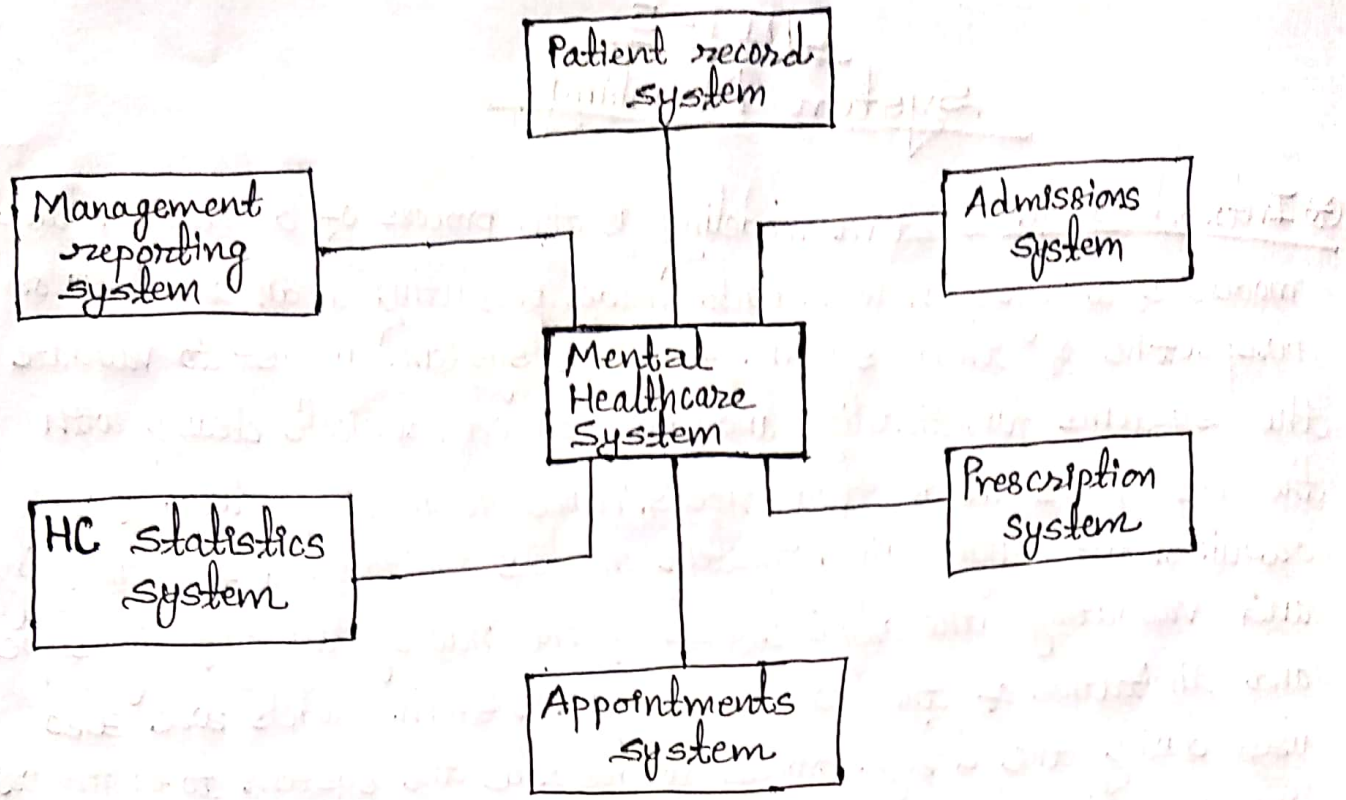
UNIT-5
System Modeling

⊛ Introduction: System modeling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system. It helps system analyst to validate the system's functionality and helps to communicate clearly with the customers about their needs. Hence, it is mainly used for requirement engineering. System models mainly help in identifying and validating the requirements of the new system by finding scope and limitation of the existing system. System models are also used during the design process to describe the system to engineers implementing the system. There are different types of models to represent the system from different perspectives: Contextual model, Interaction model, Structural model, Behavioral model.

⊛. Context Models:

Context models represents the operational environment of the system, they represent what lies outside the system boundaries. The environment of the system contains social and organizational concerns which directly or indirectly effects the position of system boundaries. Hence, architectural models are used to show the system and its relationship with other systems. Context models should be developed early in the process to reduce the system costs and the time needed for understanding the system requirements and design.

Figure below is the context model for mental healthcare system that shows the mental healthcare system and the other systems in its environment.



⊛ 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. Use case diagrams and sequence diagrams are used for interaction modeling.

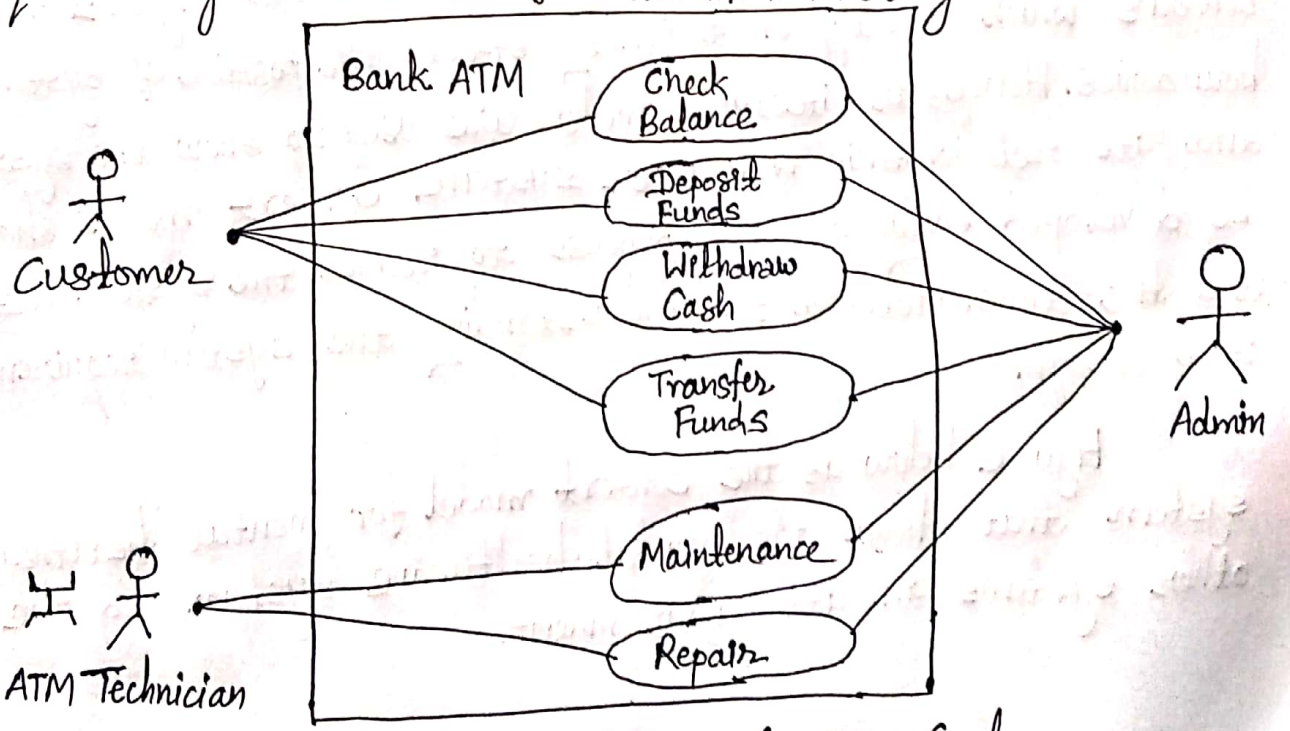


Fig: Use Case diagram for ATM System

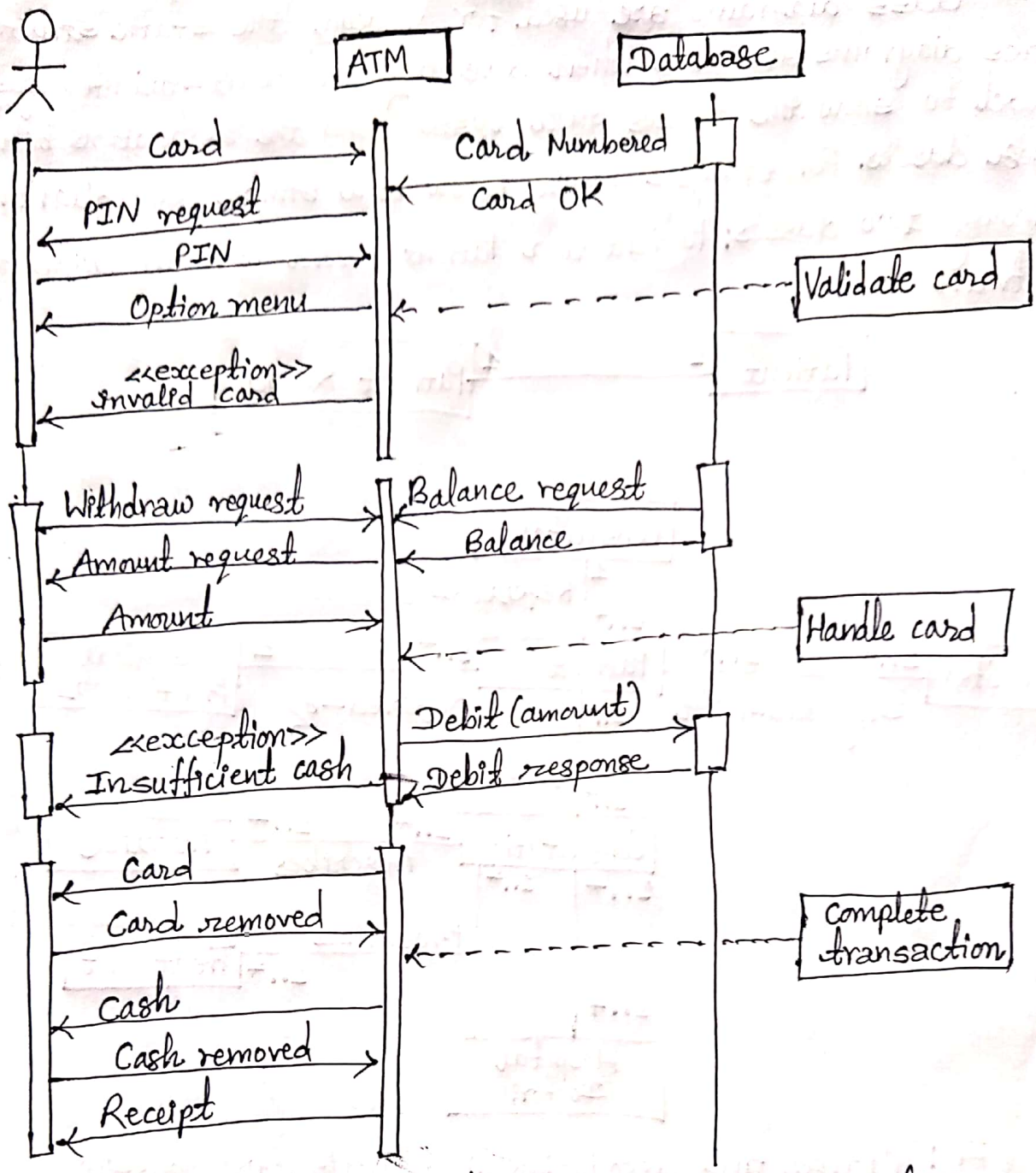
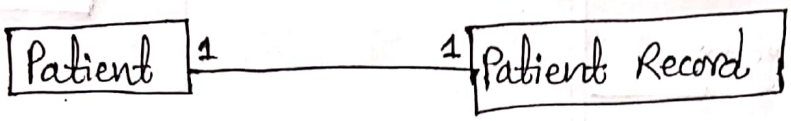


Fig: Sequence diagram of ATM withdrawl

⊛ Structural Models:

Structural models show the organization of a system in terms of the components that make up the 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. We create structural models of a system when we are discussing and designing the system architecture.

Class diagrams are used for modeling the static structure. Class diagrams are used when developing an object-oriented system model to show the classes in a system and the associations between these classes. For example, Figure below is a simple class diagram showing two classes: Patient and Patient Record with an association between them.



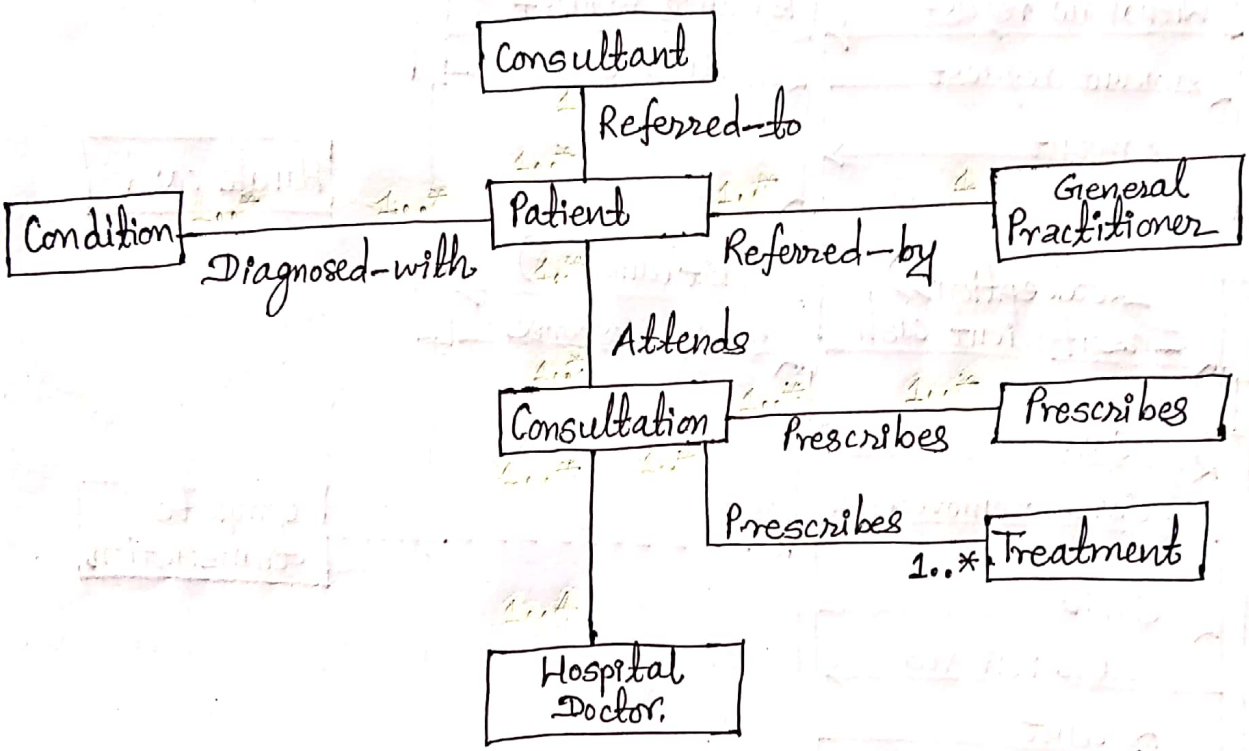
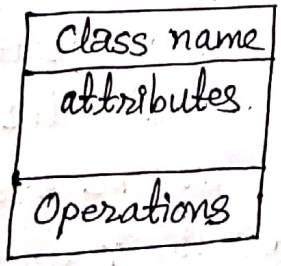


Fig: Classes and associations in a healthcare system.

To define the classes in more detail, we add information about their attributes and operations.



⊛Behavioural models:

Behavioural models are models of the dynamic behaviour 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. We can think of these stimuli as being of two types:

1) Data: Some data arrives that has to be processed by the system.
2) Events: Some event happens that triggers system processing.

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.

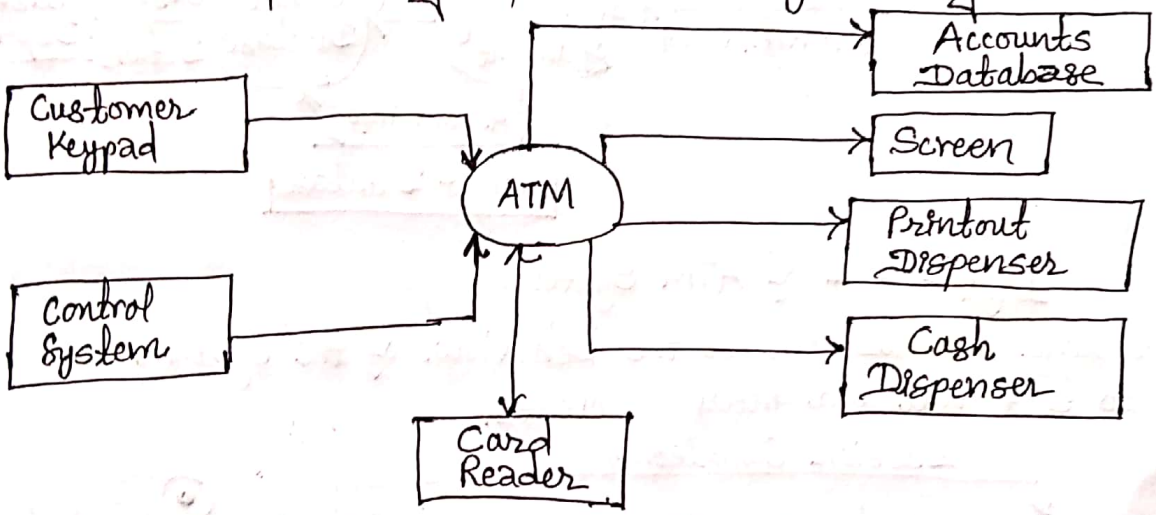


Fig: DFD 0 of ATM System

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 may cause a transition from one state to another.

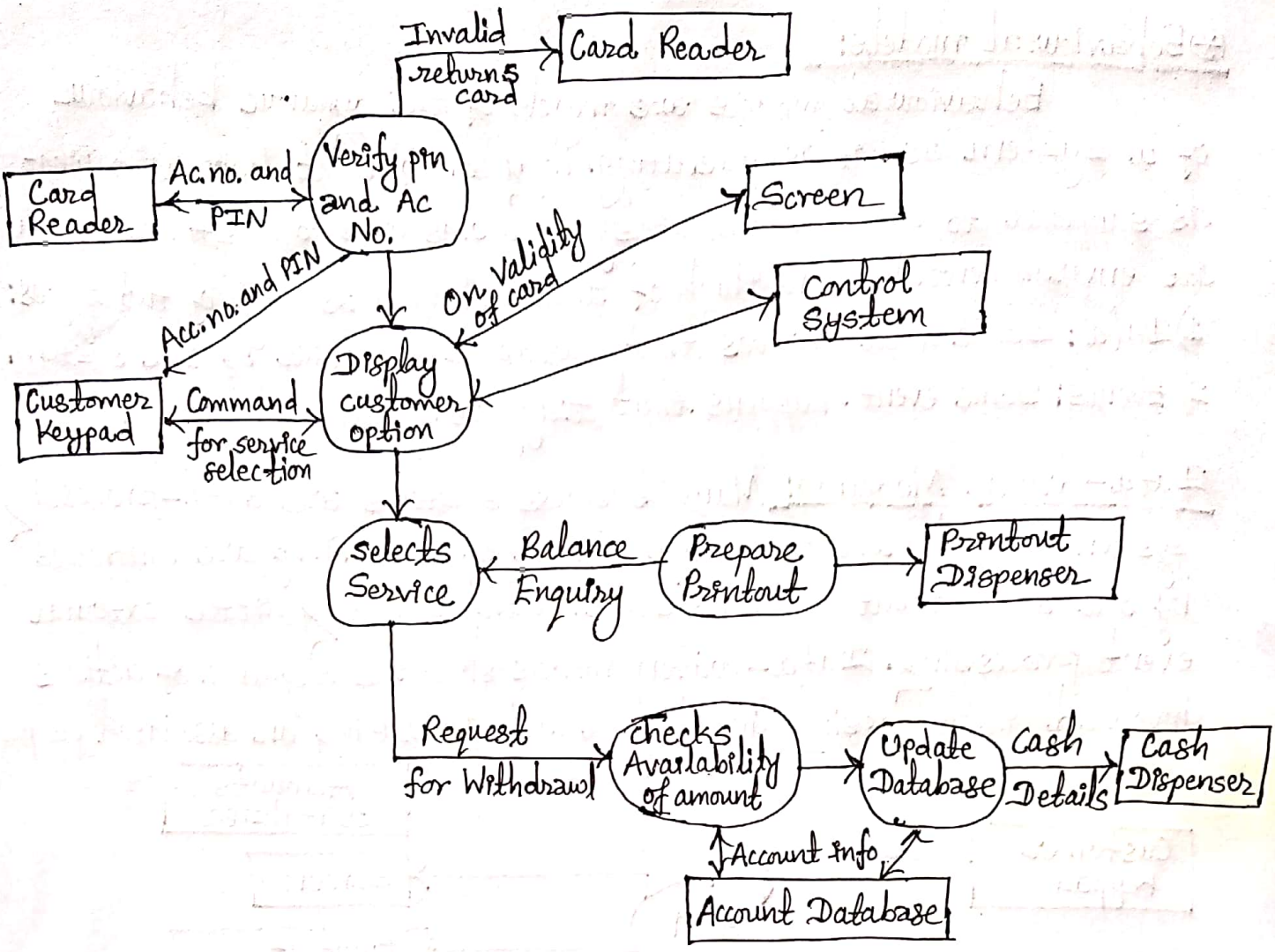


Fig: DFD 1 of ATM System

State machine models model the behaviour of the system in response to external and internal events.

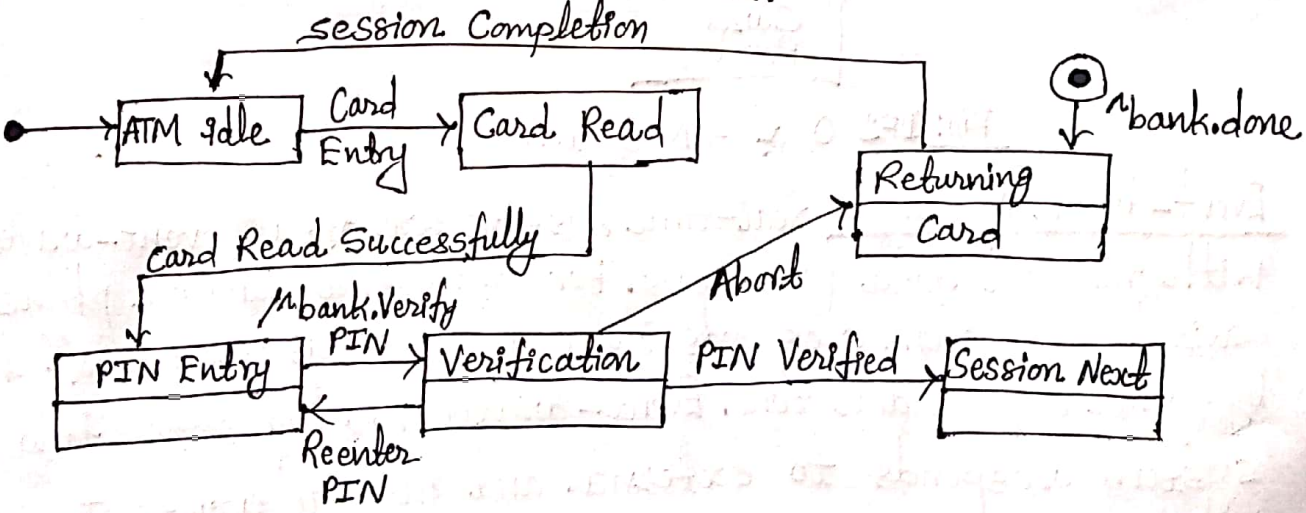


Fig: State diagram of ATM System.

⊛ Model-driven architecture:

Model-driven architecture is an approach to software design and implementation that focuses on developing models. It uses UML models to describe a system. Here, models at different levels of abstraction are created. The Model-driven architecture (MDA) recommends that three types of abstract system model should be produced:

1). A computation independent model (CIM): CIMs models the important domain abstractions used in a system and so are sometimes called domain models. We can develop several different CIMs reflecting different views of system.

2) A platform-independent model (PIM): PIMs model the operation of the system without reference to its implementation. A PIM is usually described using UML models that show the static system structure and how it responds to external and internal events.

3) Platform-specific models (PSM): PSMs are transformations of the platform-independent model with a seperate PSM for each application platform. In principle, there may be layers of PSM, with each layer adding some platform-specific detail.

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 outweighted by the costs of developing translators for new platforms.

Q1. Define software. Discuss system modeling with suitable example.

Ans: Software is an organized collection of computer programs and associated documentation. A software system consists of a number of several programs, configuration files which are used to set of these programs and system documentations which describe the structure of system and user documentation which explains how to use software.

System modeling is the process of dereloping abstract models of a system, with each model presenting a different view or perspective of that system. It is about representing a system using some kind of graphical notation, such as Unified Modeling Language (UML).

Models can explain the system from different prespectives:

External: where we model the context or environment of system.

Interaction: where we model the interactions between a system and its environment, or between the components of system.

Structural: where we model the organization of a system or the structure of the data processed by the system.

Behavioural: where we model the dynamic behaviour of the system and how it responds to events.

Five types of UML diagrams that are the most useful for system modeling:

Activity diagrams: which show activities involved in a process or data processing.

Use case diagrams: which show the interactions between the 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 associations between these classes.

State diagrams: which show how the system reacts to internal and external events.