PRACTICALS

1. [illegible]nd implementation of class diagrams.

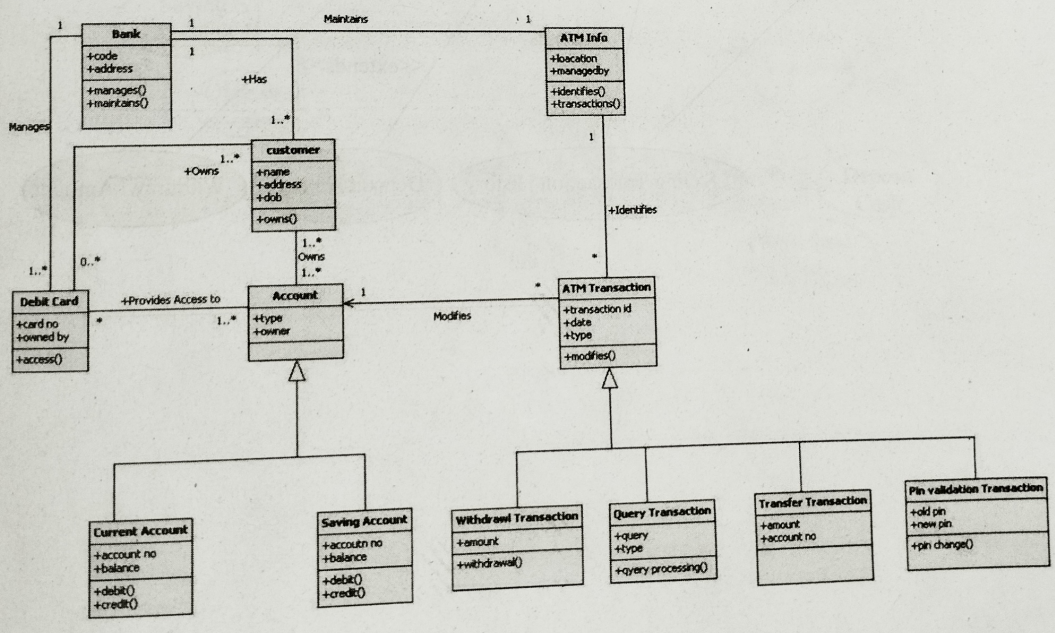
A class diagram models the static structure of a system. It shows relationships between classes, objects, attributes, and operations.

Classes represent an abstraction of entities with common characteristics. Associations represent the relationships between classes.

A classes is a rectangle divided into compartments. The name of the class is written in the first partition (centered, bolded, and capitalized), list the attributes in the second partition (left-aligned, not bolded, and lowercase), and operations into the third.

Class Name
attributes
operations()
responsibility

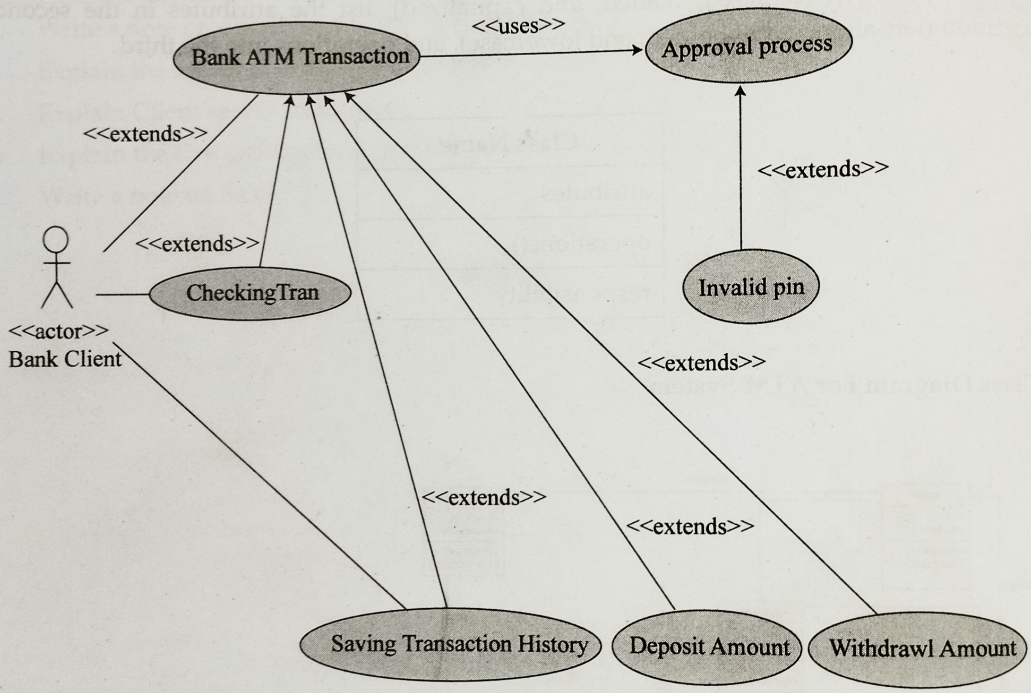
Class Diagram For ATM System



2. Study and implementation of Use Case Diagrams.

A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system.

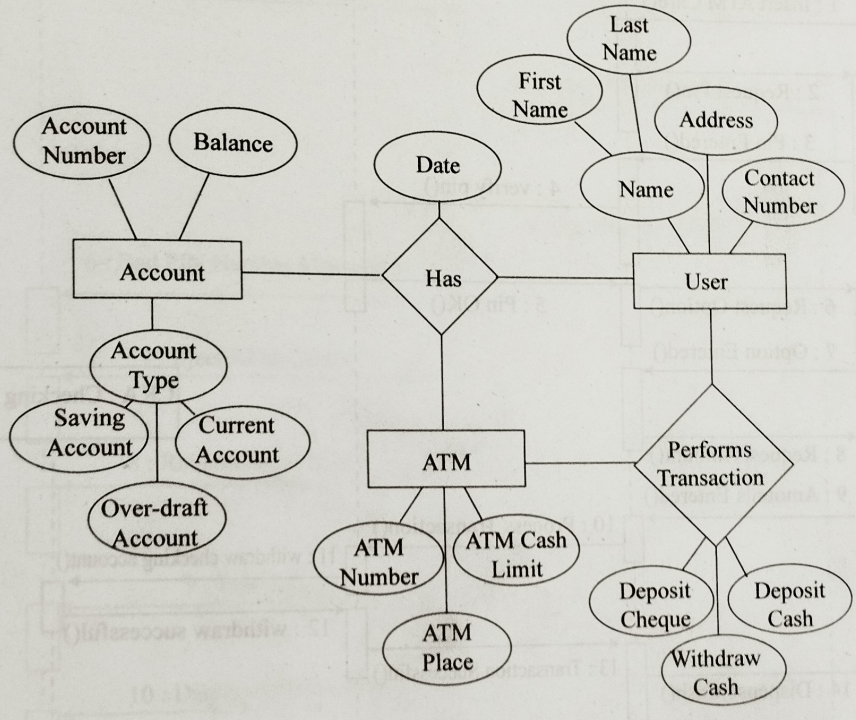
Use Diagram For ATM System



3. Study and implementation of Entity Relationship Diagrams.

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how "entities" such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes.

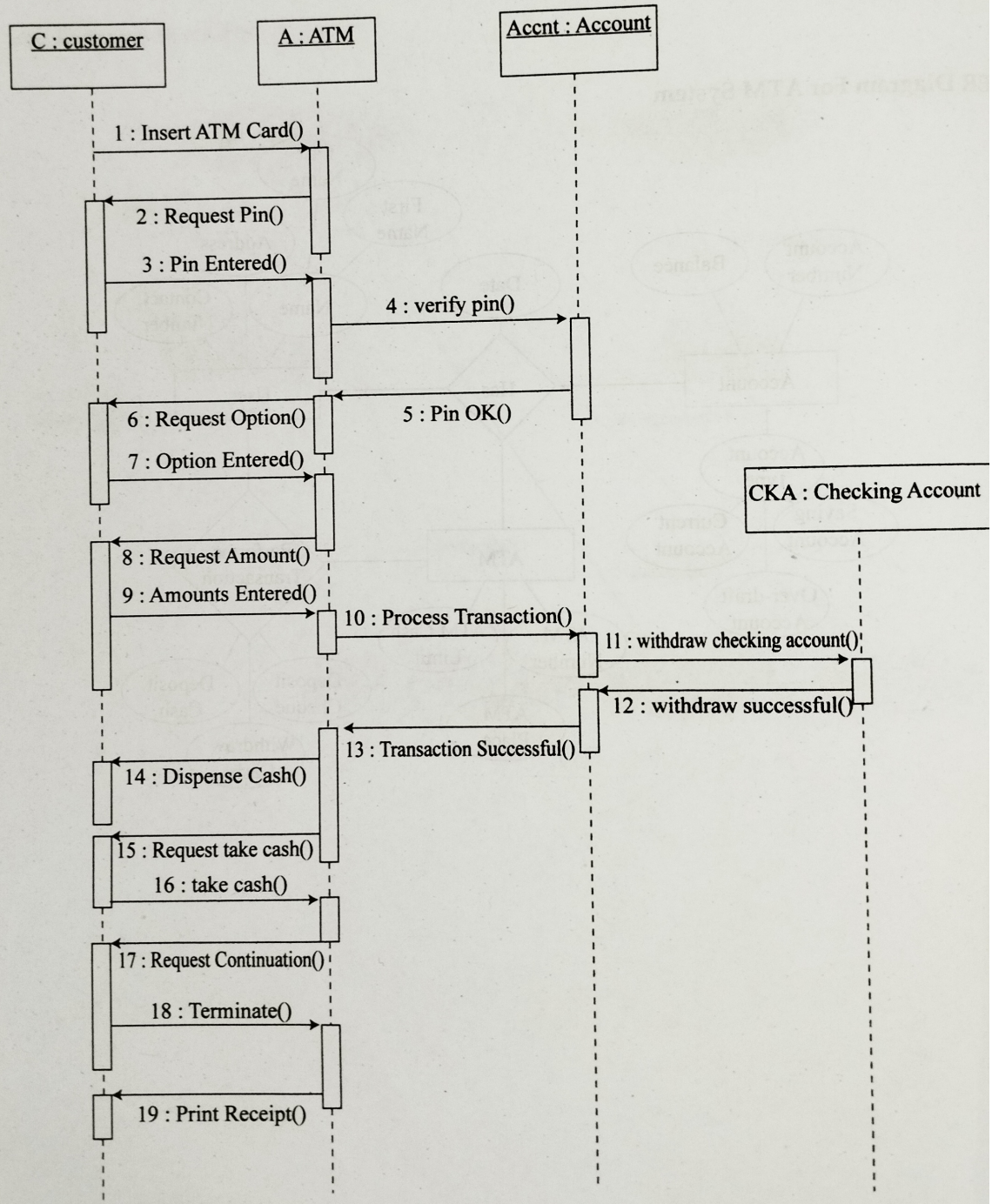
ER Diagram For ATM System



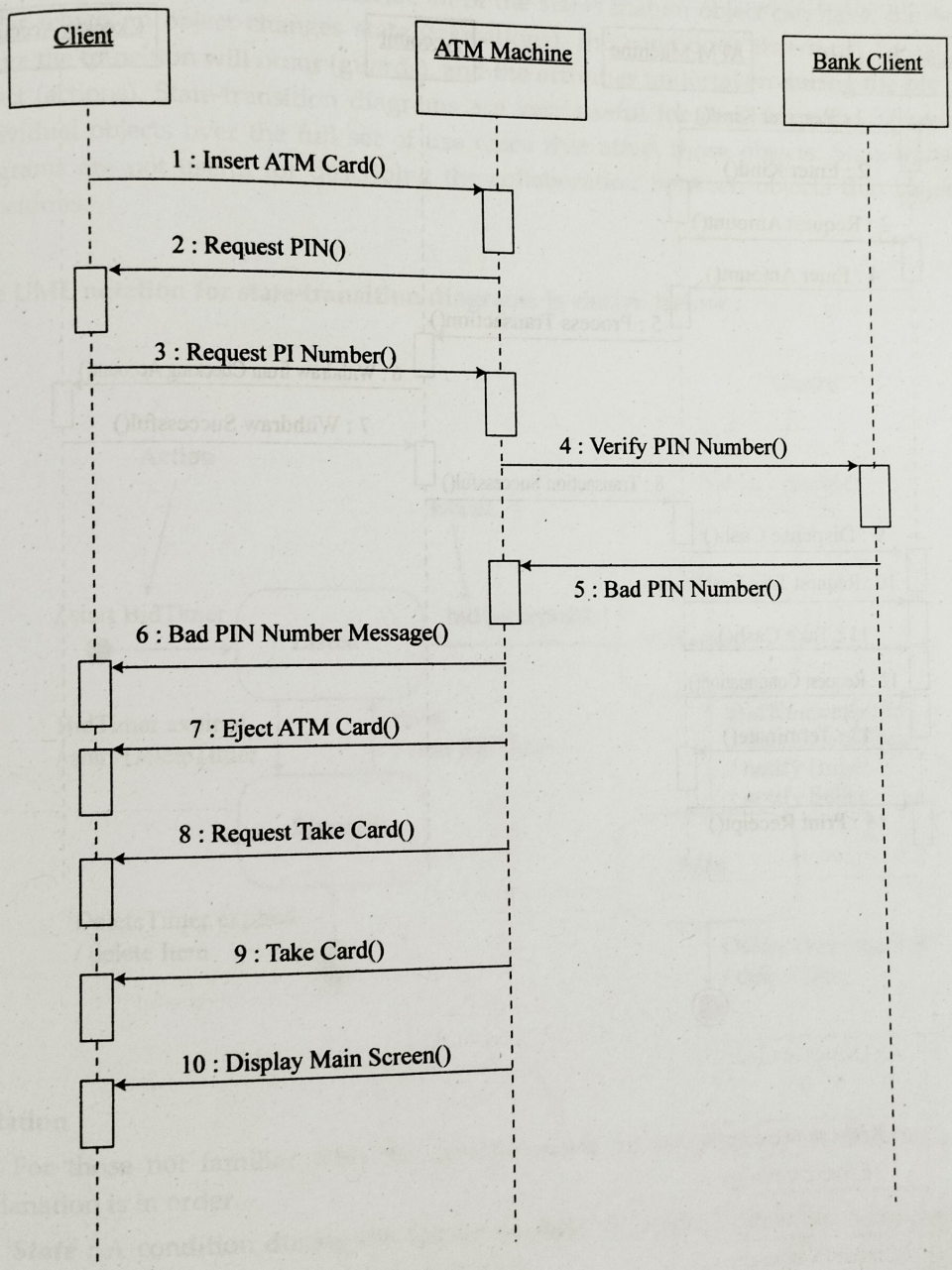
4. Study and implementation of Sequence Diagrams.

Sequence diagrams are a popular dynamic modeling solution. Dynamic modeling focuses on the interactions occurring within the system. Sequence diagrams specifically focus on the "lifelines" of an object and how they communicate with other objects to perform a function before the lifeline ends.

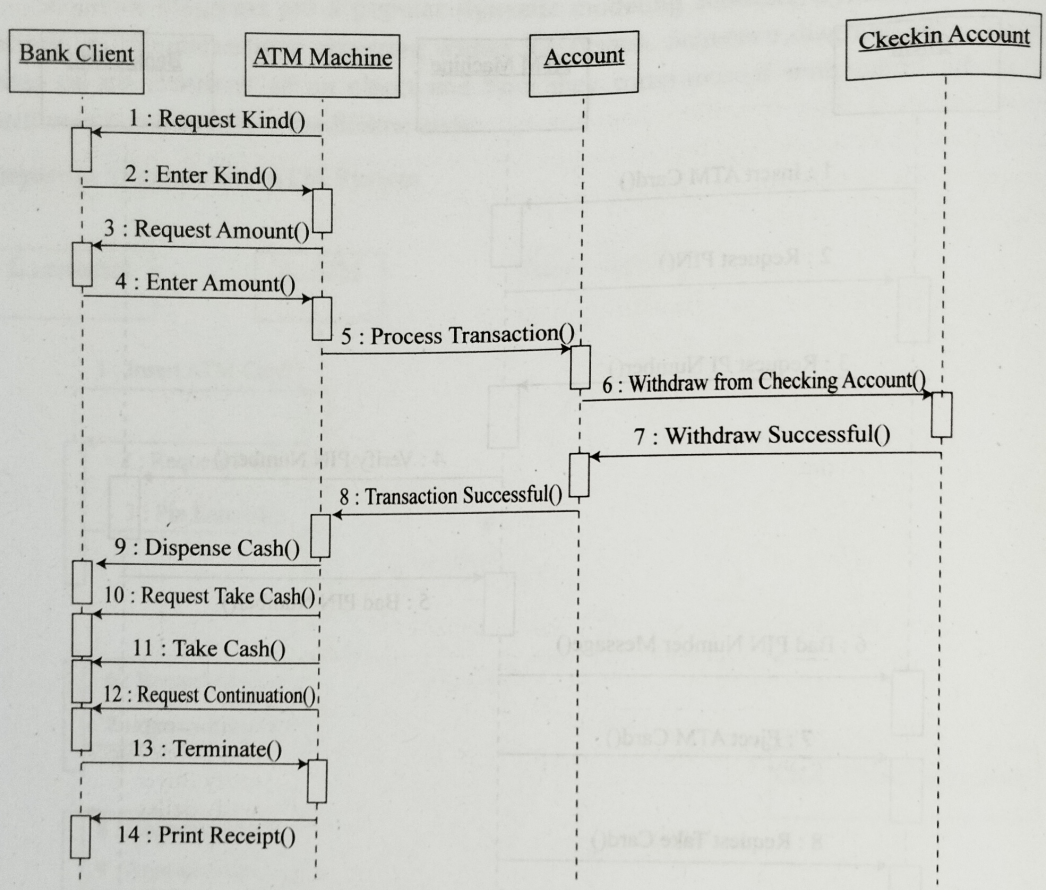
Sequence Diagram For ATM System



Sequence Diagram for Invalid Pin ATM Machine



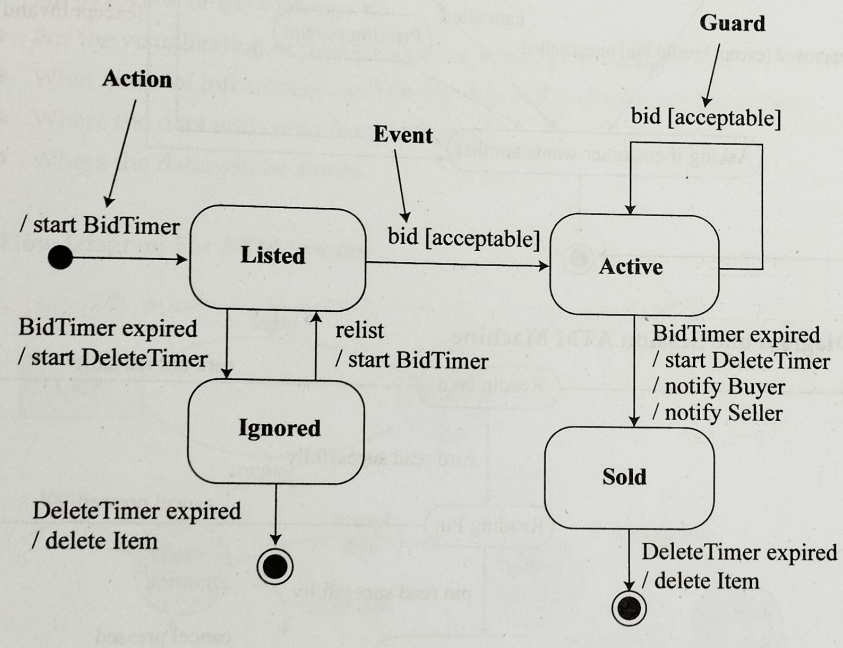
Sequence Diagram ATM withdrawal



5. Study and implementation of State Transition Diagrams.

State-transition diagrams describe all of the states that an object can have, the events under which an object changes state (transitions), the conditions that must be fulfilled before the transition will occur (guards), and the activities undertaken during the life of an object (actions). State-transition diagrams are very useful for describing the behavior of individual objects over the full set of use cases that affect those objects. State-transition diagrams are not useful for describing the collaboration between objects that cause the transitions.

The UML notation for state-transition diagrams is shown below :



Notation

For those not familiar with the notation used for state-transition diagrams, some explanation is in order.

State : A condition during the life of an object in which it satisfies some condition, performs some action, or waits for some event.

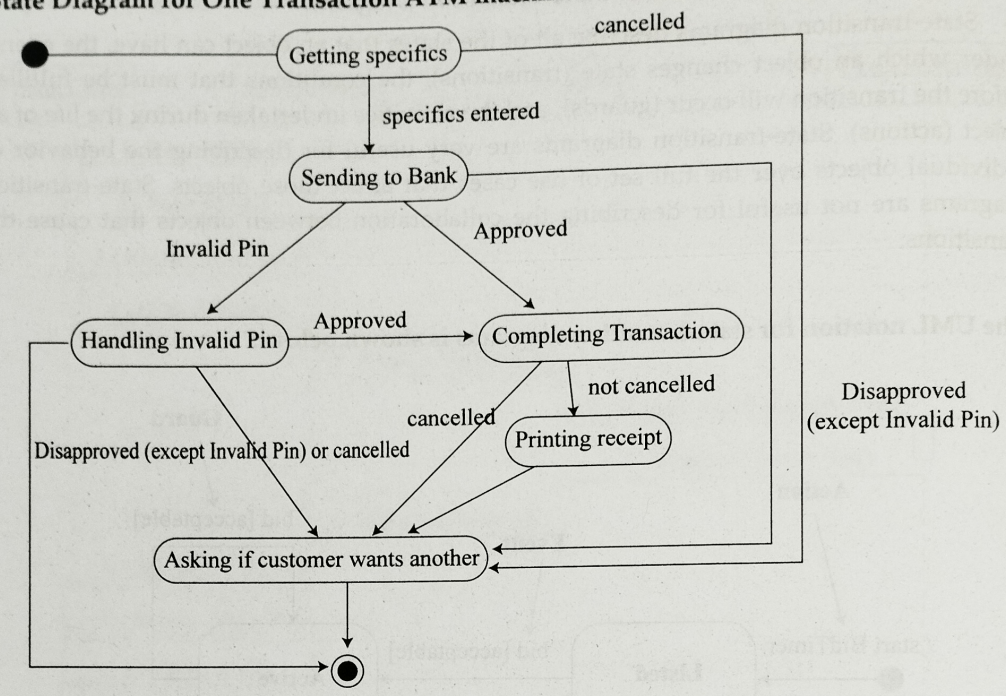
Event : An occurrence that may trigger a state transition. Event types include an explicit signal from outside the system, an invocation from inside the system, the passage of a designated period of time, or a designated condition becoming true.

Guard : A boolean expression which, if true, enables an event to cause a transition.

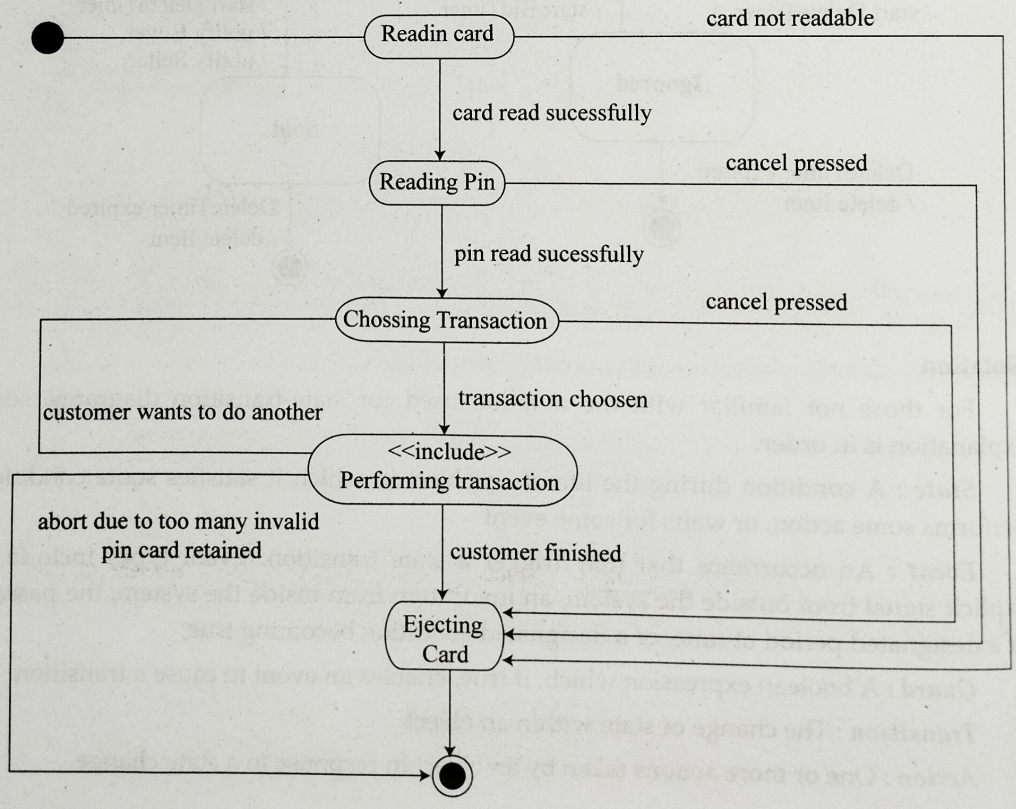
Transition : The change of state within an object.

Action : One or more actions taken by an object in response to a state change.

State Diagram for One Transaction ATM machine



State Diagram one Session ATM Machine



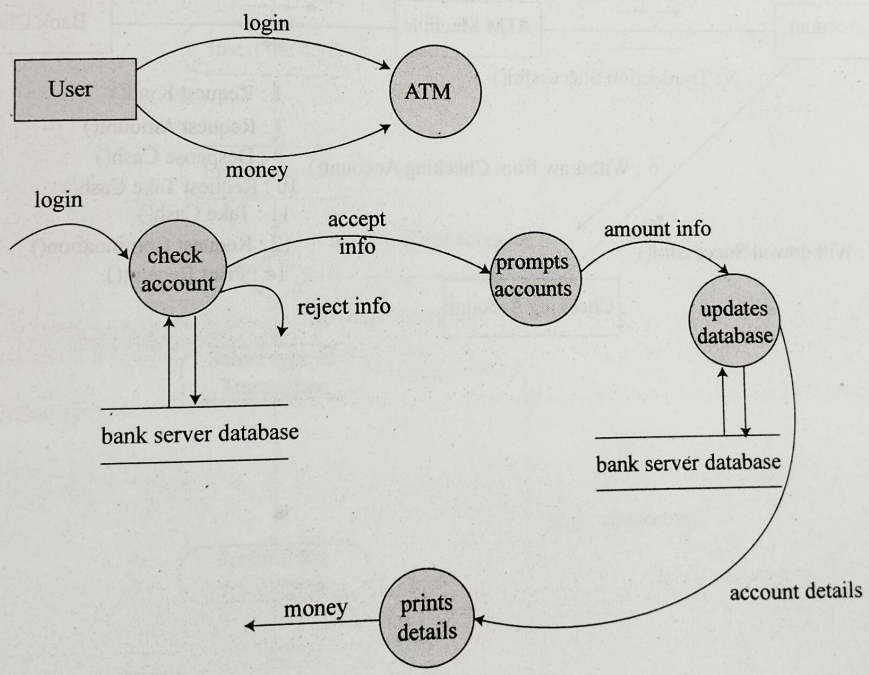
6. Study and implementation of Data Flow Diagrams.

Data Flow Diagrams show information transfers and process steps of a system. The general concept is an approach of a depicting how occurs input in a system, further processes and what runs out. The aim of DFD is in accomplishing of understanding between developers and users. Data flow diagrams are maintained with other methods of structured systems analysis.

A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.

- Graphical representation of the "flow" of data through an information system;
- Modeling process aspects;
- An overview of the system;
- For the visualization of data processing (structured design);
- What kinds of information will be input to and output from the system;
- Where the data will come from and go to;
- Where the data will be stored.

Data Flow Diagram For ATM System

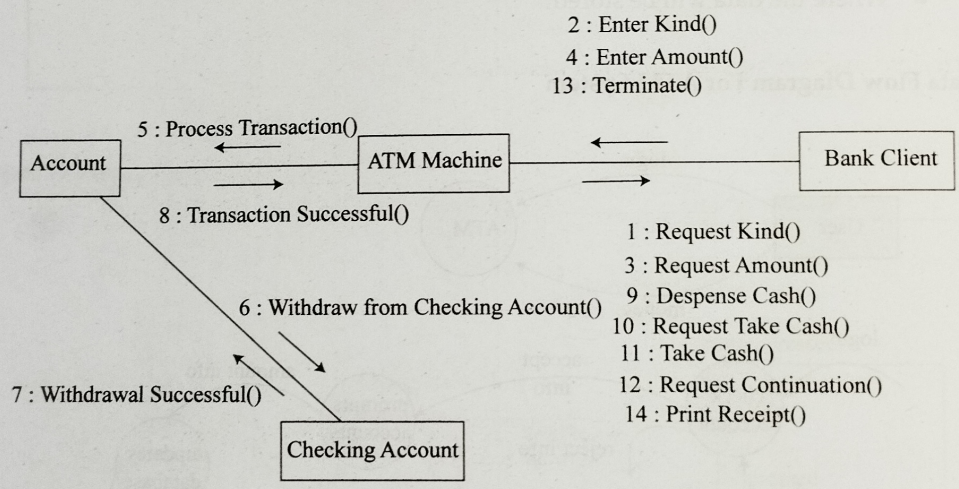


7. Study and implementation of Collaboration Diagrams.

Collaboration diagrams are used to show how objects interact to perform the behavior of a particular use case, or a part of a use case. Along with sequence diagrams, collaborations are used by designers to define and clarify the roles of the objects that perform a particular flow of events of a use case. They are the primary source of information used to determining class responsibilities and interfaces.

Unlike a sequence diagram, a collaboration diagram shows the relationships among the objects. Sequence diagrams and collaboration diagrams express similar information, but show it in different ways. Collaboration diagrams show the relationships among objects and are better for understanding all the effects on a given object and for procedural design.

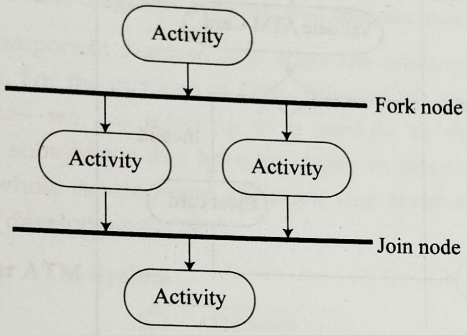
Collaboration Diagram For ATM System



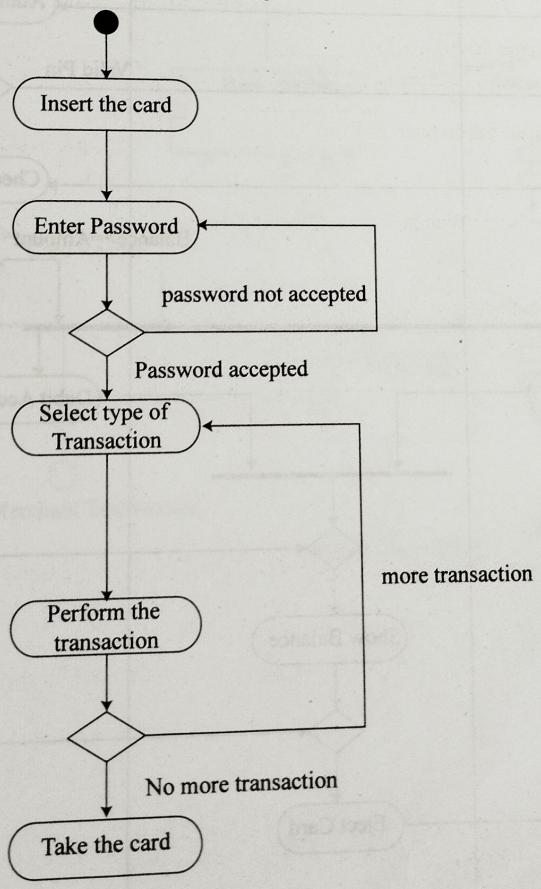
8. Study and implementation of Activity Diagrams.

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent.

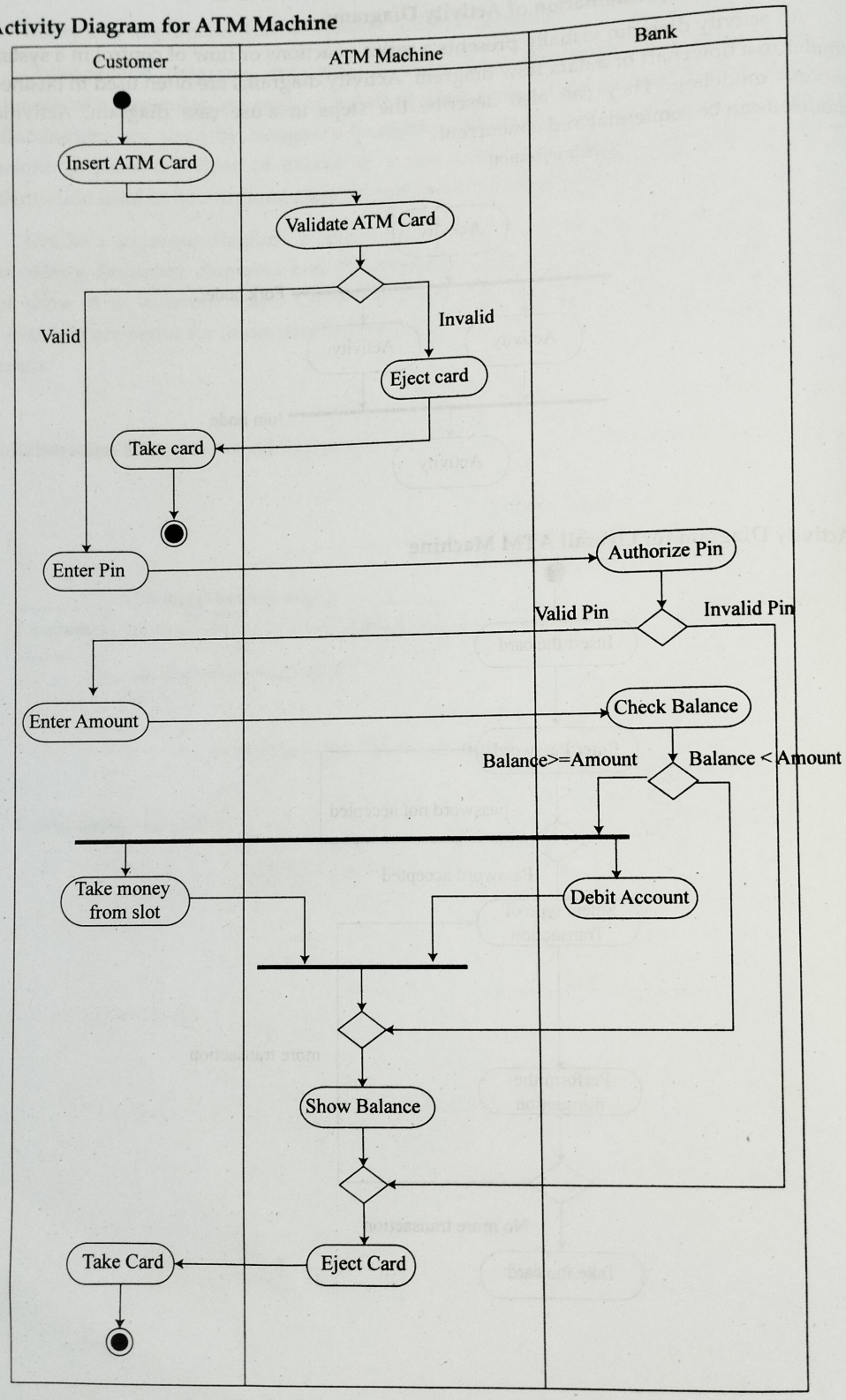
Synchronization



Activity Diagram for Overall ATM Machine



Activity Diagram for ATM Machine



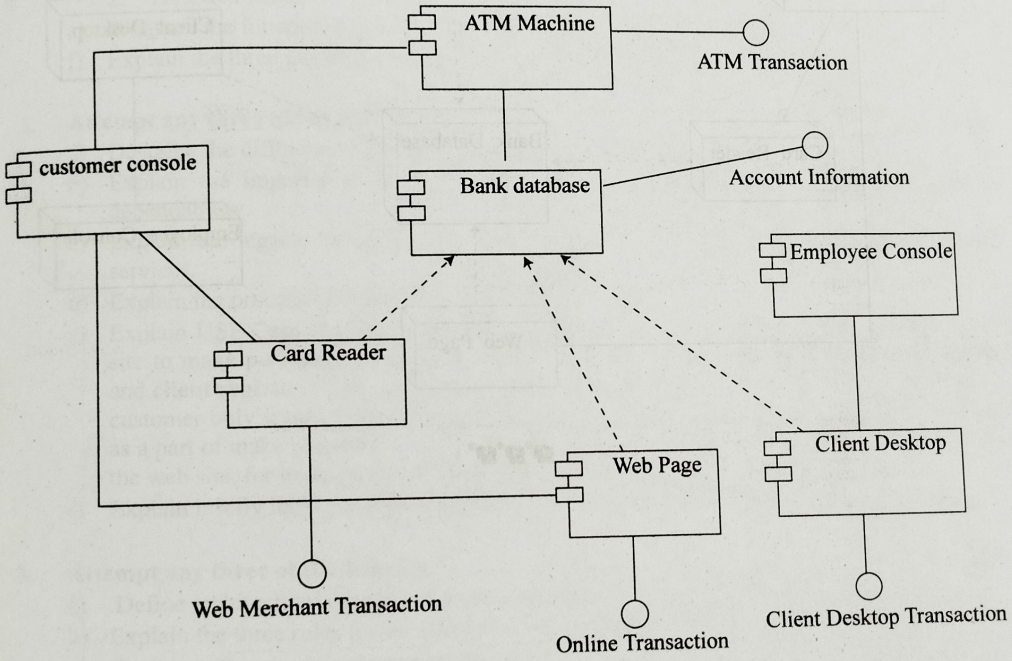
9. Study and implementation of Component Diagrams.

Component diagrams are different in terms of nature and behavior. Component diagrams are used to model the physical aspects of a system. Now the question is, what are these physical aspects? Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node.

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

The purpose of a component diagram is to show the relationship between different components in a system. For the purpose of UML 2.0--and for any time that we discuss components on this site-- we are referring to a module of classes which represent independent systems or subsystems that have an ability to interface with the rest of the system. There exists a whole development approach that revolves around components, called component-based development (CBD).

Component Diagram For ATM System



10. Study and implementation of Deployment Diagrams.

Deployment Diagram also helps to model the physical aspect of an Object-Oriented software system. It models the run-time configuration in a static view and visualizes the distribution of components in an application.

A deployment diagram can cover a wide variety of levels within a single model. For instance, one diagram might cover how the application layers are connected, another might indicate where the source code is located within the system, and another might show how an executable is used across several nodes.

Deployment Diagram For ATM System