UNIT - II

2 Requirements Analysis and Specification

Syllabus

Software requirements : Functional and non-functional, User requirements, System requirements, Software requirements document - Requirement engineering process : Feasibility studies, Requirements elicitation and analysis, Requirements validation, Requirements management-Classical analysis : Structured system analysis, Petri nets - Data dictionary.

Contents

2.1 Software Requirements

What is requirement engineering ?

Requirement engineering is the process of

- *establishing the services that the customer requires from a system*
- *and the constraints under which it operates and is developed.*

The requirements themselves are the descriptions of the system services and constraints that are generated during the requirements engineering process.

What is a requirement ?

A requirement can range from a high-level abstract statement of a service or of a system constraint to a detailed mathematical functional specification.

The requirement must be open to interpretation and it must be defined in detail.

Types of requirements

The requirements can be classified as

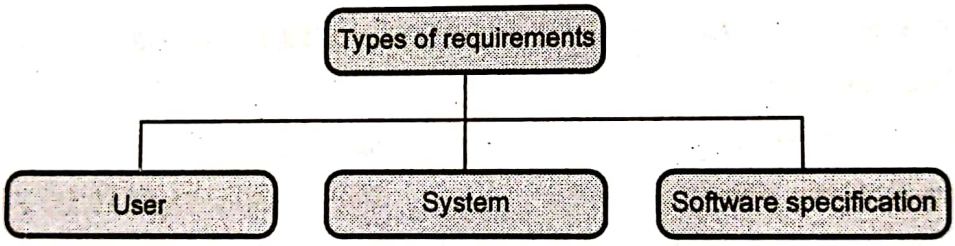


Fig. 2.1.1 Types of requirements

- **User requirements**

It is a collection of statements in natural language plus description of the services the system provides and its operational constraints. It is written for customers.

- **System requirements**

It is a structured document that gives the detailed description of the system services. It is written as a contract between client and contractor.

- **Software specification**

It is a detailed software description that can serve as a basis for design or implementation. Typically it is written for software developers.

2.2 Functional and Non Functional Requirements

AU : Dec.-13,17, May-13, Marks 11

Software system requirements can be classified as functional and non functional requirements.

2.2.1 Functional Requirements

- Functional requirements should describe all the required functionality or system services.
- The customer should provide statement of service. It should be clear how the system should react to particular inputs and how a particular system should behave in particular situation.
- Functional requirements are heavily dependant upon the type of software, expected users and the type of system where the software is used.
- Functional user requirements may be high-level statements of what the system should do but functional system requirements should describe the system services in detail.
- For example : Consider a library system in which there is a single interface provided to multiple databases. These databases are collection of articles from different libraries. A user can search for, download and print these articles for a personal study.

From this example we can obtain functional Requirements as–

1. The user shall be able to search either all of the initial set of databases or select a subset from it.
2. The system shall provide appropriate viewers for the user to read documents in the document store.
3. A unique identifier (ORDER_ID) should be allocated to every order. This identifier can be copied by the user to the account's permanent storage area.

2.2.1.1 Problems Associated with Requirements

- **Requirements imprecision**

1. Problems arise when requirements are not precisely stated.
2. Ambiguous requirements may be interpreted in different ways by developers and users.
3. Consider meaning of term 'appropriate viewers'.

- **User intention** - special purpose viewer for each different document type;
- **Developer interpretation** - Provide a text viewer that shows the contents of the document.

- **Requirements completeness and consistency -**

The requirements should be both complete and consistent. *Complete* means they should include descriptions of all facilities required. *Consistent* means there should be no conflicts or contradictions in the descriptions of the system facilities.

Actually in practice, it is impossible to produce a complete and consistent requirements document.

2.2.2 Non Functional Requirements

- The non functional requirements define system properties and constraints. Various properties of a system can be : Reliability, response time, storage requirements. And constraints of the system can be : Input and output device capability, system representations etc.
- Process requirements may also specify programming language or development method.
- Non functional requirements are more critical than functional requirements. If the non functional requirements do not meet then the complete system is of no use.

2.2.2.1 Types of Non Functional Requirements

The classification of non functional requirements is as given below.

(See Fig. 2.2.1 on next page)

Product requirements

These requirements specify how a delivered product should behave in a particular way. For instance: execution speed, reliability.

Organizational requirements

The requirements which are consequences of organizational policies and procedures come under this category. For instance: process standards used implementation requirements.

External requirements

These requirements arise due to the factors that are external to the system and its development process. For instance : interoperability requirements, legislative requirements.

In short, non functional requirements arise through

i) User needs

ii) Because of budget constraints

iii) Organizational policies

iv) The need for interoperability with other software or hardware systems

v) Because of external factors such as safety regulations.

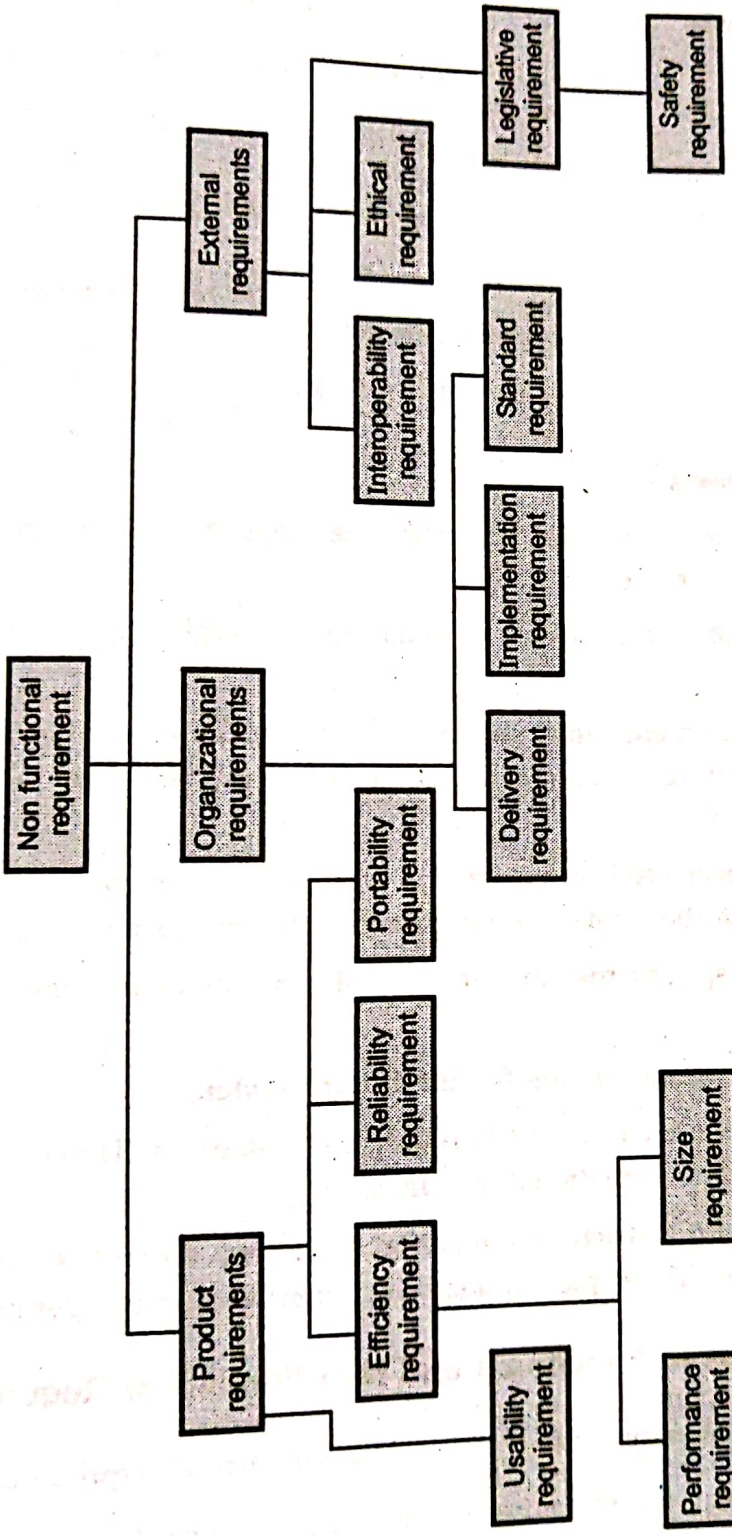


Fig. 2.2.1 Types of non functional requirement

Metrics used for specifying the non functional requirements

Property	Metric
Speed	Events per response time processed transactions per second.
Size	Kilo bytes.
Reliability	Mean time to failure. Rate of failure. Occurrence availability.
Robustness	Time to restart after failure. Probability of events causing failure.
Portability	Number of target statements.

2.2.2.2 Domain Requirements

- Domain requirements are derived from the application domain of the system instead of specific user needs.
- These requirements make use of domain terminologies specific to the existing domain concept.
- The domain requirements may be in the form of new functional requirements, constraints on existing functional requirement or guidance on how to carry out certain computation.
- These are the specialised requirements and hence software engineers find it difficult to co-relate the domain requirements with the system requirements.
- It is important to specify the domain requirements otherwise the system will not work properly.
- **Example : Domain requirements for the library system.**
- There should be user interface for handling the databases. These interfaces should be according to some international standard.
- If there is copyright restriction on some document then it should get printed locally on the server. The copies of such document should not get created.

2.2.3 Difference between Functional and Non Functional Requirements

Functional requirements	Non functional requirements
The functional requirements specify the features of the software system	The non functional requirements specify the properties of the software system.

Functional requirements describe what the product must do	Non functional requirements describe how the product should perform.
The functional requirements specify the actions with which the work is concerned.	The non functional requirements specify the experience of the user while using the system.
Example : For a library management system, allowing user to read the article online is a functional requirement.	Example : For a library management system, for a user who whishes to read the article online must be authenticated first.

Metrics used for specifying Non Functional Requirements

The metrics used for specifying the non functional requirements are -

1. **Reliability :** The application should be highly reliable and it should generate all the updated information in correct order.
2. **Availability :** Any information must be quickly available from computer to the authorized user.
3. **Security :** The application must be password protected. This feature is essential to avoid any unauthorized access to the application.
4. **Maintainability :** The application should be maintained in such a manner that if any change in requirement occurs then it should be easily incorporated in an individual module.
5. **Extensibility :** The application should be maintained in such a manner that if any new requirement occurs then it should be easily incorporated in an individual module.
6. **Portability :** The application should be portable on the desired operating system.
7. **Reusability :** The application should have the ability that a segment of source code can be used again to add new functionalities with slight or no modification. Reusable modules and classes reduce implementation time.
8. **Resource utilization :** The application should make use of resources to its maximum capability.

Example 2.2.1 For the requirement given below, identify stakeholders, functional and non-functional requirements

A software is to be built that will control an Automated Teller Machine (ATM). The ATM machine services customers 24 × 7. ATM has a magnetic stripe reader for reading an ATM card, a keyboard and display for interaction with the customer, a slot for depositing envelopes, a dispenser for cash, a printer for printing receipts and a switch that allows an operator to start/stop a machine.

The ATM services one customer at a time. When a customer inserts an ATM card and enters the Personal Identification Number (PIN), the details are validated for each transaction. A customer can perform one or more transactions. Transactions made against each account are recorded so as to ensure validity of transactions.

If PIN is invalid, customer is required to re-enter PIN before making a transaction, If customer is unable to successfully enter PIN after three tries, card is retained by machine and customer has to contact bank.

The ATM provides the following services to the customer :

1) Withdraw cash in multiples of 100
2) Deposit cash in multiples of 100
3) Transfer amount between any two accounts.
4) Make balance enquiry.
5) Print receipt.

Each of the above transactions must be made within 60 seconds. in case the time exceeds 60 seconds, then the transaction is cancelled automatically. Also, if the machine is not used for more than two minutes after entry of card, the card is retained by the machine.

An operator panel with a key-operated switch allows an operator to start and stop the servicing of customers. When the switch is moved to the "off position, the machine will shut down, so that the operator may remove deposit envelopes and reload the machine with cash, blank receipts, etc. The operator is required to verify and enter the total cash on hand before starting the system from this panel.

AU : Dec.-13, Marks 11

Ans. : **i) Stakeholders**

1. Bank customer
2. Service operator
3. Hardware and software maintenance engineers
4. Database administrators
5. Banking regulators
6. Security administrator

ii) Functional requirements

1. There should be the facility for the Customer to insert a card.
2. The system should first validate card and PIN.
3. The system should allow the customer to deposit amount in the bank.
4. The system should dispense the cash on withdrawal.
5. The system should provide the printout for the transaction.
6. The system should make the record of the transactions made by particular customer.
7. On invalid PIN entry for three times the card should be retained by the system.
8. The cash withdrawal is allowed in multiple of 100.
9. The cash deposition is allowed in multiple of 100.
10. The customer is allowed to transfer amount between the two accounts.
11. The customer is allowed to know the balance enquiry.
12. The customer is allowed to get the printout for desired transaction.
13. The system should be efficient.

iii) Non functional requirements

1. Each of the transaction should be made within 60 seconds.If the time limit is exceeded ,then cancel the transaction automatically.
2. If there is no response from the bank computer after request is made within the minutes then the card is rejected with error message.
3. The bank dispenses money only after the processing of withdrawal from the bank. That means if sufficient fund is available in user's account then only the withdrwal request is processed.
4. Each bank should process the transactions from several ATM centers at the same time.
5. The machine should be loaded with sufficient fund in it.

Example 2.2.2 *List the stakeholders and all types of requirements for an online train reservation system.* **AU : Dec.-17, Marks 7**

Solution :

1) Passenger

2) Database Administrator

3) Booking Clerk

4) Bank

Functional Requirements

1) The online booking made by the customer should be associated with the account.
2) Booking confirmation must be sent to the user on specified contact details.
3) The system should provide a search option to the user so that user can search for required train and for required number of reservations.
4) The system should allow the user to make online payments.
5) The system should
6) The system should allow the user to cancel the booking and half of the amount paid by the customer must be refunded to him.

Non Functional Requirements

1) The system must be available to the user for 24*7.
2) The system should accept the payments via different payment methods like PayPal, Wallets, Cards, vouchers etc.
3) Use of captcha and encryption to avoid bots from booking tickets.
4) The response time of the system while searching, booking or cancellation operation should be very less.

Review Question

1. *Explain the metrics used for specifying non functional requirements.* **AU : May-13, Marks 8**

2.3 User Requirements

- The user requirements should describe functional and non functional requirements in such a way that they are understandable by system users who don't have detailed technical knowledge.
- User requirements are defined using natural language, tables and diagrams because these are the representations that can be understood by all users.

Various **problems** that can arise **in the requirement specifications** when requirements are given in natural language -

Lack of clarity

Sometimes requirements are given in ambiguous manner. It is expected that text should help in clear and precise understanding of the requirements.

Requirements confusion

There may be confusion in functional requirements and non functional requirements, system goals and design information.

Requirements mixture

There may be a chance of specifying several requirements together as a single requirement.

2.3.1 Guidelines for Writing User Requirements

- Prepare a standard format and use it for all requirements.
- Apply consistency in the language. Use
  - *'shall'* for mandatory requirements
  - and *'should'* for desirable requirements.
- The text which is mentioning the key requirements should be highlighted.
- Avoid the use of computer jargon (computer terminologies). It should be written in simple language.

For example

Consider a spell checking and correcting system of a word processor. The user requirements can be given in natural language as

1. The system should posses a traditional **word dictionary and user supplied dictionary.** It shall provide a user-activated facility which checks the spelling of words in the document against spellings in the system dictionary and user-supplied dictionaries.
2. When a word is found in the document which is not given in the dictionary, then the system should suggest 10 alternative words. These alternative words should be based on a match between the word found and corresponding words in the dictionaries.
3. When a word is found in the document which is not in any dictionary, the system should propose following options to user :

1. Ignore the corresponding instance of the word and go to next sentence.
2. Ignore all instances of the word.
3. Replace the word with a suggested word from the dictionary.
4. Edit the word with user-supplied text.
5. Ignore this instance and add the word to a specified dictionary.

2.4 System Requirements

AU : May-16, Marks 4

- System requirements are more detailed specifications of system functions, services and constraints than user requirements.

- They are intended to be a basis for designing the system.
- They may be incorporated into the system contract.
- The system requirements can be expressed using system models.
- The requirements specify what the system does and design specifies how it does.
- System requirement should simply describe the external behavior of the system and its operational constraints. They should not be concerned with how the system should be designed or implemented.
- For a complex software system design it is necessary to give all the requirements in detail.
- Usually, natural language is used to write system requirements specification and user requirements.

2.4.1 Structured Language Specification

- One of the method of writing requirements is by using structured natural language.
- All the requirements should be written in a **standard way** while using structured language specification.
- The **advantage** of specifying requirements using this method is that requirement become **understandable** and **expressive.**
- The only necessary thing while writing requirements using natural language is that some degree of **uniformity must be maintained.**
- Extra information can be added when the requirements are written using natural language. This information can be represented using tables or graphical models.
- One way of using graphical model is use of **sequence diagram.**
- The sequence diagram represents the sequence of actions that user performs while interacting the system.
- Example : Following is a sequence diagram for withdrawal of cash from ATM. (See Fig. 2.4.1 on next page)

Why requirement and design are inseparable ?

- A system architecture may be designed to structure the requirements.
- The system may inter-operate with other systems and that may generate design requirements.
- The use of a specific design may be a domain requirement.

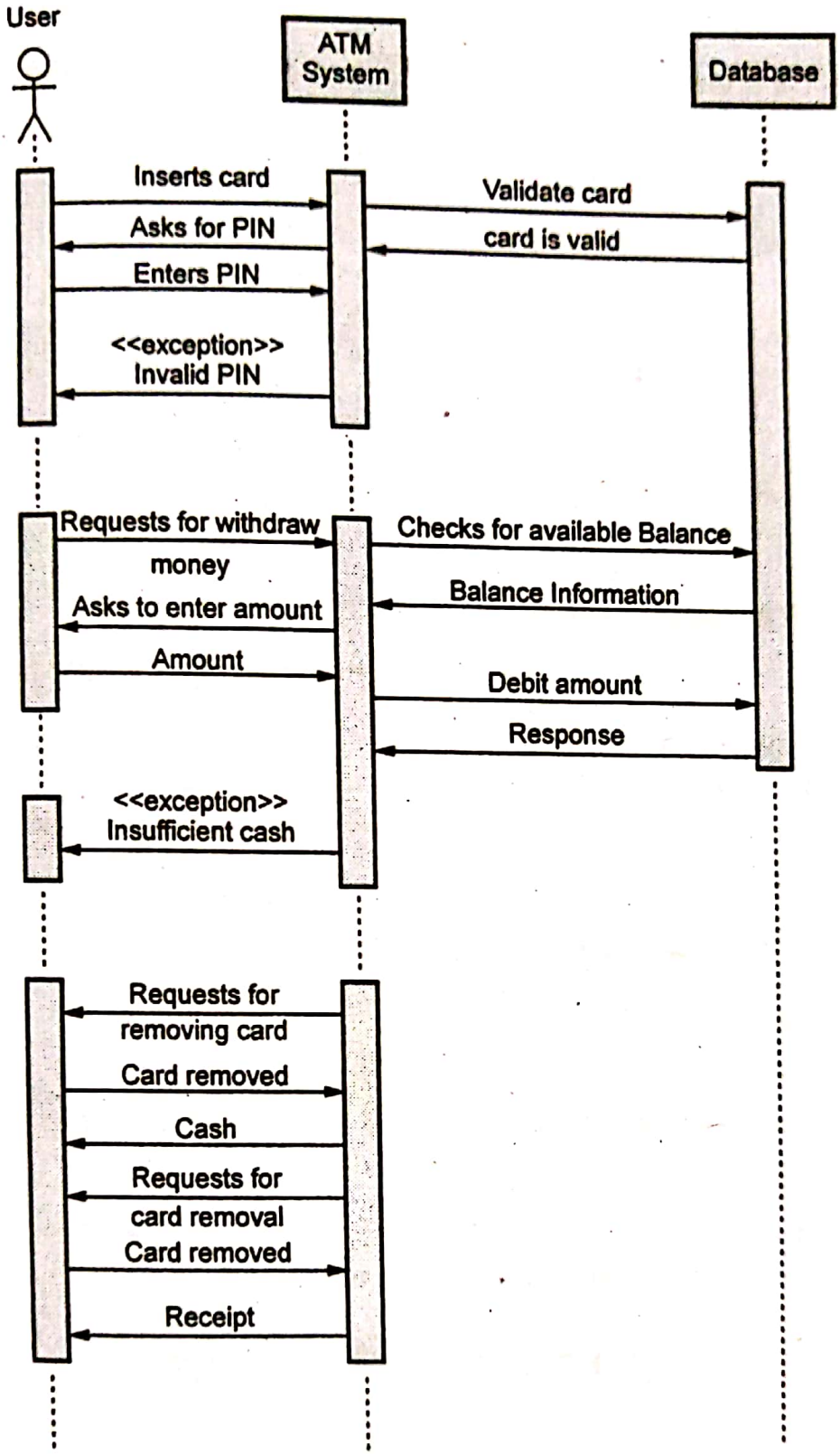


Fig. 2.4.1 Sequence diagram of ATM withdrawal

Example 2.4.1 *Differentiate between user and system requirements.* **AU : May-16, Marks 4**

Ans. :

User Requirement	System Requirement
The focus of this types of requirements is on the problem domain.	The focus of this type of requirements is on solution domain.
These requirements describe what effects need to be achieved.	These requirements describe what software the software must do.
User requirement tell what application should do to satisfy users needs.	System requirements tell a system should have to be able to run program.
The user defines his/her requirements for the system. Hence it is not necessary to incorporate all the user requirements in the system contract.	These requirements must be incorporated into system contract.

2.5 Software Requirements Document

AU : Dec.-11, May-13,14,16,18, Marks 16

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is not a design document. As far as possible, it should set of what the system should do rather than how it should do it.

Software Requirements Specification

The software requirements provide a basis for creating the Software Requirements Specifications (SRS).

The SRS is useful in estimating cost, planning team activities, performing tasks, and tracking the team's progress throughout the development activity.

Typically software designers use IEEE STD 830-1998 as the basis for the entire Software Specifications.The standard template for writing SRS is as given below.

Document Title

Author(s)
Affiliation
Address
Date
Document Version

1. Introduction

1.1 Purpose of this document

Describes the purpose of the document.

1.2 Scope of this document

Describes the scope of this requirements definition effort. This section also details any constraints that were placed upon the requirements elicitation process, such as schedules, costs.

1.3 Overview

Provides a brief overview of the product defined as a result of the requirements elicitation process.

2. General Description

- Describes the general functionality of the product such as similar system information, user characteristics, user objective, general constraints placed on design team.
- Describes the features of the user community, including their expected expertise with software systems and the application domain.

3. Functional Requirements

This section lists the functional requirements in ranked order. A functional requirement describes the possible effects of a software system, in other words, *what* the system must accomplish. Each functional requirement should be specified in following manner

- **Short, imperative sentence stating highest ranked functional requirement.**

1. **Description**
   A full description of the requirement.
2. **Criticality**
   Describes how essential this requirement is to the overall system.
3. **Technical issues**
   Describes any design or implementation issues involved in satisfying this requirement.
4. **Cost and schedule**
   Describes the relative or absolute costs of the system.
5. **Risks**
   Describes the circumstances under which this requirement might not able to be satisfied.

6. **Dependencies with other requirements**
   Describes interactions with other requirements.
7. **... any other appropriate**

4. Interface Requirements

This section describes how the software interfaces with other software products or users for input or output. Examples of such interfaces include library routines, token streams, shared memory, data streams, and so forth.

- **4.1 User Interfaces**

  Describes how this product interfaces with the user.

  - **4.1.1 GUI**

    Describes the graphical user interface if present. This section should include a set of screen dumps to illustrate user interface features.
  - **4.1.2 CLI**

    Describes the command-line interface if present. For each command, a description of all arguments and example values and invocations should be provided.
  - **4.1.3 API**

    Describes the application programming interface, if present.
- **4.2 Hardware Interfaces**

  Describes interfaces to hardware devices.
- **4.3 Communications Interfaces**

  Describes network interfaces.
- **4.4 Software Interfaces**

  Describes any remaining software interfaces not included above.

5. Performance Requirements

Specifies speed and memory requirements.

6. Design Constraints

Specifies any constraints for the design team such as software or hardware limitations.

7. Other Non-Functional Attributes

Specifies any other particular non functional attributes required by the system. Such as :

7.1 Security

7.2 Binary Compatibility

7.3 Reliability

7.4 Maintainability

7.5 Portability

7.6 Extensibility

7.7 Reusability

7.8 Application Compatibility

7.9 Resource Utilization

7.10 Serviceability

... others as appropriate

8. Operational Scenarios

This section should describe a set of scenarios that illustrate, from the user's perspective, what will be experienced when utilizing the system under various situations.

9. Preliminary Schedule

This section provides an initial version of the project plan, including the major tasks to be accomplished, their interdependencies, and their tentative start/stop dates.

10. Preliminary Budget

This section provides an initial budget for the project.

11. Appendices

11.1 Definitions, Acronyms, Abbreviations

Provides definitions terms, and acronyms, can be provided.

11.2 References

Provides complete citations to all documents and meetings referenced.

2.5.1 Characteristics of SRS

Various characteristics of SRS are

- Correct - The SRS should be made up to date when appropriate requirements are identified.
- Unambiguous - When the requirements are correctly understood then only it is possible to write an unambiguous SRS.

- Complete - To make the SRS complete, it should be specified what a software designer wants to create a software.
- Consistent - It should be consistent with reference to the functionalities identified.
- Specific - The requirements should be mentioned specifically.
- Traceable - What is the need for mentioned requirement? This should be correctly identified.

2.5.2 Example of SRS

Software Requirements Specification For Attendance Maintenance System

Prepared by Anjali
December 1, 2014
Release 1.0

Version 1.0

Table of Contents

1. Introduction

1.1 Purpose

This document gives detailed functional and non functional requirements for attendance maintenance system. The purpose of this document is that the requirements mentioned in it should be utilized by software developer to implement the system.

1.2 Scope

This system allows the Teacher to maintain attendance record of the classes to which it is teaching. With the help of this system Teacher should be in a position to send e-mail to the students who remain absent for the class. The system provides a cumulative report at every month end for the corresponding class.

1.3 Overview

This system provides an easy solution to the Teacher to keep track of student attendance, and statistics.

2. General Description

This attendance maintenance system replaces the traditional, manual attendance system by which a lot of paper work will be reduced. The Teacher should able to view photograph of a student along with his attendance in his Laptop. This is the primary feature of this system. Another feature is that Teacher can be allowed to edit particular record at desired time. The system should produce monthly attendance report. And there should be facility to send an e-mail/warning to the student remaining absent in the class.

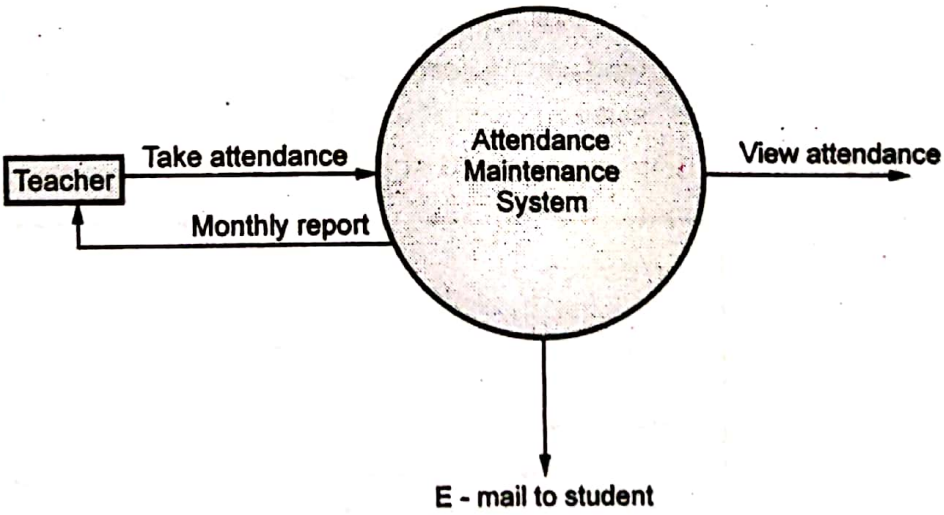


Fig. 2.5.1

Every Teacher should have Laptop with wireless internet connection. A Teacher may teach to different classes and a separate record for the corresponding classes should be maintained.

2.1 User Manual

The system should provide **Help** option in which how to operate the system should be explained. Also hard copy of this document should be given to the user in a booklet form.

3. Functional Requirements

3.1 Description

The identity of student is verified and then marked present at particular date and Time. The system should display student photograph along with their names for that corresponding class. The student may be marked present or absent depending upon his presence. The system should send e-mails to absent students. A statistical report should display individual's report or a cumulative report whenever required.

3.2 Technical Issues

The system should be implemented in VC++

4. Interface Requirements

4.1 GUI

GUI 1 : Main menu should provide options such as File, Edit, Report, Help.

GUI 2 : In File menu one can create a new record file or can open an existing record file. For example : If file *SECOMP_SEMI_07* is opened then we may view attendance record of SE COMPUTER class in year 2007 and a record for semester-I can be viewed.

GUI 3 : The display of record should be

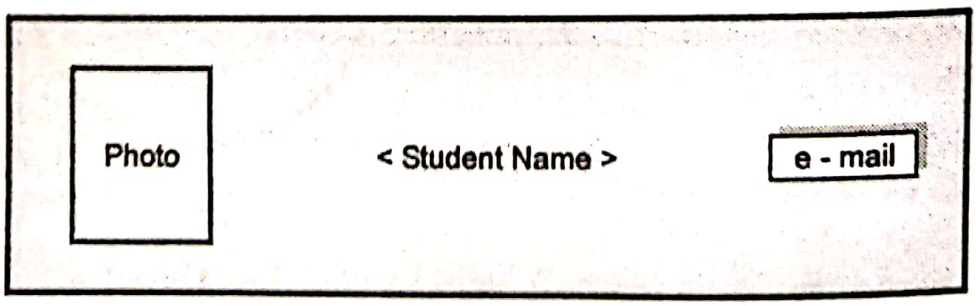


Fig. 2.5.2

The photo can be clicked to mark student present for particular class. The **e-mail** button can be clicked to send e-mail to the student being absent.

GUI 4 : Report option should display statistical report. It may be for particular student or for the whole class.

GUI 5 : Help option should describe functionality of the system. It should be written in simple HTML.

4.2 Hardware Interface

Hardware Interface 1 : The system should be embedded in the laptops.

Hardware Interface 2 : The laptop should use wireless Ethernet card to send e-mails. These Laptops should be the clients of departmental database server.

4.3 Software Interface

Software Interface 1 : Attendance maintenance system.

Software Interface 2 : The attendance database should be transmitted to departmental database server.

Software Interface 3 : E-mail message generator which generates standard message of absence.

Software Interface 4 : Report generators

5. Performance Requirements

This system should work concurrently on multiple processors between the college hours. The system should support 50 users.

The e-mail should be sent within one hour after the class gets over.

The system should support taking attendance of maximum 100 students per class.

6. Design Constraints

The system should be designed within 6 months.

7. Other Non-functional Attributes

7.1 Security

The teacher should provide password to log on to the system. He/she should be able to see the record of his/her class.

The password can be changed by the Teacher by providing existing password.

7.2 Reliability

Due to wireless connectivity , reliability can not be guaranteed.

7.3 Availability

The system should be available during college hours.

7.4 Maintenability

There should be a facility to add or delete or update Teachers and students for each semester.

7.5 Reusability

The same system will be used in each new semester.

8. Operational Scenarios

There will be student database, Teacher database. The student database will contain students name, class, attendance, e-mail address, address, phone number.

The Teacher database will contain Teacher's name, classes taught, e-mail address, phone number.

9. Preliminary Schedule

The system has to be implemented within 6 months.

Importance of SRS : The software requirements provide a basis for creating the Software Requirements Specifications (SRS).

The SRS is useful in estimating cost, planning team activities, performing tasks, and tracking the team's progress throughout the development activity.

Example 2.5.1 *An independent trunk company wants to track and record its drivers driving habits. For this purpose the company nas rented 800 phones numbers and has printed the numbers in the front, back and sides of all trucks owned by the company. Next to the 800 numbers a message is written "PLEASE REPORT ANY DRIVER OR TRUCK PROBLEM BY CALLING THIS NUMBER". The hacking company waits for you to develop a system that :*

i) Collects information from caller about the driver performance and behaviour as well as truck condition,

ii) Generates daily and monthly reports for each driver and truck management.

iii) Reports problems that require immediate action to an on-duty manager.

Analyse the problems that statement and list major functions to be incorporated with the SRS document.

AU : Dec.-11, Marks 8

Solution : The major functions to be included with SRS document are -

i) A) Creation of databases such as drivers database, trucks database, callers database.

B) Updation for above created databases.

C) Display and searching operations on database.

ii) A) Report generation (Daily) of trucks and drivers.

B) Report generation (Monthly) of trucks and drivers.

iii) A) Sending of SMS to corresponding driver about the problem.

B) Updation for duty hours if necessary.

C) Deletion of particular record (For truck database) if the truck is out of condition.

D) Insertion of new record (New truck arrival).

Review Questions

1. *Show the template of IEEE standard software requirements document.* **AU : May-13, Marks 8**
2. *What are the components of the standard structure for the software requirements document ? Explain in detail.* **AU May-14, Marks 8, May-18, Marks 16**
3. *Write the software requirement specification for a system of your choice.* **AU : May-14, Marks 8**
4. *Explain the organization of SRS and highlight the importance of each subsection.* **AU : May-16, Marks 8**

2.6 Requirement Engineering Process

AU : May-15, 17, Dec.-15, Marks 16

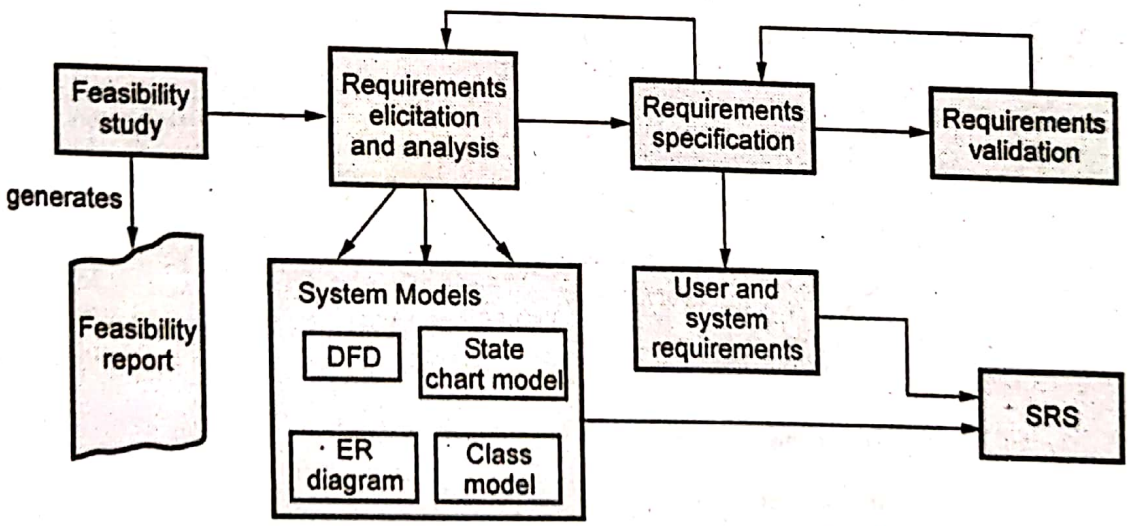


Fig. 2.6.1 Requirement engineering process

- A Requirement Engineering is a process in which various activities such as **discovery, analysis** and **validation** of system requirements are done.
- It begins with feasibility study of the system and ends up with requirement validation. Finally the requirement document has to be prepared.

- This process is a three stage activity where the activities are arranged in the iterative manner. In the early stage of this process most of the time is spent on understanding the system by understanding the high-level non functional requirements and user requirements. The spiral model of requirement engineering process is as shown in Fig. 2.6.2.

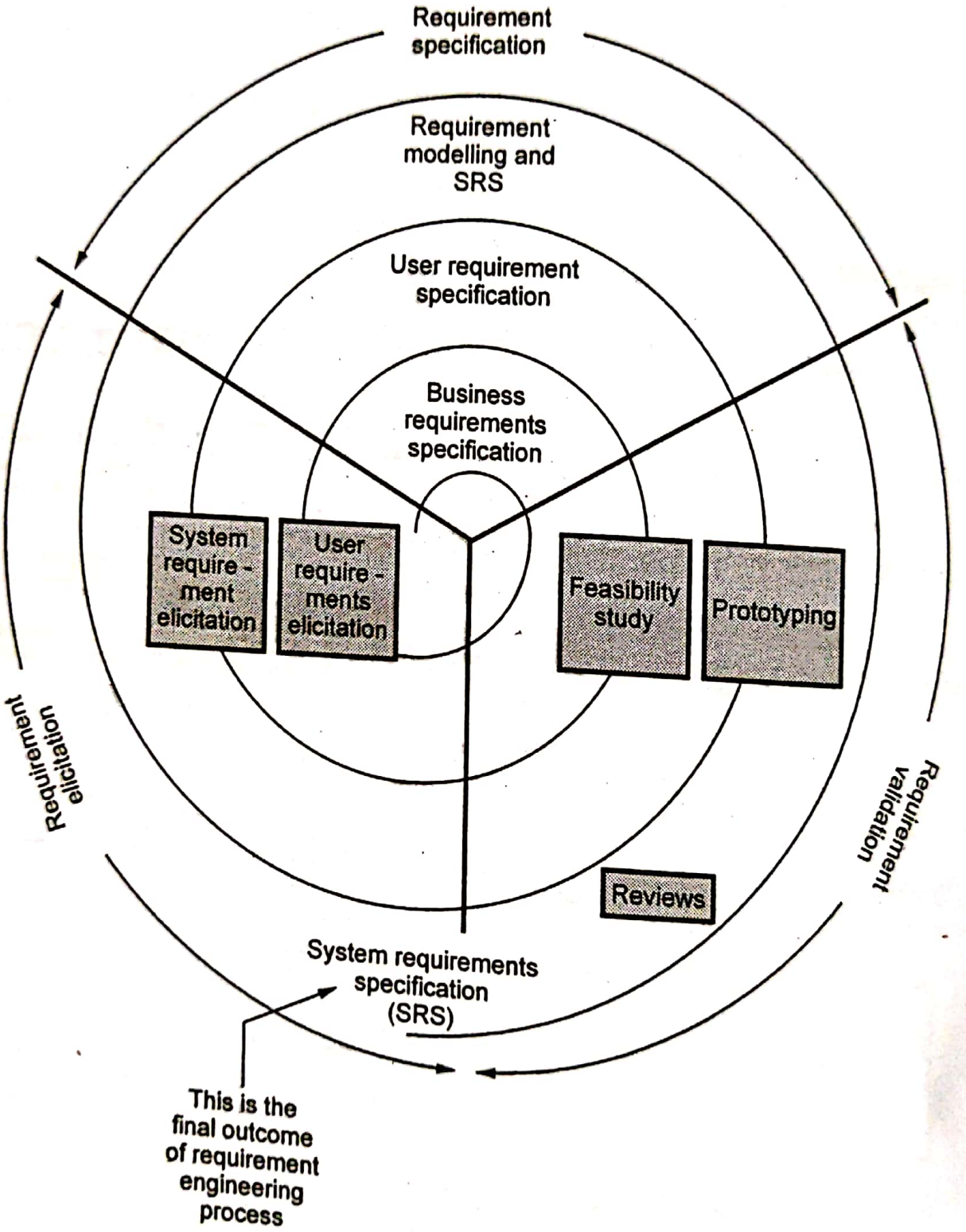


Fig. 2.6.2 Spiral model of requirements engineering process

Requirement engineering process performs following **seven distinct functions** -

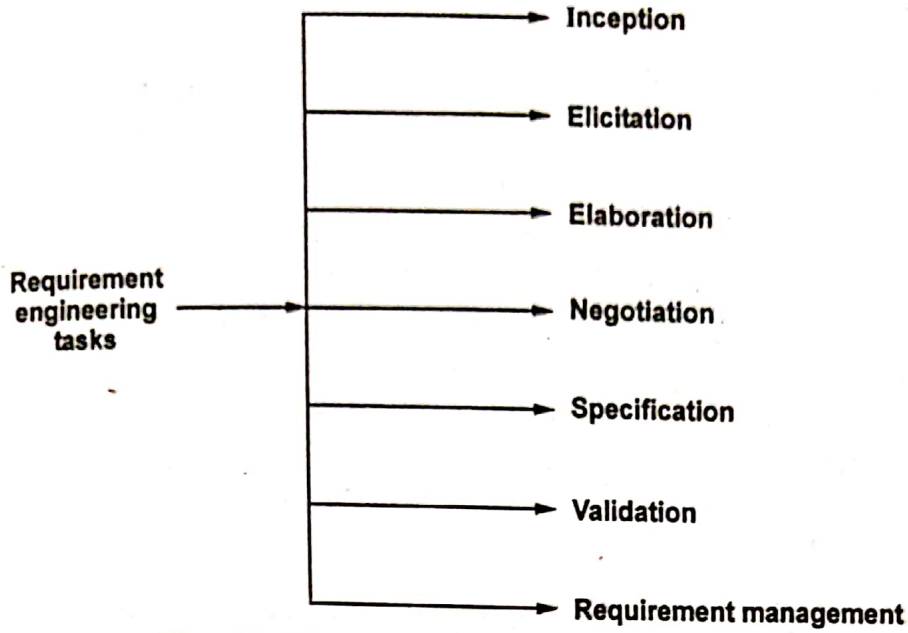


Fig. 2.6.3 Requirement engineering tasks

Let us now discuss these tasks in detail -

Inception : The inception means specifying the **beginning** of the software project. Most of the software projects get started due to business requirements. There may be potential demand from the market for a particular product.

The purpose of inception is to -

1. Establish the basic understanding of the project.
2. Find out all possible solutions and to identify the nature of the solution.
3. Establish an effective communication between developer and the customer.

Elicitation : Requirements elicitation means **requirements discovery.** Requirements elicitation is very difficult task.

Following are the reasons for : ***why it is difficult to understand customer wants?*** -

1. Customer sometimes is unable to specify the **scope of the project.** Sometimes customers specify too many technical details and this may increase the confusion.
2. There is difficulty in **understanding the problem.** Sometimes customer could not decide what are their needs and wants. Sometimes they have got poor understanding of capabilities and limitations of the existing computing environment.

   Sometimes customers find it **difficult to communicate** with the system engineer about their needs. Sometimes customers may have got **some conflicting requirements.** This ultimately results in specifying **ambiguous requirements.**

3. As project progresses the needs or requirements of the customers changes. This creates a problem of **volatility**.

To overcome these problems the requirements gathering must be done very systematically.

Elaboration : Elaboration is an activity in which the information about the requirements is **expanded** and **refined**. The **goal** of elaboration activity is to prepare a technical model of software functions, features and constraints.The elaboration consists of several **modelling and refinement tasks**. In this process several **user scenarios** are created and refined. During elaboration, each user scenario is parsed and various **classes** are identified. These classes are nothing but the business entities that are visible to end user. Then the **attributes** and **services** (functions) of these classes are defined. Then the relationship among these classes is identified. Finally the **analysis model** gets developed during the elaboration phase.

Negotiation : Sometimes customer may **demand for more** than that is achieved or there are certain situations in which customer demands for something which cannot be achieved in **limited business resources**. To handle such situations requirement engineers must convince the customers or end users by solving various **conflicts**.

Specification : A specification can be a **written document, mathematical** or **graphical model, collection of use case scenarios** or may be the **prototypes.** There is a need to develop a **standard specification** in which requirements are presented in consistent and understandable manner.

Validation : Requirement validation is an activity in which requirement specification is analyzed in order to ensure that the requirements are specified unambiguously. If any inconsistencies, omissions and errors are identified then those are corrected or modified during the validation.

Requirement management : Requirements management is the process of managing changing requirements during the requirements engineering process and system development.

Review Questions

1. *What is requirements engineering ? Explain in detail the various processes in requirements engineering.* **AU : May-17, Marks 13**
2. *Write about the following requirements engineering activities.*
   (i) Inception (ii) Elicitation (iii) Elaboration (iv) Negotiation (v) Specification
   (vi) Validation (vii) Requirements management **AU : May-15, Marks (2+3+3+2+2+2+2)**
3. *Explain the software requirement engineering process with neat diagram.* **AU : Dec.-15, May-17, Marks 16**

2.7 Feasibility Studies

AU : May-17,Dec.-17, Marks 8

Definition : *A feasibility study is a study made to decide whether or not the proposed system is worthwhile.*

The **focus** of feasibility study is to check

- If the system contributes to **organizational objectives.**
- If the system can be engineered using **current technology.**
- If the system is within the given **budget.**
- If the system can be **integrated** with other useful systems.

The implementation of feasibility study is based on the information assessment (what is required), information collection and report writing.

While performing the feasibility study, following questionnaires to the people in the organization should be asked -

1. What if the system wasn't implemented ?
2. What are current process problems ?
3. How will the proposed system help ?
4. What will be the integration problems ?
5. Is new technology needed? With what skills ?
6. What facilities must be supported by the proposed system ?

Feasibility study should be done with the help of project managers who is going to handle that particular project, software engineers who are about to develop that system, technical experts and customers who will be using the system. Typically the feasibility study should be completed within **two to three weeks.**

Finally the **feasibility study report** has to be prepared.

Types of Feasibility Study

1. **Technical Feasibility :** It is the measure of technical resources such as hardware components, software techniques and skilled persons.
2. **Economical Feasibility :** It is the measure whether finances or funds are available for proposed system.
3. **Operational Feasibility :** It is a measure of how well the solution of problems or a specific alternative solution will work in the organization.
4. **Schedule Feasibility :** It is the establishment of time limit for completion of the project. This kind of feasibility is dependent upon available manpower and economical support for the project.

Thus the feasibility study helps in understanding the requirements of the system. Hence feasibility study affects the requirement collection implicitly or explicitly.

Review Questions

1. *Explain the feasibility studies. What are the outcomes? Does it have implicit or explicit effects on software requirement collection?* **AU : May-17, Marks 8**
2. *What is feasibility study ? How it helps in requirement engineering process ?* **AU : Dec.-17, Marks 3**

2.8 Requirements Elicitation and Analysis

AU : Dec.-13,16, May-08,17,18, Marks 16

After performing feasibility study the requirements elicitation and analysis can be done. Requirement **elicitation means discovery** of all possible requirements. After identifying all possible requirements the analysis on these requirements can be done. Software engineers communicate the end-users or customers in order to find out certain information such as: application domain, expected services from the system, the expected performance level of the system. From this information even constraints of the system can be decided.

2.8.1 Stakeholders

This is a commonly used term in software engineering. The stakeholder means the person(s) who will be affected by the system. For example : end-user, system maintenance engineers or software engineers can be stakeholders. Following is the list of **problems** encountered in understanding the requirements of the system.

Problems	Description
Unrealistic expectations	Stakeholders sometimes unable to specify what they want exactly. Sometimes they specify unrealistic demands.
Differences in the requirements	Different stakeholders specify different requirements. The requirement engineer has to resolve the conflicts in the requirements with proper communication with the stakeholders
Economic and business environment	The economic and business environment is dynamic due to which there may be change in the requirements or change in the stakeholders. Both these things may affect the requirement analysis process.
Political changes	Sometimes political factors affect heavily on the need for the system. Hence there may be change in the requirements or stakeholders which ultimately affect the requirement elicitation and analysis.

2.8.2 Requirement Elicitation and Analysis Process

The spiral model as given below depicts the requirement elicitation and analysis process.

The process activities are -

1. **Requirement discovery :** By having effective communication with the customers the requirements can be identified.
2. **Requirements classification and discovery :** All the unstructured requirements can be categorised systematically depending upon their nature. And they are arranged in groups.
3. **Requirement prioritisation :** There are some conflicting requirements. Hence the requirements are prioritised first. If there are some unrealistic requirements then negotiations are made and only realistic prioritised requirements are collected. If any conflict occurs then it its resolved by requirement engineers.
4. **Requirement documentation :** This is the specification of all the requirements. And an important requirement document is created.

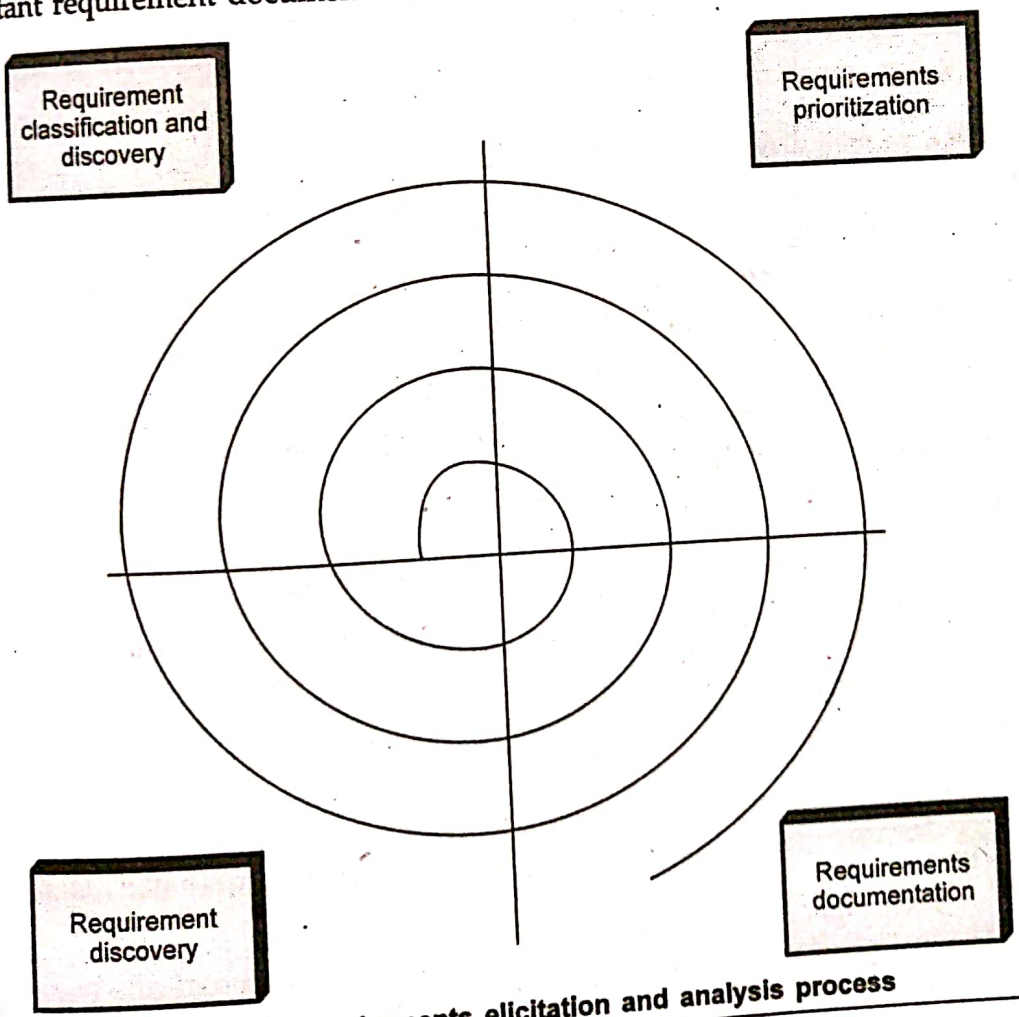


Fig. 2.8.1 Requirements elicitation and analysis process

2.8.3 Requirement Discovery

Requirement discovery means finding all relevant information about the system. The **sources of information** for requirement gathering are: documentation, system stakeholders and specification some other system which is of similar kind. Various **methods** of requirement discovery are conducting interviews, observations, creating use case scenarios. Let us discuss various methods of requirements discovery.

1. Interviewing

This is an effective method of requirement gathering. The requirement engineering team communicates the stakeholders by asking them various questions about the system and its use. From the answers the requirements can be identified. There are two types of interviews. (Refer Fig. 2.8.2.)

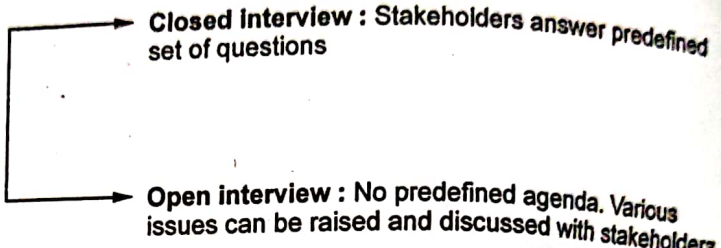


Fig. 2.8.2 Types of interview

Interviews are useful for understanding stakeholders but they are not much useful for understanding the application domain.

Characteristics of effective interviews

1. The interviews should be conducted in a free environment and they should be conducted with open minded approach. The requirement engineers should listen to stakeholders with patience. Similarly if stakeholders are expecting some unrealistic things about the system they should be ready to change their mind and ideas about the system.
2. Interviewee should start the discussion by asking questions and the requirements should be gathered together.

2. Scenario

- Scenario is the sequence of interactions made by the user with the software systems. Requirements engineers can use the information gained from this scenarios to formulate actual system requirements. The scenarios describe the interaction sessions.
- Each scenario covers one or more interaction sessions.
- Each scenario includes -
  - A description of what the system and the users want at the beginning of the scenario
  - A description of normal flow of events
  - A description of something that went wrong and handling of it

- A description of other activities occurring in parallel
- A description of finish state

For example : **Scenario for an article downloading in the library system.**

Initial Assumption	User logs in to the library system and searches for the desired article
Normal flow of events	1. When user find the required article and clicks the button for downloading the article, he might be displayed for either provide the subscriber information or request for payment for purchasing the article. 2. User either submits the subscription information or pays online for the article. 3. He might be displayed with copyright information and might be requested to fill up the copyright related form 4. User fills up the form and submits it. 5. The user is authenticated an required document can be presented to the user in pdf form. 6. User might select the print option for printing the required article
What went wrong	1. User fails to pay for the article 2. User makes mistakes in filling up the copyright form Both these events are handled by rejecting the download request.
Other activities	User can download other articles He might search for other articles
Finish State	User gets log out the library system.

3. View point

Viewpoint provides the perspective to the system and using these perspectives the requirements can be discovered. The viewpoint is also useful in classifying the stakeholders. Following are the types of viewpoints -

Interactor viewpoint : This viewpoint is useful for finding the interaction one system with other system. For example in ATM system the user of ATM is the interactor for the bank.

Indirect viewpoints : The user who is not using the system directly but its existence reflects on the requirements of the system. Such stakeholders form indirect viewpoint.

Domain viewpoints : The domain characteristics and constraints that affect on requirements of the system sets the domain viewpoints. For example in Library system the rules that are to be followed for reserving the book(the fine or dues should be paid already or book must be returned within 1 week etc.) form the domain view point.

From these viewpoints the requirements can be discovered. From interactor viewpoint the system requirements can be obtained. The indirect viewpoints help in getting the

organizational requirements and constraints. The domain requirements provide domain constructs that need to be applied for the system.

View points in library management system is given by following Fig. 2.8.3

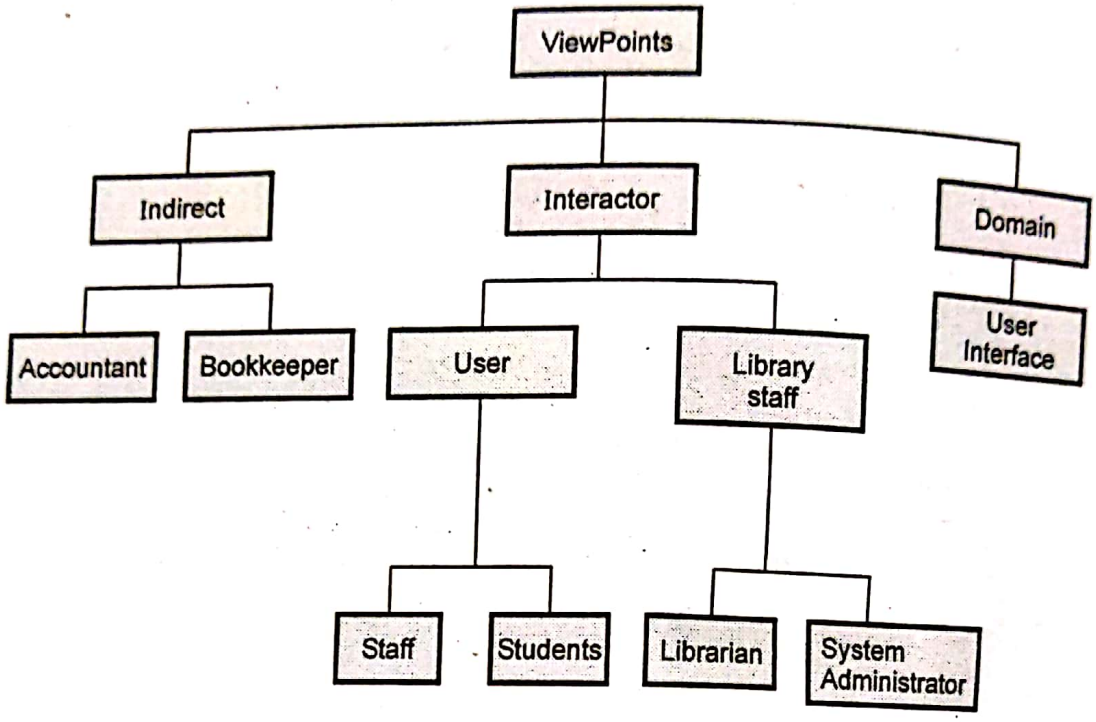


Fig. 2.8.3 Viewpoints for library system

4. Use cases

Use cases are the fundamental units of modelling language, in which functionalities are distinctly presented. The use case is a scenario based technique. Use cases help to identify individual interactions with the system. Use-cases are extensively used for requirements elicitation. By designing the proper use cases for different scenarios major reuirements of the system can be identified. The typical notations used in the use cases are as shown in Fig. 2.8.4.

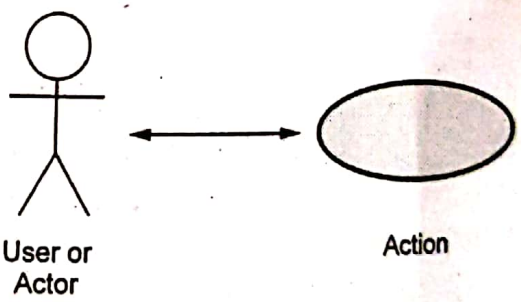


Fig. 2.8.4 Use-case notation

The use case for ATM system is as shown below.

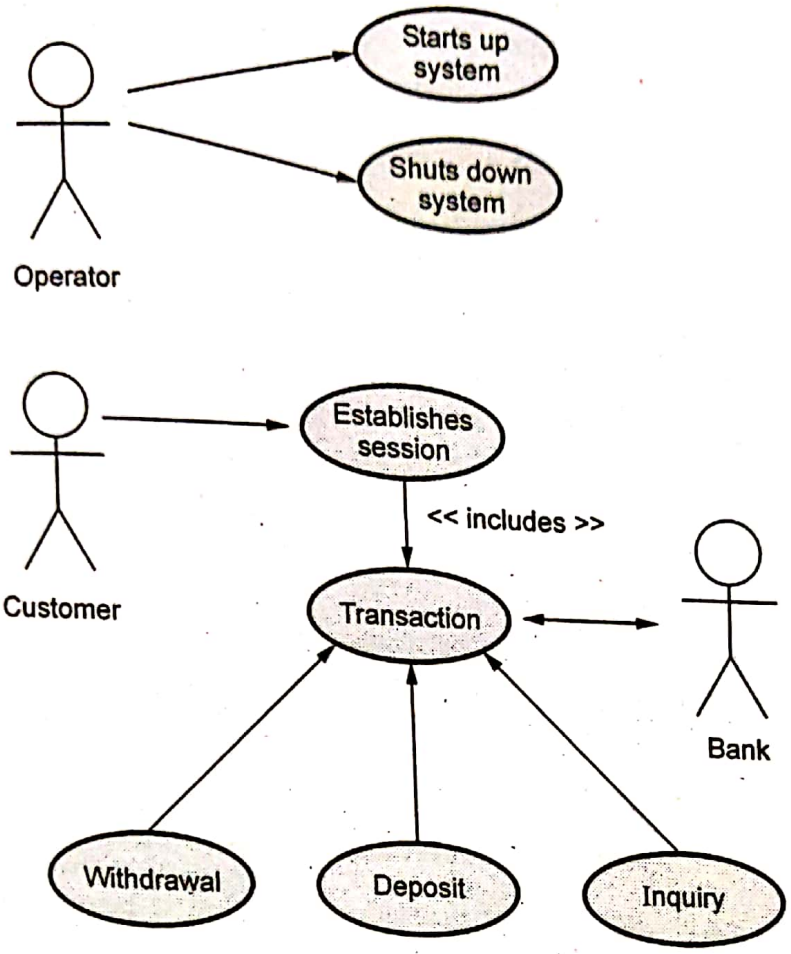


Fig. 2.8.5 Use cases for ATM system

Flow of events can be described as

System start up : The system is started when the operator turns on the switch. Initially the operator has to enter some amount of money in the cash dispenser. The connection with the bank gets established. Then only the servicing customer can begin.

System shutdown : The operator can shutdown the system only after confirming that there is no customer currently operating the ATM system. Then the connection to the bank gets disconnected.

Session : This use case starts when the user inserts the card. ATM pulls the card inside and reads it. Then further inquiry information is displayed on the display panel.

Transaction : The transaction use case is comprised of amount withdrawal, deposit and inquiry.

The interaction diagram for system startup is as given below.

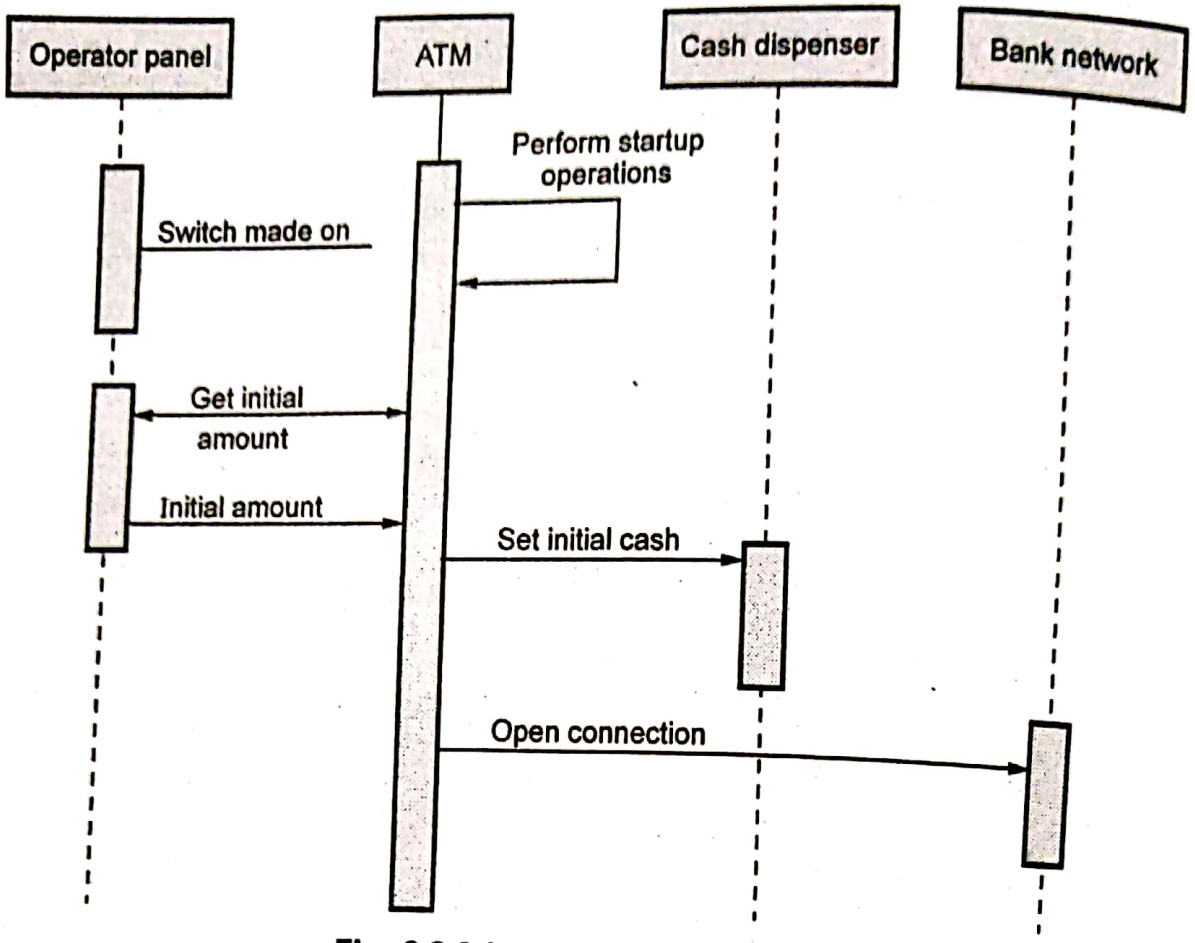


Fig. 2.8.6 Interaction diagram

This diagram depicts the sequence of operations that must be performed in order to accomplish the task.

5. Ethnography :

Ethnography is a technique of observation which is used to understand social and organizational requirements.

The system analyst imagines himself as a part of the working environment in which the system will be used. He then observes the day-to-day activities and notes down the needs and tasks required by the organization. The analysts thus finds the implicit requirements of the system.

People understand their own work, but they do not understand the relationship of that work with organization. Hence by ethnographic technique such requirements can be exposed.

Two types of requirements can be discovered by ethnography.

1. Requirements obtained from working style of people

These are the requirements that can be identified from the sequence of actions that a user is performing. For example in ATM system when the user enters the PIN, the card

validity action takes place. Unless and until the card gets validated there should not be any transaction processing. That means, it is required that a valid card, valid PIN must be entered for getting the money from ATM.

2. Requirements obtained from inter-activities performed by the people

Sometimes for finding the social requirements the other people's activities should be known. For example in ATM system the operator can not shutdown the system if some transaction is in processing.

Enthography can be combined with prototyping

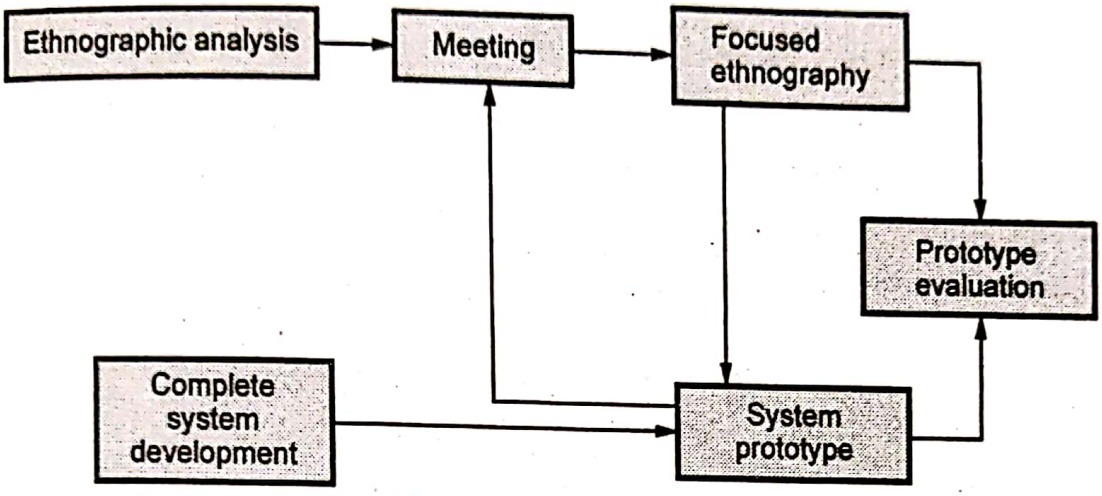


Fig. 2.8.7 Ethnographic process

- The ethnography informs the development of protoype using this information prototype refinement cycles can be used.
- Prototype focuses on ethnography by identifying problems and questions discussed by ethnographer.

Requirement elicitation problem

1. Stakeholders don't know what they want.
2. Stakeholders may have unrealistic expectations.
3. Stakeholders use their own language.
4. Different stakeholders may have different requirements.
5. Certain political factors may affect the requirements.
6. Business or economic changes create dynamic environment.

Example 2.8.1 *Perform requirements elicitation for watch system that facilitates to set time and alarm.*

Solution : Requirements elicitation is done for watch system by using following steps

1) **Requirements discovery :** By effective communication with customers the requirements can be identified.

2) **Requirements classification :** All unstructured requirements can be classified systematically and can be arranged in functional and non-functional requirements.

3) **Requirements prioritisation :** Some requirements are conflicting requirements. Hence they need to be priortised first. If some requirements are un-realistic, then they must be eliminated and only realistic requirements are collected.

The functional and nonfunctional requirements for watch system are -

Functional requirements

1) System allows to set time in 12:00 or 24:00 hour display format.
2) It allows to set alarm.
3) System allows to delete the already set alarm.
4) System allows to set timings for snooze.
5) System allows to stop alarm, during alarming.
6) The volume of alarm can be set by user.
7) It should allow to set more than one alarms.

Non-functional requirements

1) It should indicate battery low message if battery is low.
2) The system must be reliable and nobody should hack the system.

Use case diagram

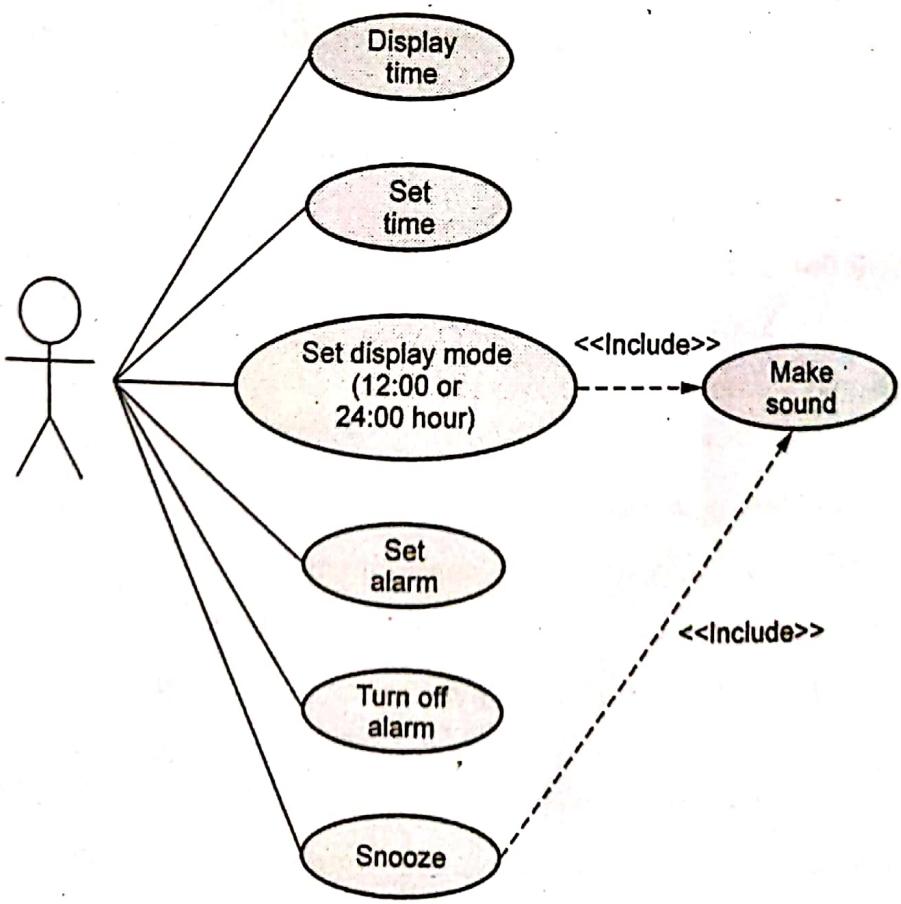


Fig. 2.8.8

Example 2.8.2 *Develop complete use case for the following :*
Making a withdrawal from ATM in a bank

Solution :

Use Case	ATM system
Primary Actor	Customer
Goal in Context	To monitor all the functionalities required to establish the session
Preconditions	ATM system has to be programmed and as to recognise and validate the PIN
Trigger	On completion of every activity a beep has to be generated by the system
Scenario	1. Customer observes the control panel 2. Customer enters the ATM card 3. Customer enters the PIN 4. Customer selects the operation - withdrawal 5. Customer collects the cash, statement, card etc
Exceptions	1. The control panel is not ready 2. PIN is incorrect 3. Card is not recognised 4. Insufficient balance 5. Limit for the transaction exceeds 6. Total number of allowed transactions per day
Priority	Essential and must be implemented in banking system
Secondary Actor	Administrator
Open Issues	1.Is there a need to display any additional information by the control panel ? 2. For how many time the PIN is allowed to enter on incorrect supplement. 3. How much time is allotted to the customer to pick the items like cash or card after completion of operation ?

Example 2.8.3 *Develope complete use case for the following : Searching for books based on title in an on-line book store.*

Solution :

Use Case	On-line book store
Primary Actor	Customer
Goal in Context	To monitor all the functionalities required to establish the session

Scenario	1. Customer logs in to purchase the book online 2. Customer browses through the catalog based on title of the book 3. Customer selects the book and adds it to the shopping cart. 4. The contents of the shopping cart are displayed. 5. The book details and shipping information is confirmed by the customer. 6. The customer pays for the book
Extensions	6a The customer pays by credit card (include relationship) 6b The customer pays by the cheque (include relationship) 6c The customer pays by cash (include relationship)

Example 2.8.4 *A coffee vending machine dispenses coffee to customers. Customers order coffee by selecting a recipe from a set of recipes. Customers pay for the coffee using coins. Change is given back, if any, to the customers. The 'Service Assistant' loads ingredients (coffee. powder, milk, sugar, water, chocolate) into the coffee machine. The 'Service Assistant' adds a recipe by indicating the name of the coffee, the units of coffee powder, milk, sugar, water and chocolate to be added as well as the cost of the coffee.*

The Service Assistant can also edit and delete a recipe.

i) Develop the use case diagram for the specification above. (Marks 6)

ii) For any two scenarios draw an activity diagram and sequence diagram. (5+5 Marks)

AU : Dec.-13, Marks 16

Solution : i) The use case diagram is as given below -

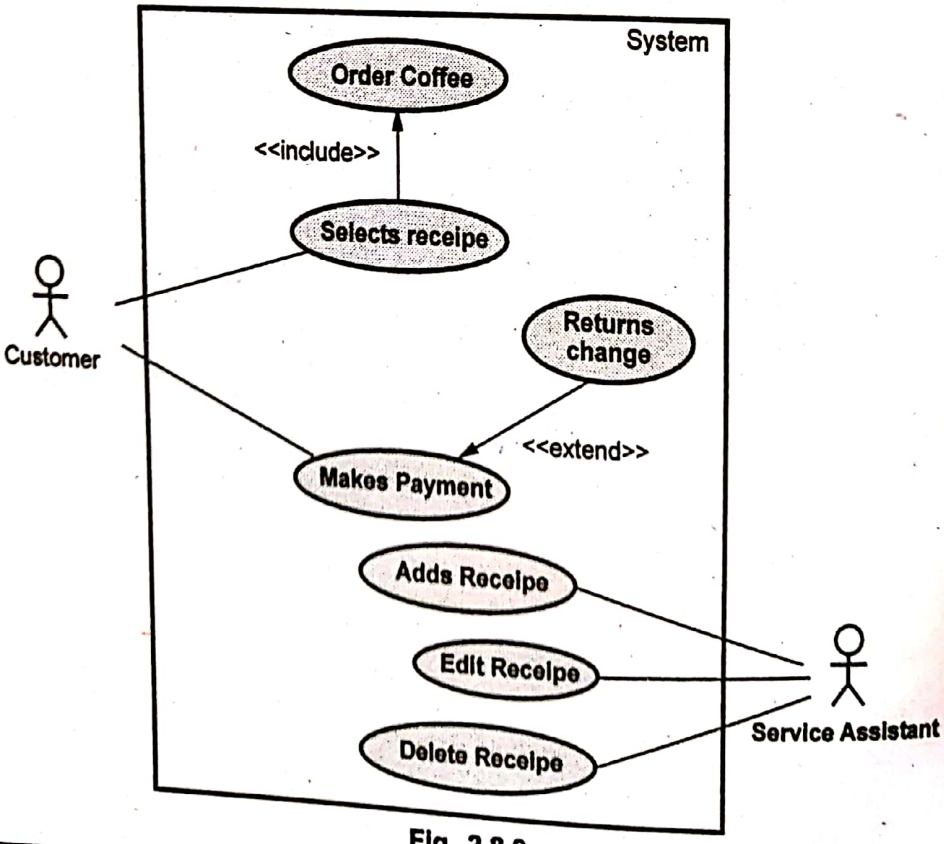


Fig. 2.8.9

ii) Activity Diagram

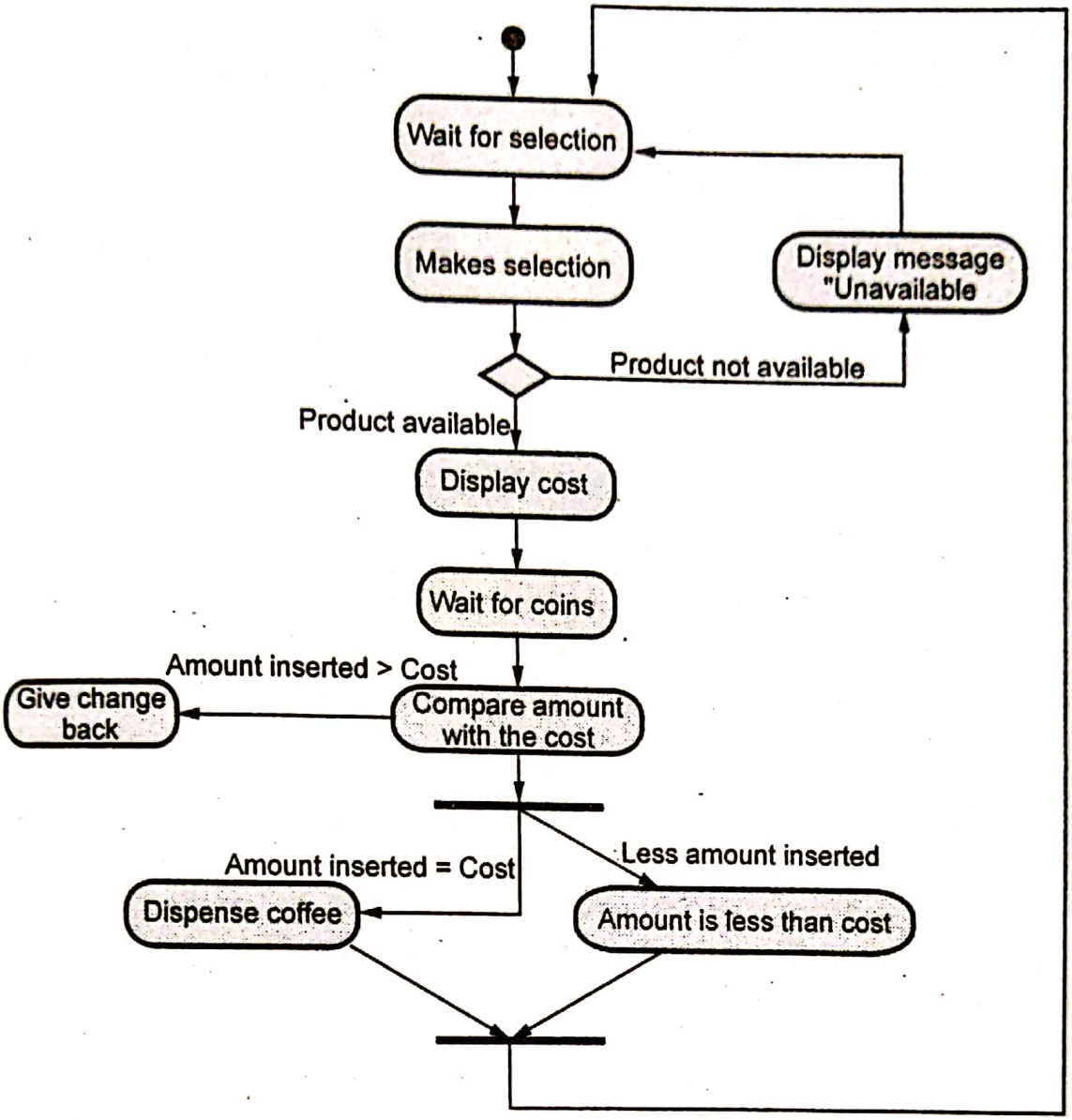


Fig. 2.8.10 Activity diagram for customer orders for the coffee

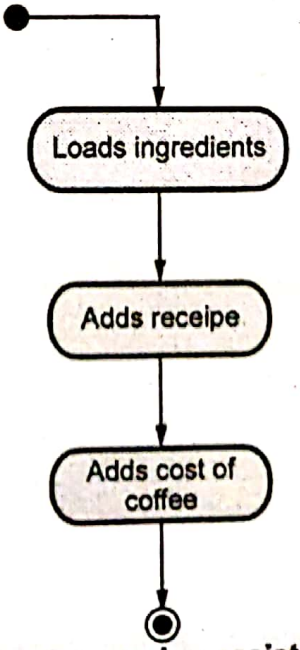


Fig. 2.8.11 Activity diagram for service assistant for adding the receipe

Sequence Diagram :

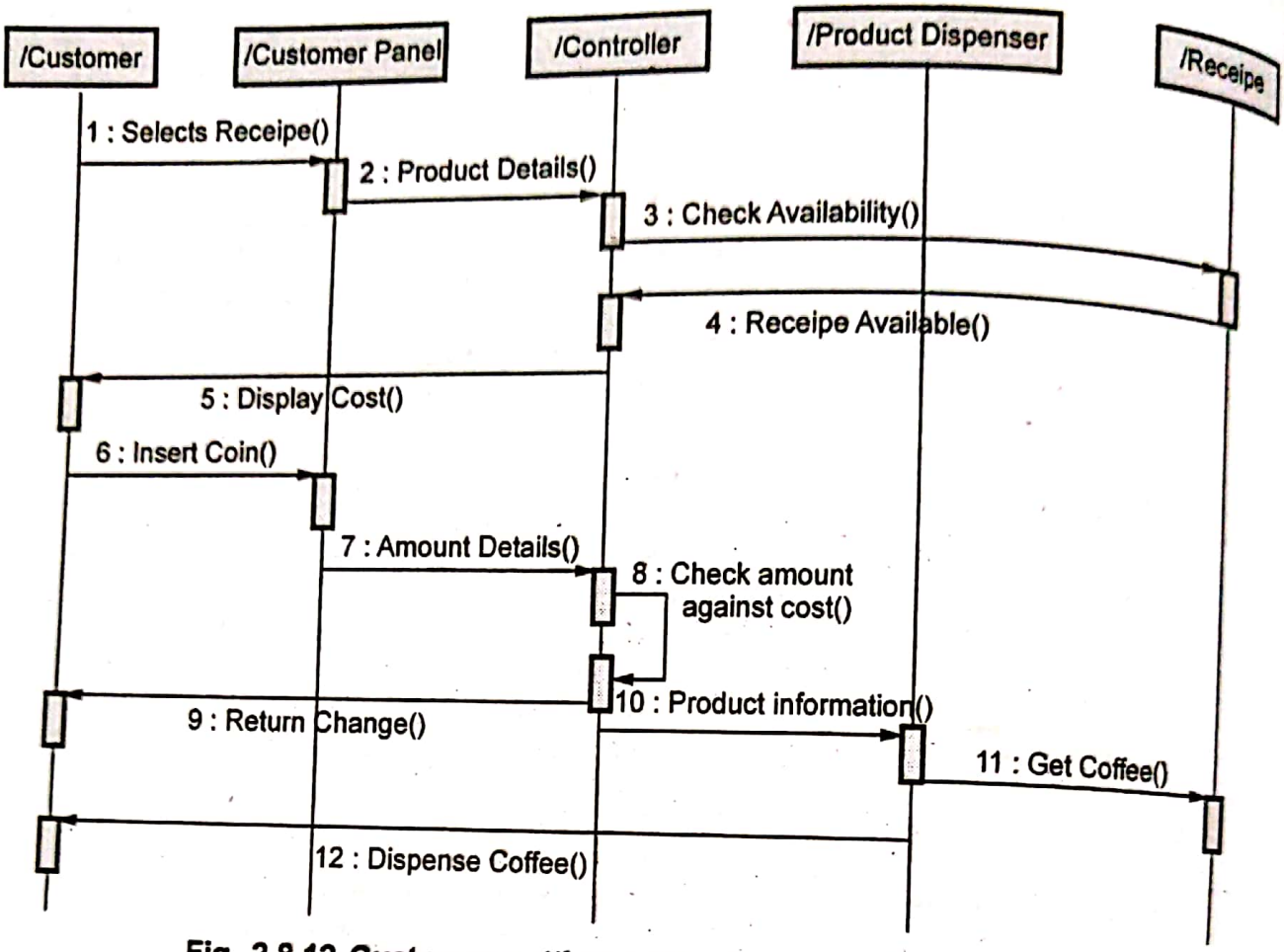


Fig. 2.8.12 Customer getting coffee from vending machine

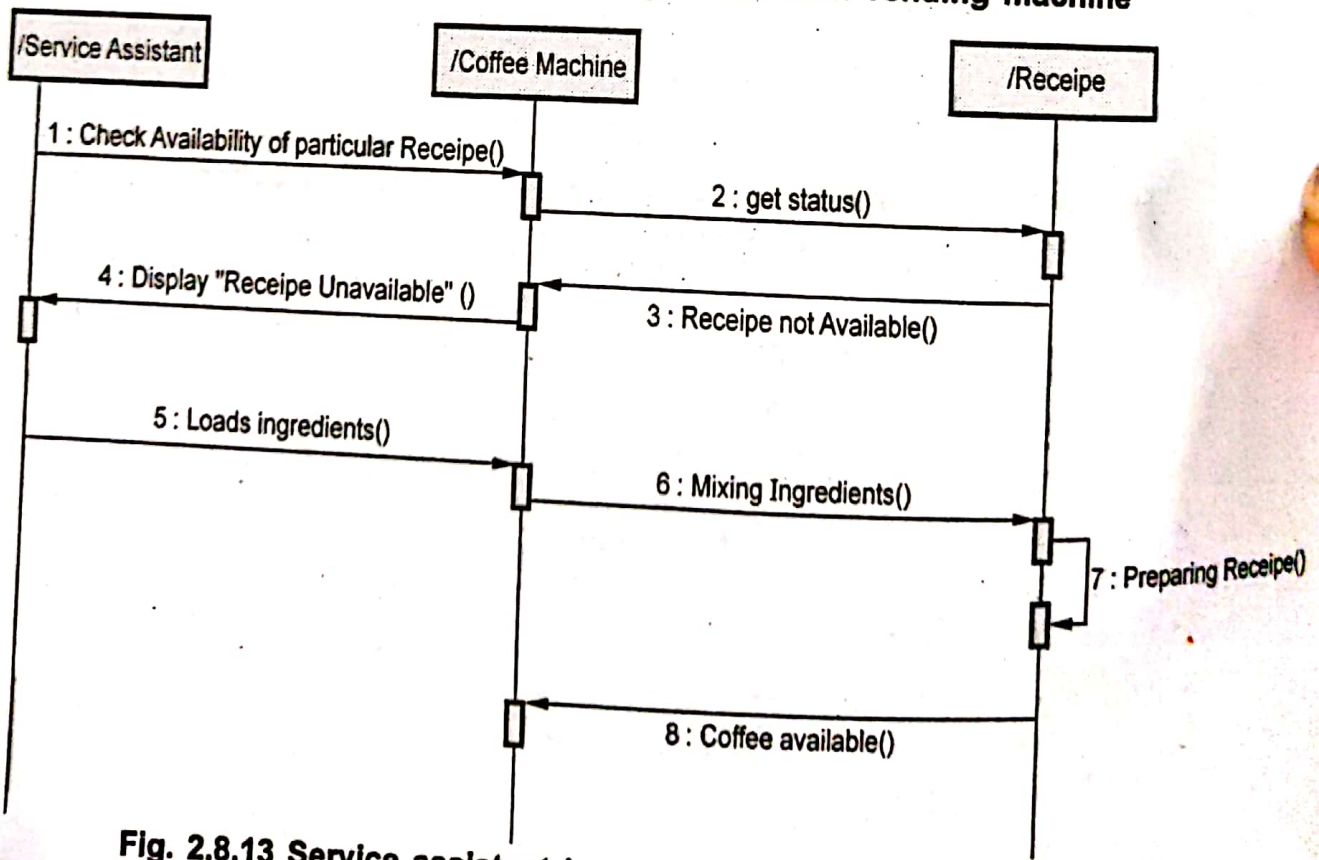


Fig. 2.8.13 Service assistant loads the ingredient and prepare coffee

Review Questions

1. *What is requirements elicitation ? Briefly describe the various activities performed in requirements elicitation phase with an example of a watch system that facilitates to set time and alarm.*
   AU : Dec.-16, Marks 16
2. *What is requirement engineering ? State its process and explain requirements elicitation problem.*
   AU : May-08, Marks 8
3. *What is requirement elicitation ? Briefly describe various activities performed in requirements elicitation phase with an example of watch system that facilitates to set time and alarm.*
   AU : Dec.-16, May-18, Marks 16
4. *Write a note on what are difficulties in elicitation, requirement elicitation.*
   AU : May-17, Marks 5

2.9 Requirement Validation

Requirement validation is a process in which it is checked that whether the gathered requirements represent the same system that customer really wants.

- In requirement validation the requirement errors are fixed. Requirements error costs are high so validation is very important. Fixing a requirements error after delivery may cost up to 100 times the cost of fixing an implementation error.
- Requirement checking can be done in following manner
  1. **Validity :** Does the system provide the functions which best support the customer's needs?
  2. **Consistency :** Are there any requirements conflicts ?
  3. **Completeness :** Are all functions required by the customer included ?
  4. **Realism :** Can the requirements be implemented according to budget and technology ?
  5. **Verifiability :** Can the requirements be checked ?

Requirements validation techniques

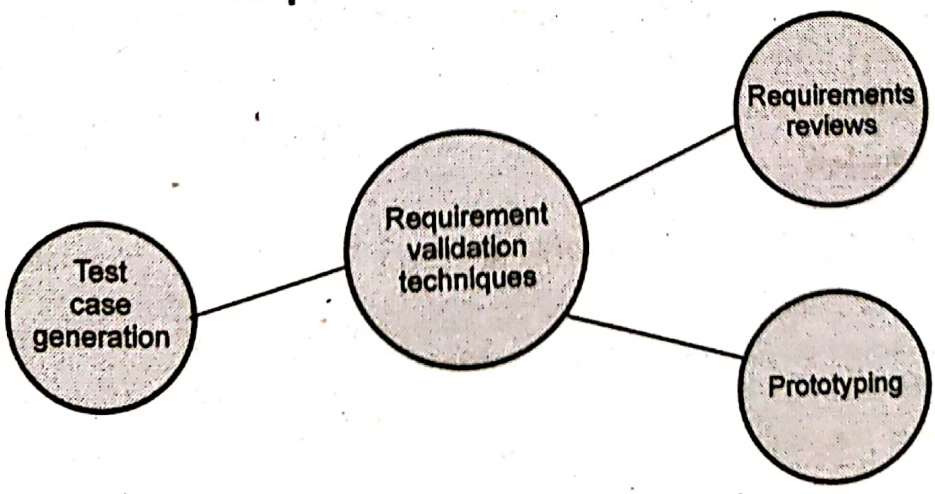


Fig. 2.9.1 Requirement validation technique

1. Requirements reviews : Requirement review is a systematic manual analysis of the requirements.

- The requirement review should be taken only after formulation of requirement definition. And both the customer and contractor staff should be involved in reviews.
- Reviews may be formal (with completed documents) or informal.
- Good communications should take place between developers, customers and users. Such a healthy communication helps to resolve problems at an early stage.

2. Prototyping : The requirements can be checked using executable model of system.

3. Test-case generation : In this technique, the various tests are developed for requirements. The requirement check can be carried out with -

- **Verifiability :** Is the requirement realistically testable ?
- **Comprehensibility :** Is the requirement properly understood ?
- **Traceability :** Is the origin of the requirement clearly stated ?
- **Adaptability :** Can the requirement be changed without a large impact on other requirements ?

2.10 Requirement Management

AU : May-16, Marks 12

Definition : Requirements management is the process of managing changing requirements during the requirements engineering process and system development.

Why requirements get change ?

- Requirements are always incomplete and inconsistent. New requirements occur during the process as business needs change and a better understanding of the system is developed.
- System customers may specify the requirements from business perspective that can conflict with end user requirements.
- During the development of the system, its business and the technical environment may get changed.

2.10.1 Enduring and Volatile Requirements

Eduring requirements : These are the stable requirements that are derived from the core activity of the organization. These requirements are dependent upon the application domain of the software. For example: For banking system, transfer of money from one account to another is the enduring requirement.

Volatile requirements : For certain requirements if there is a possibility that those requirements may get changed during the development stage or after the system becomes operational, then such requirements are called volatile requirements.

Types of volatile requirements

1. Mutable requirements : If due to change in the environment, if the requirements get changed then such requirements are called as mutable requirements. For example: In the library management system, if the traditional library is turned into a digital library(containing e-books,on-line articles,e-learning or video conferencing facilities) then requirements for the library automation software will be changed.

2. Consequential requirements : Requirements that gets changed due to introduction of computer based systems, such requirements are called consequential requirements.

For example : In a tours and travel agencies, due to on-line ticket booking facilities requirements get changed. Such changed requirements are consequential requirements.

3. Emergent requirements : Due to customer's understanding of the system during the development stage certain requirements may get changed. Such types of requirements are called emergent requirements.

4. Compatibility requirements : Requirements which depend upon the specific system or business process in an organization are called compatible requirements

2.10.2 Requirements Management Planning

Following things should be planned during requirement process.

- Traceability is concerned with relationship between requirements their sources and the system design. Using traceability the requirement finding becomes easy.

Various types of traceability are

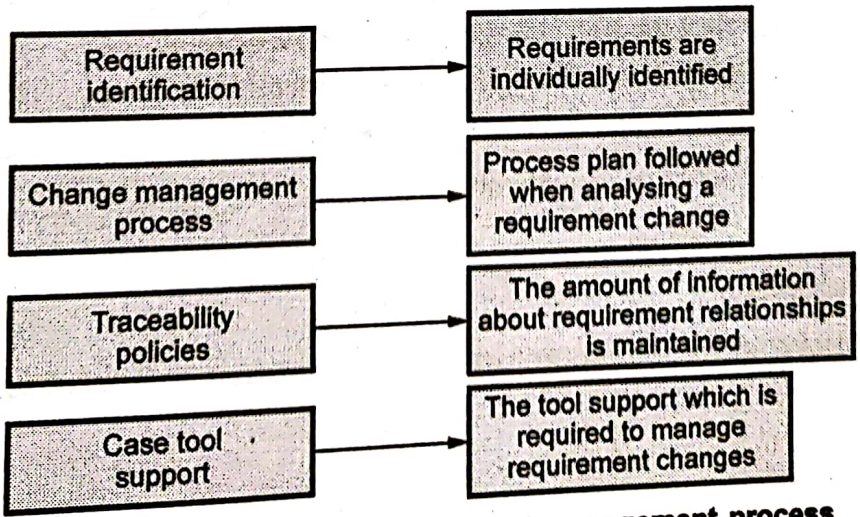


Fig. 2.10.1 Planning in requirement management process

1. **Source traceability :** These are basically the links from requirement to stakeholders who propose these requirements.
2. **Requirements traceability :** These are the links between dependent requirements.
3. **Design traceability :** These are the links from requirements to design.

- Traceability information is typically represented by a data structure **Traceability matrix.** If one requirement is dependant upon the other requirement then in that row-column cell 'D' is mentioned and if there is a weak relationship between the requirements then corresponding entry can be denoted by 'R'.

For example

Requirement ID	A	B	C	D	E	F
A		D			R	
B			D			
C				R		
D			D			R
E						
F	R			D		

For mentioning the traceability of small systems usually the traceability matrix is maintained.

- Case tool support is required for
  1. Requirement storage 2. Change management 3. Traceability management

2.10.3 Requirement Change Management

The requirement change management is a technique that can be applied to the processes in which requirements may get changed. The need for requirement change management is that even though the changes are made consistently in the requirements it is possible to incorporate those changes in a controlled manner. The requirement change management process can be applied in three stages.

1. Problem analysis and change specification
2. Change analysis and costing
3. Change implementation

Fig. 2.10.2 shows the stages of requirement change management process.

1. Problem analysis and change specification : When requirement change request is made for some particular problem then the problem with older requirement is mentioned or sometimes simply change specification is given. Then first of all, problem analysis or change specification is analysed in order to validate the required change. If necessary the feedback of this analysis is given to the person who is demanding such change.

2. Change analysis and costing : Following actions are carried out in this stage :

i) The effect of change is assessed using traceability information.

ii) The cost of such change is estimated.

iii) After getting the cost of changes the decision is made on whether to go for implementation of these changes or not.

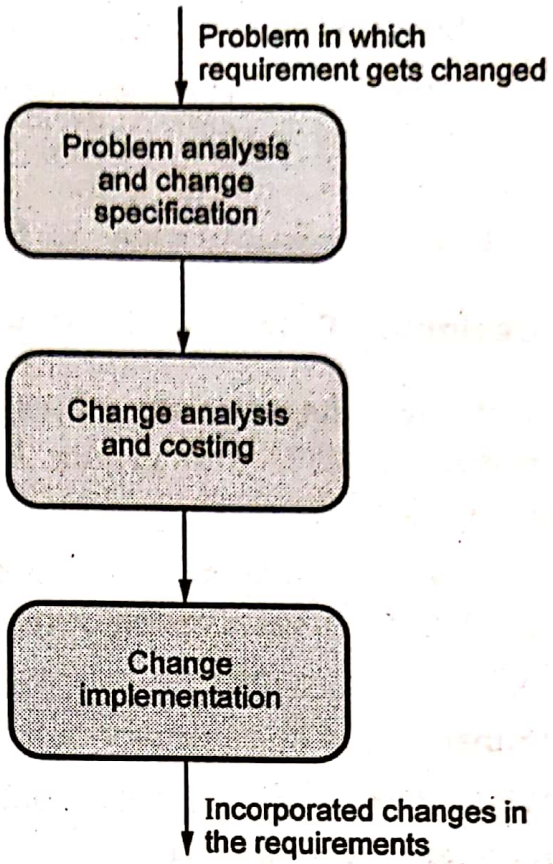


Fig. 2.10.2 Requirement change management process

3. Change implementation : Once it is decided to implement the proposed changes in the requirement, the requirement document has to be modified. The requirement document has to either re-written or re-organised. This can be achieved by making the modularity in the requirement specifications, so that it becomes easy to change individual section without affecting other part of requirement document.

Review Question

1. *Describe the requirements change management process in detail.* **AU : May-16, Marks 12**

2.11 Structured System Analysis

AU : Dec.-13,15,16,17, May-08,10,15,17,18, Marks 16

The structured system analysis is a technique in which the system requirements are converted into specifications and then into computer programs, hardware configurations and related manual procedures.

- The structured analysis is mapping of problem domain to flows and transformations.

- The system can be modeled using :
  - Entity relationship diagram are used to represent the data model.
  - Data flow diagram and control flow diagrams are used to represent the functional model.
- Along with system modeling the specification can be written for the system using
  1. Process Specification 2. Control Specification

2.11.1 Designing Data Flow Diagrams

- The data flow diagrams are used to model the information and function domain. Refinement of DFD into greater levels helps the analyst to perform functional decomposition.
- The guideline for creating a DFD is as given below –
  1. Level 0 DFD i.e. Context level DFD should depict the system as a single bubble.
  2. Primary input and primary output should be carefully identified.
  3. While doing the refinement isolate processes,data objects and data stores to represent the next level.
  4. All the bubbles (processes) and arrows should be appropriately named.
  5. One bubble at a time should be refined.
  6. Information flow continuity must be maintained from level to level.

Symbols used in DFD

Symbol	Notation
External Entity : External entities are objects outside the system, with which the system communicates. External entities are sources and destinations of the system's inputs and outputs.	External Entity
Process : A process transforms incoming data flow into outgoing data flow.	Level / Process
Transition : It represents the flow of information from one entity to another.	→
Data Store : Datastores are repositories of data in the system. They are sometimes also referred to as files or databases.	Data Store

- A simple and effective approach to expand the level 0 DFD to level 1 is to perform "grammatical parse" on the problem description. Identify nouns and verbs from it. Typically nouns represent the external entities, data objects or data stores and verbs represent the processes. Although grammatical parsing is not a foolproof but we can gather much useful information to create the data flow diagrams.

2.11.2 Rules for Designing DFD

1. No process can have only outputs or only inputs. The proces must have both outputs and inputs.

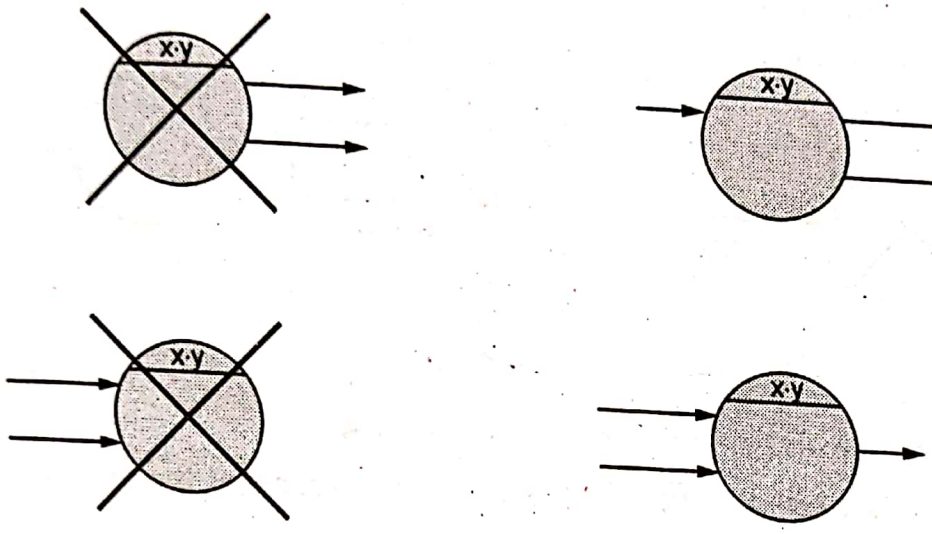


Fig. 2.11.1

2. The verb phrases in the problem description can be identified as processes in the system.
3. There should not be a direct flow between data stores and external entity. This flow should go through a process.

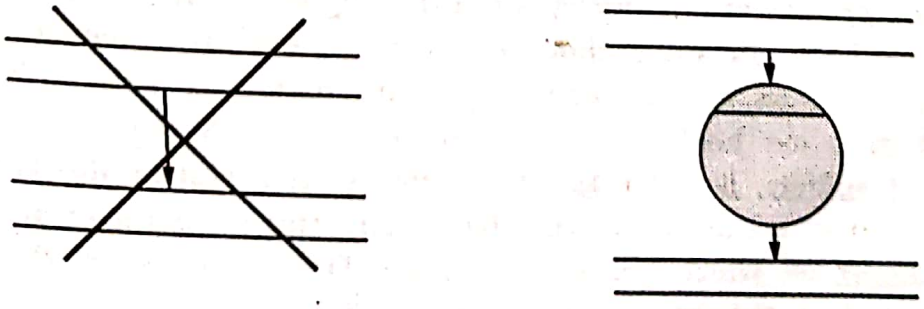


Fig. 2.11.2

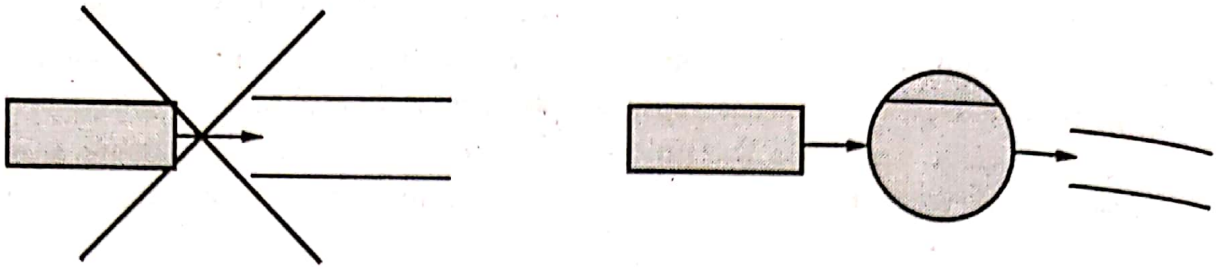


Fig. 2.11.3

4. Data store labels should be noun phrases from problem description.
5. No data should move directly between external entities. The data flow should go through a process.

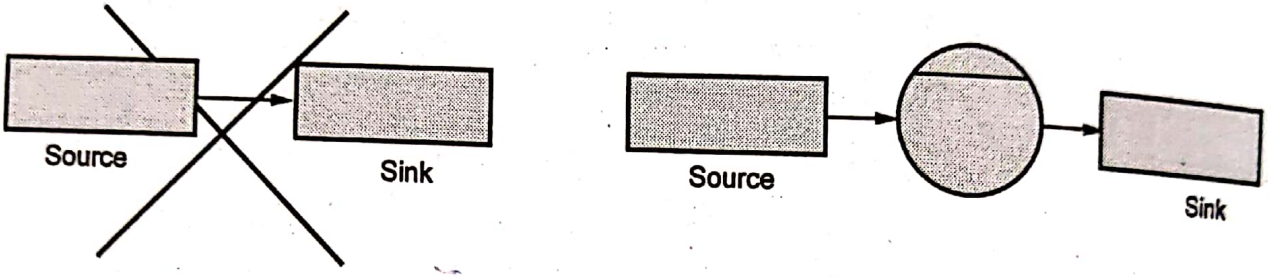


Fig. 2.11.4

6. Generally source and sink labels are noun phrases.
7. Interactions between external entities is outside scope of the system and therefore not represented in DFD.
8. Data flow from process to a data store is for updation/insertion/deletion.
9. Data flow from data store to process is for retrieving or using the information.
10. Data flow labels are noun phrases. from problem description.

Example 2.11.1 *Design DFD for library management system for level 0 DFD and level 1 DFD.*

AU : Dec.-16, Marks 8

Solution : A student comes to a library for borrowing book. The student makes the book request by giving book title and author name. The student has to submit his library card to the library. Sometimes student may simply give topic and demand for a book (For example "just give me book on data structure"). The library information system maintains list of authors, list of titles, list of topics. This system also keeps record of topics on which books are available with the system. This system maintains information about shelf number on which books are located. Finally the list of demanded book should be displayed, on the console for ease of selection.

The first level of DFD can be

Solution :

In this DFD the whole system is represented with the help of input, processing and output. The input can be -

i) Student requests for a book hence **Book request.**

ii) To show identity of the student he/she has to submit his/her Library card, hence **Library card.** The processing unit can be globally given as

Library information system

The system will produce following outputs-

i) The demanded book will be given to student. Hence **Book** will be the output.

ii) The library information system should display **demanded book information** which can be used by customer while selecting the book.

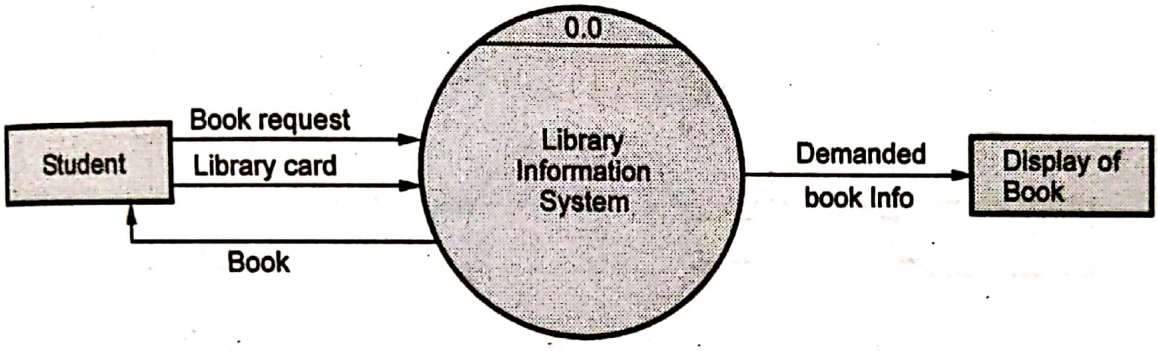


Fig. 2.11.5 Level 0 DFD (Context level DFD)

Level 1 DFD

In this level, the system is exposed with more processing details. The processes that need to carried out are -

i) Delivery of Book ii) Search by Topic

These processes require certain information such as List of Authors, List of Titles, List of Topics, the book selves from which books can be located. This kind of information is represented by **data store.**

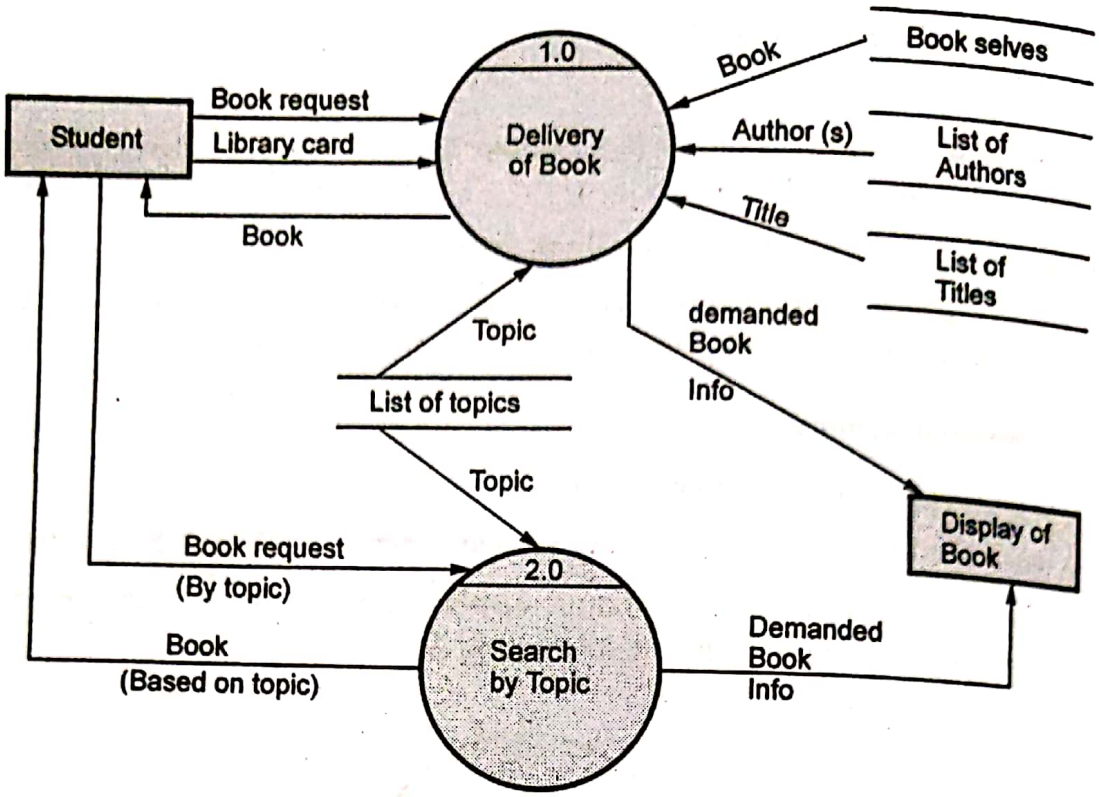


Fig. 2.11.6 Level 1 DFD

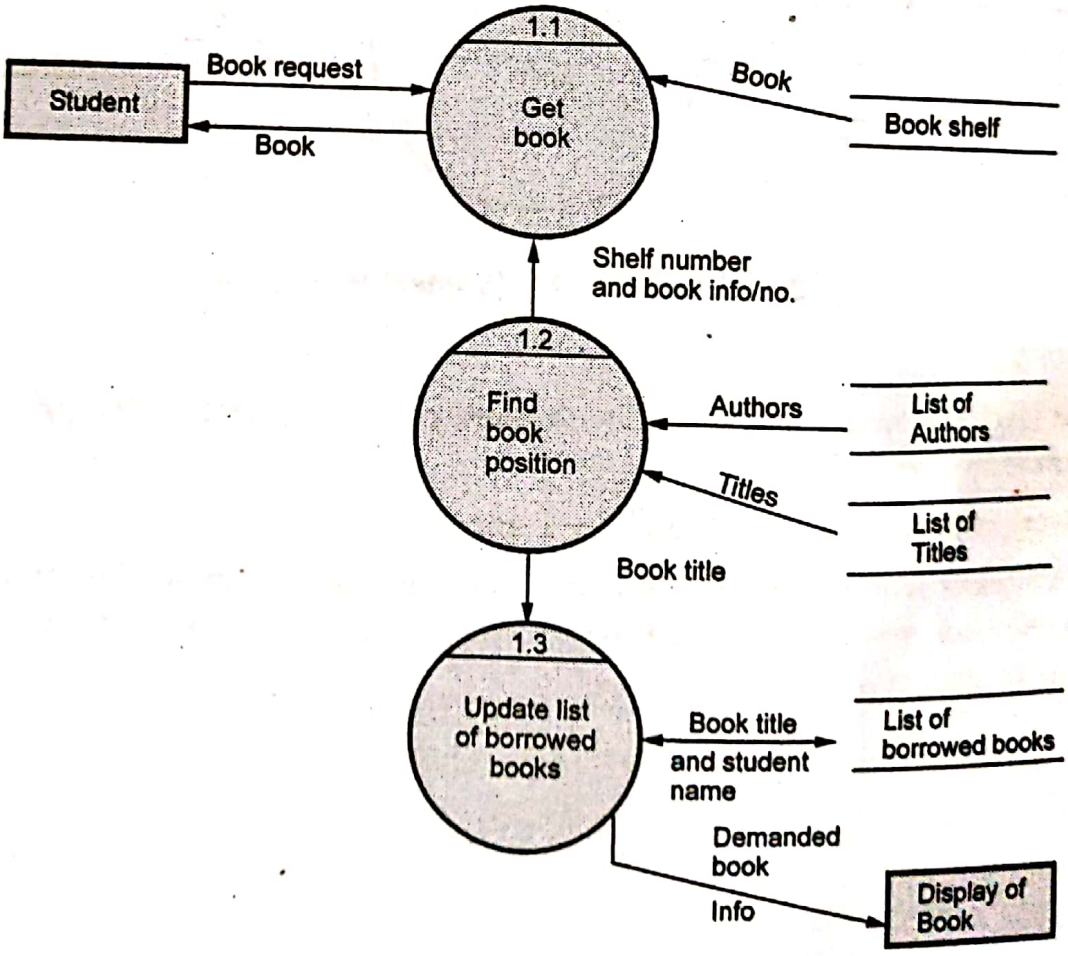


Fig. 2.11.7 Level 2 DFD

Example 2.11.2 *Draw use case and data flow diagrams for a restaurant system. The activities of the restaurant system are listed below.*

Receive the customer food orders, produce the customer ordered foods, serve the customer with their ordered foods, collect payment from customers, store customer payment details, order raw materials for food products, pay for r **AU : Dec.-16, Marks 8, May-15, Marks 16**

Solution : A customer goes to a restaurant and orders for the food. The food order is noted down carefully and this order is sent to kitchen for preparing the required food.

This restaurant has to manage one housekeeping department which maintains sold items and inventory data. The daily information about sold items and inventory deplition amount is used to generate a management report. Finally this management report is given to restaurant manager.

In this level, the system is designed globally with input and output. The input to Food ordering system are -

1. As customer orders for the food. Hence food order is an input.

Level 0 DFD

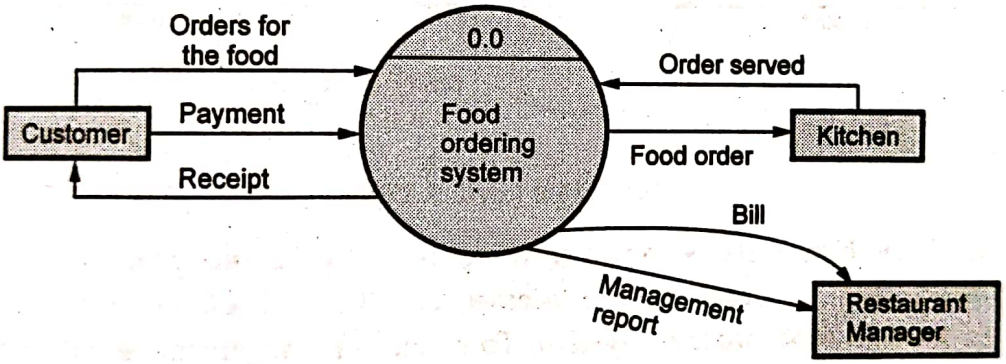


Fig. 2.11.8 Level 0 DFD (Context level DFD)

The output to food ordering system are -

1) Receipt.

2) The food order should be further given to kitchen for processing the order.

3) Bill and management report is given to restaurant manager.

Level 1 DFD

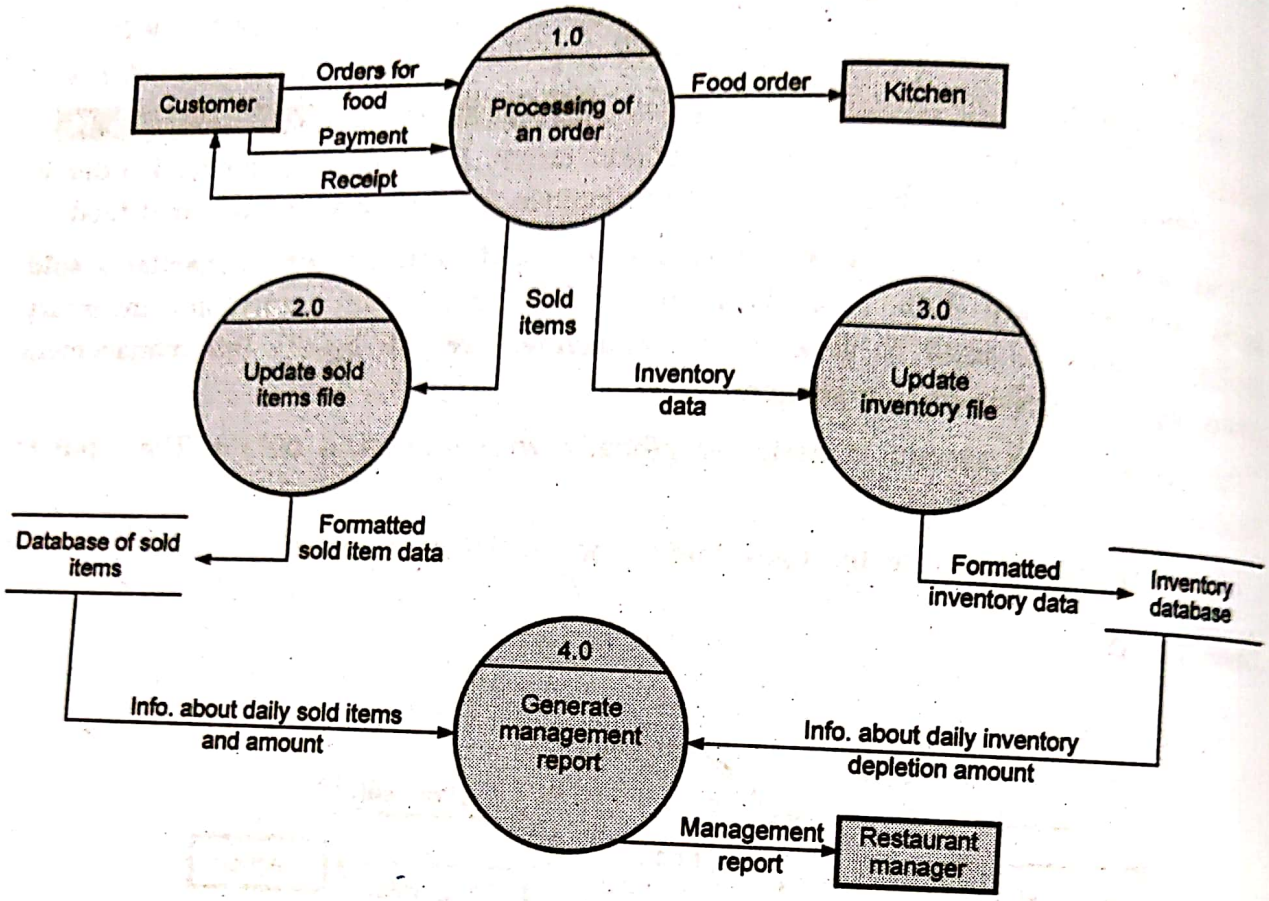


Fig. 2.11.9 Level 1 DFD

In this level, the bubble 0.0 is shown in more detail by various processes. The process 1.0 is for processing an order. And processes 2.0, 3.0 and 4.0 are for housekeeping activities involved in food ordering system. To create a management report there should be some information of daily sold items. At the same time inventory data has to be maintained for keep track of 'instock' items. Hence we have used two **data stores** in this DFD -

1. Database of sold items
2. Inventory database

Finally management report can be prepared using daily sold details and daily inventory deplition amount. This management report is given to restaurant manager.

In this DFD we will elabortate "processing of order".

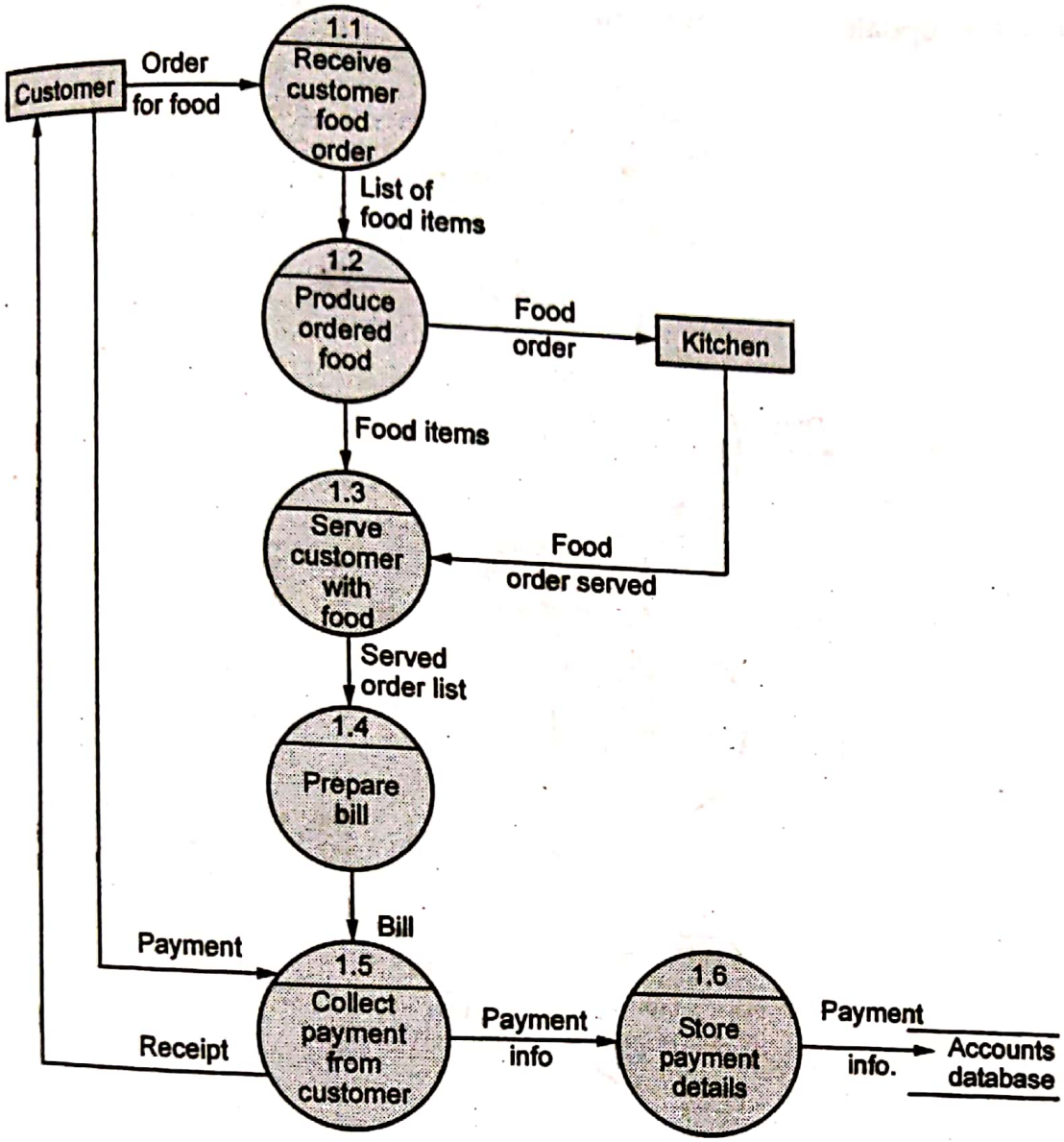


Fig. 2.11.10

The DFA for "update for inventory file " is shown in detail as follows -

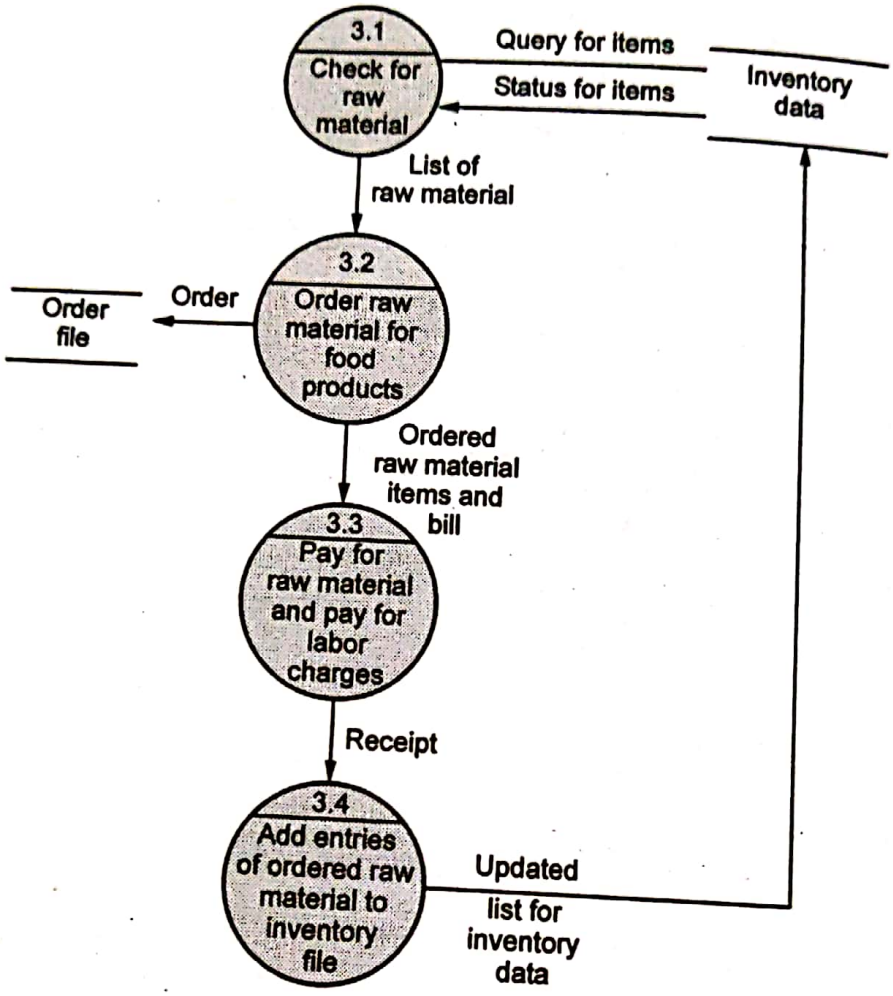


Fig. 2.11.11

Level 2 DFD :

In this DFD we will elaborate "Generate management report" activity in more detail. For generating management report we have to access sold items data and inventory data. Then aggregate both solid items data and inventory data. Total price of each item has to be computed. Then from these calculation a management report has to be prepared and given to the restaurant manager. These details can be shown in this DFD. (See Fig. 2.11.12 on next page.)

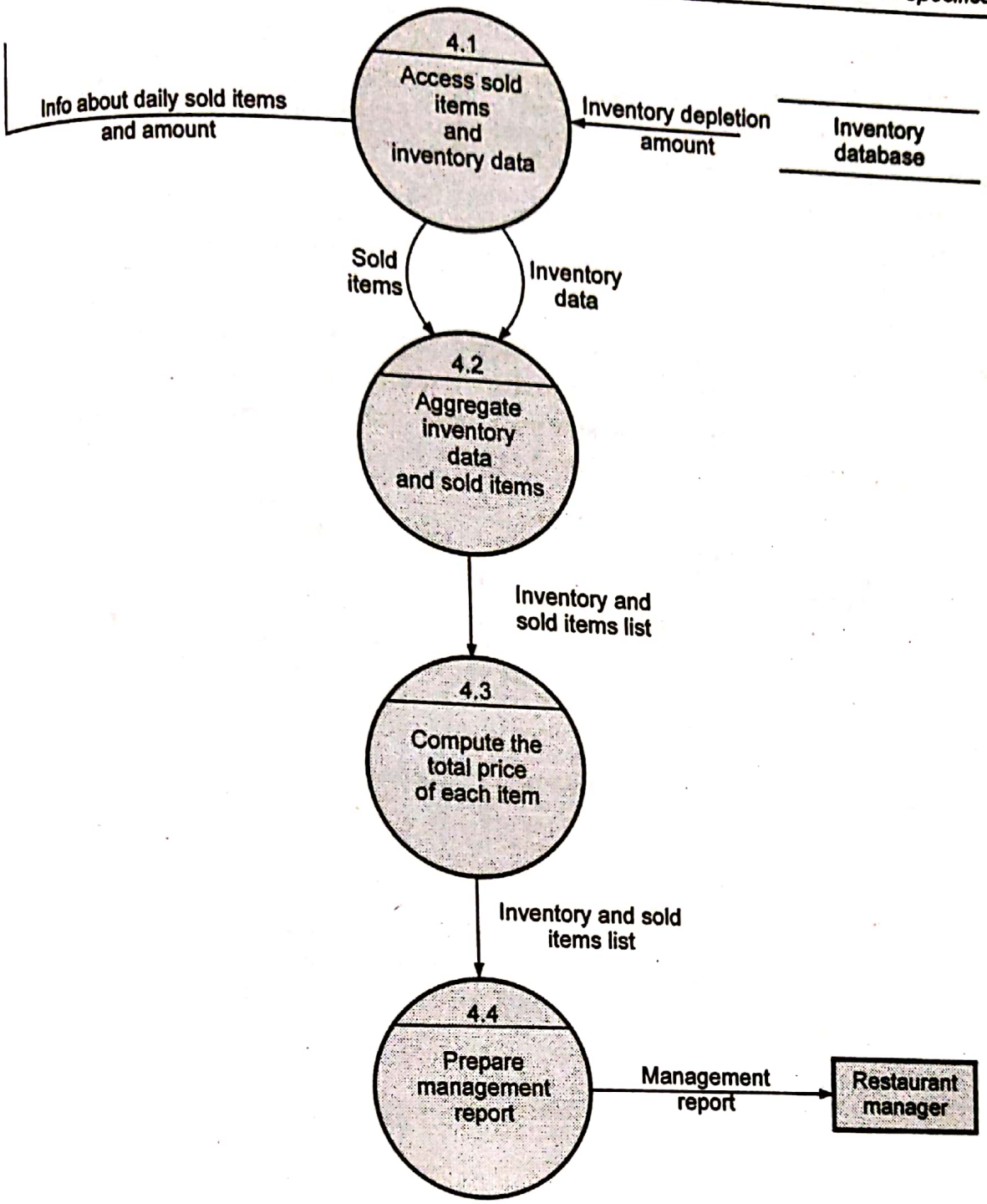


Fig. 2.11.12 Level 3 DFD

Example 2.11.3 *Prepare a CFD for a Books order processing system.*

Solution : The **customers** in this system are book sellers who do not make a stock of books. As the **orders** come the books are demanded from **publishers** directly.

As per the problem description it is clear that the input to this system is 'customer' who places an order and output will be purchase order given to publisher.

Now, we will do more detailing for order processing when an order is received by the system, it will be first verified using the books database. Credit rating will be decided by checking customer details (i.e. whether the customer is regular customer or

whether he is new). For submitting a batch of orders we have to maintain pending orders list as well. There may be the order for books which are published by different publishers, in such a case database for different publishers need to be maintained by the system. This list of purchase orders is maintained by the system and then after verifying the shipment a shipping note will be given to the customer. The level 1 decomposition is as shown in Fig. 2.11.14.

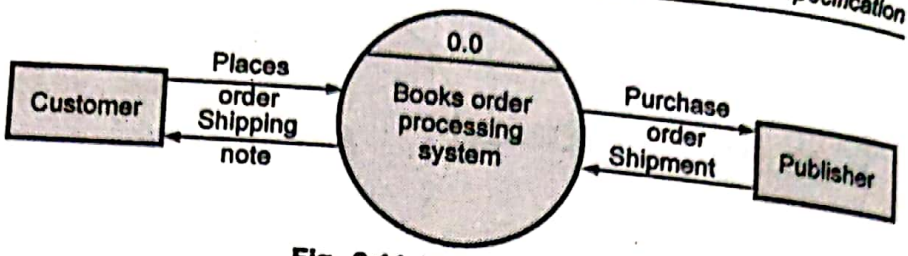


Fig. 2.11.13 Level 0 CFD

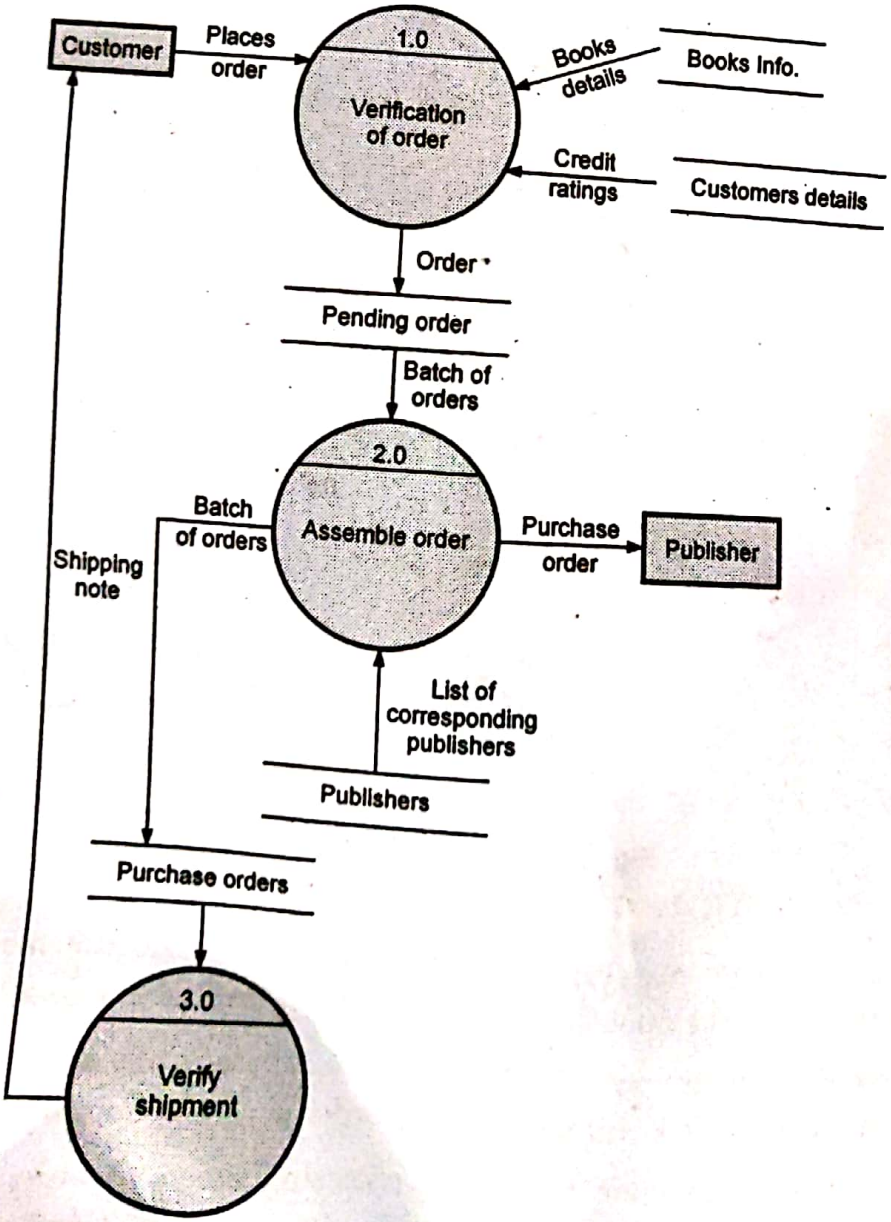


Fig. 2.11.14 Level 1 CFD

We can further decompose the system by elaborating the process **Assemble order.** In assemble order we

- First get details of total copies per title.
- Then prepare purchase order accordingly for corresponding publishers.
- Send these purchase orders to publishers.
- These purchase order details should be stored in the system. Let us draw level 2 CFD for these details.

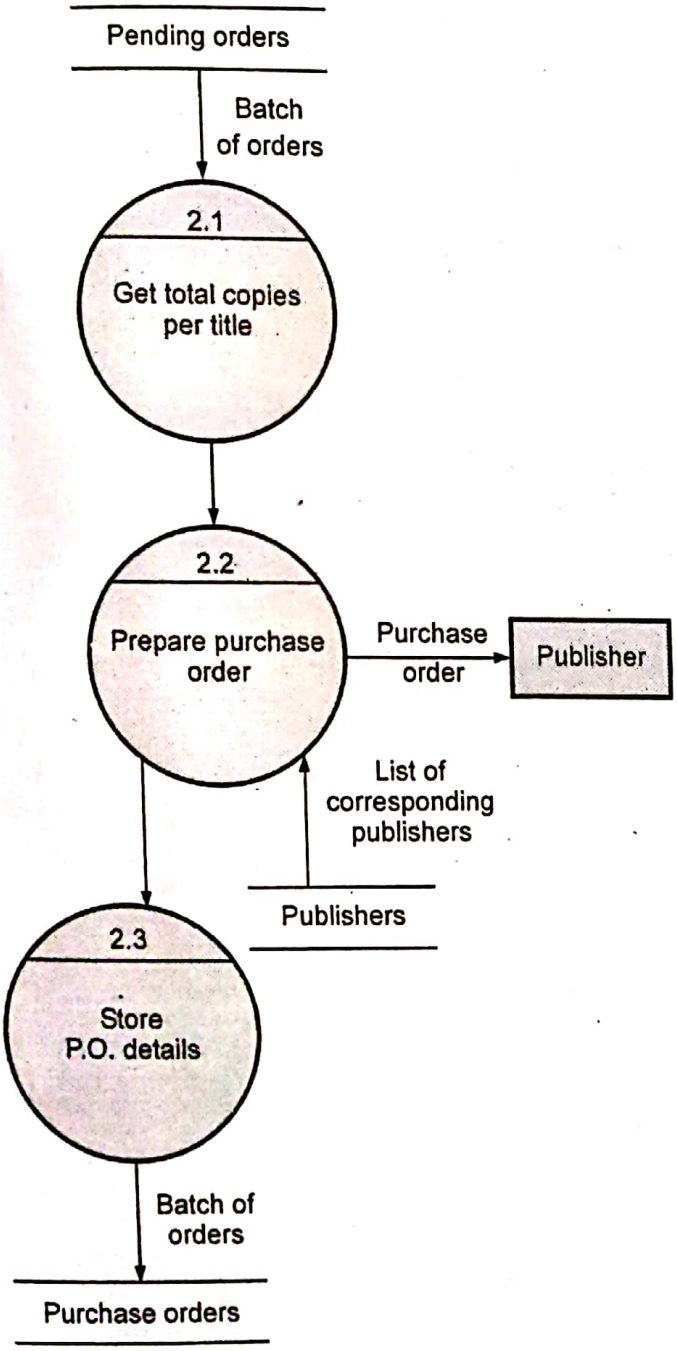


Fig. 2.11.15

Example 2.11.4 *Design DFD for Hotel reservation system.*

Solution : In a hotel reservation system, a **customer** can make online booking for a hotel, by specifying the accomodation requirements such as type of room (AC/Non AC/One bed/Two bed), total number of rooms, duration of stay. The system then **selects** a suitable hotel as per customer's requirements. If such a hotel is found then the **availability** of rooms in that hotel is **checked.** The **charges are calculated** for the selected requirement and these are acknowledged to the customer. If the customer is satisfactory about the selection made by the system then he **confirms** the reservation.

The system then gives these booking details to the corresponding **hotel.** Design DFD for this system.

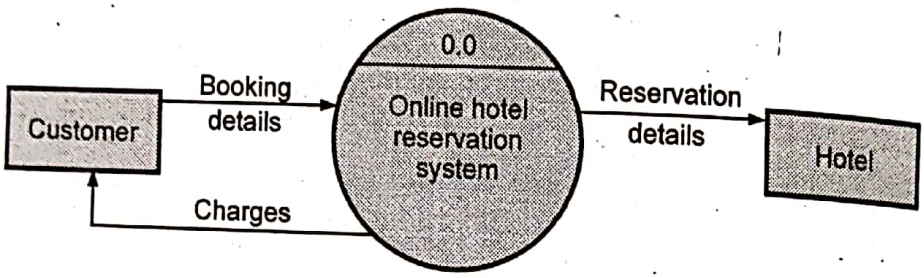


Fig. 2.11.16 Level 0 DFD

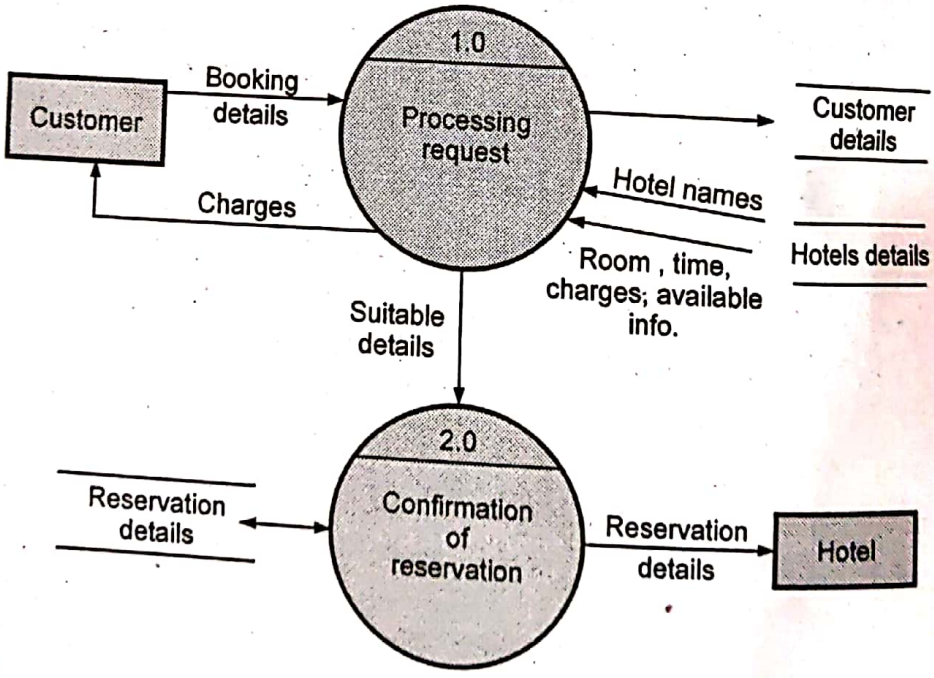


Fig. 2.11.17 Level 1 DFD

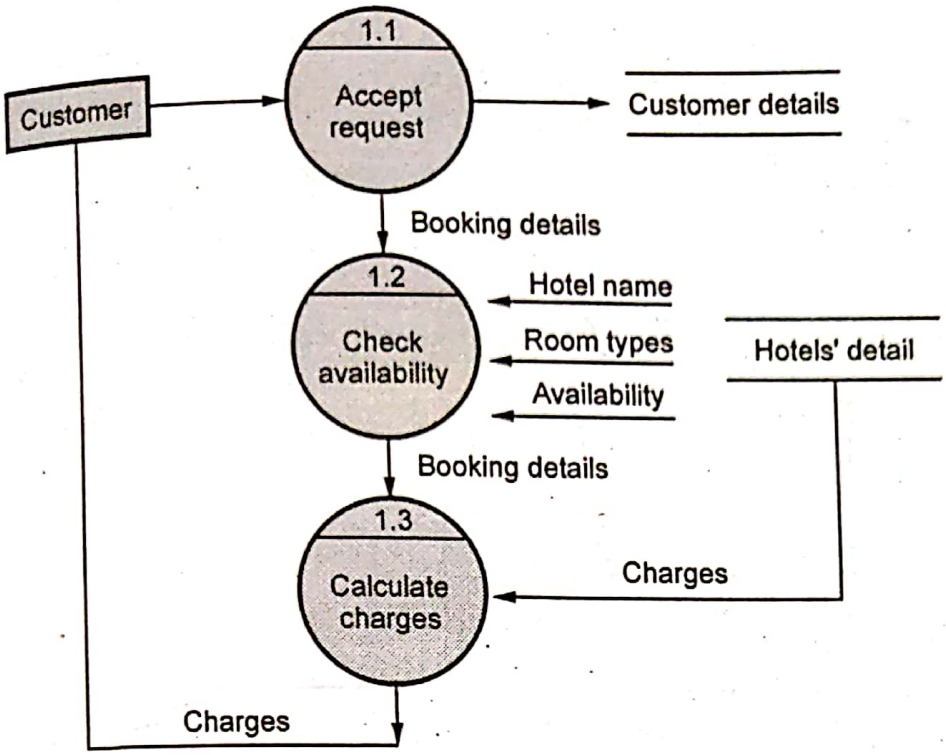


Fig. 2.11.18 Level 2 DFD

Example 2.11.5 *Tamil Nadu Electricity Board (TNEB) would like to automate its billing process. Customers apply for a connection-(domestic/commercial). EB staff take readings and update the system each customer is required to pay charges bi-monthly according to the rates set for the type of connection. Customers can choose to pay either by cash/card. A bill is generated on payment. Monthly reports are provided to the EB Manager.*

i) Give a name for the system. (Mark 1)

ii) Draw the Level - 0 DFD (Context Flow Diagram) (Marks 5)

iii) Draw the Level - 1 DFD. (Marks 10) **AU : Dec.-13, Marks 16**

Solution : i) Electricity Bill Management System

ii)

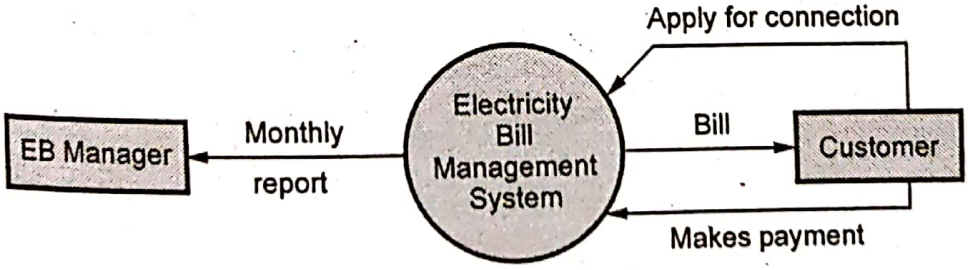


Fig. 2.11.19 Level 0 DFD

iii)

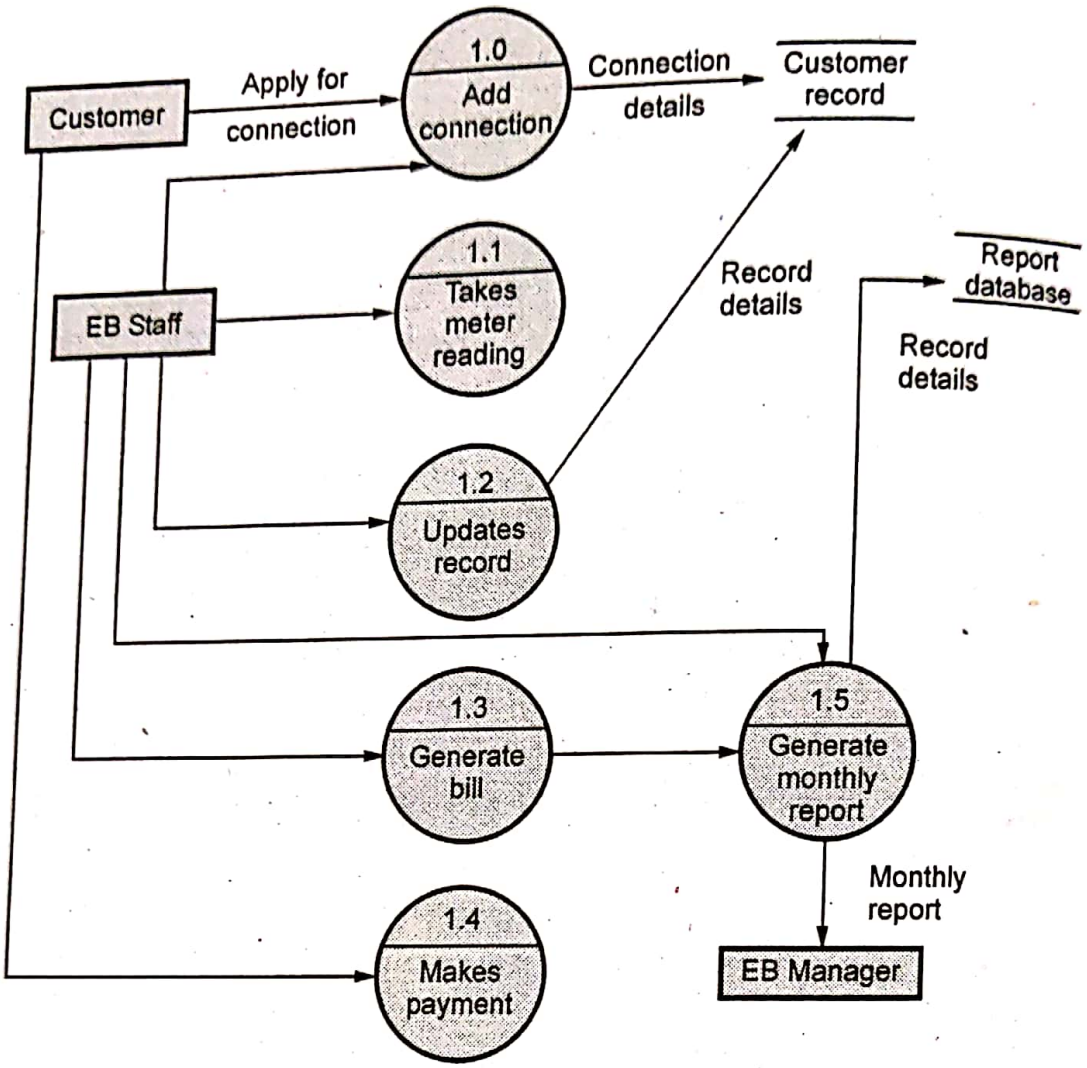


Fig. 2.11.20 Level 1 DFD

Example 2.11.6 *Develop a dataflow diagram for burglar alarm system along with entity relationship diagram.* **AU : May-10, Marks 8**

Solution : 1) DFD :

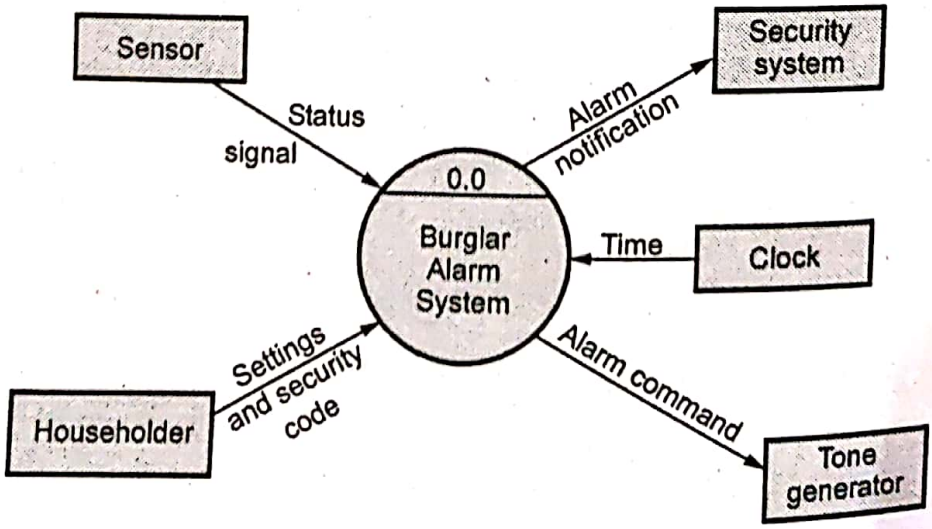


Fig. 2.11.21 Context diagram

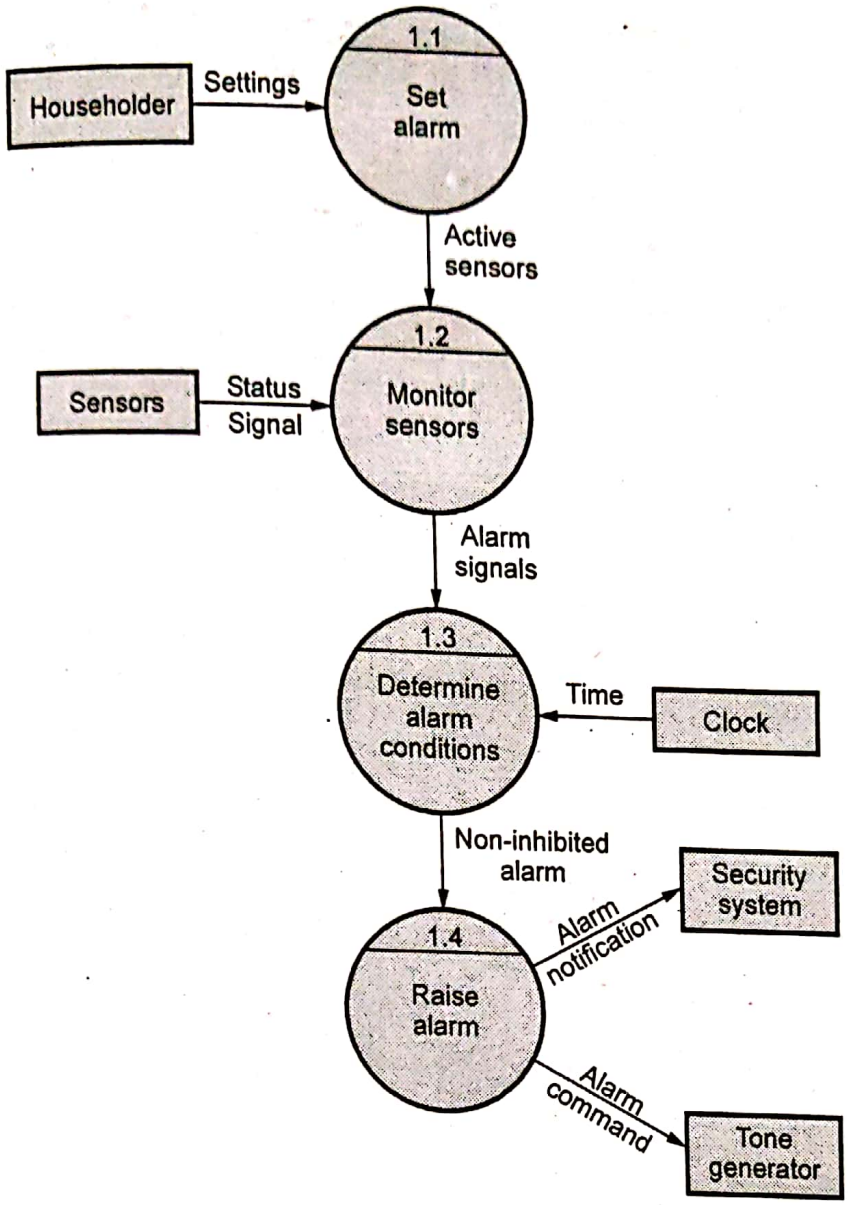


Fig. 2.11.22 Level 1 DFD

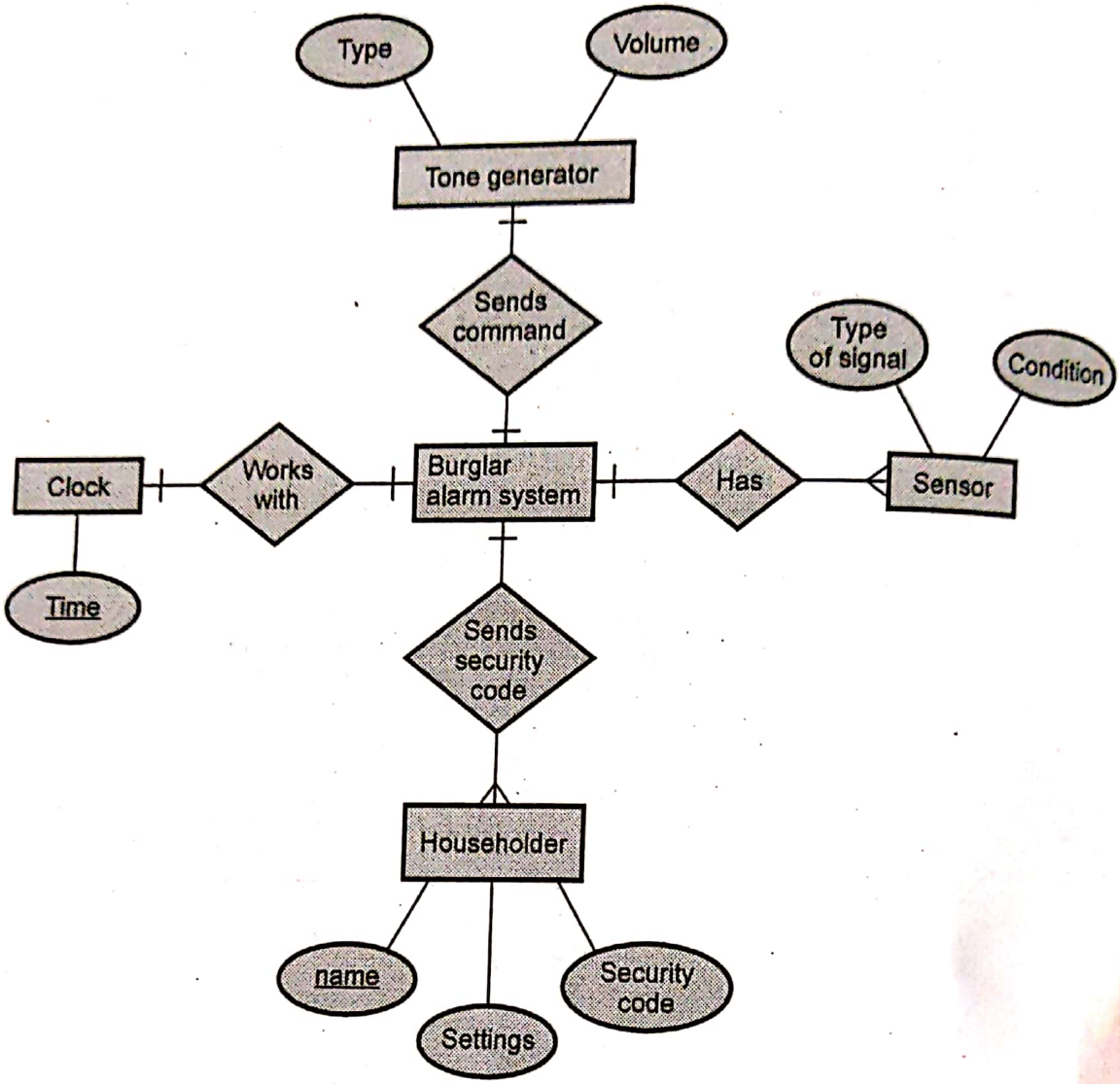


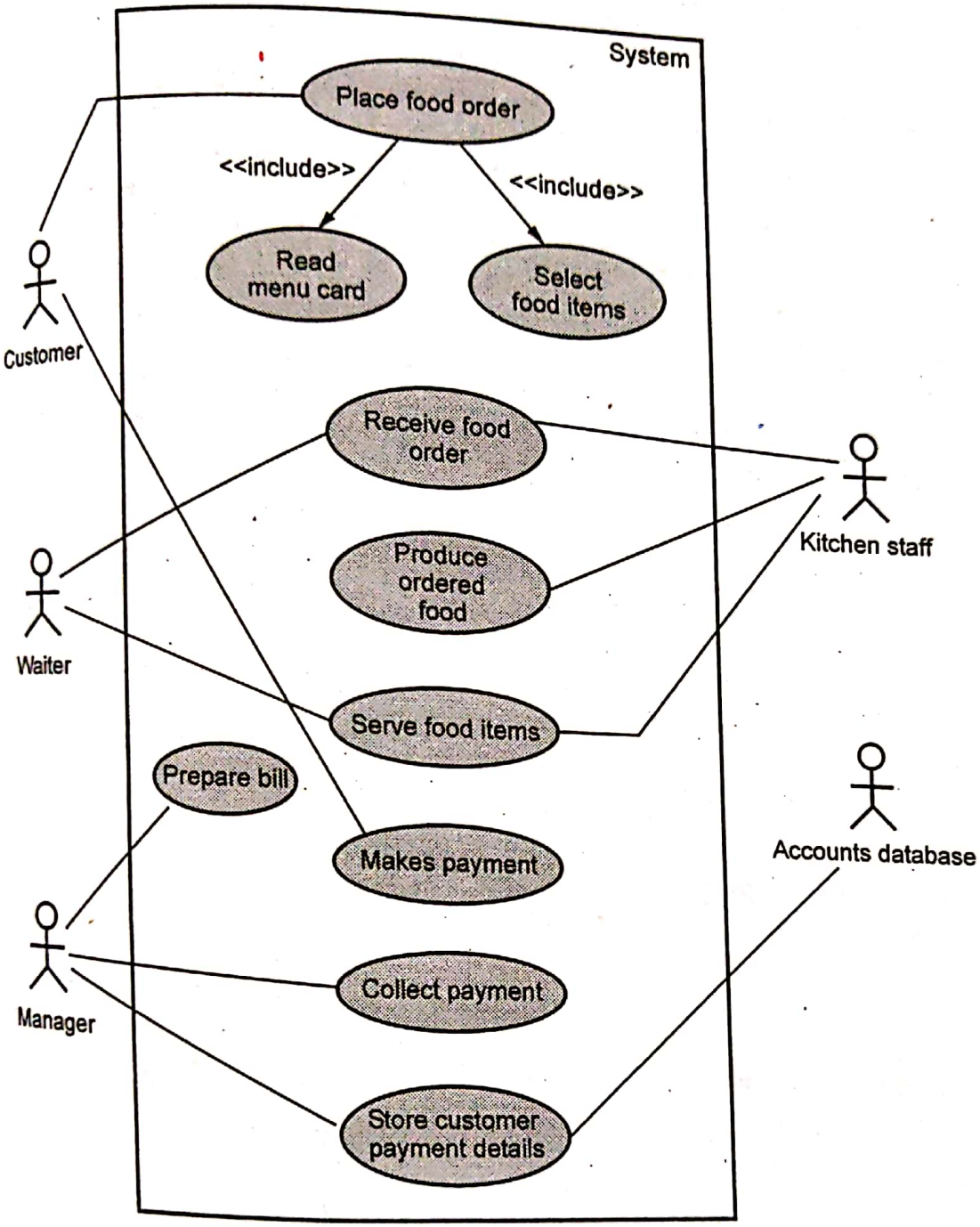
Fig. 2.11.23 ER diagram

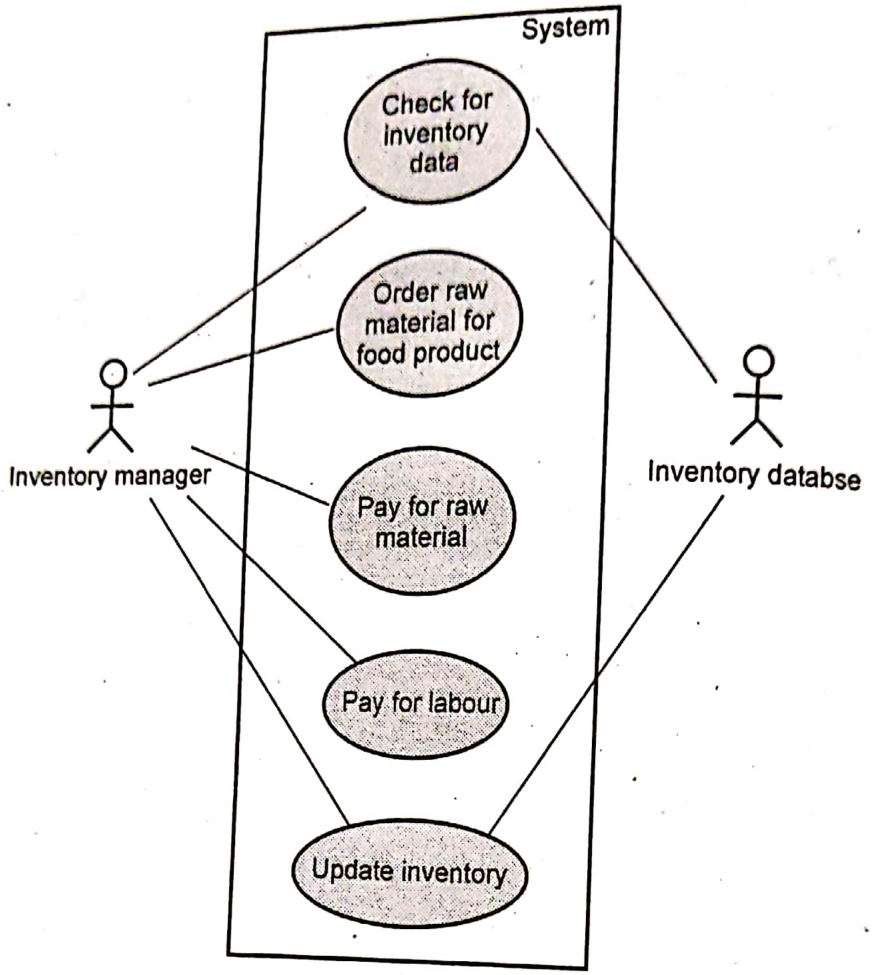
Example 2.11.7 *Draw the use case and data flow diagrams for a restaurant system. The activities of restaurant system are listed below -*

Receive the customer food orders, produce the customer ordered foods, serve the customer with their ordered foods, collect payment from customers, store customer payment details, order raw materials for food products, pay for raw materials and pay for labour.

AU : May-15, Marks 16

Solution : Part I : Use case diagram :





Part II : Data flow diagram : Refer example 2.11.2.

Example 2.11.8 *Consider an online railway reservation system, which allows the user to select route, book/cancel tickets using net banking/credit/Debit cards. The site also maintains the history of the passengers. For the above system, list and draw the use case scenario and model the above specification using data flow diagram.* **AU : Dec.-15, Marks 16**

Solution : The use cases for the given railway reservation system are -

1. logs in
2. select route
3. book or cancel tickets
4. view status
5. logs out

The use case diagram is as follows -

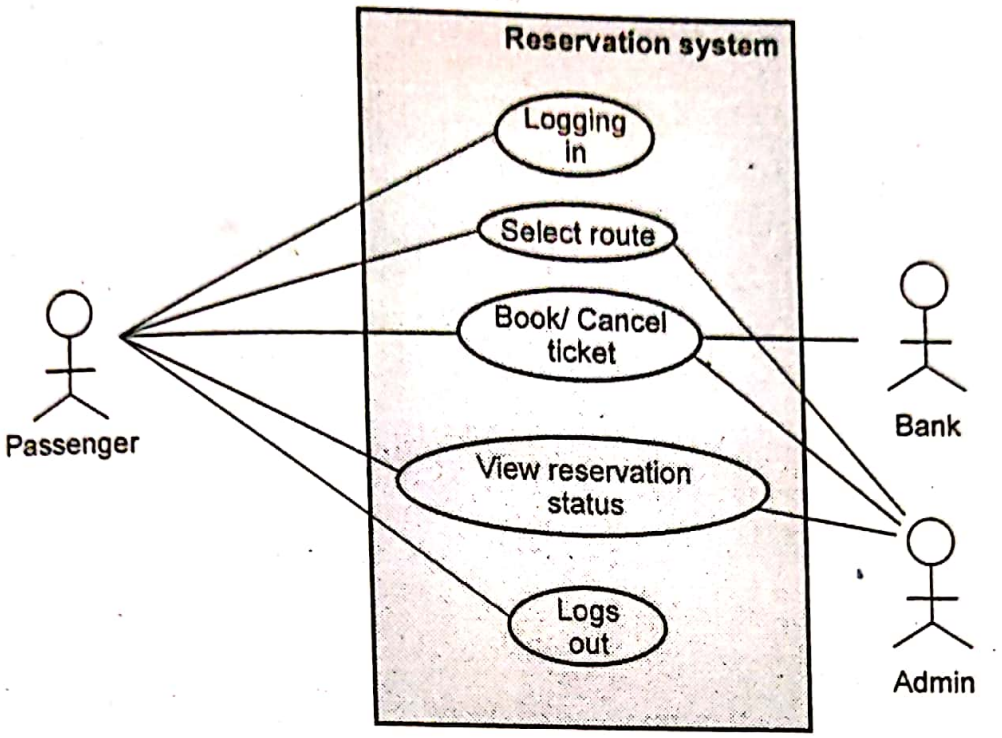


Fig. 2.11.24 Use case diagram

The level 0 and level 1 DFD can be as shown below –

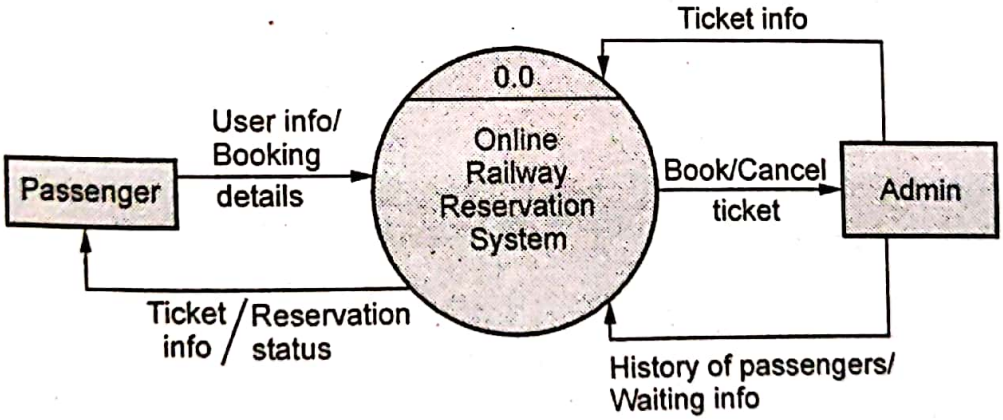


Fig. 2.11.25 Level 0 DFD

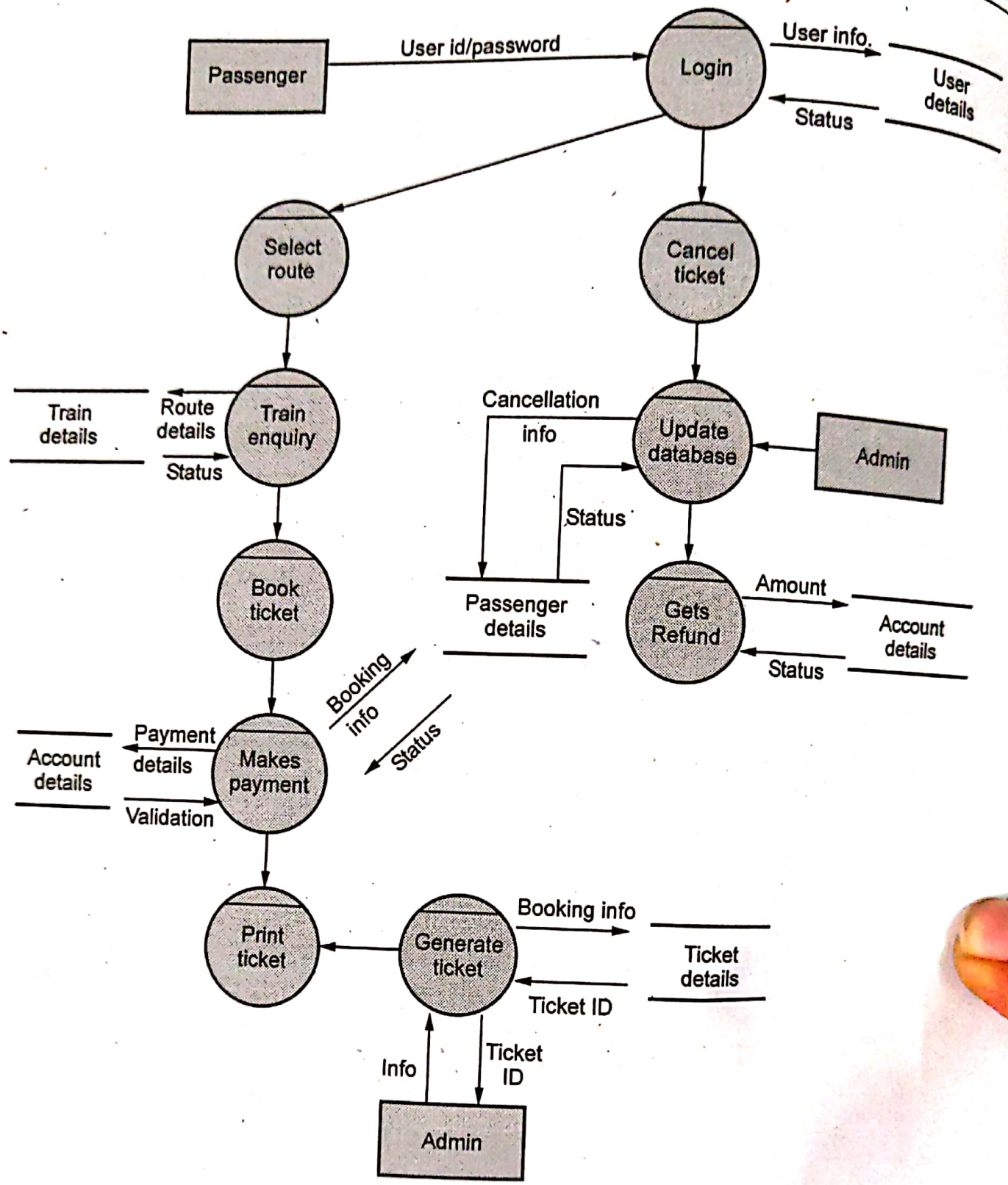


Fig. 2.11.26 Level 1 DFD

Example 2.11.9 *Consider an online book stores. It accepts individual/bulk orders, process payments, triggers delivery of the books. Some of the major features of the system include :*

- *Order books*
- *User friendly online shopping cart function*
- *Create, view, modify and delete books to be sold*
- *To store inventory and sales information in database*
- *To provide an efficient inventory system*
- *Register for book payment options*
- *Request book delivery*
- *Add a wish list*
- *Place request for books not available*
- *To be able to print invoices to members and print a set of summary reports*
- *Internet access.*

Analyse the system using the context diagram and level 1 DFD for the system. Explain the components of DFD.

AU : May-17, Marks 15

Solution :

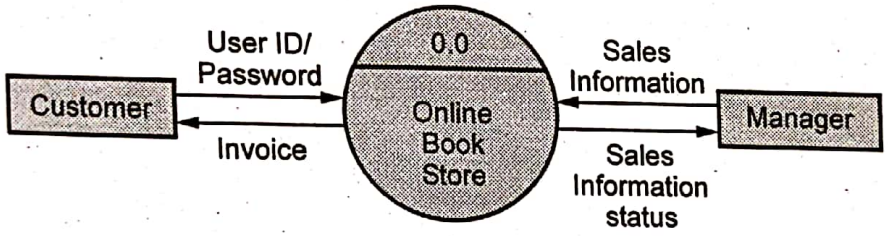


Fig. 2.11.27 Context level DFD

Refer Fig. 2.11.28 on next page.

Example 2.11.10 *Consider the process of ordering a pizza over the phone. Draw the use case diagram and also sketch the activity diagram representing each step of the process, from the moment you pick up the phone to the point where you start eating pizza. Include activities that others need to perform. Add exception handling to the activity diagram you developed. Consider at least two exceptions (e.g. delivery person wrote down wrong address, deliver person brings wrong pizza).*

AU : Dec.-17, Marks 13

Solution : Use case diagram

Refer Fig. 2.11.29 on page 2 - 69.

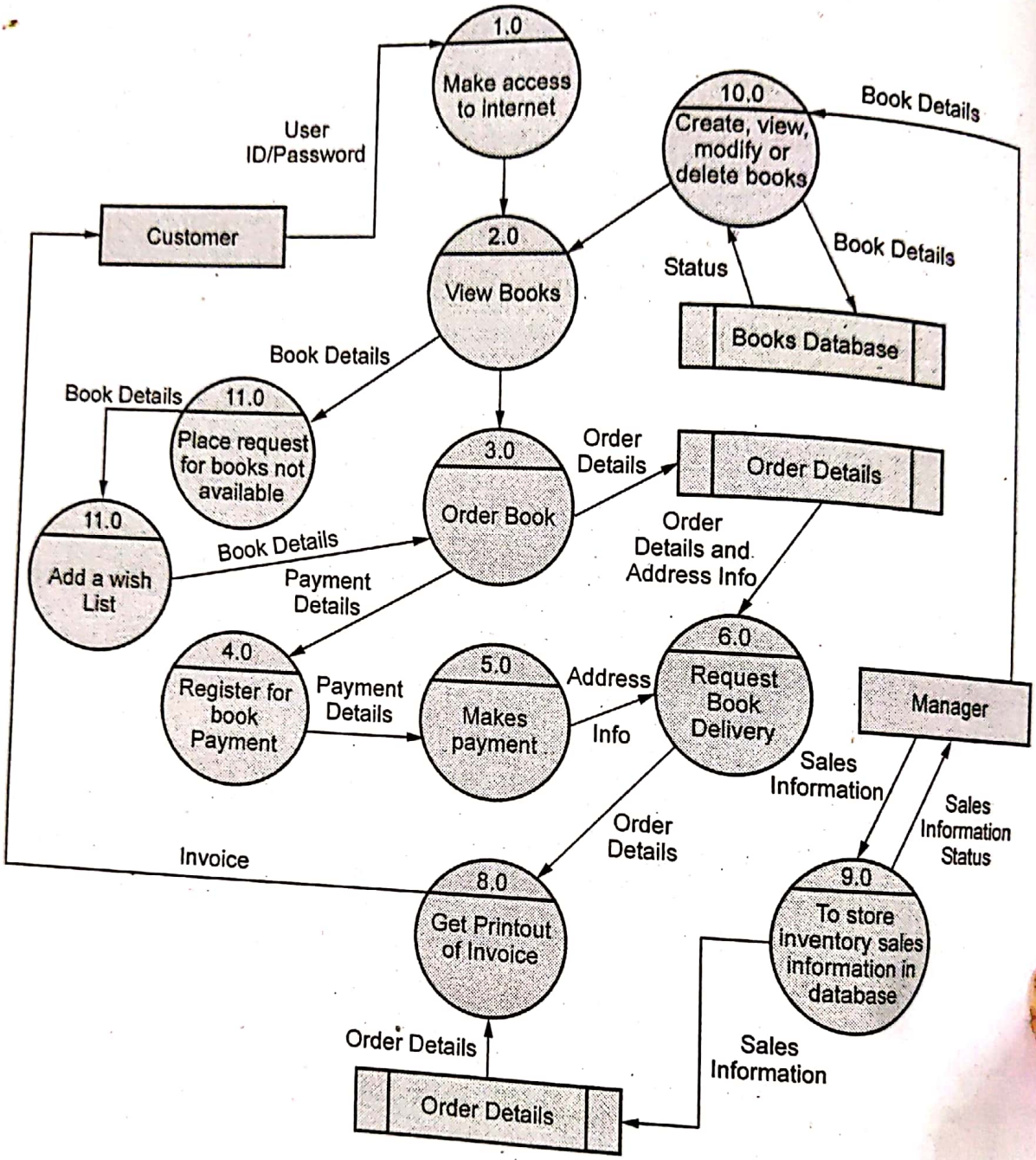


Fig. 2.11.28 Level 1 DFD

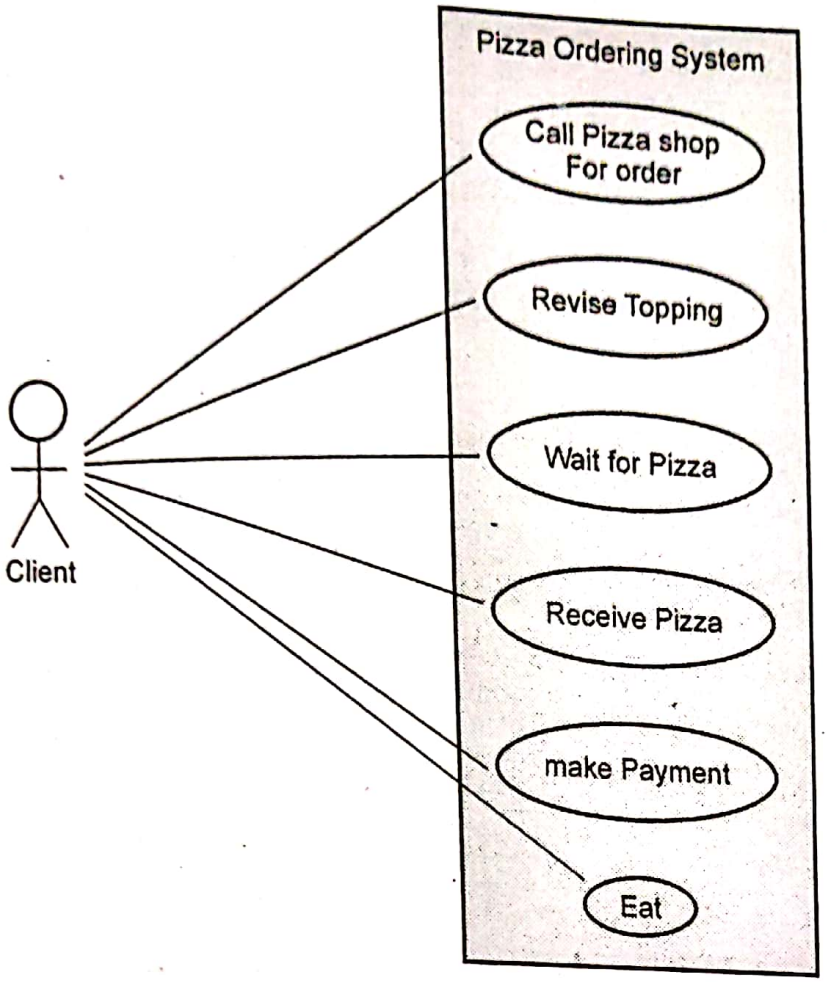


Fig. 2.11.29 Use case diagram for pizza ordering system

Activity Diagram

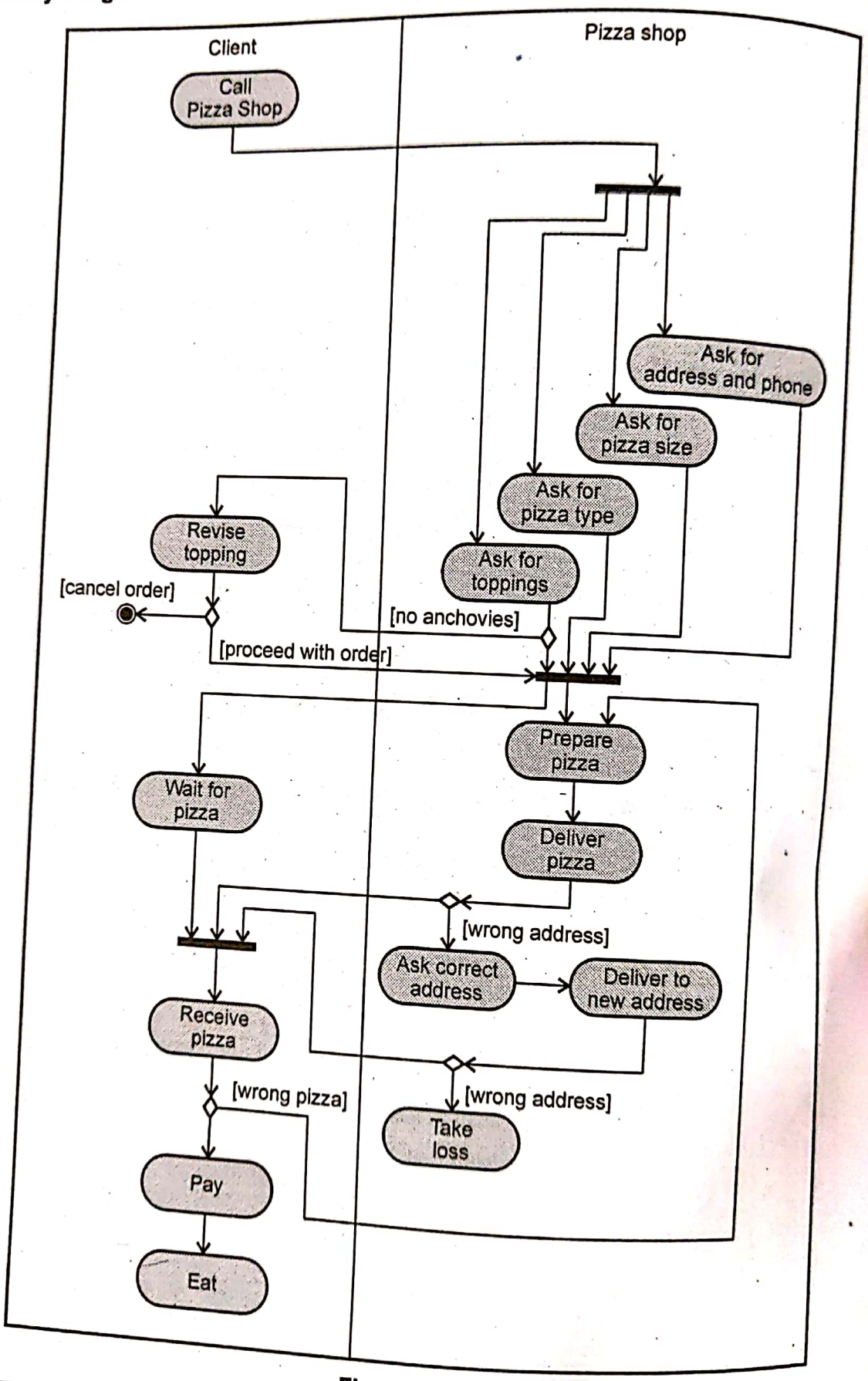


Fig. 2.11.30

Example 2.11.11 *What is the purpose of DFD ? What are the components of DFD ? Construct DFD for the following system : An on-line shopping system for XYZ provides many services and benefits to its members and staffs. Currently, XYZ staffs manually handle the purchasing information with the use of basic office software, such as Microsoft Offices Word and Excel. It may results in having mistakes easily and the process is very inconvenient. XYZ needs an online shopping system has five key features :*

i) to provide the user friendly online shopping cart function to members to replace hardcopy ordering form ;

ii) to store inventory and sales information in database to reduce the human mistakes, increase accuracy and enhance the flexibility of information processing ;

iii) to provide an efficient inventory system which can help the XYZ staffs to gain enough information to update the inventory ;

iv) to be able to print invoices to members and print a set of summary reports for XYZ's internal usage ;

v) to design the system that is easy to maintain and upgrade. **AU : May-18, Marks 15**

Solution : Level 0 DFD

Fig. 2.11.31

Level 1 DFD

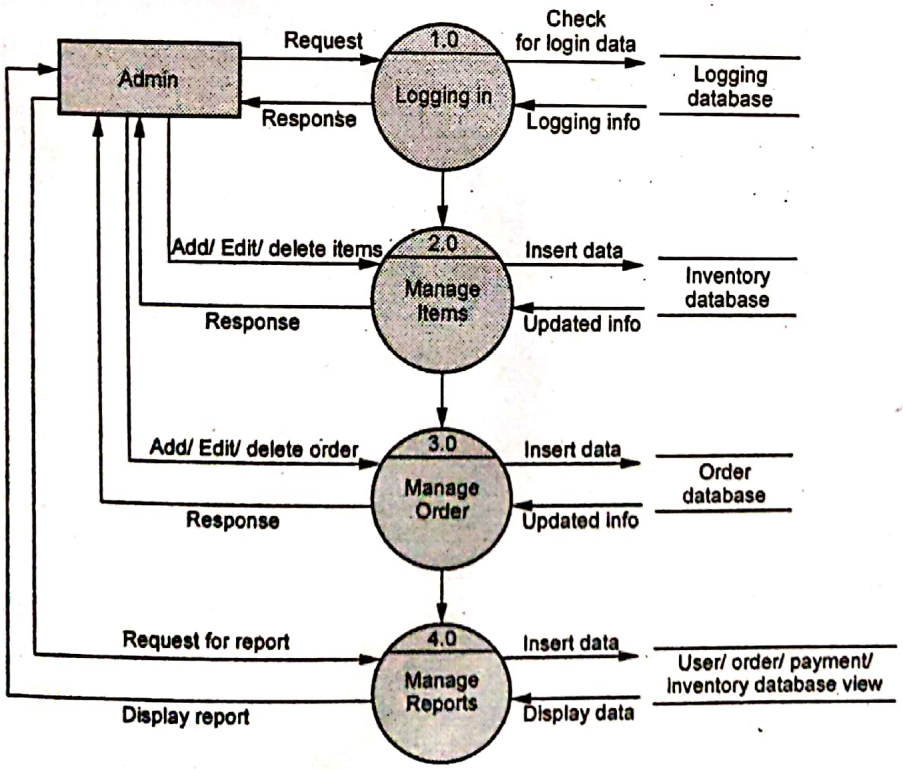


Fig. 2.11.32

Level 2 DFD : For manage order

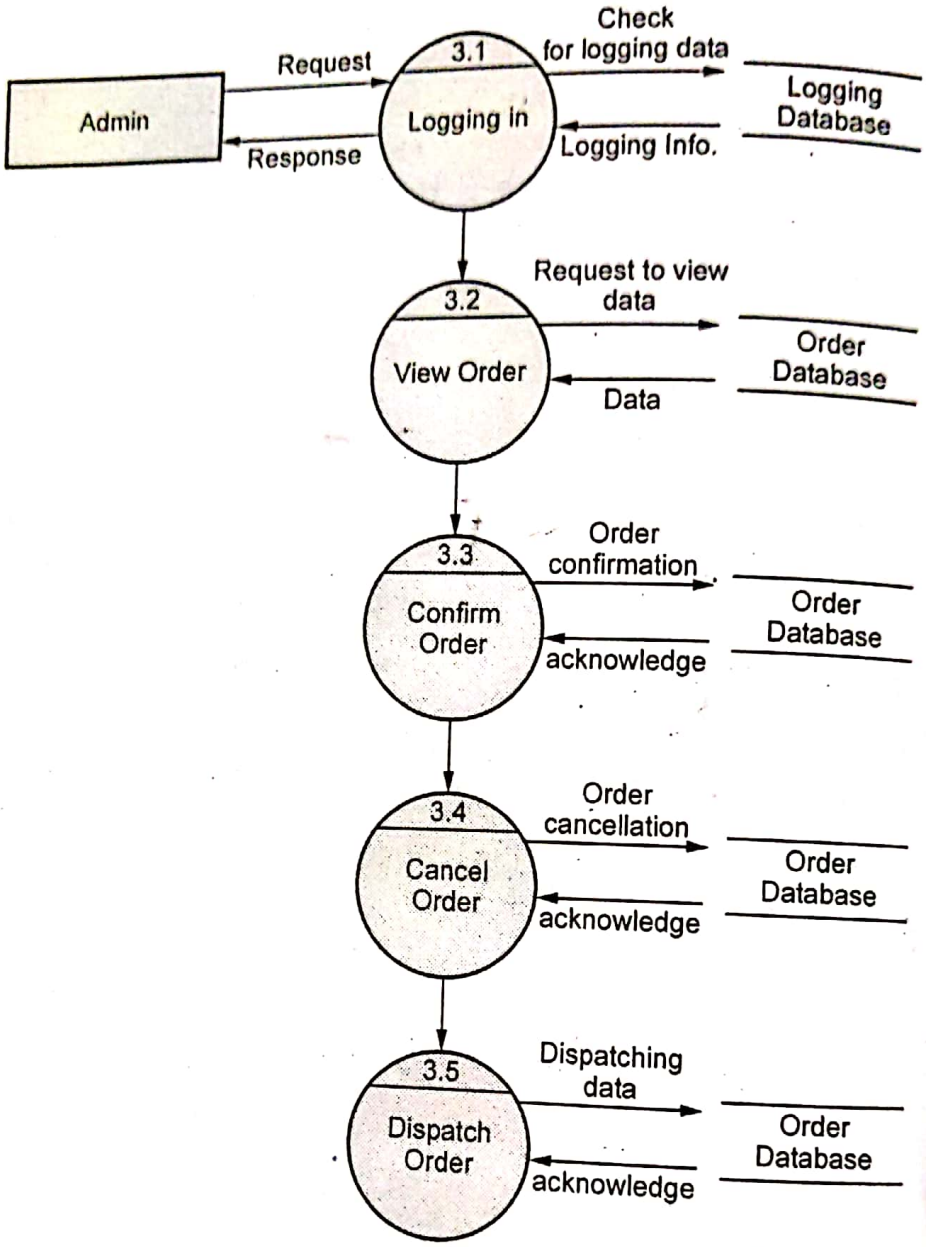


Fig. 2.11.33

Level 2 DFD : For manage items

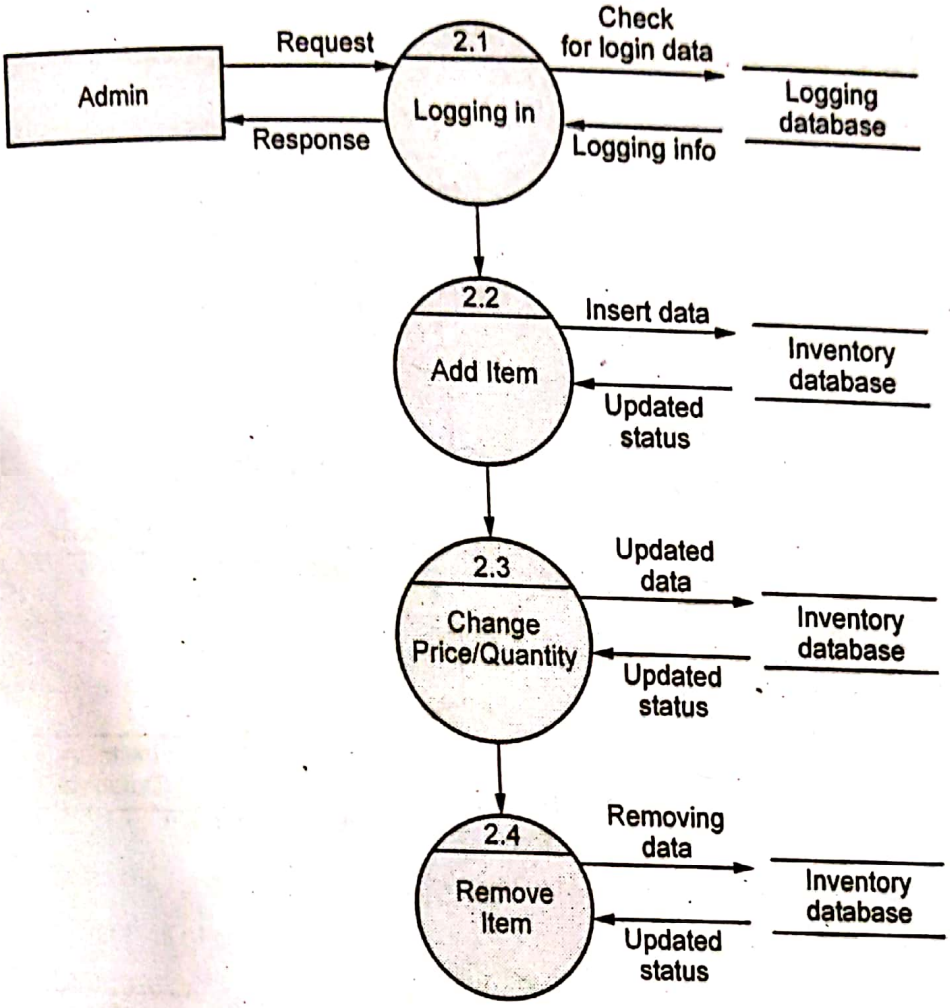


Fig. 2.11.34

Level 1 DFD : From user's perspective

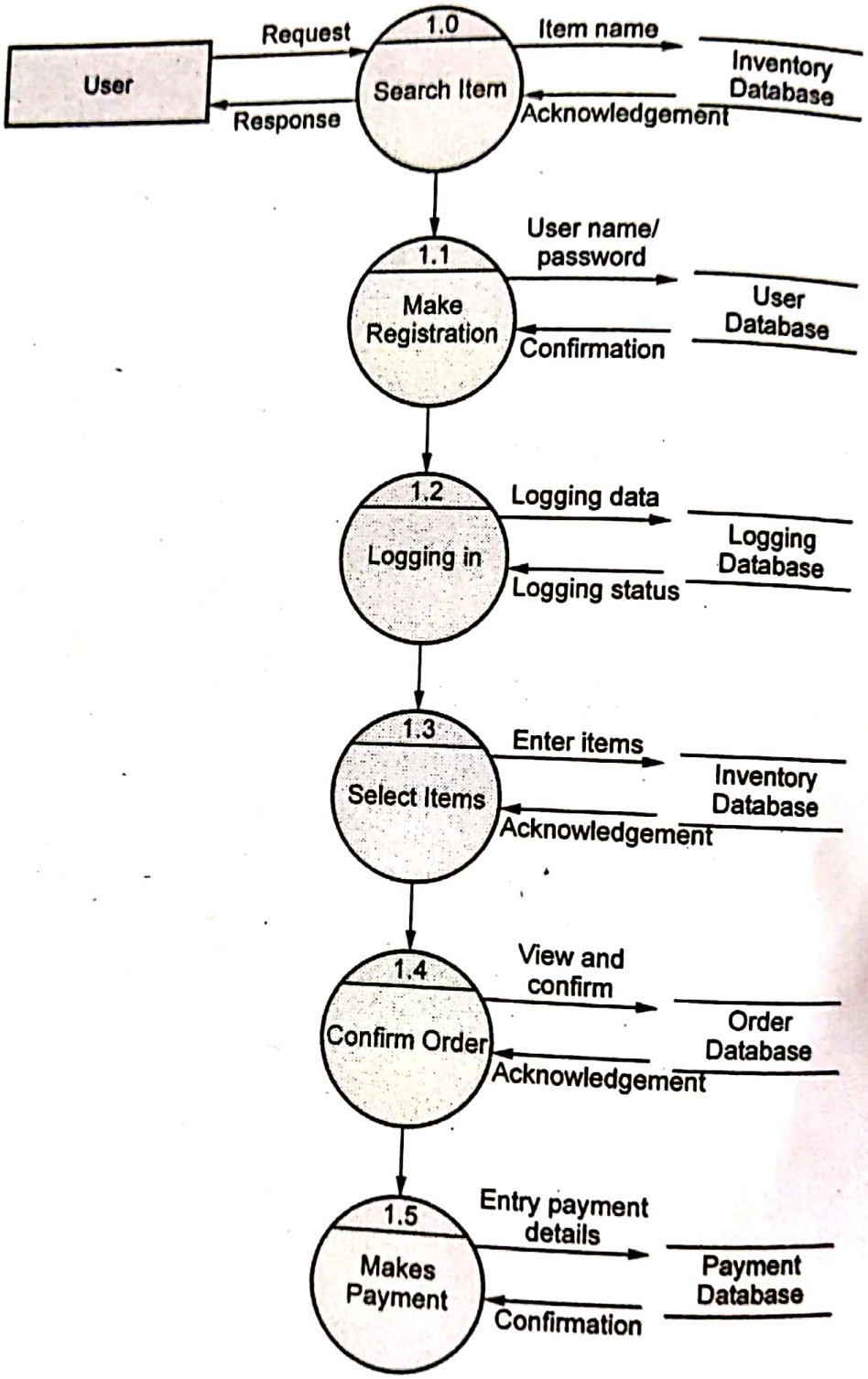


Fig. 2.11.35

2.11.3 Difference between Structured Analysis and Object Oriented Analysis

Structured analysis	Object oriented analysis
In structured analysis the process and data are separately treated.	In object oriented analysis the process(methods) and data are encapsulated in the form of object.
Various models that are getting developed in structured analysis are DFD,CFD, ER diagram and Data dictionary.	Various models that are getting developed in object oriented analysis are class diagram, use case diagram, activity diagram and deployment diagram.
Simpler to design.	Modular systems can be developed using this kind of analysis.
It is not suitable for large and complex projects.	For large and complex system object oriented approach is more suitable.

Review Question

1. *Describe primary differences between structured analysis and object oriented analysis.*

AU : May 08, Marks 6

2.12 Petri Nets

- The Petri Nets are invented by Carl Adam Petri in 1962.
- The Petri Nets can be rigorously used to define the system. The Petri nets are more popularly used for distributed systems and systems with resource sharing.
- In Petri Nets there are three types of components - places (circles), transitions (rectangles) and arcs (arrows)

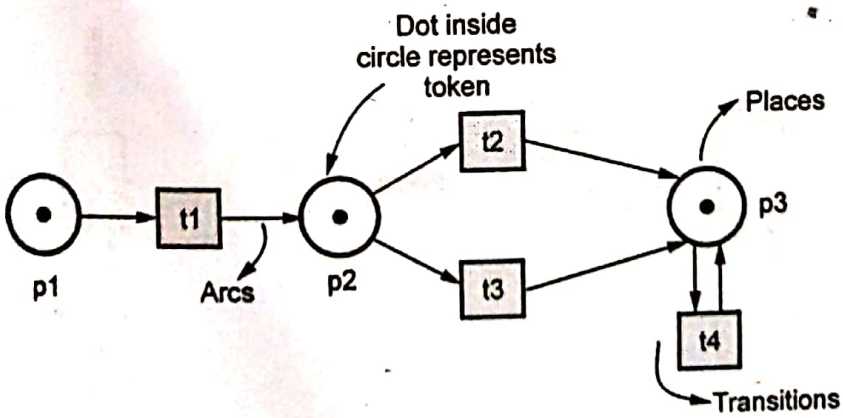


Fig. 2.12.1 Petri nets

- **Places :** Represent possible states of the system.
- **Transitions :** Cause the change in the states

- **Arc :** Connects a place with transition or transition with a place.

For example -

- Another equivalent notation used is a solid bar. This solid bar is used to denote transitions.
- The token represents the dynamic objects. The states having block dots are called marked Petri token. If there is a single dot inside the circle then that means it represent the one unit of certain resource.
- **Firing a transition :** Transitions take input from the input places and produce tokens in the output places. This is called firing a transition. For example

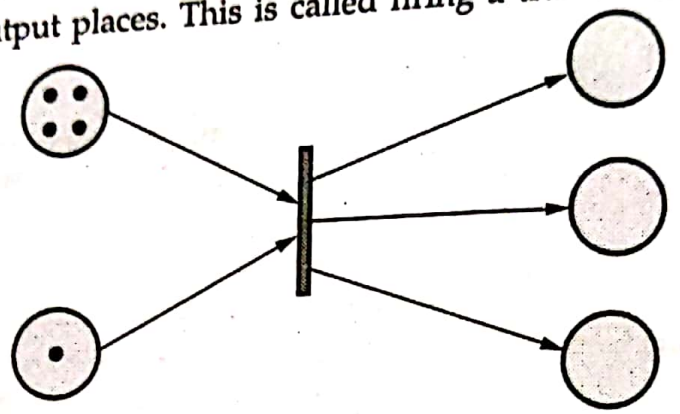


Fig. 2.12.2 Firing transitions

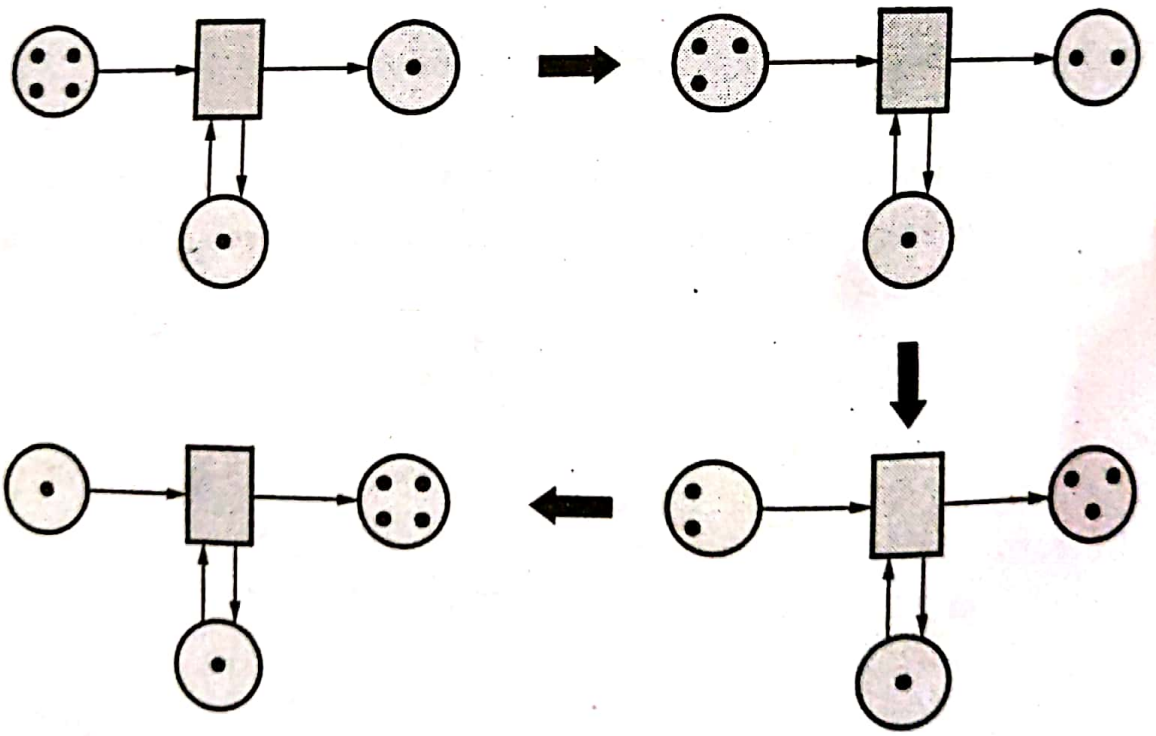


Fig. 2.12.3 Firing transitions in succession

Example 2.12.1 *Draw a petrinet for a traffic light switching system.*

Solution :

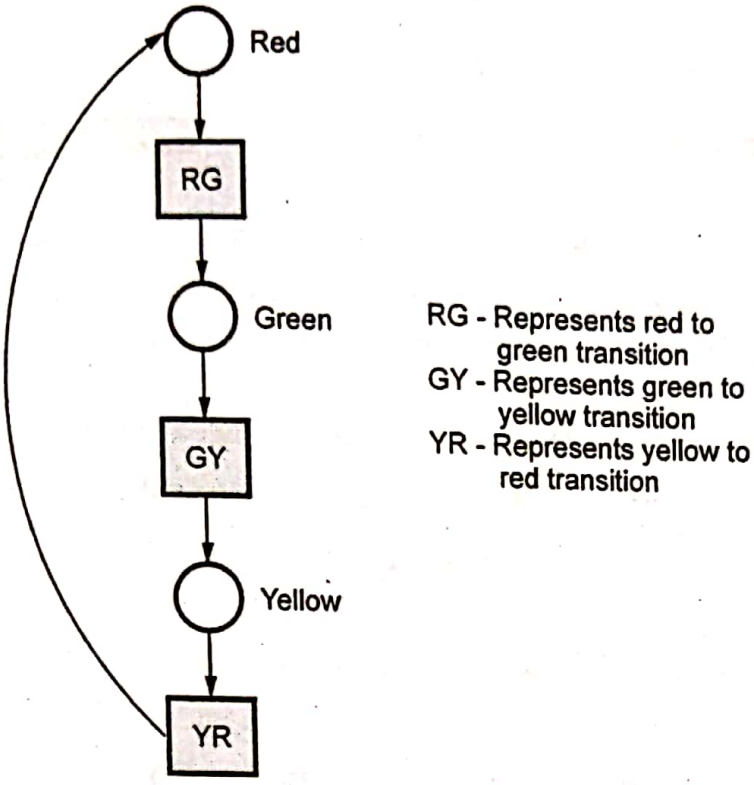
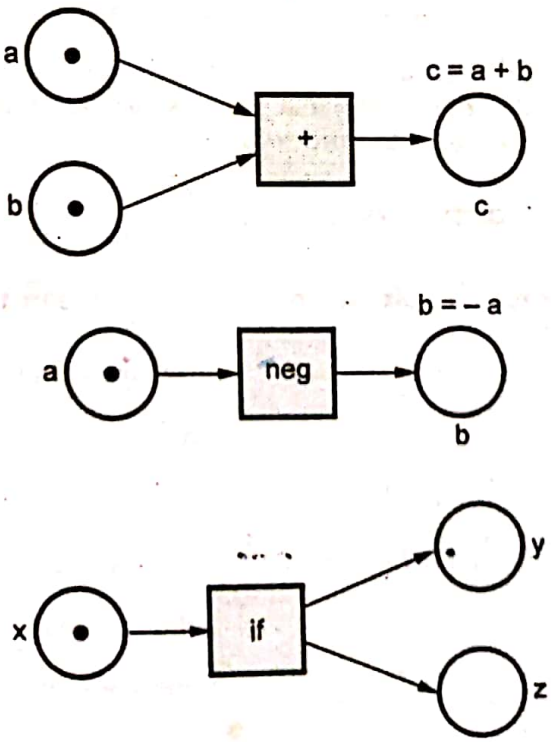


Fig. 2.12.4 Petrinet for traffic light signaling

Example 2.12.2 *Represent the operations i) $c = a + b$ ii) $b = -a$ iii) if $x > 0$ then $y = x$ else $z = x$*

Solution :



Advantages of using Petri Nets

1. It specifies the precise definition of problem.
2. It represents the graphical workflow.
3. It is useful in allocating resources to the tasks.

2.13 Data Dictionary

AU : Dec.-06, 08, 09, Marks 16

The data dictionary can be defined as an organized collection of all the data elements of the system with precise and rigorous definitions so that user and system analyst will have a common understanding of inputs, outputs, components of stores and intermediate calculations.

- The data models are less detail hence there is a need for data dictionary.
- Data dictionaries are lists of all of the names used in the system models.
- Descriptions of the entities, relationships and attributes are also included in data dictionary.
- Typically, the data dictionaries are implemented as a part of structured analysis and design tool
- The data dictionary stores following type of information

Name	Description
Name	The primary name of data or control item, the data store or external entity.
Alias	Other name used for the Name
Where-used or how is used	It describes where the data or control item is used. It also describes how that item is used(that means input to the process, output to the process)

The notations used in data dictionary are

Data construct	Notation	Meaning
Composition	=	Is composed of
Sequence	+	And
Selection	[|]	Or
Repetition	$\{ \}^n$	Repetition for n times
	( )	Optional data
	...	Commented information

For example :

Consider the some reservation system. The data item "passenger" can be entered in the data dictionary as

name :	passenger
alias :	none
where used/	
how used :	passenger's query (input) ticket (output)
description :	
passenger =	passenger_name + passenger_address
passenger_name =	passenger_last_name + passenger_first name + passenger_middle_initial
passenger_address =	local_address + community_address + zip_code
local_address =	house_number + street_name + (apartment_number)
community_address =	city_name + [state_name | province_name]

- The data dictionary defines the data items unambiguously
- One can give the detailed description of data items using data dictionary
- For large computer based system the size of data dictionary is very huge. It is also complex to maintain such a data dictionary manually. Hence automated (CASE) tools can be used to maintain the data dictionary.

Advantages :

1. Data dictionary support name management and avoid duplication
2. It is a store of organisational knowledge linking analysis, design and implementation.

Review Question

1. *What is data dictionary ? Give an example.*

AU : Dec.-06, Marks 8 , IT, Dec.-09, Marks 16; IT, Dec.-08, Marks 2 + 8

Two Marks Questions with Answers

Q.1 Write distinct steps in requirement engineering process. AU : IT, May-06

Ans. : The distinct steps in requirement engineering process are

1. Requirements elicitation.
2. Requirements analysis.
3. Requirements validation.
4. Requirements management.

Q.2 Why SRS must be traceable? What is traceability requirement ? AU : IT, Dec.-05

Ans. : There may be chances of having new requirements during development of the software. Traceability is concerned with relationship between requirements their sources and the system design. Thus change in requirement is manageable if SRS is traceable.

The traceability requirement is the relationship between dependent requirements.

Q.3 What are non-functional requirements for a software ? AU : IT, May-03, May-05

Ans. : The non-functional requirements define system properties and constraints. Various properties of a system can be reliability, response time, storage requirements, and constraints of the system can be input and output device capability, system representations.

Q.4 What is the outcome of feasibility study ? AU : IT, Dec.-03

Ans. : The feasibility study decides whether the proposed system is worthwhile or not. The outcome of feasibility study is the results obtained from following questions :

1. Whether system contributes to organizational objectives ?
2. Whether the system can be engineered? Is it within the budget ?
3. Whether the system can be integrated with other existing system ?

After performing the feasibility study a feasibility study report is prepared.

Q.5 Distinguish between expected requirements and exciting requirements. AU : IT, May-04

Ans. : Expected requirements are essentially basic functions or features that customers normally expect from the product. Expected requirements are usually invisible unless they become visible when they are unfulfilled. Their absence will cause customer's dissatisfaction. For example : ease of software installation.

Exciting requirements are sort of "out of ordinary" functions or features of a product. Exciting requirements are also usually invisible unless they become visible when they are fulfilled and result in customer satisfaction; they do not leave customers dissatisfied when left unfulfilled. For example : In word processing software if the product contains the number of page layout capability then that becomes an exciting requirement.

Q.6 What is data dictionary ? Where it is used in software engineering ?

AU : IT, Dec.-04 , May-09, 13

Ans. : The data dictionary represents all of the names used in the system models. The descriptions of the entities, relationships and attributes are also included in data dictionary. The data dictionary is used during structured analysis for providing the detail information of data objects being used in DFD. Nowaday, the data dictionary is implemented as a part of CASE.

Q.7 How requirements are collected for "user-interface" of software ?

AU : IT, Dec.-04

Ans. : While designing the user-interface of software the requirement collection can be done by focusing on the profile of user who will interact with the system. Skill level, business understanding and general grasping to the new system are recorded. Users can be categorized into different categories and for each category of user requirements are elicitations. The most commonly used elicitation technique is to conduct meeting software developer and the end user.

Q.8 Differentiate data flow diagram and state transition diagram. AU : IT, Dec.-06

Ans. :

Data flow diagram	State transition diagram
Data flow diagram is a graphical representation for representing the information flow and the transforms that are applied as data move from input to output.	State transition diagram is a graphical representation for representing the behaviour of a system by depicting its states and the events that cause the system to change state.
The data flow diagram is a collection of process, data store, flow of data (transitions) and external entity.	The state transition diagram is a collection of states and events.
The data flow diagrams are used to represent the system at any level of abstraction, and the increasing levels are used to expose more and more functionalities in the system.	The state transition diagrams are typically drawn at single level. They are intended to expose the overall behaviour of the system.

Q.9 Why it is so difficult to gain a clear understanding of what customer wants ?

AU : CSE, Dec.-07

OR

Write a note on what are the difficulties in elicitation, requirement elicitation.

AU : May-17, Marks 5

OR

Requirements analysis is unquestionably the most communication intensive step in the software engineering process. Why does the communication path frequently breaks down ?

AU : May-16, Marks 8

Ans. : Following are the reasons for : why it is difficult to understand customer wants -

1. Customer sometimes is unable to specify the **scope of the project**. Sometimes customers specify too many technical details and this may increase the confusion.
2. There is difficulty in **understanding the problem**. Sometimes customer could not decide what are their *needs and wants*. Sometimes they have got poor understanding of capabilities and limitations the existing computing environment.

   Sometimes customers find it difficult to communicate with the system engineer about their needs. Sometimes customers may have got some conflicting requirements. This ultimately results in specifying ambiguous requirements.
3. As project progresses the needs or requirements of the customers changes. This creates a problem of volatility.

Q.10 Create a data dictionary that provides with a precise definition of telephone number, it should indicate, where and how this data item is used and why supplementary information that is relevant to it ? AU : CSE, Dec.-07

Ans. : The data dictionary for telephone number is as given below.

name :	**telephone number**
aliases :	**none**
where used/how used :	**assess against the set-up(o/p)**
	dialled number (i/p)
description :	
telephone number=[local number+STD]	
local number=prefix+access number	
STD=+91+area code+local number	
area code=[020 | 022]	
prefix=* a four digit number that starts with 2*	
access number=*any four digit number*	

For example : Consider a telephone number +91 (020)24495496 of **Technical Publications** in Pune.

Here **+91** then followed by the **area code** 020 then comers **prefix=2449** which starts with 2 and finally the **access number** is 5496. This is the number in city Pune (the area code for Pune is 20). If we want to dial this number in city Pune itself then the local number is 24495496.

Q.11 What is the major distinction between user requirements and system requirements ? AU : IT, May-08, 16, Marks 4

Ans. : The user requirements describe both the functional and non-functional requirements in such a way that they are understandable by the users who do not have

detailed technical knowledge. On the other hand the system requirements are more detailed specifications of system functions, services and constraints than user requirements.

The user requirements are specified using natural languages, tables and diagrams because these representations can be understood by all users.

The system requirements can be expressed using system models.

Q.12 Identify ambiguities and omissions in the functional requirements. What questions would you ask to clarify these functional requirements ?

AU : IT, May-08

Ans. : Following are the questions that must be asked to clarify the functional requirements -

1. Is the requirement specification clear ? Can it be misinterpreted ?
2. What are the sources of requirements ? Is the final requirement specification as per the original source ?
3. Is there any requirement violation for domain constructs ?
4. What are the other related requirements of the current requirement ? And are they cross verified ?
5. For tracing the requirement is there any system model ?
6. Is the requirement testable ?
7. Has an index specification created for the requirements ?
8. Is the requirement satisfying the system objective.

Q.13 How do we use the models that we create during requirement analysis ?

AU : CSE, Dec.-08

Ans. : Various models and their use during requirement analysis phase are given as below -

1. **Data model -** The entity relationship model is created during the data modelling. The ERD defines the relationship between data objects.
2. **Functional model -** The data flow diagrams are created in this modelling. The DFD depicts the information flow in three phases input, output and process.
3. **Behavioural model -** The behavioural models are used to describe the overall behaviour of a system. The state transition diagrams are used to represent the behavioural model.

Q.14 List out requirements engineering.

AU : CSE, May-09

Ans. : Various activities in requirement engineering are -

1. Requirements elicitation 2. Requirements analysis
3. Requirements validation 4. Requirements management

Q.15 Define functional and non functional requirements. AU : Dec.-10, May-14

(Refer sections 2.2.1 and 2.2.2)

Q.16 What do requirement process involve ? AU : May-12

Ans. : Requirements engineering process involves elicitation, analysis and validation.

Q.17 List two advantages of traceability tables in the requirements management phase AU : Dec.-13

Ans. : The two advantages of traceability tables are as given below -

1. The traceability matrix ensures the coverage between different types of requirements.
2. The quick change impact analysis can be made using traceability matrix.

Q.18 List the notations used in data flow models.

(Refer section 2.11.1) AU : May-14

Q.19 Give two examples of non functional requirements. AU : Dec.-14

Ans. : Consider the Library Management System for which the two non functional requirements can be-

1. The user who wishes to read the article on-line must be authenticated first.
2. The article must be displayed within 5 seconds.

Q.20 What is data dictionary ? AU : Dec.-14, 15, 16, May-17

Ans. : The data dictionary represents all of the names used in the system models. The descriptions of entities, relationships and attributes are also included in the data dictionary.

Q.21 What is the need for feasibility analysis ? AU : May-15

Ans. : The feasibility study is required to decide whether or not the proposed system is worthwhile to implement.

Q.22 How are the requirements validated ? AU : May-15

Ans. : Requirements are validated with the help of following validation techniques -

1. Requirements reviews : It is a manual analysis of requirements with the participation of customer and contractor staff.

2. Prototyping : The requirements can be checked using executable model of system.

3. Test case generation : In this case, various tests are developed for requirements.

Q.23 Define feasibility study and list the types. AU : Dec.-15

Ans. : Definition : A feasibility study is a study made to decide whether or not the proposed system is worthwhile.

Types : 1) Technical feasibility

2) Economic feasibility

3) Schedule feasibility

4) Operational feasibility

Q.24 List the characteristics of good SRS. AU : May-16

Ans. : 1) SRS must be **correct.**

2) SRS must be **unambiguous.**

3) SRS must be **complete.**

4) SRS must be **consistent.**

5) SRS must be **traceable.**

Q.25 What are the linkages between data flow and E-R diagram ? AU : May-16

Ans. : 1) Both the Data Flow Diagrams (DFD) and ER diagram are system modeling techniques used during structured system analysis.

2) The ER diagram is used to represent data model while data flow diagrams are used to represent the functional model.

Q.26 Classify the following as functional / non functional requirements for banking system.

a) Verifying back balance

b) Withdrawing money from bank

c) Completion of transactions in less than one second.

d) Extending the system by providing more tellers for customers. AU : Dec.-16

Ans. : a) Functional requirement

b) Functional requirement

c) Non-functional requirement

d) Non-functional requirement.

Q.27 What is the purpose of petrinets ? AU : May-17

Ans. : Petrinets are rigorously used to define the system. There are three types of components in petrinets. These are

i) Places : Represent possible states of the system.

ii) Transitions : Causes the change in the states.

iii) Arc : Connects a place with transition or transition with places.

Q.28 Differentiate between normal and exciting requirements. AU : May-17

Ans. : Nornal requirements : The requirements as per goals and objectives of the system are called normal requirements. For example - Handling mouse and keyboard events for any GUI based system.

Exciting requirements : When certain requirements are satisfied by the software beyond customer's expectations then such requirements are called exciting requirement. For

example - In word processing software system if there are some exceptional page layout facilities then it falls in exciting requirements category.

Q.29 Draw a use case diagram for an online shopping which should provide provisions for registering, authenticating the customers and also for online payment through any payment gateway like Paypal. AU : Dec.-17

Ans. :

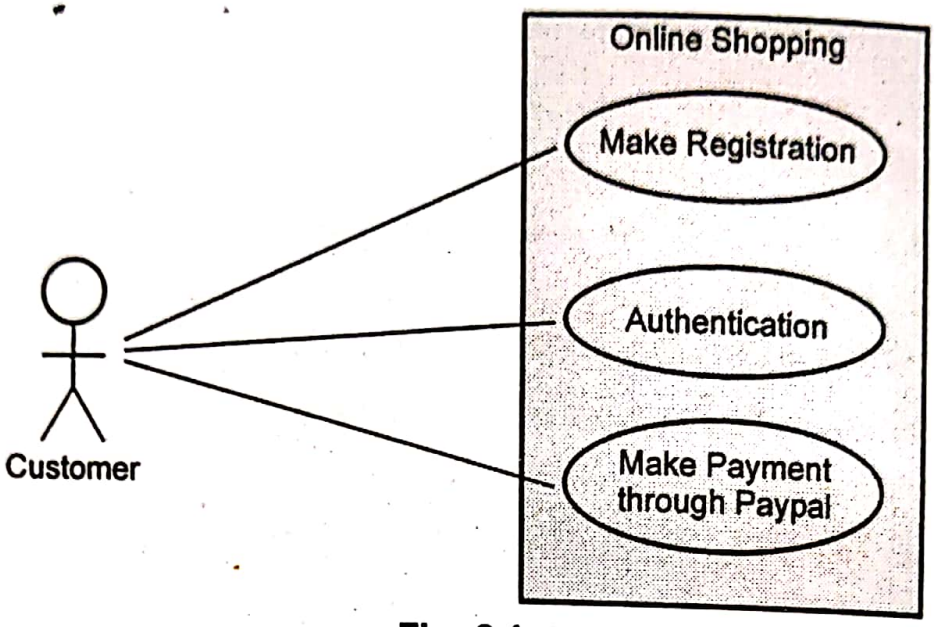


Fig. 2.1

Q.30 Define Quality Function Development (QDF). AU : Dec.-17

Ans. : Quality function deployment is a quality management technique which translates the customer needs and wants into technical requirements. This technique was introduced in Japan.

Q.31 Draw the context flow graph of a ATM automation system. AU : Dec.-17

Ans. :

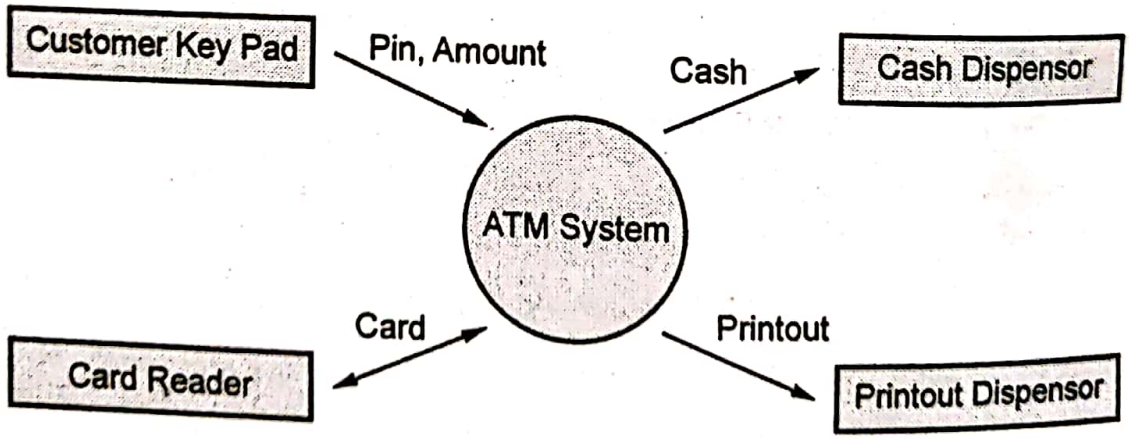


Fig. 2.2

Q.32 What are the various types of traceability in software engineering ? AU : May-17

Ans. : Various types of traceability in software engineering are -

(1) Source traceability (2) Requirements traceability (3) Design traceability