Sequence diagrams are a type of UML (Unified Modeling Language) diagram that visually represent the interactions between objects or components in a system over time. They focus on the order and timing of messages or events exchanged between different system elements. The diagram captures how objects communicate with each other through a series of messages, providing a clear view of the sequence of operations or processes.

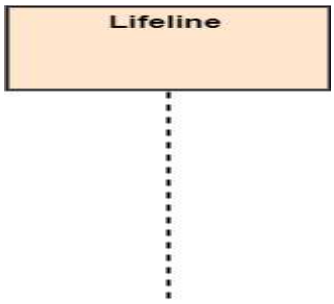
Notations of a Sequence Diagram

Object

It represents the existence of an object of a particular time.

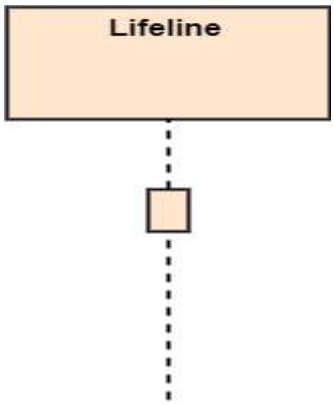
Lifeline

An individual participant in the sequence diagram is represented by a lifeline. It is positioned at the top of the diagram.



Activation

It is represented by a thin rectangle on the lifeline. It describes that time period in which an operation is performed by an element, such that the top and the bottom of the rectangle is associated with the initiation and the completion time, each respectively.



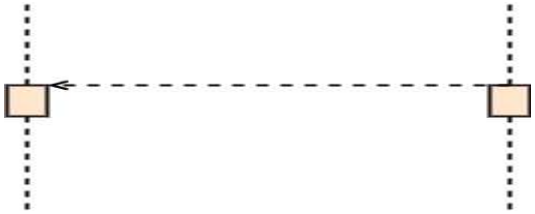
Messages

The messages depict the interaction between the objects and are represented by arrows. They are in the sequential order on the lifeline. The core of the sequence diagram is formed by messages and lifelines.

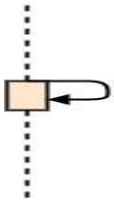
Following are types of messages enlisted below:

- **Call Message:** It defines a particular communication between the lifelines of an interaction, which represents that the target lifeline has invoked an operation.

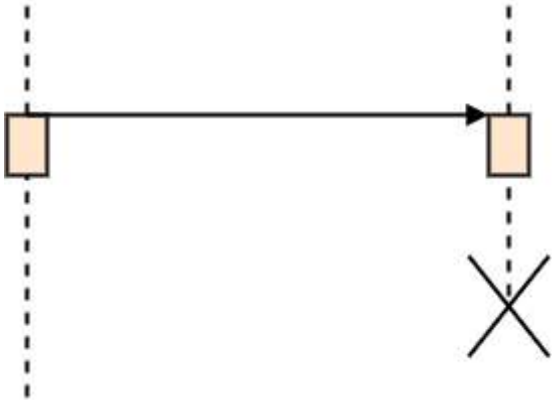
Return Message: It defines a particular communication between the lifelines of interaction that represent the flow of information from the receiver of the corresponding caller message.



Self Message: It describes a communication, particularly between the lifelines of an interaction that represents a message of the same lifeline, has been invoked.



Destroy Message: It describes a communication, particularly between the lifelines of an interaction that depicts a request to destroy the lifecycle of the target.



The **activity diagram** helps in envisioning the workflow from one activity to another. It put emphasis on the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc

Notation of an Activity diagram

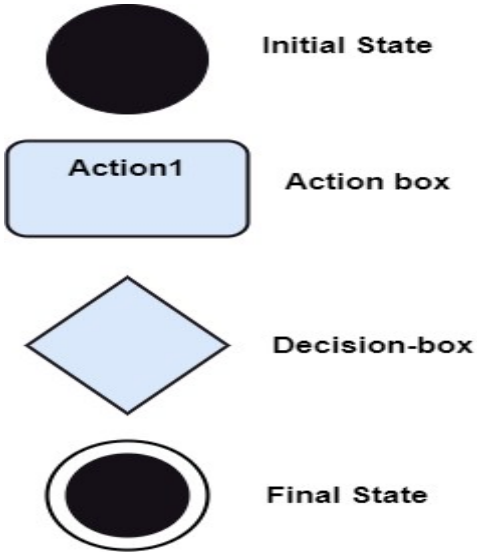
Activity diagram constitutes following notations:

Initial State: It depicts the initial stage or beginning of the set of actions.

Final State: It is the stage where all the control flows and object flows end.

Decision Box: It makes sure that the control flow or object flow will follow only one path.

Action Box: It represents the set of actions that are to be performed.

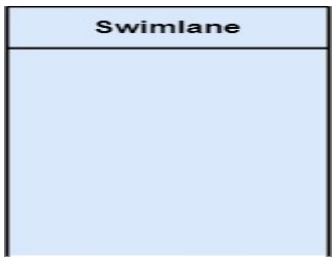


Activity

The control flow of activity is represented by control nodes and object nodes that illustrates the objects used within an activity. The activities are initiated at the initial node and are terminated at the final node.

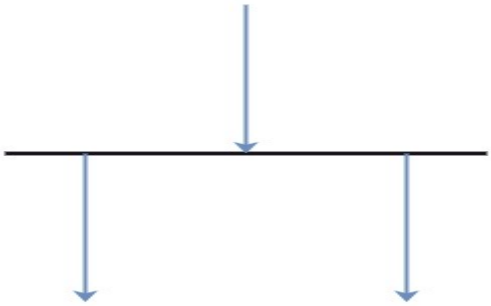
swimlane

The swimlane is used to cluster all the related activities in one column or one row. It can be either vertical or horizontal.



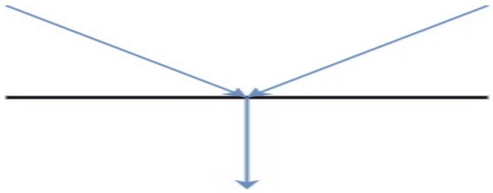
Forks

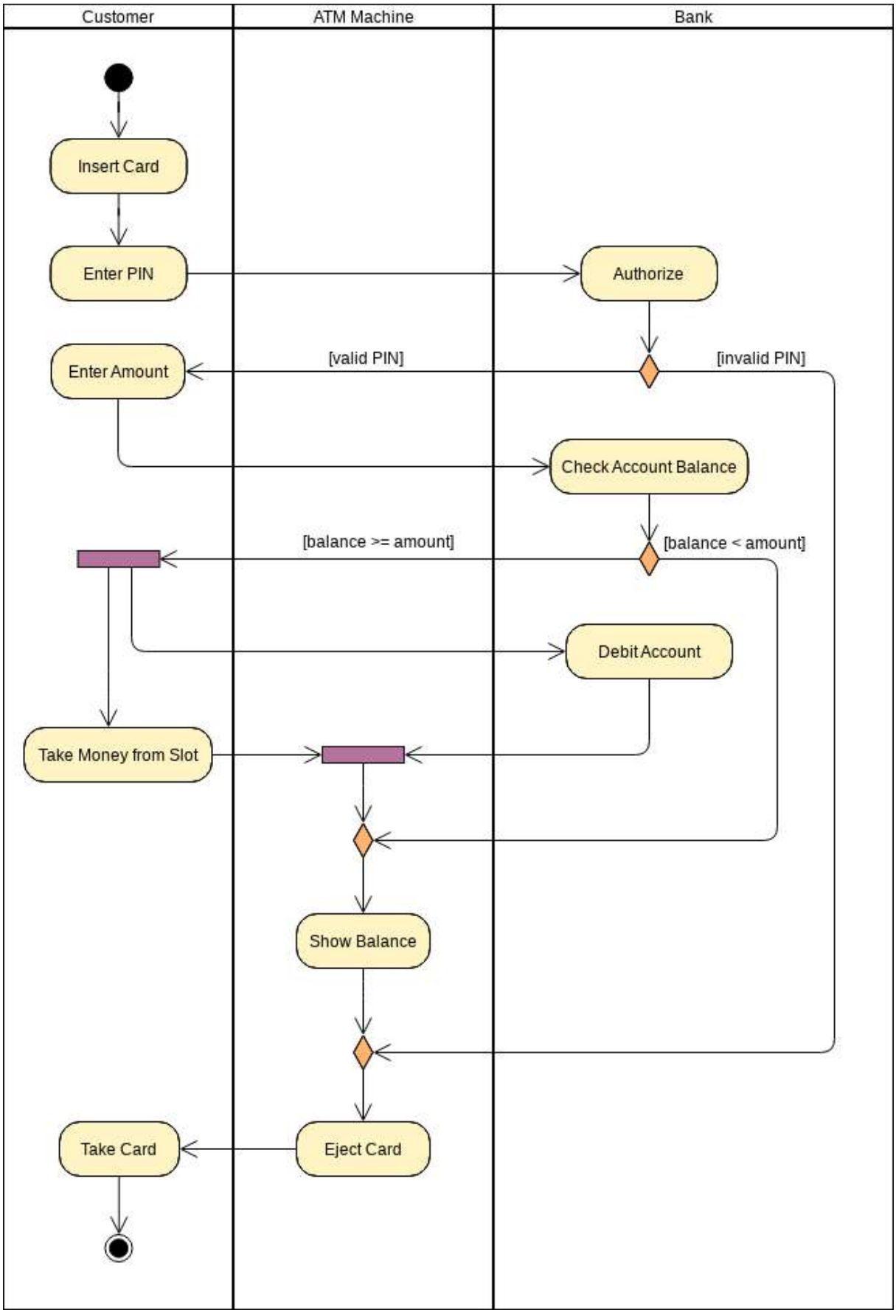
A fork node consists of one inward edge and several outward edges.

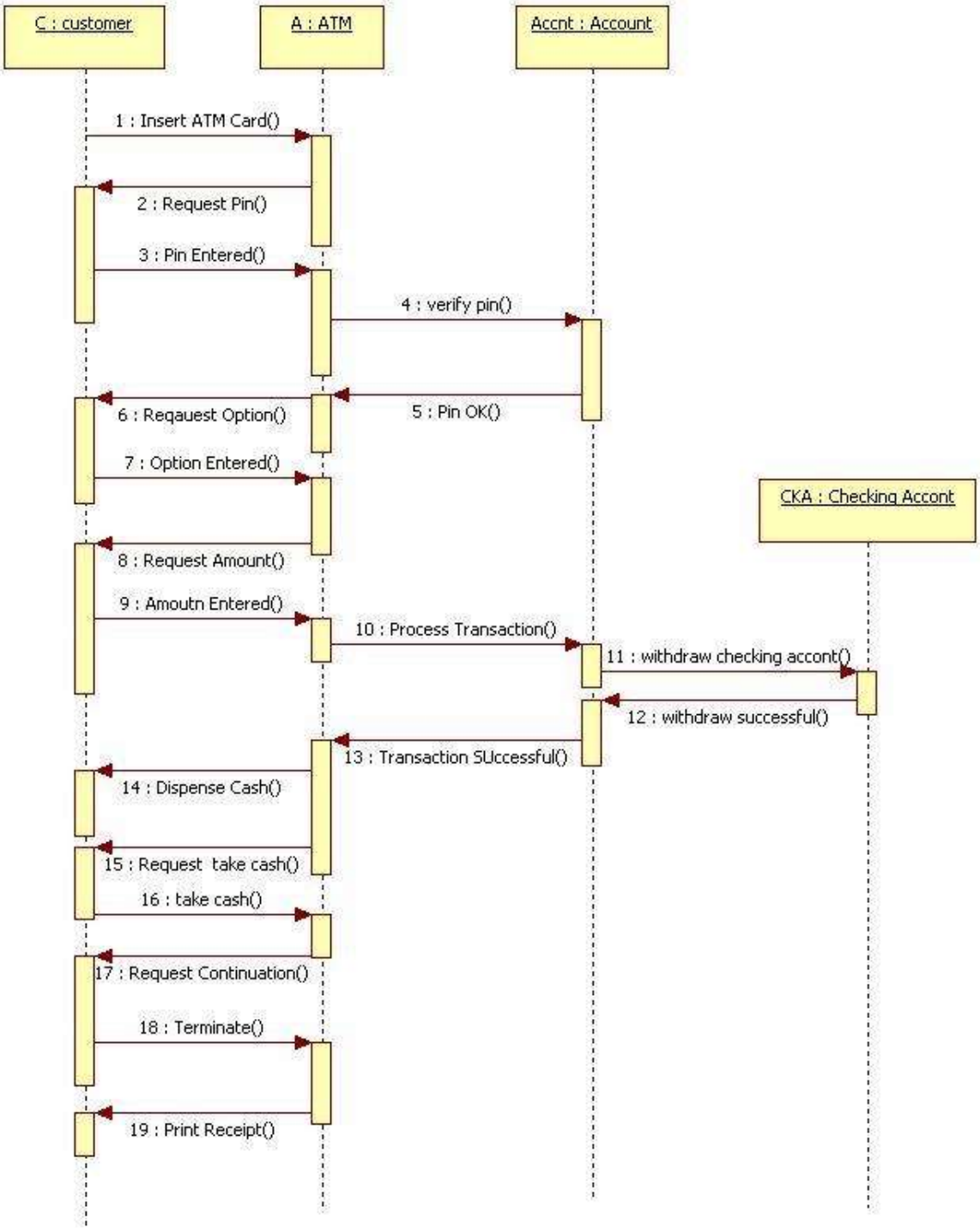


Join Nodes

Join nodes are the opposite of fork nodes. A Logical AND operation is performed on all of the inward edges as it synchronizes the flow of input across one single output (outward) edge.




Customer
ATM Machine
Bank
Insert Card
Enter PIN
Authorize
Enter Amount
[valid PIN]
[invalid PIN]
Check Account Balance
[balance >= amount]
[balance < amount]
Debit Account
Take Money from Slot
Show Balance
Take Card
Eject Card

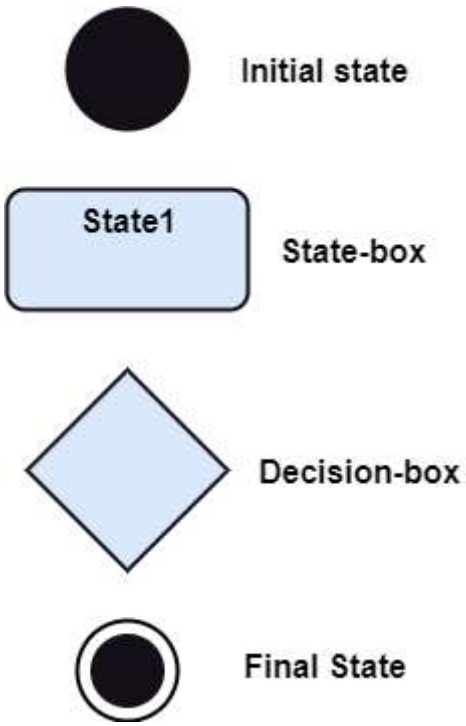

C : customer
A : ATM
Accnt : Account
CKA : Checking Accont
1 : Insert ATM Card()
2 : Request Pin()
3 : Pin Entered()
4 : verify pin()
5 : Pin OK()
6 : Reqauest Option()
7 : Option Entered()
8 : Request Amount()
9 : Amoutn Entered()
10 : Process Transaction()
11 : withdraw checking accont()
12 : withdraw successful()
13 : Transaction SUccessful()
14 : Dispense Cash()
15 : Request take cash()
16 : take cash()
17 : Request Continuation()
18 : Terminate()
19 : Print Receipt()

Statechart

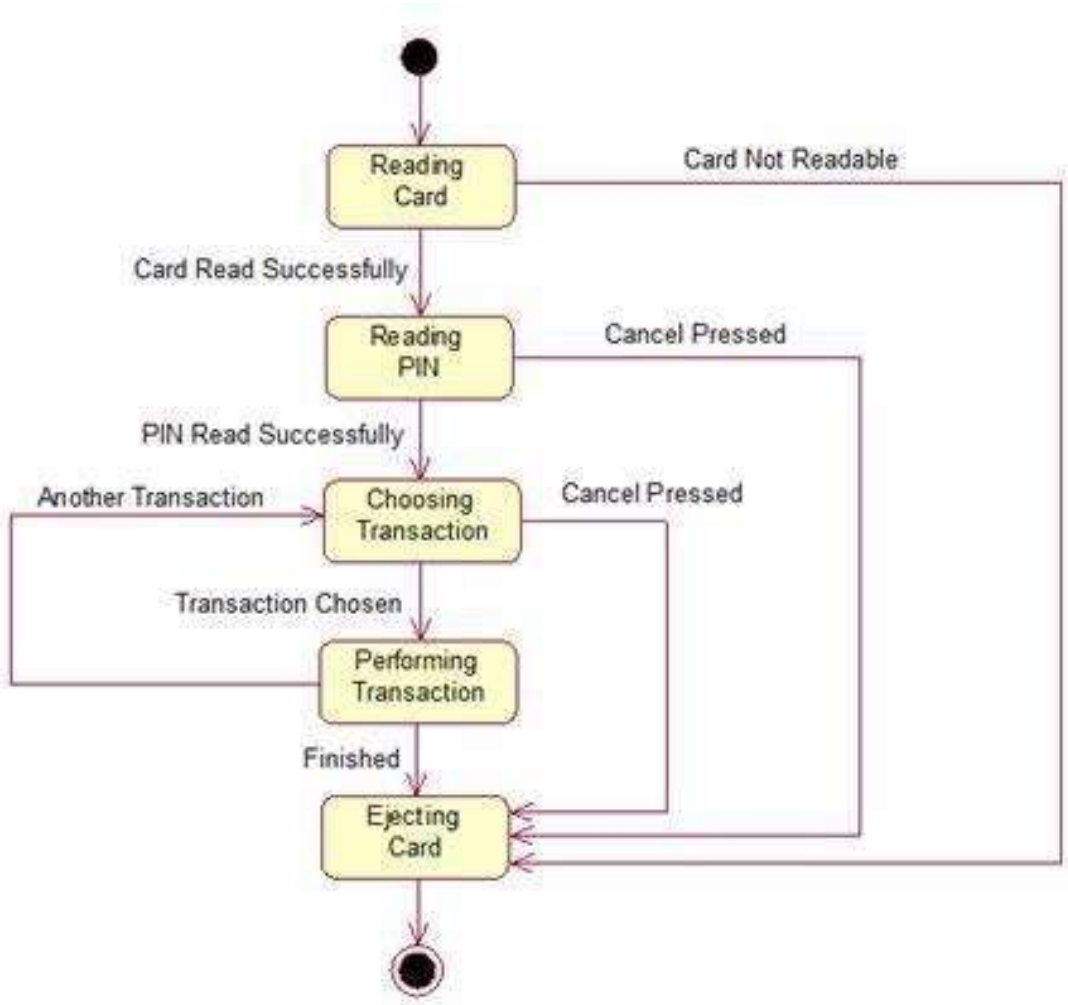
The state machine diagram is also called the Statechart or State Transition diagram, which shows the order of states underwent by an object within the system. It captures the software system's behavior. It models the behavior of a class, a subsystem, a package, and a complete system.

Notation of a State Machine Diagram

Following are the notations of a state machine diagram enlisted below:



1. **Initial state:** It defines the initial state (beginning) of a system, and it is represented by a black filled circle.
2. **Final state:** It represents the final state (end) of a system. It is denoted by a filled circle present within a circle.
3. **Decision box:** It is of diamond shape that represents the decisions to be made on the basis of an evaluated guard.
4. **Transition:** A change of control from one state to another due to the occurrence of some event is termed as a transition. It is represented by an arrow labeled with an event due to which the change has ensued.
5. **State box:** It depicts the conditions or circumstances of a particular object of a class at a specific point of time. A rectangle with round corners is used to represent the state box.



Collaboration Diagram

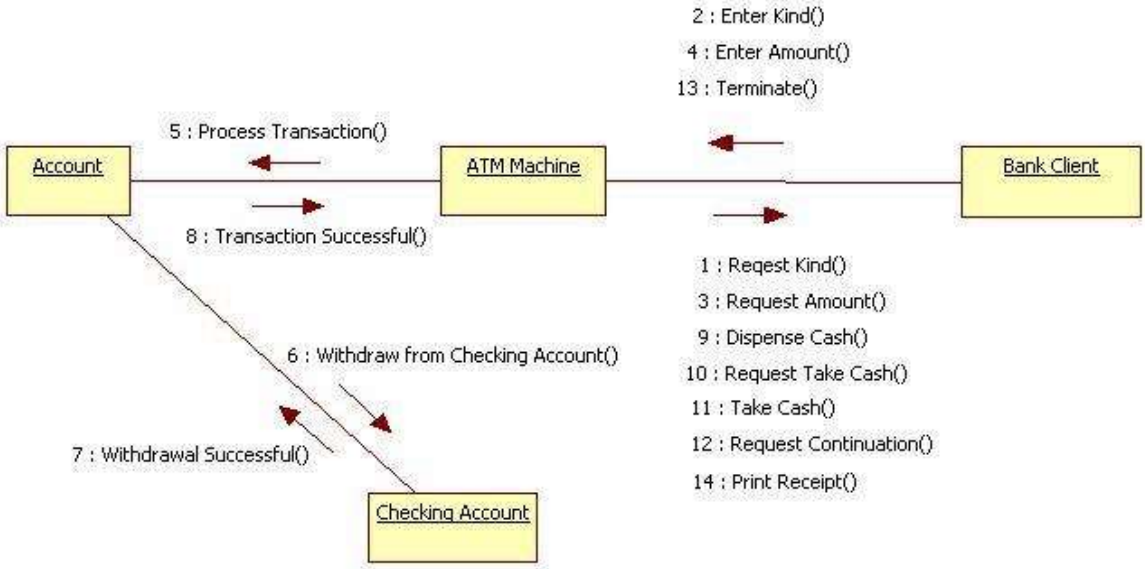
The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.

Notations of a Collaboration Diagram

Objects: The representation of an object is done by an object symbol with its name and class underlined, separated by a colon.

Links: The link is an instance of association, which associates the objects and actors. It portrays a relationship between the objects through which the messages are sent. It is represented by a solid line.

Messages: It is a communication between objects which carries information and includes a sequence number, so that the activity may take place. It is represented by a labeled arrow, which is placed near a link.



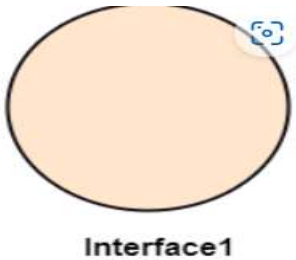
component diagram

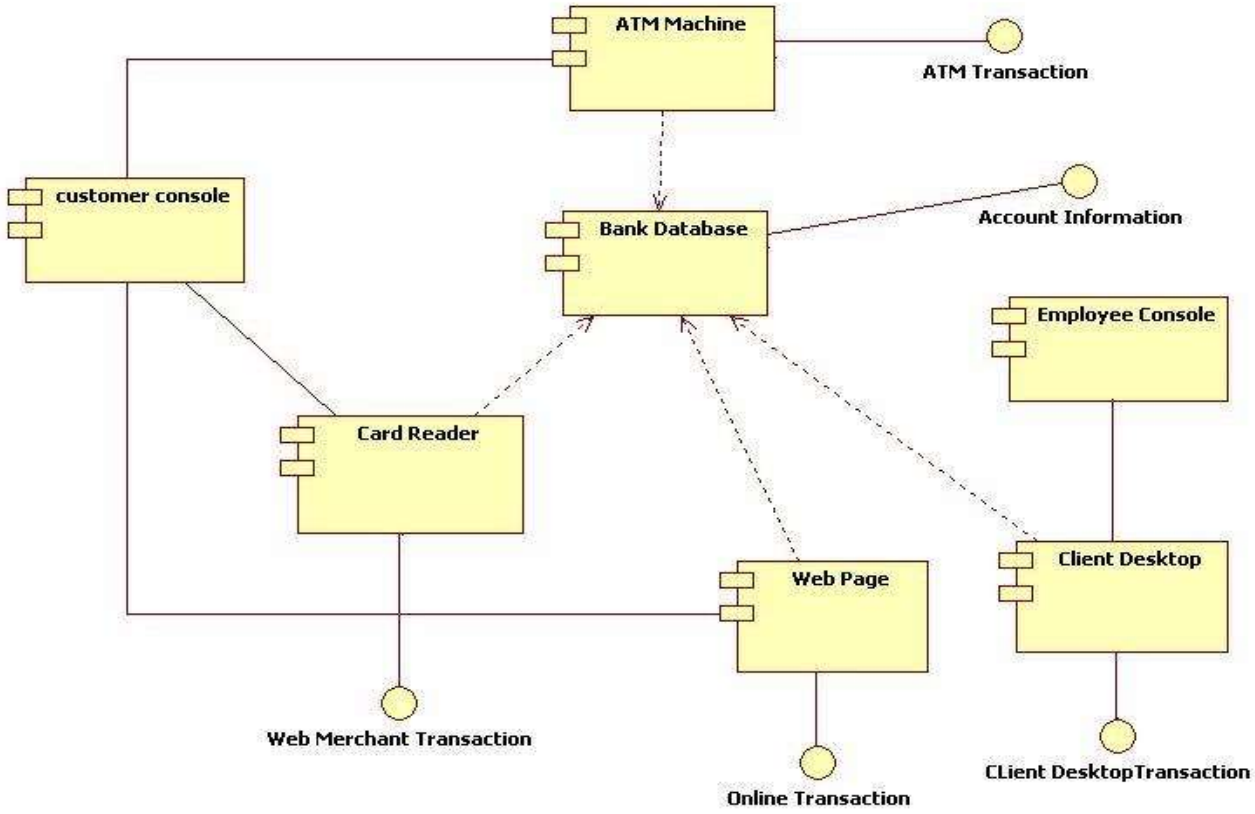
A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node.

Notation of a Component Diagram

Component: Represented as a rectangle with the component's name. Components are modular units of a system, such as a web server, database, or API.

Interface shown as a circle (lollipop symbol) or a rectangle with a line. Represents the methods or operations exposed by a component.

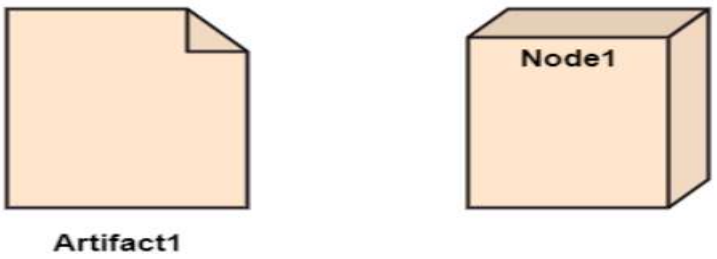




Deployment diagram

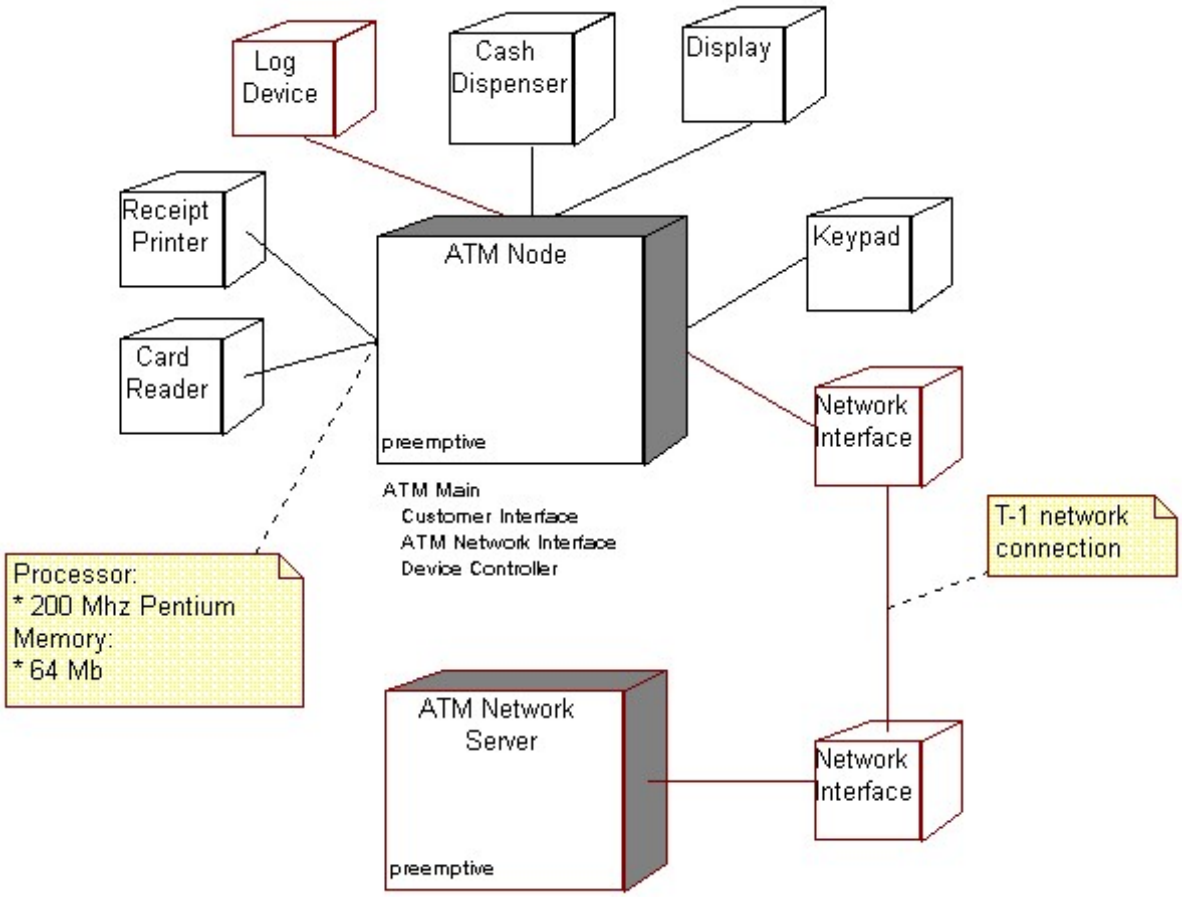
The deployment diagram visualizes the physical hardware on which the software will be deployed. It portrays the static deployment view of a system. It involves the nodes and their relationships.

Symbol and notation of Deployment diagram



Node: Represent physical hardware like servers, computers, or IoT devices.

Artifact : An artifact represents a physical piece of information that is deployed onto a node. Artifacts can include files, executables, scripts, or any deliverable that gets installed or deployed on hardware or a virtual environment. An artifact is depicted as a rectangle with a folded corner.


Log Device
Cash Dispenser
Display
Receipt Printer
ATM Node
Keypad
Card Reader
Network Interface
preemptive
ATM Main
Customer Interface
ATM Network Interface
Device Controller
T-1 network connection
Processor:
* 200 Mhz Pentium
Memory:
* 64 Mb
ATM Network Server
Network Interface
preemptive