INDEX

EXP NO.	NAME OF THE PROGRAM
1	WRITE A PROBLEM STATEMENT TO DEFINE A TITLE OF THE PROJECT WITH BOUNDED SCOPE OF PROJECT
2	PREPARE BROAD SRS (SOFTWARE REQUIREMENT SPECIFICATION) FOR THE ABOVE SELECTED PROJECTS
3	PREPARE USE CASES AND DRAW USE CASE DIAGRAM USING MODELLING TOOL
4	DEVELOP THE ACTIVITY DIAGRAM TO REPRESENT FLOW FROM ONE ACTIVITY TO ANOTHER FOR SOFTWARE DEVELOPMENT
5	DEVELOP DATA DESIGNS USING DFD DECISION TABLE & ER DIAGRAM
6	DRAW CLASS DIAGRAM, SEQUENCE DIAGRAM, COLLABORATION DIAGRAM, STATE TRANSITION DIAGRAM FOR THE ASSIGNED PROJECT
7	WRITE TEST CASES TO VALIDATE REQUIREMENTS OF ASSIGNED PROJECT FROM SRS DOCUMENT
8	EVALUATE SIZE OF THE PROJECT USING FUNCTION POINT METRIC FOR THE ASSIGNED PROJECT

EXNO: 1	Write a Problem Statement to define a title of the project with bounded scope of project

Aim:

To Write a Problem Statement to define ATM System of the project with bounded scope of project.

Problem analysis and Project planning:

1.1 Introduction

Banking is one of the common and day to day attribute of life. Nowadays it is totally different from that existed a few years ago banking has become completely computerized new facilities such as credit cards, debit cards & ATM has been introduced. ATM is automatic teller machine which is basically used to withdraw money from an account.

1.2 Objectives

The objective of this software is similar to ATM software installed in ATM center. It should first validate the pin in the ATM card. Then the type of transaction is enquired and the information from the customer is validated. If it is a withdrawal the amount is asked. After the money is delivered the transaction just made is updated in the database where the customer's information is stored.

1.3 Scope

The scope of the project is to design an ATM system that will help in completely automatic banking this software is going to be designed for withdrawal and deposit of money and register the transaction in the database where the customer's information is stored.

1.4 Problem Statement

ATM is another type of banking where the most frequently type of transaction made is withdrawal. A user may withdraw as much as many amount as he wants until his account holds a sum greater than his withdrawal amount. ATM is completely automated and there is no necessity of the ATM center being placed at the bank itself. It can be placed in the shopping malls, airports, railway stations etc. This ATM system can use any kind of interface. But it should be user friendly and not confusing. Help manuals should be provided in case any customer has problem working with the software. The system will retain information on the entire customer who has necessity rights to access the service.

It will contain the balance amount in the account, rate of interest, any special allowance for that customer and most of all pin number of the customer.

The ATM system should be compatible with any kind of database such as MS-ACCESS, DB2,ORACLE, SQL, SERVER etc. the emphasis here is on consistency. Some customer could

have availed some special offers on his ATM cards. So this must be taken care of and the appropriate data should be dealt with. The ATM should provide easy access to the data for the customer. It should also have a highly secure interface so that one can take money one behalf of others. So the security is one of the main aspects in ATM.

The problem statement is the initial starting point for a project. It is basically a one to three page statement that everyone on the project agrees with that describes what will be done at a high level. The problem statement is intended for a broad audience and should be written in non technical terms. It helps the non-technical and technical personnel communicate by providing a description of a problem. It doesn't describe the solution to the problem.

The input to requirement engineering is the problem statement prepared by customer. It may give an overview of the existing system along with broad expectations from the new system.

The first phase of requirements engineering begins with requirements elicitation i.e. gathering of information about requirements. Here, requirements are identified with the help of customer and existing system processes. So from here begins the preparation of problem statement. So, basically a problem statement describes what needs to be done without describing how.

PRATICAL SIGNIFICANCE:

To analyze the basic requirement of software product and to generate problem statement and to analyze the bounded scope of the software product.

RELEVANT AFFECTIVE DO MAIN RELATED OUTCOMES:-

1. Follow precaution measured.
2. Demonstrate working as a leader is a team member.
3. Follow ethical practices.

MINIMUM THEROTICAL BACKGROUND:-

A problem statement is clear consist description of the issue that needs to be addressed by a problem solving team. It is used to center and focus the team at the beginning keep the team on track during the effects delivered on outcomes that solves the problem statement.

□ How to write problem statement: - Write down your vision to in order to decided what must, be done while, solving. The pattern it’s important to understand the vision. It’s an initial starting point for a project. A problem statement expressed the words that will be used to keep the efforts focused and it should represent a solvable problem. The file can be used to spark the discussion about the problem.

1) How to write a program statement.

2) How to get it started.

3) Who does the problem affect?

4) What are the boundaries of the problem?

5) When the issue does occurs.

6) Where is the issue programming.

7) Why it is important to fix the program.

PROBLEM STATEMENTS FOR A PROJECT

1. Describe ideal state of affairs.
2. Explain your problem
3. Explain your problem financial cost
4. Backup your assertions
5. Purpose your solution
6. Explain the benefits of solution
7. Conclude by summarizing the problems and solutions
8. Write down the thesis statements

Result:

The problem statement was written successfully by following the steps described above.

EXNO: 2	Prepare broad SRS (Software Requirement Specification) for the above selected projects

Aim:

To Prepare broad SRS (Software Requirement Specification) for the above selected projects.

Software Requirement Analysis:

1. Introduction

The software ATM Excl 3.0TM version1.0 is to be developed for Automated Teller Machines (ATM). An automated teller machine (ATM) is computerized telecommunications device that provides a financial institution's customers a secure method of performing financial transactions, in a public space without the need for a human bank teller. Through ATM Excl 3.0TM , customers interact with a user-friendly interface that enables them to access their bank accounts and perform various transactions.

1.1 Purpose

This SRS defines External Interface, Performance and Software System Attributes requirements of ATM Excl 3.0TM. This document is intended for the following group of people:- Developers for the purpose of maintenance and new releases of the software.

1.2 Scope

This document applies to Automated Teller Machine software ATM 3.0TM. This software facilitates the user to perform various transaction in his account without going to bank. This software offers benefits such cash withdrawals, balance transfers, deposits, inquiries, credit card advances and other banking related operations for customers. It also allows the administrator to fix the tariffs and rules as and when required. The software takes as input the login Id and the bank account number of the user for login purposes. The outputs then comprise of an interactive display that lets the user select the desirable function that he wants to perform.. The software is expected to complete in duration of six months and the estimated cost is Rs18 lakhs.

1.3 Overview

Section 1.0 discusses the purpose and scope of the software.

Section 2.0 describes the overall functionalities and constraints of the software and user characteristics.

Section 3.0 details all the requirements needed to design the software.

2. The Overall Description

2.1 Product Perspective

A single functional unit consisting of various sub-components. This software allows the user to access their bank accounts remotely through an ATM without any aid of human bank teller. This software also allows the perform various other functions apart from just accessing his bank account such as mobile bill clearings etc. Some of its hardware components are cassettes, memory, drives, dispensers i.e. for receipts and cash, a card reader, printer, switches, a console, a telephone dialer port, a networking port and disks. The ATM communicates with the bank's central server through a dialup communication link. The Memory of the system shall be 20MB. The Cassette capacity shall be at least 2000 notes.

2.2 Product Functions

The major functions that ATM Excel 3.0TM performs are described as follows:-

Language Selection:- After the user has logged in, the display provides him with a list of languages from which he can select any one in order to interact with the machine throughout that session. After the language selection the user is prompted with an option that whether he wants the selected language to be fixed for future use so that he is not offered with the language selection menu in future thus making the transaction a bit faster. User also has the freedom to switch to a different language mentioned in the list in between that session. **Account Maintenance:-** The various functions that a user can perform with his account are as follows:- Account Type:-The user has the freedom to select his account type to which all the transactions are made.

2.3 User Characteristics

There are different kind of users that will be interacting with the system. The intended user of the software are as follows:-

- User A: A novice ATM customer. This user has little or no experience with electronic means of account management and is not a frequent user of the product. User A will find the product easy to use due to simple explanatory screens for each ATM function. He is also assisted by an interactive teaching mechanism at every step of the transaction, both with the help of visual and audio help sessions.
- User B: An experienced customer. This user has used an ATM on several occasions before and does most of his account management through the ATM. There is only a little help session that too at the beginning of the session thus making the transaction procedure more faster.
- Maintenance Personnel: A bank employee. This user is familiar with the functioning of the ATM. This user is in charge of storing cash into the ATM vault and repairing the

ATM in case of malfunction. This user is presented with a different display when he logs in with the administrator's password and is provided with options different from that of normal user. He has the authority to change or restrict various features provided by the software in situations of repairing.

2.4 Constraints

The major constraints that the project has are as follows:-

- The ATM must service at most one person at a time.
- The number of invalid pin entries attempted must not exceed three. After three unsuccessful login attempts, the card is seized/blocked and need to be unlocked by the bank.
- The simultaneous access to an account through both, the ATM and the bank is not supported.
- The minimum amount of money a user can withdraw is Rs 100/- and the maximum amount of money a user can withdraw in a session is Rs.10,000/- and the maximum amount he can withdraw in a day is Rs 20,000/-

Before the transaction is carried out, a check is performed by the machine to ensure that a minimum amount of Rs 1000/- is left in the user‘s account after the withdrawal failing which the withdrawal is denied.

- The minimum amount a user can deposit is Rs 100/- and the maximum amount he can deposit is Rs 10,000/-. A user can select only that cellular operator for mobile bill clearings that is supported by the bank.
- The software requires a minimum memory of 20GB
- The database used should be Oracle7.0.
- There shall be a printer installed with the machine to provide the user with the printed statement of the transaction.
- For voice interactions, speakers should also be there to accompany the machine.

2.5 Assumptions and Dependencies

The requirements stated in the SRS could be affected by the following factors:

- One major dependency that the project might face is the changes that need to be incorporated with the changes in the bank policies regarding different services. As the policies changes the system needs to be updated with the same immediately. A delay in doing the same will result to tremendous loss to the bank. So this should be changed as and when required by the developer.
- Another constraint relating to the operating environment is that we are specific to Oracle Database.

- The project could be largely affected if some amount is withdrawn from the user‘s account from the bank at the same time when someone is accessing that account through the ATM machine. Such a condition shall be taken care of.
- At this stage no quantities measures are imposed on the software in terms of speed and memory although it is implied that all functions will be optimized with respect to speed and memory.

It is furthermore assumed that the scope of the package will increase considerably in the future.

3. External Interface Requirements

3.1.1 User Interface Requirements

The interface provided to the user should be a very user-friendly one and it should provide an optional interactive help for each of the service listed. The interface provided is a menu driven one and the following screens will be provided:-

1. A login screen is provided in the beginning for entering the required username/pin no. and account number.

2. An unsuccessful login leads to a reattempt(maximum three) screen for again entering the same information. The successful login leads to a screen displaying a list of supported languages from which a user can select any one.

3. In case of administrator, a screen will be shown having options to reboot system, shut down system, block system, disable any service.

4. In case of reboot/ shut down, a screen is displayed to confirm the user‘s will to reboot and also allow the user to take any backup if needed.

5. In case of blocking system, a screen is provided asking for the card no. By entering the card no of a particular user, system access can be blocked for him.

6. Administrator is also provided with a screen that enables him to block any service provided to the user by enter in the name of the service or by selecting it from the list displayed.

7. After the login, a screen with a number of options is then shown to the user. It contains all the options along with their brief description to enable the user to understand their functioning and select the proper option.

8. A screen will be provided for user to check his account balance.

9. A screen will be provided that displays the location of all other ATMs of same bank elsewhere in the city.

10. A screen will be provided for the user to perform various transactions in his account.

The following reports will be generated after each session dealt with in the machine:-

1. The login time and logout time along with the user‘s pin no and account number is registered in the bank‘s database.

2. The ATM‘s branch ID through which the session is established is also noted down in the bank‘s database.

3. Various changes in the user‘s account after the transactions, if any, are reported in the database.

4. A printed statement is generated for the user displaying all the transactions he performed.

Other various user interface requirements that need to be fulfilled are as follows:-

- The display screen shall be of 10" VGA color type.
- The display screen shall have 256 color resolution.
- The display screen shall also support touch screen facility.
- The speakers shall support Yamaha codes.
- The keypad shall consist of 16 tactile keys.
- There shall be 8 tactile function keys.
- The keyboard will be weather resistant.
- The transaction receipt shall be 3.1" × 6".
- The statement receipt shall be 4.2" × 12".
- The deposit envelopes shall be 9" long and 4" wide.

3.1.2 Hardware Interface Requirements

There are various hardware components with which the machine is required to interact. Various hardware interface requirements that need to be fulfilled for successful functioning of the software are as follows:-

- The ATM power supply shall have a 10/220 V AC manual switch.
- The ATM card should have the following physical dimensions:- o Width - 85.47mm-85.72mm o Height - 53.92mm-54.03mm o Thickness - 0.76mm+0.08mm
- The card reader shall be a magnetic stripe reader
- The card reader shall have Smart card option.
- The slot for a card in thye card reader may include an extra indentation for the embossed area of the card. In effect it acts as a polarization key and may be used to aid the correct insertion orientation of the card. This is an additional characteristic to the magnetic field sensor which operates off the magnetic stripe and is used to open a mechanical gate on devices such as ATMs.
- There shall be a 40 column dot matrix receipt printer.
- There shall be a 40 column dot matrix statement printer.
- The receipt dispenser shall be a maximum of 4" width and 0.5" thickness.
- The statement dispenser shall be a maximum of 5" width and 0.5" thickness.

- The envelope depository shall be a maximum of 4.5" width, 10" length and 0.5" thickness.
- Screen resolution of at least 800X600-required for proper and complete viewing of screens. Higher resolution would not be a problem.

3.1.3 Software Interface Requirements

In order to perform various different functions, this software needs to interact with various other software. So there are certain software interface requirements that need to be fulfilled which are listed as follows:- The transaction management software used to manage the transaction and keep track of resources shall be BMS version 2.0.

- The card management software used to verify pin no and login shall be CMS version 3.0.
- Yamaha codes 367/98 for active speakers.
- The database used to keep record of user accounts shall be Oracle version7.0.

3.1.4 Communication Interface Requirements

The machine needs to communicate with the main branch for each session for various functions such as login verification, account access etc. so the following are the various communication interface requirements that are needed to be fulfilled in order to run the software successfully:- The system will employ dial-up POS with the central server for low cost communication.

- The communication protocol used shall be TCP/IP.
- Protocol used for data transfer shall be File Transfer Protocol.(FTP)

4. System Features

1. Remote Banking and Account Management Description

- The system is designed to provide the user with the facility of remote banking and perform various other functions at an interface without any aid of human bank teller. The functioning of the system shall be as follows:-
- At the start, the user is provided with a log in screen and he is required to enter his PIN NO. and Account details which are then verified by the machine. In case of an unsuccessful attempt a user is asked again for his credentials but the maximum number of attempt given to the user is limited to 3 only, failing which his card is blocked and need to be unblocked by the bank for any future use. After a successful log in, the user is presented with a list of language. The user can select anyone in the list for interaction with the machine for the entire session.

- After the language selection the user is also asked whether he wants to fix that language for future use also so that he is never asked for language in future. In addition there is also a facility for the user to switch to any other language during that session.

- After the language selection, the user is directed towards a main page that displays a set of options/services along with their brief description, enabling the user to understand their functioning. The user can select any of the listed option and can continue with the transaction.

- The machine also provides the user with a number of miscellaneous services such as:

- The machine lists a set of operators that are supported by the bank. A user can clear off his pending mobile phone bills be selecting his operator.
- The machine also has the facility to display a map that marks the location of other ATMs of the same bank in the city. This may help the user to look for the ATM nearest to his destination.
- At any moment if the user wants to abort the transaction, he is provided with an option to cancel it. Just by pressing the abort button he can cancel all the changes made so far and can begin with a new transaction.
- After the user is finished with his work, for security purpose, he is required to log out and then take his card out of the slot.

Validity Checks

- In order to gain access to the system, the user is required to enter his/her correct user id/pin no and account no failing which his card may be blocked.
- The user can access only one account at a time and can enter only one account no.
- Also if the user is an administrator, he is required to enter his login id in order to access and change the facilities provided by the system.

Sequencing Information

The information about the users and their account should be entered into the database prior to any of the transactions and the backup be maintained for all account information

Error Handling/ Response to Abnormal Situations

If any of the above validation/sequencing flow does not hold true, appropriate error messages will be prompted to the user for doing the needful.

Receipt Generation

After etch transaction user has performed, a receipt is generated that contains all the information about the transaction. The format of the generated receipt.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The following list provides a brief summary of the performance requirements for the software:

5.1.1 Capacity

The ATM shall provide customers a 24 hour service.

5.1.2 Dynamic requirements

- The card verification time must not exceed 0.8 sec. under normal server workload and 1 sec. under peak server workload.
- The pin number verification time must not exceed 0.3 sec. under normal server workload and 0.5 sec. under peak server workload.
- Account balance display time must not exceed 2 sec. under normal server workload and 3 sec. under peak server workload.
- Account balance transfer time must not exceed 3 sec. under normal server workload and 4 sec. under peak server workload.
- Cash withdrawal transaction time must not exceed 4 sec. under normal server workload and 5 sec. under peak server workload.
- Deposit transaction time after insertion of the deposit envelope must not exceed 5 sec. under normal server workload and 6 sec. under peak server workload.
- Receipt printing time after must not exceed 3 sec. under normal server and peak server workload.
- Touch screen and button response time must not exceed 5000ms.
- Credit card advance time must not exceed 6 sec. under normal traffic and server and peak traffic and server workload.

5.1.3 Quality

The primary objective is to produce quality software. As the quality of a piece of software is difficult to measure quantitatively, the following guidelines will be used when judging the quality of the software:

1. Consistency – All code will be consistent with respect to the style.

(This is implied when adhering to the standard).

2. Test cases – All functionality will be thoroughly tested

5.2 Software System Attributes

5.2.1 Reliability

- The data communication protocol shall be such that it ensures reliability and quality of data and voice transmission in a mobile environment. For example, CDMA.
- The memory system shall be of non-volatile type.

5.2.2 Availability

- The product will have a backup power supply incase of power failures.
- Any abnormal operations shall result in the shutting down of the system.
- After abnormal shutdown of the ATM, the system shall have to be manually restarted by a maintenance personnel.
- There should be no inconsistency introduced in the account during whose transaction the system is abnormally shut down.

5.2.3 Security

- The system shall be compatible with AIMS security standards.
- The system shall have two levels of security i.e. ATM card and pin verification both authenticated by the CMS software.
- The Encryption standard used during pin transmission shall be triple DES.
- The password shall be 6-14 characters long.
- Passwords shall not contain name of customers as they are easy to be hacked.
- Passwords can contain digit, hyphen and underscore.
- User should be provided with only three attempts for login failing which his card needs to be blocked.
- There shall be a security camera installed near the ATM.
- There shall be a secured cash vault with a combination locking system.
- The product cabinet cover shall be manufactured using Fiber glass for security purposes.

5.2.4 Maintainability

- The system components i.e. modem, memory, disk, drives shall be easily serviceable without requiring access to the vault.
- The system should have the mechanism of self-monitoring periodically in order to detect any fault.
- The system should inform the main branch automatically as soon as it detects any error. The kind of fault and the problem being encountered should also be mentioned by the system automatically.

5.3 Business Rules

The business rules for the software are as follows:

- The Administrator has the authority to fix the rules and regulations and to set or update the policies as and when required.
- The staff at the bank performs the following:
    - Making the entries in the system regarding all the details of the bank account of the user.
    - Keeping the bank account of the user updated as soon as changes are encountered so that the data is in consistent state.
    - Blocking or seizing of the account of user on discovery of any illegal transaction.
    - Unblocking of ATM card that got blocked due to more than three unsuccessful login attempts.
    - Blocking of a lost/stolen ATM card on complaint of the user, only if he presents his verification and a FIR filed for that case.
    - Constantly monitor all the ATMs in the city to check whether any one of them is encountering any fault.
    - Immediately correct any fault discovered in any of the ATM.
    - Maintain the backup of all the accounts for reliability purposes.
    - Rollback all the changes made in an account during whose transaction an ATM
- got abnormal shutdown.
- In case of loss of the ATM card. The user has to lodge a First Investigation Report(FIR) and present its one copy to bank officials for card blocking purposes.
- A log of the following annexures is generated by the system:
- User bank account details.
- Updations made in the user account along with date, time and the changes made.
- Schedule of fixed assets.

6 Other Requirements

None.

Appendix A: Glossary

AIMS - ATM Information Management System.

ATM - An unattended electronic machine in a public place, connected to a data system and related equipment and activated by a bank customer to obtain cash withdrawals and other banking services

Braille- A system of writing and printing for blind or visually impaired people, in which varied arrangements of raised dots representing letters and numerals are identified by touch.

CDMA - Code Division Multiple Access, a reliable data communication protocol.

CMS - Card Management Software developed by KPM Bank.

Dial-Up - A message format for low cost communications.

POS Internet - An interconnected system of networks that connects computers around the world via the TCP/IP protocol.

Smart Card - Card without hardware which stores the user‘s private keys within a tamper proof software guard.

Tactile- Special keyboard designed to aid the visually impaired.

Keyboard

TCP/IP - Transmission Control Protocol/Internet Protocol.

Result:

The SRS was made successfully by following the steps described above.

EXNO: 3	Prepare USE Cases and Draw Use Case Diagram using modelling Tool

Aim:

To Prepare USE Cases and Draw Use Case Diagram using modeling Tool.

Theory:

According to the UML specification a use case diagram is ―a diagram that shows the relationships among actors and use cases within a system.‖ Use case diagrams are often used to:

- Provide an overview of all or part of the usage requirements for a system or organization in the form of an essential model or a business model
  - Communicate the scope of a development project
  - Model your analysis of your usage requirements in the form of a system use case model
- Use case models should be developed from the point of view of your project stakeholders and not from the (often technical) point of view of developers. There are guidelines for:
  - Use Cases
  - Actors
  - Relationships
- System Boundary Boxes

1. Use Cases

A use case describes a sequence of actions that provide a measurable value to an actor. A use case is drawn as a horizontal ellipse on a UML use case diagram.

1. Use Case Names Begin With a Strong Verb
2. Name Use Cases Using Domain Terminology
3. Place Your Primary Use Cases In The Top-Left Corner Of The Diagram
4. Imply Timing Considerations By Stacking Use Cases.

2. Actors

An actor is a person, organization, or external system that plays a role in one or more interactions with your system (actors are typically drawn as stick figures on UML Use Case diagrams).

1. Place Your Primary Actor(S) In The Top-Left Corner Of The Diagram
2. Draw Actors To The Outside Of A Use Case Diagram
3. Name Actors With Singular, Business-Relevant Nouns
4. Associate Each Actor With One Or More Use Cases
5. Actors Model Roles, Not Positions
6. Use <<system>> to Indicate System Actors

7. Actors Don‘t Interact With One Another

8. Introduce an Actor Called ―Time‖ to Initiate Scheduled Events

3. Relationships

There are several types of relationships that may appear on a use case diagram:

- An association between an actor and a use case
- An association between two use cases
- A generalization between two actors
- A generalization between two use cases

Associations are depicted as lines connecting two modeling elements with an optional open headed arrowhead on one end of the line indicating the direction of the initial invocation of the relationship. Generalizations are depicted as a close-headed arrow with the arrow pointing towards the more general modeling element.

1. Indicate An Association Between An Actor And A Use Case If The Actor Appears Within The Use Case Logic

2. Avoid Arrowheads On Actor-Use Case Relationships

3. Apply <<include>> When You Know Exactly When To Invoke The Use Case

4. Apply <<extend>> When A Use Case May Be Invoked Across Several Use Case Steps

5. Introduce <<extend>> associations sparingly

6. Generalize Use Cases When a Single Condition Results In Significantly New Business Logic

7. Do Not Apply <<uses>>, <<includes>>, or <<extends>>

8. Avoid More Than Two Levels Of Use Case Associations

9. Place An Included Use Case To The Right Of The Invoking Use Case

10. Place An Extending Use Case Below The Parent Use Case

11. Apply the ―Is Like‖ Rule to Use Case Generalization

12. Place an Inheriting Use Case Below The Base Use Case

13. Apply the ―Is Like‖ Rule to Actor Inheritance

14. Place an Inheriting Actor Below the Parent Actor

4. System Boundary Boxes

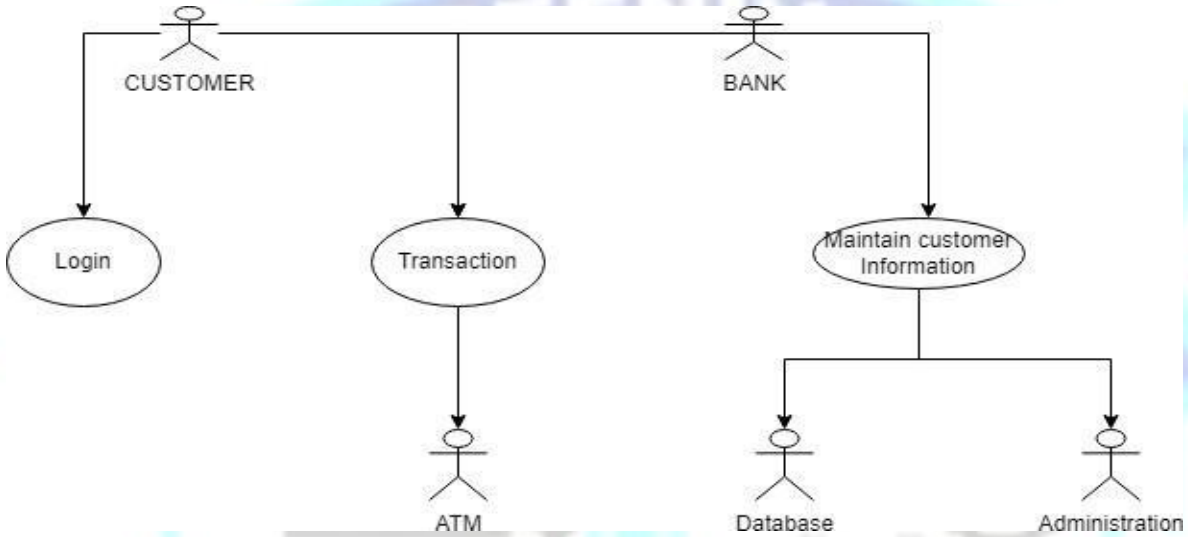
The rectangle around the use cases is called the system boundary box and as the name suggests it indicates the scope of your system – the use cases inside the rectangle represent the functionality that you intend to implement.

1. Indicate Release Scope with a System Boundary Box.

2. Avoid Meaningless System Boundary Boxes.

Creating Use Case Diagrams

we start by identifying as many actors as possible. You should ask how the actors interact with the system to identify an initial set of use cases. Then, on the diagram, you connect the actors with the use cases with which they are involved. If actor supplies information, initiates the use case, or receives any information as a result of the use case, then there should be an association between them.



Procedure (for rational rose):

- Click on the File menu and select New.
- Now from the Dialogue Box that appears, select the language which you want to use for creating your model.
- In the left hand side window of Rational Rose right click on Use Case view‖ and select New>Use Case Diagram.
- Enter the name of new Use Case file in the space provided, and then click on that file name.

You can now use the window that appears on right hand side to draw your Use Case diagram using the buttons provided on the vertical toolbar.

Result:

The Use case diagram was made successfully by following the steps described above.

EXNO: 4	**Develop the activity diagram to represent flow from one activity to another for software**

Aim:

To Develop the activity diagram to represent flow from one activity to another for software.

Theory:

Activity diagrams are typically used for business process modeling, for modeling the logic captured by a single usecase or usage scenario, or for modeling the detailed logic of a business rule. Although UML activity diagrams could potentially model the internal logic of a complex operation it would be far better to simply rewrite the operation so that it is simple enough that you don't require an activity diagram. In many ways UML activity diagrams are the object-oriented equivalent of flow charts and data flow diagrams (DFDs) from structured development. Let's start by describing the basic notation :

- Initial node. The filled in circle is the starting point of the diagram. An initial node isn't required although it does make it significantly easier to read the diagram.
- Activity final node. The filled circle with a border is the ending point. An activity diagram can have zero or more activity final nodes.
- Activity. The rounded rectangles represent activities that occur. An activity may be physical, such as Inspect Forms, or electronic, such as Display Create Student Screen.
- Flow/edge. The arrows on the diagram. Although there is a subtle difference between flows and edges, never a practical purpose for the difference although.
- Fork. A black bar with one flow going into it and several leaving it. This denotes the beginning of parallel activity.
- Join. A black bar with several flows entering it and one leaving it. All flows going into the join must reach it before processing may continue. This denotes the end of parallel processing.
- Condition. Text such as [Incorrect Form] on a flow, defining a guard which must evaluate to true in order to traverse the node.
- Decision. A diamond with one flow entering and several leaving. The flows leaving include conditions although some modelers will not indicate the conditions if it is obvious.
- Merge. A diamond with several flows entering and one leaving. The implication is that one or more incoming flows must reach this point until processing continues, based on any guards on the outgoing flow.

- Partition. If figure is organized into three partitions, it is also called swim lanes, indicating who/what is performing the activities (either the Applicant, Registrar, or System).
- Sub-activity indicator. The rake in the bottom corner of an activity, such as in the Apply to University activity, indicates that the activity is described by a more finely detailed activity diagram.
- Flow final:The circle with the X through it. This indicates that the process stops at this point.

GUIDELINES ASSOCIATED FOR DRAWING AN ACTIVITY DIAGRAM

1. General Guidelines
2. Activities
3. Decision Points
4. Guards
5. Parallel Activities
6. Swimlane Guidelines
7. Action-Object Guidelines

1. General Guidelines

1. Place The Start Point In The Top-Left Corner. A start point is modeled with a filled in circle, using the same notation that UML State Chart diagrams use. Every UML Activity Diagram should have a starting point, and placing it in the top-left corner reflects the way that people in Western cultures begin reading. Figure1, which models the business process of enrolling in a university, takes this approach.

2. Always Include an Ending Point. An ending point is modeled with a filled in circle with a border around it, using the same notation that UML State Chart diagrams use is interesting because it does not include an end point because it describes a continuous process – sometimes the guidelines don‘t apply.

3. Flowcharting Operations Implies the Need to Simplify. A good rule of thumb is that if an operation is so complex you need to develop a UML Activity diagram to understand it that you should consider refactoring it.

2. Activities

An activity, also known as an activity state, on a UML Activity diagram typically represents the invocation of an operation, a step in a business process, or an entire business process.

1. Question Black Hole Activities. A black hole activity is one that has transitions into it but none out, typically indicating that you have either missed one or more transitions.

2. Question ―Miracle‖ Activities. A miracle activity is one that has transitions out of it but none into it, something that should be true only of start points.

3. Decision Points

A decision point is modeled as a diamond on a UML Activity diagram.

1. Decision Points Should Reflect the Previous Activity. In figure1 we see that there is no label on the decision point, unlike traditional flowcharts which would include text describing the actual decision being made, we need to imply that the decision concerns

2. whether the person was enrolled in the university based on the activity that the decision point follows. The guards, depicted using the format [description], on the transitions leaving the decision point also help to describe the decision point.

3. Avoid Superfluous Decision Points. The Fill Out Enrolment Forms activity in FIGURE1 includes an implied decision point, a check to see that the forms are filled out properly, which simplified the diagram by avoiding an additional diamond.

4. Guards

A guard is a condition that must be true in order to traverse a transition.

1. Each Transition Leaving a Decision Point Must Have a Guard

2. Guards Should Not Overlap. For example guards such as x <0, x = 0, and x > 0 are consistent whereas guard such as x <= 0 and x >= 0 are not consistent because they overlap – it isn‘t clear what should happen when x is 0.

3. Guards on Decision Points Must Form a Complete Set. For example, guards such as x < 0 and x >0 are not complete because it isn‘t clear what happens when x is 0.

4. Exit Transition Guards and Activity Invariants Must Form a Complete Set. An activity invariant is a condition that is always true when your system is processing an activity.

5. Apply a [Otherwise] Guard for ―Fall Through‖ Logic.

6. Guards Are Optional. It is very common for a transition to not include a guard, even when an activity includes several exit transitions.

5. Parallel Activities

It is possible to show that activities can occur in parallel, as you see in depicted using two parallel bars. The first bar is called a fork, it has one transition entering it and two or more transitions leaving it. The second bar is a join, with two or more transitions entering it and only one leaving it.

1. A Fork Should Have a Corresponding Join. In general, for every start (fork) there is an end (join). In UML 2 it is not required to have a join, but it usually makes sense.

2. Forks Have One Entry Transition.

3. Joins Have One Exit Transition

4. Avoid Superfluous Forks depicts a simplified description of the software process of enterprise architectural modelling a part of the Enterprise Unified Process (EUP). There is significant opportunity for parallelism in this process, in fact all of these activities could happen in parallel, but forks were not introduced because they would only have cluttered the diagram.

6. Swim lane Guidelines

A swim lane is a way to group activities performed by the same actor on an activity diagram or to group activities in a single thread includes three swim lanes, one for each actor.

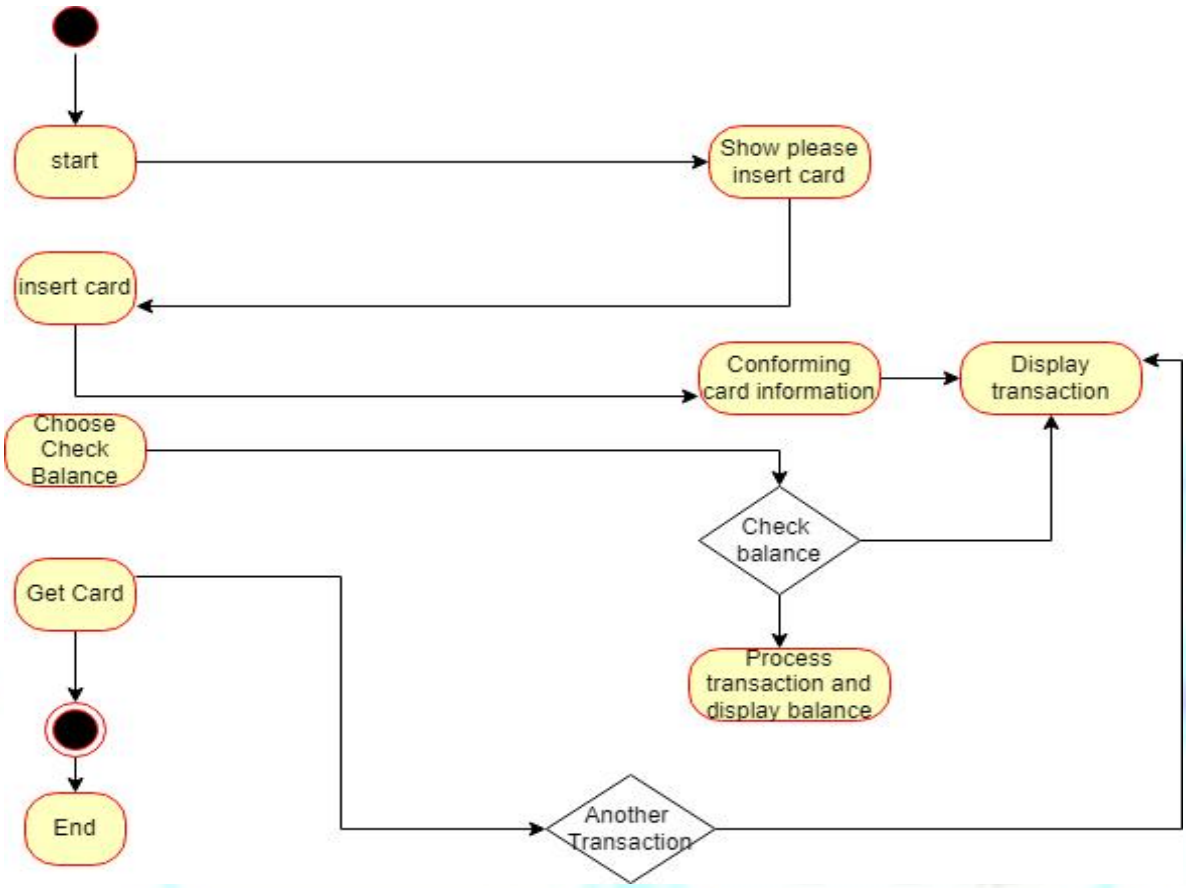
Apply Swim Lanes To Linear Processes. A good rule of thumb is that swim lanes are best applied to linear processes, unlike the one depicted.

1. Have Less Than Five Swim lanes.
2. Consider Swim areas For Complex Diagrams.
3. Swim areas Suggest The Need to Reorganize Into Smaller Activity Diagrams.
4. Consider Horizontal Swim lanes for Business Processes. you see that the

Swim lanes are drawn horizontally, going against common convention of drawing them vertically.

5. Action-Object Guidelines
6. Activities act on objects, In the strict object-oriented sense of the term an action object is a system object, a software construct. In the looser, and much more useful for business application modeling, sense of the term an action object is any sort of item. For example in the Expense Form action object is likely a paper form.

1. Place Shared Action Objects on Swim lane Separators.
2. When An Object Appears Several Time Apply State Names.
3. State Names Should Reflect the Lifecycle Stage of an Action Object.
4. Show Only Critical Inputs and Outputs.
5. Depict Action Objects As Smaller Than Activities.



Result:

The activity diagram was made successfully by following the steps described above.

EXNO: 5	Develop data Designs using DFD Decision Table & ER Diagram

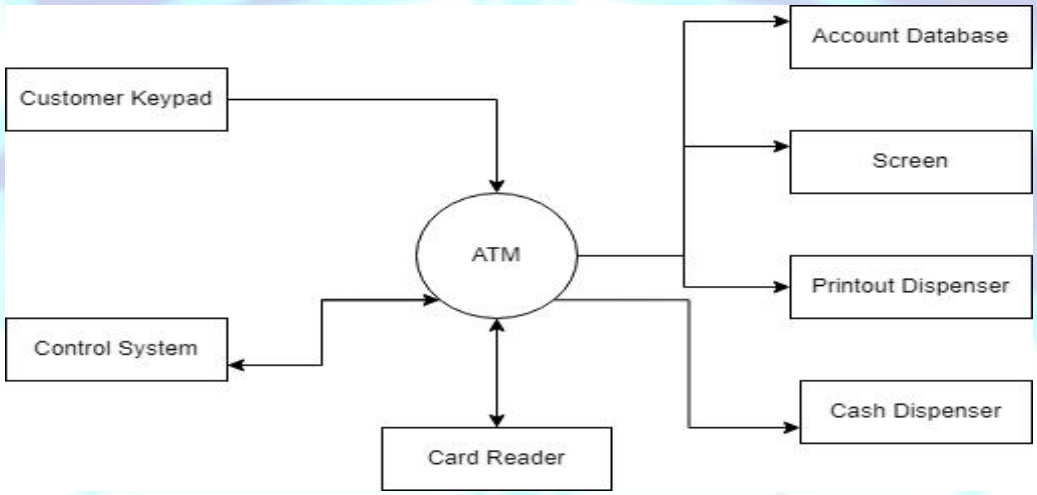
Aim:

To Develop data Designs using DFD Decision Table & ER Diagram.

DFD of an ATM system:

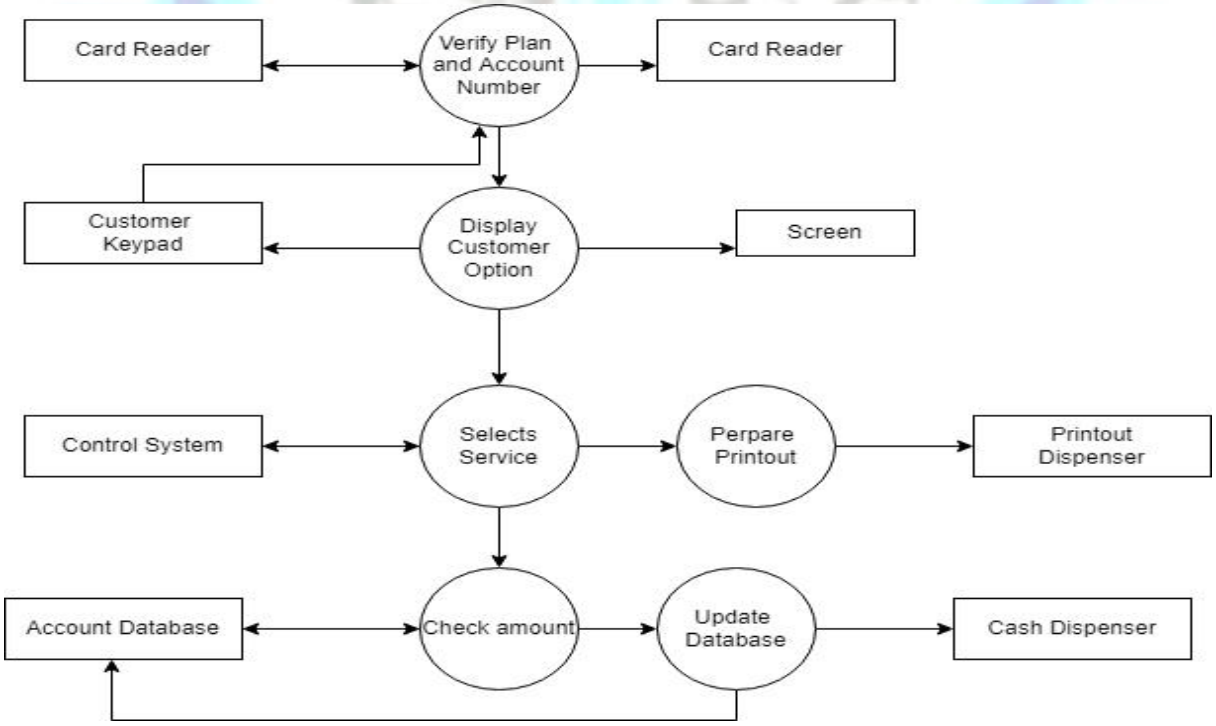
Level 0 DFD :

This level is also known as Context Level DFD. At this level, only the interacting inputs and outputs with a system are described. The DFD of this level is shown below:



Level 1 DFD:

At this level, more detailed information is given about the processing of the ATM system. The DFD of this level is shown below:



ER Diagram:

The ERD or ER Diagram for ATM System shows the system entity relationships in each entity and their supposed functions in each relationship. Now here's the sample ER Diagram of ATM Management System.

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. VII Resources required Sr No Instrument Specification Quantity Remarks

1 Computer System Any Desktop PC with attached 10 No. Whichever is available HardDisk 2 Any UML Software - 1 No. - VIII

Procedure

1. Select Diagram > New from the application toolbar.
2. In the New Diagram window, select Data Flow Diagram.
3. Click Next.
4. Enter the diagram name and description. The Location field enables you to select a model to store the diagram.
5. Click OK.

Precautions

1. Change the requirement carefully.
2. Save changes if required under the guidance of teacher.

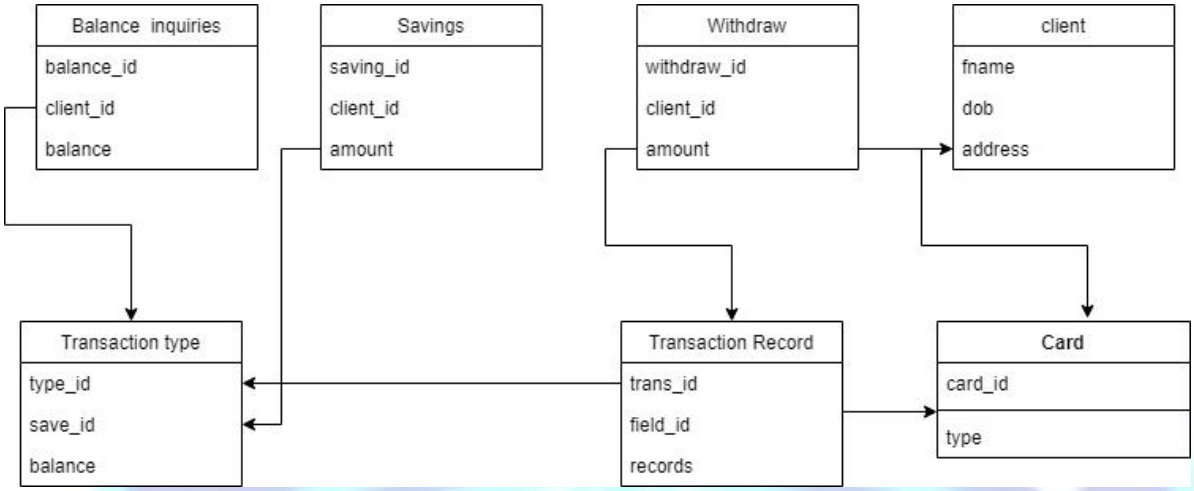
Description

Data Flow Diagrams are either Logical or Physical. Logical DFD - This type of DFD concentrates on the system process, and flow of data in the system.

for example in a Banking software system, how data is moved between different entities. Physical DFD - This type of DFD shows how the data flow is actually implemented in the system. It is more specific and close to the implementation. DFD Components DFD can represent Source, destination, storage and flow of data using the following set of components.

Entities

Entities are source and destination of information data. Entities are represented by a rectangles with their respective names. Process - Activities and action taken on the data are represented by Circle or Round-edged rectangles. Data Storage - There are two variants of data storage - it can either be represented as a rectangle with absence of both smaller sides or as an open-sided rectangle with only one side missing. Data Flow - Movement of data is shown by pointed arrows. Data movement is shown from the base of arrow as its source towards head of the arrow as destination.



Result:

The Develop data Designs using DFD Decision Table & ER Diagram was made successfully by following the steps described above.

EXNO: 6	Draw class diagram, sequence diagram, Collaboration Diagram, State Transition Diagram for the assigned project

Aim:

To Draw class diagram, sequence diagram, Collaboration Diagram, State Transition Diagram for the assigned project.

Theory:

UML sequence diagrams model the flow of logic within the system in a visual manner, enabling the user both to document and validate the logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system. Sequence diagrams, along with class diagrams and physical data models are the most important design-level models for modern application development.

Sequence diagrams are typically used to model:

1. Usage scenarios. A usage scenario is a description of a potential way the system is used. The logic of a usage scenario may be part of a use case, perhaps an alternate course. It may also be one entire pass through a use case, such as the logic described by the basic course of action or a portion of the basic course of action, plus one or more alternate scenarios. The logic of a usage scenario may also be a pass through the logic contained in several use cases. For example, a student enrolls in the university, and then immediately enrolls in three seminars.

2. The logic of methods. Sequence diagrams can be used to explore the logic of a complex operation, function, or procedure. One way to think of sequence diagrams, particularly highly detailed diagrams, is as visual object code.

3. The logic of services. A service is effectively a high-level method, often one that can be invoked by a wide variety of clients. This includes web-services as well as business transactions implemented by a variety of technologies such as CICS/COBOL or CORBA-compliant object request brokers (ORBs)

The dashed lines hanging from the boxes are called object lifelines, representing the life span of the object during the scenario being modeled. The long, thin boxes on the lifelines are activation boxes, also called method-invocation boxes, which indicate processing is being performed by the target object/class to fulfill a message.

How to Draw Sequence Diagrams

Sequence diagramming really is visual coding, even when you are modeling a usage scenario via a system-level sequence diagram. While creating a sequence diagram,start by

identifying the scope of what you are trying to model. You should typically tackle small usage scenarios at the system level or a single method/service at the detailed object level.

You should then work through the logic with at least one more person, laying out classifiers across the top as you need them. . The heart of the diagram is in the messages, which you add to the diagram one at a time as you work through the logic. You should rarely indicate return values, instead you should give messages intelligent names which often make it clear what is being returned. It is interesting to note that as you sequence diagram you will identify new responsibilities for classes and objects, and, sometimes, even new classes. The implication is that you may want to update your class model appropriately, agile modelers will follow the practice Create Several Models in Parallel, something that CASE tools will do automatically. Remember, each message sent to a class invokes a static method/operation on that class each message sent to an object invokes an operation on that object.

Regarding style issues for sequence diagramming, prefer drawing messages going from left-toright and return values from right-to-left, although that doesn‘t always work with complex objects/classes. Justify the label on messages and return values, so they are closest to the arrowhead. Also prefer to layer the sequence diagrams: from left-to-right. indicate the actors, then the controller class(es), and then the user interface class(es), and, finally, the business class(es). During design, you probably need to add system and persistence classes, which you should usually put on the right-most side of sequence diagrams. Laying your sequence diagrams in this manner often makes them easier to read and also makes it easier to find layering logic problems, such as user interface classes directly accessing persistence .

Design Documentation

1. Login

1.1 Brief description:

This use case describes how the user logs into the System.

1.2 Flow of events:

1.2.1 Basic flow:

This use case starts with the actor wishes to log in to the ATM System.

1. The system requests the user to enter the name and PIN.
2. The actor enters the name and PIN.
3. The system validates the name and the PIN and logs the user into the system.

1.2.2 Alternative flow:

1. If the user enters the wrong name and the PIN then the system displays an error message.

2. The actor can either return to the basic flow or cancel login at which point use case ends.

1.3 Pre conditions:

None

1.4 Post conditions:

User will perform corresponding transaction.

2. Transaction

2.1 Brief description:

This describes the transaction that the user is doing.

2.2 Flow of events:

2.2.1 Basic flow:

This use case starts after the user has logged on to the system.

1. The system requests the user to enter the type of transaction of either withdrawal or deposit and asks for customer information.

2. The actor enters the type of transaction and the customer information.

3. The system displays the corresponding transaction screen.

2.2.2 Alternative flow:

If the customer enters any wrong information then the system displays an error message.

2.3 Pre Condition:

The user logs on to the system.

2.4 Post Condition:

Based on the transaction he gets the transaction screen.

3. Maintain Information about Customer

3.1 Brief description:

This describes how administrator takes care of customer information.

3.2 Flow of events:

3.2.1 Basic flow:

This use case starts after the administrator has logged into the system.

1. The system asks the administrator whether he wants to add or delete customer information.

2. The administrator then enters the type of maintenance.

3.2.2 Alternative flow:

None

3.3 Pre Condition:

The administrator logs on to the system before this use case begin.

3.4 Post Condition:

Administrator gets the corresponding maintenance screen according to his choice.

3.2.1.1 Adding Customer

3.2.1.1.1 Basic flow:

1. This use case starts when the administrator has chosen to add customer's information.
2. The system asks the administrator to enter customer information.
3. The administrator enters the customer information.
4. The system displays the updated information.

3.2.1.1.2 Alternative flow:

If the administrator enters any wrong information the system displays an error message.

3.2.1.2 Deleting Customer

3.2.1.2.1 Basic flow:

1. This use case starts when the administrator has chosen to delete an existing customer from the system.
2. The system asks the administrator to enter the customer information.
3. Administrator enters the corresponding user information.
4. The system then displays updated results.

3.2.1.2.2 Alternative flow:

If the administrator has entered any wrong information then the system displays administrator error message.

3.2.1.3 Updating an existing Customer account

3.2.1.3.1 Basic flow:

1. This use case starts when the administrator has chosen to update the customer's information.
2. The system asks the administrator to enter the customer information.
3. The administrator enters the customer information.
4. The system displays the updated information.

3.2.1.3.2 Alternative flow:

If the administrator has entered any wrong information then the system displays administrator error message.

COLLABORATION DIAGRAM:

Collaboration diagrams are relatively easy to draw. They show the relationship between objects and the order of messages passed between them. The objects are listed as icons and arrows indicate the messages being passed between them. The numbers next to the messages are called sequence numbers. As the name suggests, they show the sequence of the messages as they

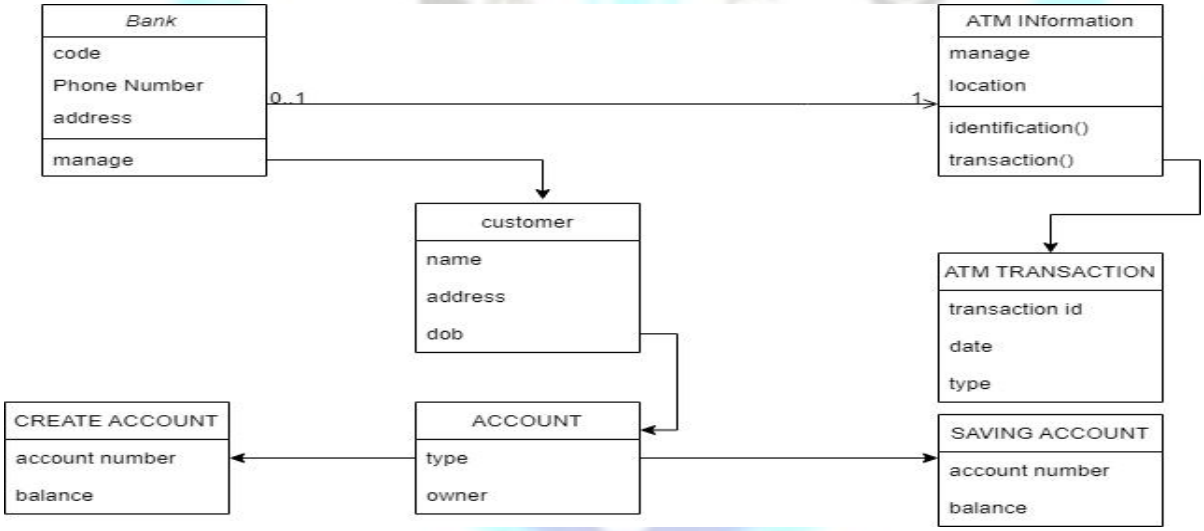
are passed between the objects. There are many acceptable sequence numbering schemes in UML. A simple 1, 2, 3... format can be used, as the example below shows, or for more detailed and complex diagrams.

User Characteristics

There are different kind of users that will be interacting with the system. The intended user of the software are as follows:-

User A: A novice ATM customer. This user has little or no experience with electronic means of account management and is not a frequent user of the product. User A will find the product easy to use due to simple explanatory screens for each ATM function. He is also assisted by an interactive teaching mechanism at every step of the transaction, both with the help of visual and audio help sessions. User B: An experienced customer. This user has used an ATM on several occasions before and does most of his account management through the ATM. There is only a little help session that too at the beginning of the session thus making the transaction procedure more faster. Maintenance Personnel: A bank employee. This user is familiar with the functioning of the ATM. This user is in charge of storing cash into the ATM vault and repairing the ATM in case of malfunction. This user is presented with a different display when he logs in with the administrator's password and is provided with options different from that of normal user. He has the authority to change or restrict various features provided by the software in situations of repairing.

CLASS DIAGRAM:



Constraints:

The major constraints that the project has are as follows:-

The ATM must service at most one person at a time. The number of invalid pin entries attempted must not exceed three. After three unsuccessful login attempts, the card is seized/blocked and

need to be unlocked by the bank. The simultaneous access to an account through both, the ATM and the bank is not supported.

The minimum amount of money a user can withdraw is Rs 100/- and the maximum amount of money a user can withdraw in a session is Rs.10,000/- and the maximum amount he can withdraw in a day is Rs 20,000/- fore the transaction is carried out, a check is performed by the machine to ensure that a minimum amount of Rs 1000/- is left in the user's account after the withdrawal failing which the withdrawal is denied. The minimum amount a user can deposit is Rs 100/- and the maximum amount he can deposit is Rs 10,000.

Assumptions and Dependencies

The requirements stated in the SRS could be affected by the following factors: One major dependency that the project might face is the changes that need to be incorporated with the changes in the bank policies regarding different services. As the policies changes the system needs to be updated with the same immediately. A delay in doing the same will result to tremendous loss to the bank. So this should be changed as and when required by the developer.

The project could be largely affected if some amount is withdrawn from the user's account from the bank at the same time when someone is accessing that account through the ATM machine. Such a condition shall be taken care of.

At this stage no quantities measures are imposed on the software in terms of speed and memory although it is implied that all functions will be optimized with respect to speed and memory. It is furthermore assumed that the scope of the package will increase considerably in the future.

User Interface Requirements

The interface provided to the user should be a very user-friendly one and it should provide an optional interactive help for each of the service listed. The interface provided is a menu driven one and the following screens will be provided:-

1. A login screen is provided in the beginning for entering the required username/pin no. and account number.

2. An unsuccessful login leads to a reattempt(maximum three) screen for again entering the same information. The successful login leads to a screen displaying a list of supported languages from which a user can select any one.

3. In case of administrator, a screen will be shown having options to reboot system, shut down system, block system, disable any service.

4. In case of reboot/ shut down, a screen is displayed to confirm the user's will to reboot and also allow the user to take any backup if needed.

5. In case of blocking system, a screen is provided asking for the card no. By entering the card noof a particular user, system access can be blocked for him.

6. Administrator is also provided with a screen that enables him to block any service provided to the user by entering the name of the service or by selecting it from the list displayed.

7. After the login, a screen with a number of options is then shown to the user. It contains all the options along with their brief description to enable the user to understand their functioning and select the proper option.

8. A screen will be provided for user to check his account balance.

9. A screen will be provided that displays the location of all other ATMs of same bank elsewhere in the city.

10. A screen will be provided for the user to perform various transactions in his account.

The following reports will be generated after each session dealt with in the machine:-

1. The login time and logout time along with the user's pin no and account number is registered in the bank's database.

2. The ATM's branch ID through which the session is established is also noted down in the bank's database.

3. Various changes in the user's account after the transactions, if any, are reported in the database.

4. A printed statement is generated for the user displaying all the transactions he performed.

Other various user interface requirements that need to be fulfilled are as follows:-

The display screen shall have 256 color resolution.

The display screen shall also support touch screen facility.

The speakers shall support Yamaha codes.

The keypad shall consist of 16 tactile keys.

There shall be 8 tactile function keys.

The keyboard will be weather resistant. The transaction receipt shall be 3.1" × 6".

The statement receipt shall be 4.2" × 12". The deposit envelopes shall be 9" long and 4" wide

Software Interface Requirements

In order to perform various different functions, this software needs to interact with various other software. So there are certain software interface requirements that need to be fulfilled which are listed as follows:-

The transaction management software used to manage the transaction and keep track of resources shall be BMS version 2.0. The card management software used to verify pin no and login shall be CMS version 3.0. Yamaha codes 367/98 for active speakers. The database used to keep record of user accounts shall be Oracle version7.0.

Communication Interface Requirements

The machine needs to communicate with the main branch for each session for various functions such as login verification, account access etc. so the following are the various communication interface requirements that are needed to be fulfilled in order to run the software successfully:- The system will employ dial-up POS with the central server for low cost communication. The communication protocol used shall be TCP/IP. Protocol used for data transfer shall be File Transfer Protocol.(FTP)

system Features:

Remote Banking and Account Management

Description

The system is designed to provide the user with the facility of remote banking and perform various other functions at an interface without any aid of human bank teller. The functioning of the system shall be as follows:-

At the start, the user is provided with a log in screen and he is required to enter his PIN NO. and Account details which are then verified by the machine. In case of an unsuccessful attempt a user is asked again for his credentials but the maximum number of attempt given to the user is limited to 3 only, failing which his card is blocked and need to be unblocked by the bank for any future use. After a successful log in, the user is presented with a list of language. The user can select anyone in the list for interaction with the machine for the entire session. After the language selection the user is also asked whether he wants to fix that language for future use also so that he is never asked for language in future. In addition there is also a facility for the user to switch to any other language during that session.

After the language selection, the user is directed towards a main page that displays a set of options/services along with their brief description, enabling the user to understand their functioning. The user can select any of the listed option and can continue with the transaction. The machine also provides the user with a number of miscellaneous services such as:

The machine lists a set of operators that are supported by the bank. A user can clear off his pending mobile phone bills be selecting his operator. The machine also has the facility to display a map that marks the location of other ATMs of the same bank in the city. This may help the user to look for the ATM nearest to his destination. At any moment if the user wants to abort the transaction, he is provided with an option to cancel it. Just by pressing the abort button he can cancel all the changes made so far and can begin with a new transaction. After the user is

finished with his work, for security purpose, he is required to log out and then take his card out of the slot.

Validity Checks

In order to gain access to the system, the user is required to enter his/her correct user id/pin no and account no failing which his card may be blocked. The user can access only one account at a time and can enter only one account no. Also if the user is an administrator, he is required to enter his login id in order to access and change the facilities provided by the system.

Sequencing Information

The information about the users and their account should be entered into the database prior to any of the transactions and the backup be maintained for all account information

Error Handling/ Response to Abnormal Situations

If any of the above validation/sequencing flow does not hold true, appropriate error messages will be prompted to the user for doing the needful.

Dynamic Requirements

The card verification time must not exceed 0.8 sec. under normal server workload and 1 sec. under peak server workload. The pin number verification time must not exceed 0.3 sec. under normal server workload and 0.5 sec. under peak server workload. Account balance display time must not exceed 2 sec. under normal server workload and 3 sec. under peak server workload. Account balance transfer time must not exceed 3 sec. under normal server workload and 4 sec. under peak server workload. Cash withdrawal transaction time must not exceed 4 sec. under normal server workload and 5 sec. under peak server workload. Deposit transaction time after insertion of the deposit envelope must not exceed 5 sec. under normal server workload and 6 sec. under peak server workload. Receipt printing time after must not exceed 3 sec. under normal server and peak server workload

Quality

The primary objective s to produce quality software. As the quality of a piece of software is difficult to measure quantitatively, the following guidelines will be used when judging the quality of the software:

Software Quality Attributes:

Reliability The data communication protocol shall be such that it ensures reliability and quality of data and voice transmission in a mobile environment. For example, CDMA. The memory system shall be of non-volatile type.

Availability

The product will have a backup power supply incase of power failures. Any abnormal operations shall result in the shutting down of the system. After abnormal shutdown of the ATM, the system shall have to be manually restarted by a maintenance personnel.

There should be no inconsistency introduced in the account during whose transaction the system is abnormally shut down.

Security

The system shall be compatible with AIMS security standards. The system shall have two levels of security i.e. ATM card and pin verification both authenticated by the CMS software. The Encryption standard used during pin transmission shall be triple DES. The password shall be 6-14 characters long. Passwords shall not contain name of customers as they are easy to be hacked. Passwords can contain digit, hyphen and underscore. User should be provided with only three attempts for login failing which his card needs to be blocked. There shall be a security camera installed near the ATM. There shall be a secured cash vault with a combination locking system. The product cabinet cover shall be manufactured using Fiber glass for security purposes.

Maintainability

The system components i.e. modem, memory, disk, drives shall be easily serviceable without requiring access to the vault. The system should have the mechanism of self-monitoring periodically in order to detect any fault. The system should inform the main branch automatically as soon as it detects any error. The kind of fault and the problem being encountered should also be mentioned by the system automatically.

Business Rules

The business rules for the software are as follows: The Administrator has the authority to fix the rules and regulations and to set or update the policies as and when required. The staff at the bank performs the following:

a. Making the entries in the system regarding all the details of the bank account of the user.

b. Keeping the bank account of the user updated as soon as changes are encountered so that the data is in consistent state.

c. Blocking or seizing of the account of user on discovery of any illegal transaction.

d. Unblocking of ATM card that got blocked due to more than three unsuccessful login attempt.

e. Blocking of a lost/stolen ATM card on complaint of the user, only if he presents his verification and a FIR filed for that case.

f. Constantly monitor all the ATMs in the city to check whether any one of them is encountering any fault.

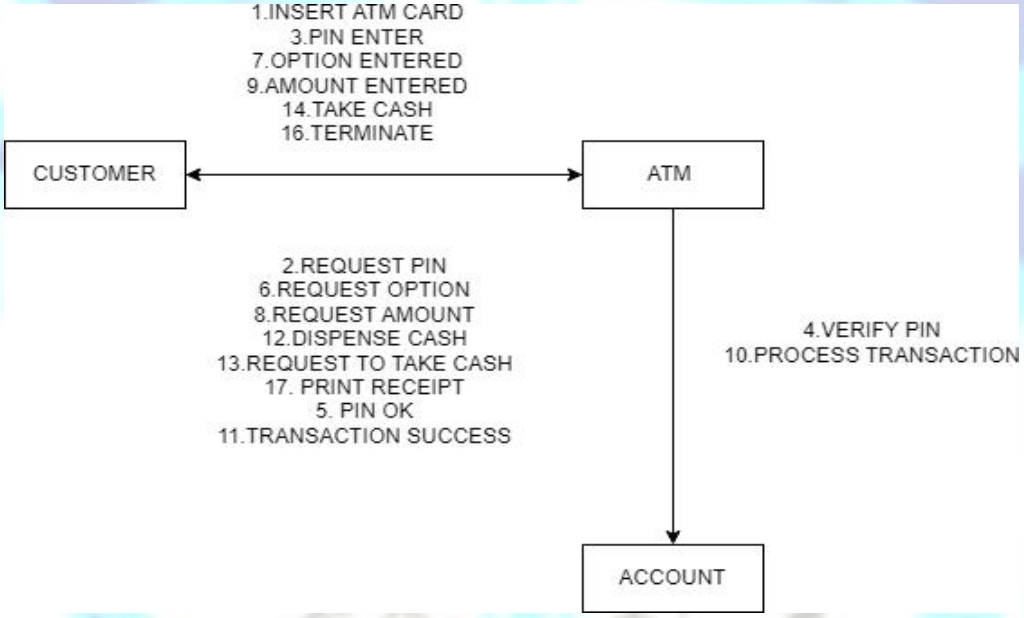
g. Immediately correct any fault discovered in any of the ATM.

h. Maintain the backup of all the accounts for reliability purposes.

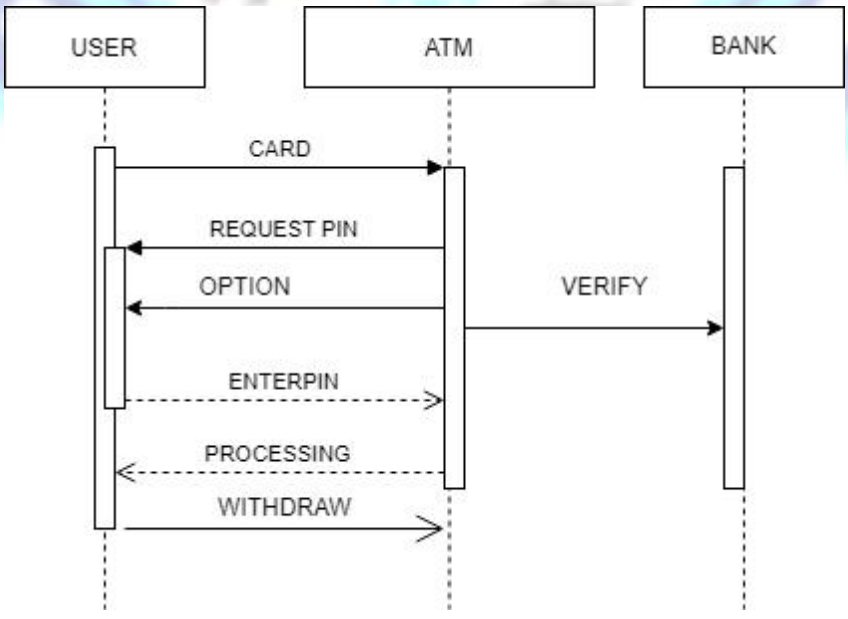
i. Rollback all the changes made in an account during whose transaction an ATM got abnormal shutdown.

In case of loss of the ATM card. The user has to lodge a First Investigation Report(FIR) and present its one copy to bank officials for card blocking purposes. A log of the following annexure is generated by the system: User bank account details. Updating made in the user account along with date, time and the changes made. Schedule of fixed assets.

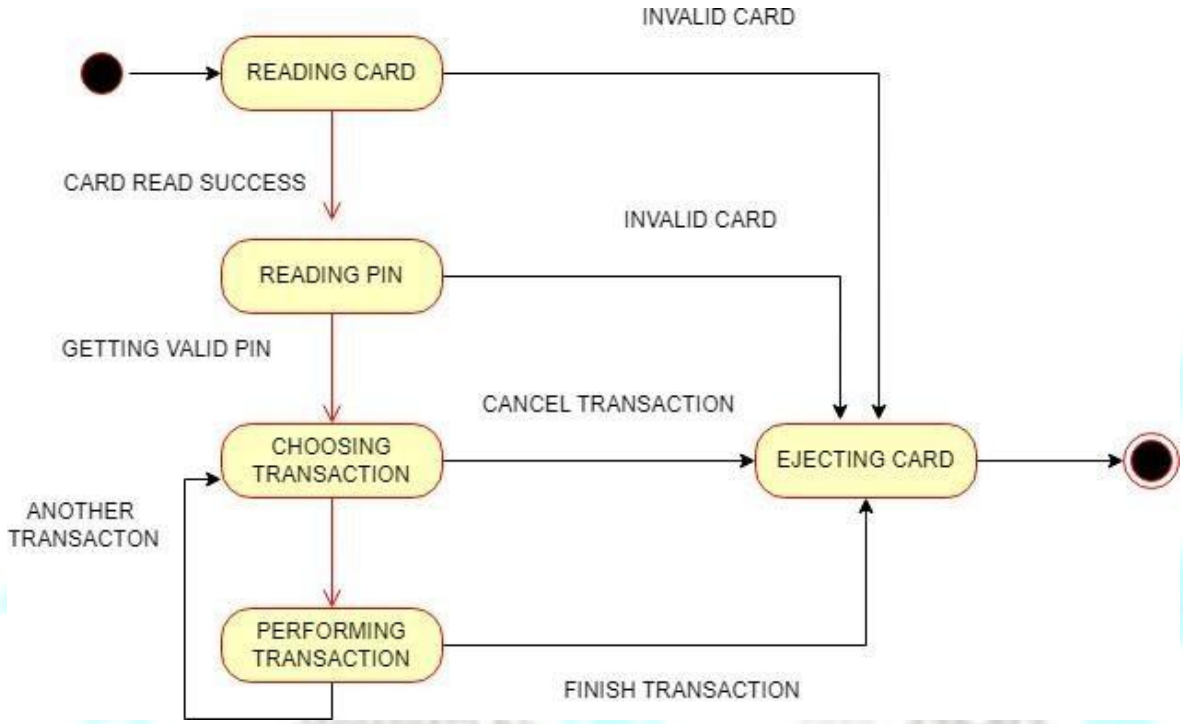
COLLABORATION DIAGRAM:



SEQUENCE DIAGRAM:



STATE DIAGRAM:



CONCLUSION:

The Draw class diagram, sequence diagram, Collaboration Diagram, State Transition Diagram was made successfully by following the steps described above.

EXNO: 7	Write Test Cases to Validate requirements of assigned project from SRS Document

Aim:

To Write Test Cases to validate requirements of assigned project from SRS Document.

THEORY:

1) Verify if the card reader is working correctly. A screen should ask you to insert the pin after inserting the valid card.

2) Verify if the cash dispenser is working as expected.

3) Verify if the receipt printer is working correctly. Which means it can print the data on the paper and the paper comes out properly.

4) Verify if the Screen buttons are working correctly.

5) Verify if the text on the screen button is visible clearly.

6) Verify the font of the text on the screen buttons.

7) Verify each number button on the Keypad.

8) Verify the functionality of the Cancel button on the Keypad.

9) Verify the text color of the keypad buttons. The numbers should be visible clearly.

10) Verify the text color and font of the data on the screen. The user should be able to read it clearly.

11) Verify the language selection option. If the messages or data are displayed in the selected language.

12) Insert the card, the correct pin, and print the receipt for available balance.

13) Verify the receipt printing functionality after a valid transaction. Whether the printed data is correct or not.

14) Verify how much time the system takes to log out.

15) Verify the timeout session functionality.

16) Verify the deposit slot functionality depending on its capability (Cash or cheque or both) by inserting a valid cheque.

17) Verify using different cards (Cards of different banks).

Verifying the Message:

18) Insert the card and an incorrect PIN to verify the message.

19) Verify the message when there is no cash in the ATM.

20) Verify the messages after a transaction.

21) Verify if a user will get a correct message if a card is inserted incorrectly.

Cash Withdrawal:

22) Verify the cash withdrawal functionality by inserting some valid amount.

23) Verify if a user can perform only one cash withdrawal transaction per PIN insert.

24) Verify the different combinations of operation and check if there will be a power loss in the middle of the operation.

Negative Test cases:

25) Verify the functionality by entering a wrong pin number for 3 or more times.

26) Verify the card reader functionality by inserting an expired card.

27) Verify the deposit slot functionality by inserting an invalid cheque.

28) Verify the cash withdrawal functionality by inserting invalid numbers like 10, 20, 50 etc.

29) Verify the cash withdrawal functionality by entering an amount greater than the per day limit,

30) Verify the cash withdrawal functionality by entering an amount greater than per transaction limit.

31) Verify the cash withdrawal functionality by entering an amount greater than the available balance in the account.

The other fields can be explained as follows:

Precondition: State of the AUT (the state in which the AUT needs to be for us to get started).

Input: Data entry steps. For these steps, it is important to note what kind of input info is required – Test data.

Validation point/trigger/action: What is causing the validation to happen? (Click of a button or toggle or the link access. Make sure there is at least one validation point to a test case- otherwise it is all going to be data entry with nothing to look for. Also to ensure that we have enough modularity, try not to combine too many validation points into one test case. 1 per test case is optimum.)

Output: Expected result.

Postcondition: This is additional information that is provided for the benefit of the tester, just to make the test case more insightful and informative. This includes an explanation of what happens or what can be expected of the AUT once all the test case steps are done.

Test Case Management Tools

Test management tools are the automation tools that help to manage and maintain the Test Cases.

Main Features of a test case management tool are

For documenting Test Cases: With tools, you can expedite Test Case creation with use of templates

Execute the Test Case and Record the results: Test Case can be executed through the tools and results obtained can be easily recorded.

Automate the Defect Tracking: Failed tests are automatically linked to the bug tracker, which in turn can be assigned to the developers and can be tracked by email notifications.

Traceability: Requirements, Test cases, Execution of Test cases are all interlinked through the tools, and each case can be traced to each other to check test coverage.

Protecting Test Cases: Test cases should be reusable and should be protected from being lost or corrupted due to poor version control. Test Case Management Tools offer features like

Naming and numbering conventions

Versioning

Read-only storage

Controlled access

Off-site backup

CONCLUSION:

To Write Test Cases to validate requirements of assigned project from SRS Document is successfully completed.

EXNO: 8	Evaluate Size of the project using function point metric for the assigned project

AIM:

To Evaluate Size of the project using function point metric for the assigned project.

THEORY:

FPA is used to make estimate of the software project, including its testing in terms of functionality or function size of the software product. However, functional point analysis may be used for the test estimation of the product. The functional size of the product is measured in terms of the function point, which is a standard of measurement to measure the software application.

Objectives of FPA

The basic and primary purpose of the functional point analysis is to measure and provide the software application functional size to the client, customer, and the stakeholder on their request. Further, it is used to measure the software project development along with its maintenance, consistently throughout the project irrespective of the tools and the technologies.

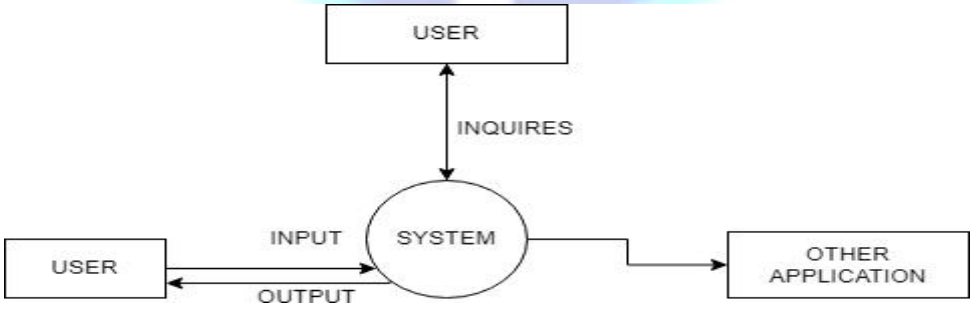
Following are the points regarding FPs

1. FPs of an application is found out by counting the number and types of functions used in the applications. Various functions used in an application can be put under five types, as shown in Table:

Measurements Parameters	Examples
1.Number of External Inputs(EI)	Input screen and tables
2. Number of External Output (EO)	Output screens and reports
3. Number of external inquiries (EQ)	Prompts and interrupts.
4. Number of internal files (ILF)	Databases and directories
5. Number of external interfaces (EIF)	Shared databases and shared routines.

All these parameters are then individually assessed for complexity.

The FPA functional units are shown in Fig:



2. FP characterizes the complexity of the software system and hence can be used to depict the project time and the manpower requirement.

3. The effort required to develop the project depends on what the software does.

4. FP is programming language independent.

5. FP method is used for data processing systems, business systems like information systems.

6. The five parameters mentioned above are also known as information domain characteristics.

Here that weighing factor will be simple, average, or complex for a measurement parameter type.

The Function Point (FP) is thus calculated with the following formula.

FP = Count-total * [0.65 + 0.01 * ∑(fi)]

= Count-total * CAF

where Count-total is obtained from the above Table.

CAF = [0.65 + 0.01 *∑(fi)]

and ∑(fi) is the sum of all 14 questionnaires and show the complexity adjustment value/ factor-CAF (where i ranges from 1 to 14). Usually, a student is provided with the value of ∑(fi)

Also note that ∑(fi) ranges from 0 to 70, i.e.,

0 <= ∑(fi) <=70

and CAF ranges from 0.65 to 1.35 because

When ∑(fi) = 0 then CAF = 0.65

When ∑(fi) = 70 then CAF = 0.65 + (0.01 * 70) = 0.65 + 0.7 = 1.35

Based on the FP measure of software many other metrics can be computed:

Errors/FP

$/FP.

Defects/FP

Pages of documentation/FP

Errors/PM.

Productivity = FP/PM (effort is measured in person-months).

$/Page of Documentation.

8. LOCs of an application can be estimated from FPs. That is, they are interconvertible. This process is known as backfiring. For example, 1 FP is equal to about 100 lines of COBOL code.

9. FP metrics is used mostly for measuring the size of Management Information System (MIS) software.

10. But the function points obtained above are unadjusted function points (UFPs). These (UFPs) of a subsystem are further adjusted by considering some more General System Characteristics

(GSCs). It is a set of 14 GSCs that need to be considered. The procedure for adjusting UFPs is as follows:

Degree of Influence (DI) for each of these 14 GSCs is assessed on a scale of 0 to 5. (b) If a particular GSC has no influence, then its weight is taken as 0 and if it has a strong influence then its weight is 5.

The score of all 14 GSCs is totaled to determine Total Degree of Influence (TDI).

Then Value Adjustment Factor (VAF) is computed from TDI by using the formula: VAF = (TDI * 0.01) + 0.65.

CONCLUSION:

The Evaluate Size of the project using function point metric for the assigned project is successfully completed.