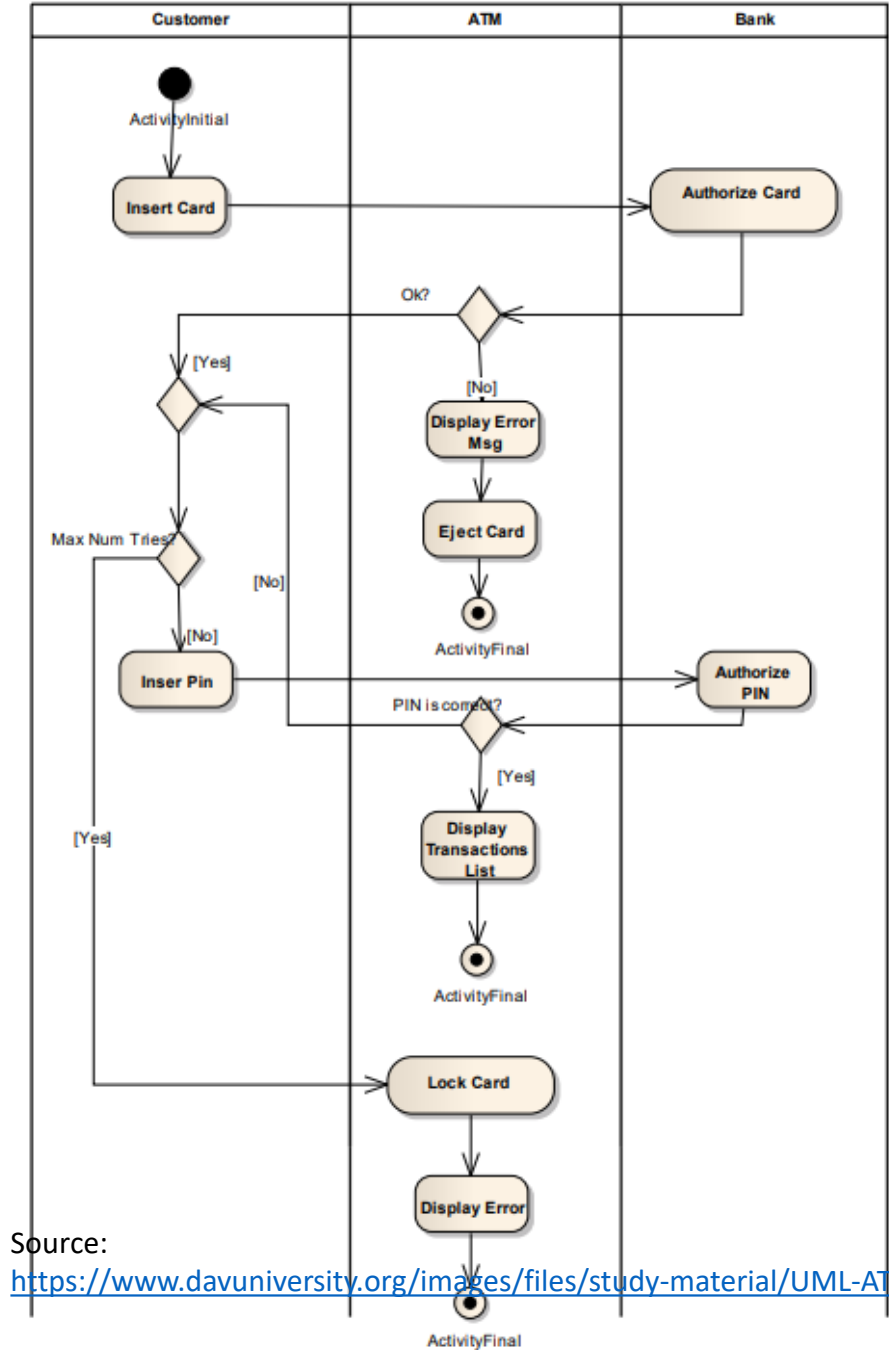
Samples:

ATM SYSTEM for ABC BANK

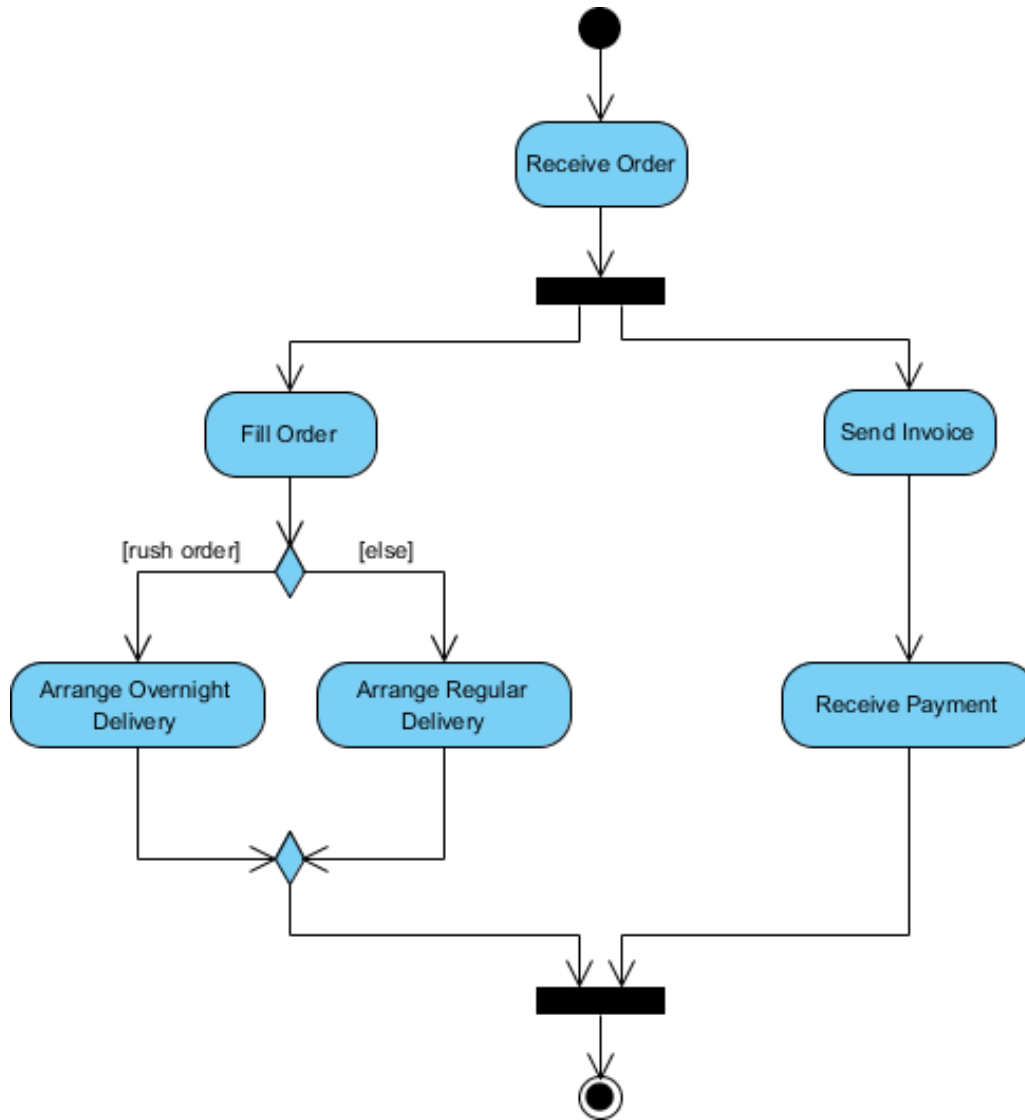


Source:
<https://creately.com/blog/diagrams/activity-diagram-tutorial/>

act Login Activity Diagram



Source:
<https://www.davuniversity.org/images/files/study-material/UML-ATM.pdf>



Order Management System

Source:

<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>

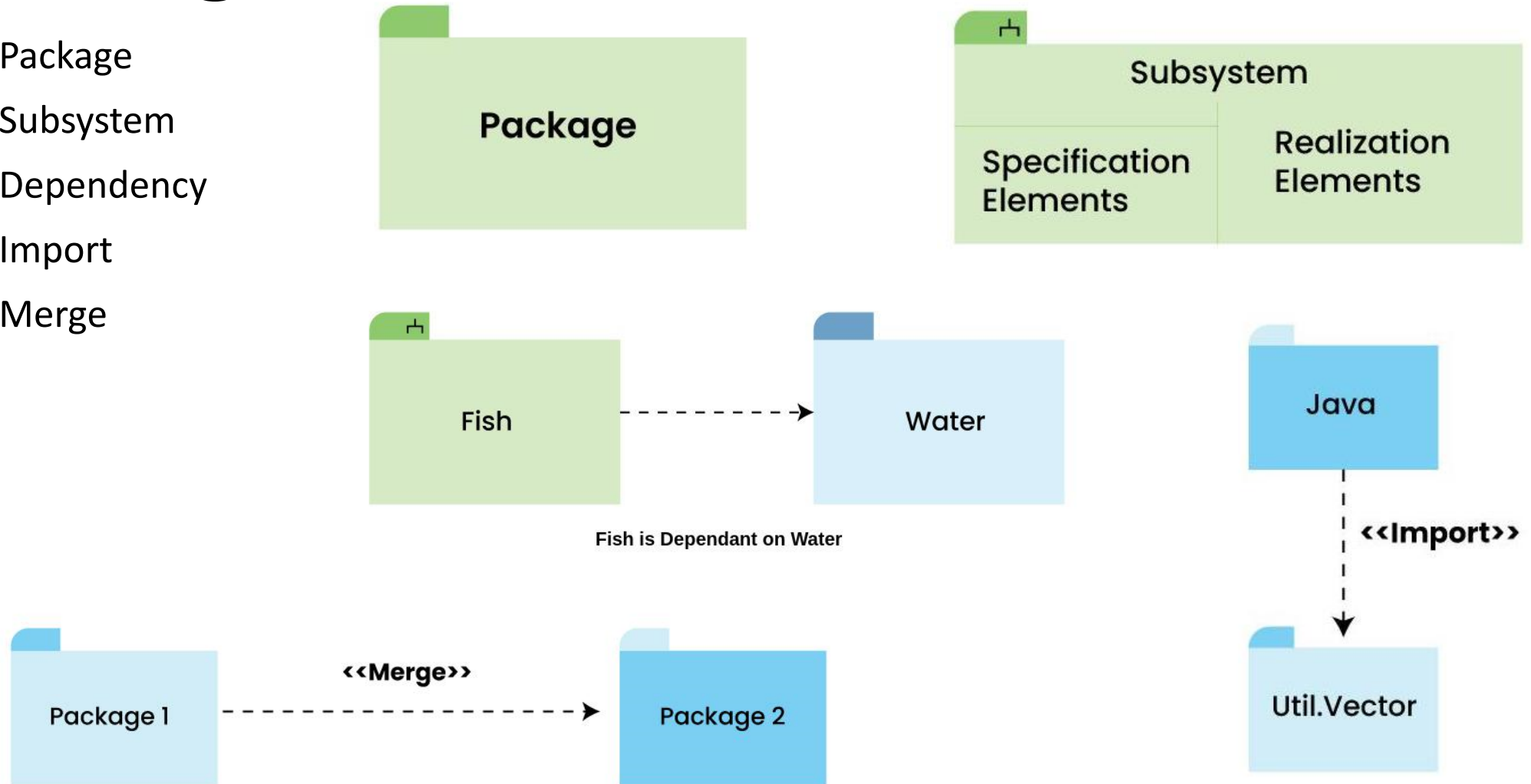
4.5 Package Structure

4.5 Package Structure

- In system modeling in software engineering, the package structure refers to the organization and arrangement of software components into logical and hierarchical packages.
- A package diagram is a type of Unified Modeling Language (UML) diagram mainly used to represent the organization and the structure of a system in the form of packages.
- *A package is used as a container to organize the elements present in the system into a more manageable unit. It is very useful to represent the system's architecture and design as a cohesive unit and a concise manner.*

Package Structure and notation

- Package
- Subsystem
- Dependency
- Import
- Merge



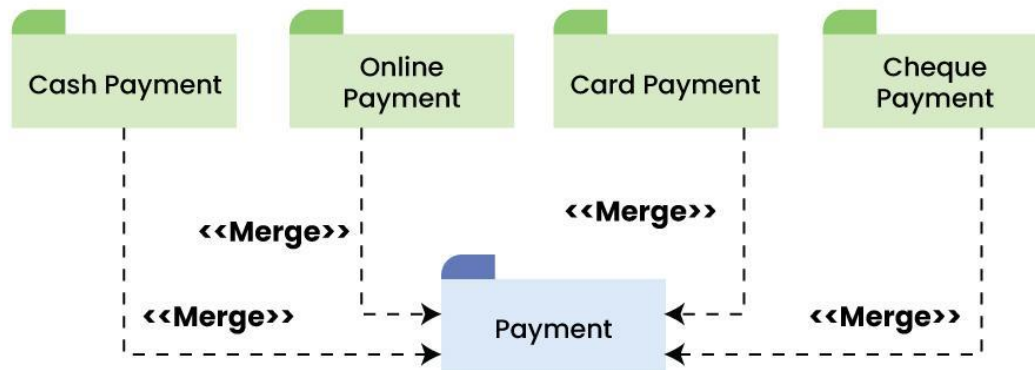
Source: <https://www.geeksforgeeks.org/package-diagram-introduction-elements-use-cases-and-benefits/>

Package Relationships

1. Package Merge Relationship

- used to represent that the contents of a package can be merged with the contents of another package.
- This implies that the source and the target package has some elements common in them, so that they can be merged together.

Package Merge Relationship



Package Diagram



2. Package Dependency Relationship

- A package can be dependent on other different packages, signifying that the source package is somehow dependent on the target package.
- The above diagram depicts that the online payment package is dependent on the Internet package and uses “need” dependency.

Package Dependency Relationship



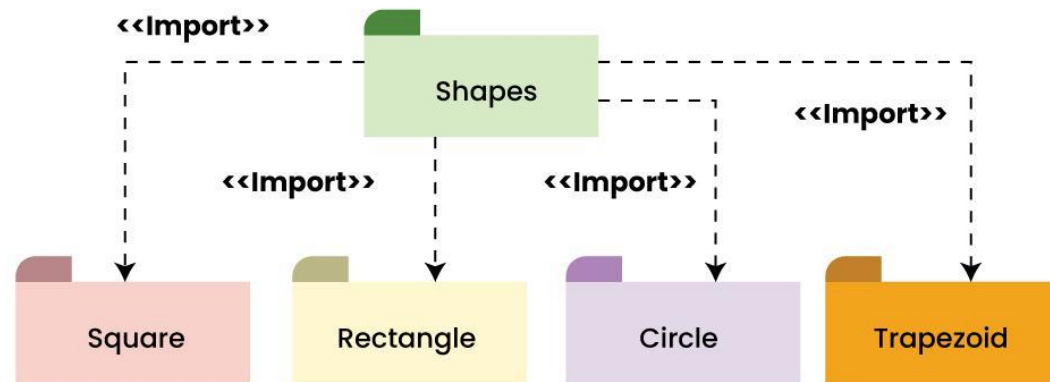
Package Diagram



3. Package Import Relationship

- This relationship is used to represent that a package is importing another package to use.
- It signifies that the importing package can access the public contents of the imported package.

Package Import Relationship



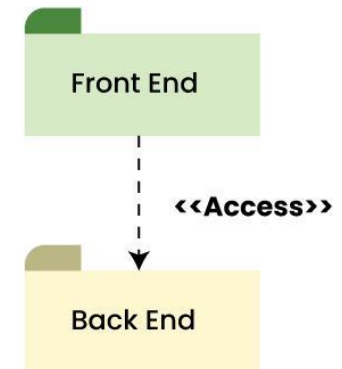
Package Diagram



4. Package Access Relationship

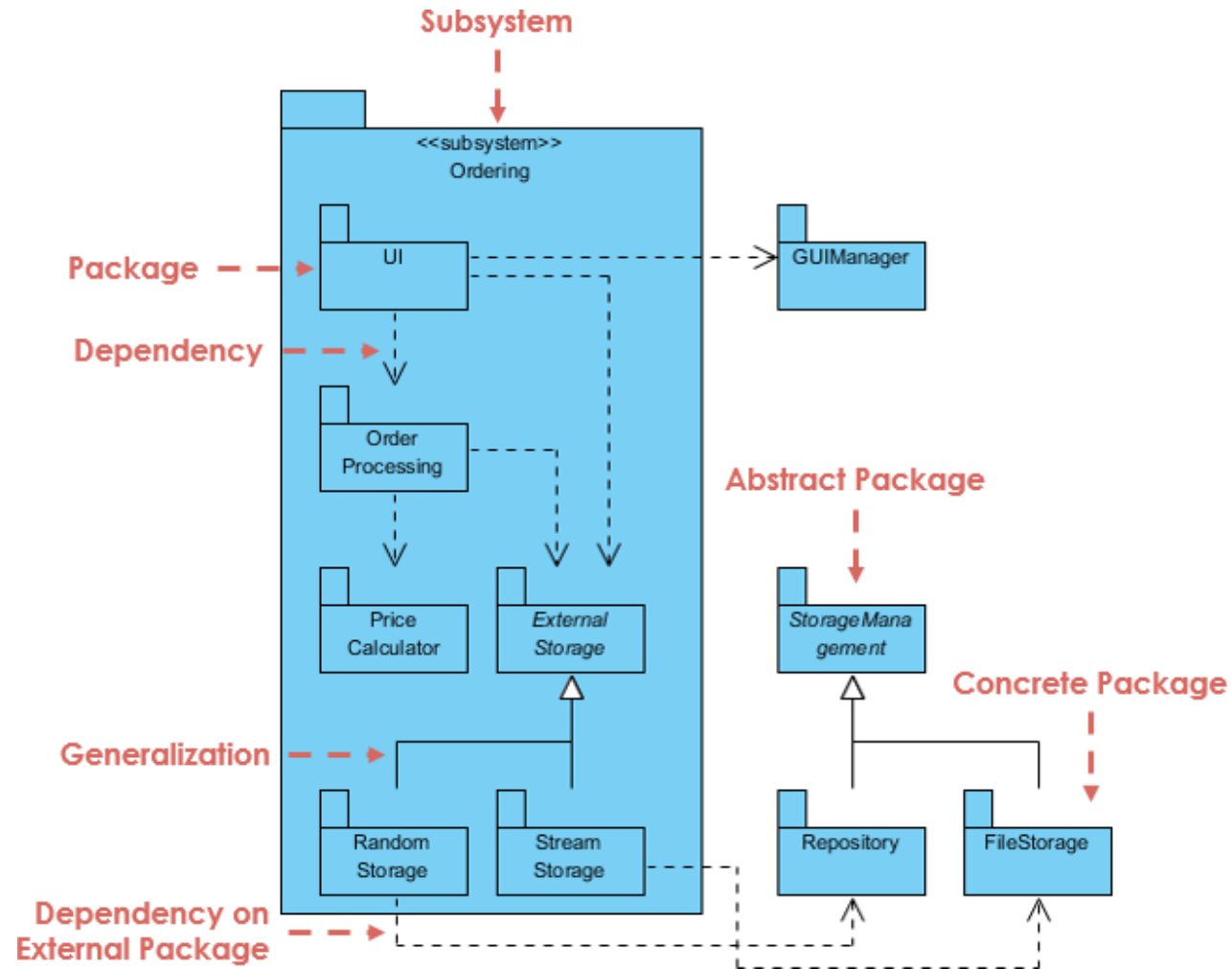
- This type of relationship signifies that there is an access relationship between two or more packages, meaning that one package can access the contents of another package without importing it.

Package Access Relationship



Package Diagram

Package diagram: Order Subsystem



Source: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-package-diagram/>

Keypoints

- 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.
- 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.

References for Complete UML diagrams:

- ATM Machine:
 - [https://www.startertutorials.com/uml/uml-diagrams-atm-application.html#Class diagram](https://www.startertutorials.com/uml/uml-diagrams-atm-application.html#Class_diagram)
 - <https://www.davuniversity.org/images/files/study-material/UML-ATM.pdf>
- Library Management System:
 - <https://www.startertutorials.com/uml/uml-diagrams-library-management-system.html>
- Online Banking System:
 - <https://www.startertutorials.com/uml/uml-diagrams-online-banking-system.html>
- Railway reservation system:
 - <https://www.startertutorials.com/uml/uml-diagrams-online-banking-system.html>

Brief Answer Questions:

1. Define system modeling.
2. What are the different perspectives with which we can view a system? List them.
3. Define UML. Why is it used in Software engineering?
4. What are structural diagrams? Give two examples.
5. What are behavioral diagrams? Give two examples.
6. Define Context diagram. What does it show?
7. What sorts of diagram can be used in Interaction modeling a system?
8. Mention the symbols used in use-case diagram.
9. When do we use extends and includes notations in use-case diagrams?
10. Mention the purposes of usecase diagram.
11. How are access specifiers shown for the attributes and methods in a class diagram?
12. Define generalization. Show it in a diagram.
13. How aggregation and composition are shown in class diagram.
14. Write the main differences between State diagram and activity diagram.
15. What is a package diagram?

Short Answer Questions:

1. Why system modeling is important in software engineering? What does it show? Explain.
2. What are structural and behavioral diagrams? Explain.
3. Gandaki University is about to develop a Library Management System. Draw the use case diagram to gather requirements.
4. Draw the context diagram for the above Library Management System.
5. Draw the sequence diagram for the above system.
6. Draw the class diagram for above system.
7. Draw the state diagram for above system.
8. Draw activity diagram for above system.
9. What is a package diagram? Explain the different package relationships.
10. Draw sequence diagram OR use case diagram that explains the deposit/withdrawal to/from Automated Teller Machine maintained by Financial Institution. You can make necessary assumptions.

11. In library system any person can apply for membership. Once the person gets membership, books can be issued. To issue book, member must provide membership card. Library system verifies validity of the card and if the card is valid, book is issued. While returning books, library system checks due date of the books and charges fines for overdue books. Members can borrow, search and reserve books by providing card number to the system. Draw Use Case diagram for above scenario.
12. Draw a use case diagram from below case study:

A customer visits online shopping portal. A customer may buy item or just visit the page and logout. The customer can select a segment, then a category and brand to get different products in the desired brand. The customer can select product for purchasing. The process can be repeated for more items. Once the customer finishes selecting the product/s, the cart can be viewed. If the customer wants to edit the final cart it can be done here. For final payment, the customer has to login to portal. If the customer is visiting for the first time, he must register with the site, else the customer use login page to proceed. Final cart is submitted for payment and card details and address details are to be confirmed with customer. Customer is confirmed with the shipment id and delivery of goods within 15 days.

End of Chapter