1.SYSTEM ENVIRONMENT

Question 1. ***What do you mean by information system and what is its importance to the organization?***

Answer:

- Information system is a collection of components working together to provide information to people in an organization.
- There are many people working in an organization. Similarly, there are many other people interacting with the organization.
- All such people need information about the system.
- The information provided to such people should be meaningful and should serve the purpose of making the organization work in a way that is beneficial to it.
- The information system provides such information.
- Hence, we can say that information system is important to an organization because it performs the task of providing the required information to the people requiring it.

Question 2. ***How many types of information system are there in an organization?***

Answer:

There can be various kinds of information systems in an organization. Some of them are:

i. **Transaction Processing system**
   - Computer programs that can be used to allow people to access the database, make any necessary changes to it and use them to initiate a further transaction are called Transaction Processing Systems (TPS).
   - The transaction processing system provides responses to the user as the transaction progresses through the system.
   - Any errors and inconsistencies, as well as the result of the final updated database, are reported.
   - Responses can be provided in a number of ways, depending upon the transaction system mode. Transactions can be input in on-line or batch mode.

ii. **<u>Management Information System</u>**

- Management Information System (MIS) takes relatively raw data available through a TPS and converts them into a meaningful aggregated form that managers need in order to conduct their responsibilities.
- Developing an MIS requires good understanding of what kind of information managers require and how managers use information in their jobs.
- For this it is important to draw on data from various subject areas and hence developing a comprehensive and accurate model of data is essential in building an MIS.

iii. **<u>Decision Support System</u>**

Decision support systems assist groups to make complex decisions. Some decisions require an optimization algorithm, while many decision support systems are experimental in nature, where the user tries different inputs to see their effects. A third kind of decision is one of policy nature where one develops alternate positions and then justifies them by argumentation. Decision support usually assumes asynchronous interaction, although there can be some advantage in synchronous discussion to resolve conflicts.

iv. **<u>Expert System</u>**

An Expert System (ES) attempts to manipulate knowledge rather than information. Users communicate with an ES through an interactive dialogue. The ES asks questions (that an expert would ask) and the end user supplies the answers. The answers are then used to determine which rules apply and the ES provides a recommendation based on the rules. The focus while developing an ES is on acquiring the knowledge of the expert in the particular problem domain.

v. **<u>Office Automation System</u>**

Office Automation (OA) Systems support the wide range of business office activities that provide for improved workflow and communication between workers, regardless of whether or not those workers are located in the same office. Office Automation functions include word processing, electronic messages, work group computing, fax processing, etc. OA systems can be designed to support both individuals and work groups.

Question 3. ***What do you know about methods and tools to build an information system?***

Answer:

- Methods and tools are used to build a system effectively.
- System development methodology defines a set of steps followed to build the systems.
- It also provides a variety of supporting methods and tools.
- It includes modeling methods used to produce models that help us to understand the system and its requirements and then to develop system specifications and are primarily used in analysis.
- Productivity tools that help people develop models and convert them to working systems are also available.
- It is always necessary to choose the right methods and tools to build a quality system.

Question 4. ***What are the different approaches based on the methods for developing the system?***

Answer:

The different approaches based on methods for developing the system are:

i. **Prototyping**

Prototyping is an iterative process of systems development in which requirements are converted to a working system that is continually revised through close work between an analyst and users. In other words, building a scaled down but functional version of a desired system is called prototyping. A prototype can be developed with some fourth-generation languages, with the query and screen and report design tools of a database management system, and with Computer-Aided Software Engineering (CASE) tools. Prototyping is a form of rapid application development (RAD).

ii. **Joint Application design**

Joint Application Design (JAD) is a structured process in which users, managers and analysts work together for several days in a series of intensive meetings to specify or review system requirements. Because of bringing the people directly affected by the system in one place and time, time and organizational resources are better managed. Also, group members develop a shared understanding of what the system is supposed to do.

iii. **<u>Participatory design</u>**

Participatory Design (PD), the end users of the system and improvements in their work is given central focus. PD may involve either the entire user community or an elected group of users in the development process. The organization’s management and outside consultants provide advice rather than control.

Question 5. ***How is system structure defined with the view of Information System?***

Answer:

- ◈ System structure defines the boundary of the system and the environment in which the system works.
- ◈ System boundary defines the components that make up a system.
- ◈ Anything outside the system boundary is the system environment.
- ◈ Within the system boundary there can be a number of subsystems, which carry out parts of the system function.

The system structure of an information system can be viewed as a collection of people, process and data.

- **<u>People</u>**

  The information produced by information systems is used by people in the organization in their everyday activities, such as in making decisions.

- **<u>Process</u>**

  In order to support the user activities and the interaction between the various users, information systems include processes that ensure that the right people receive the right information at the right time. These processes determine what is to be done with data as it enters and passes through the system.

- **Data**

  Data in used in the information systems for generating meaningful information. Data is stored in various equipments such as hard disks. Similarly, data needs to be transferred from one place to another through communication links.

Question 6. ***Describe Centralized system.***

Answer:

- In a centralized system, users are connected to a computer system through terminals or workstations.
- The computer system supports a number of databases.
- The computer system contains the program that allows the users to access the database.
- Many centralized systems support structured processes made up of a predefined sequence of steps.
- An example of centralized systems is data warehousing and data mining.
- ***Data warehousing*** refers to maintaining a central repository of records.
- Access to historical information that helps analysts to study patterns in past activities is called ***data mining*****.**

Question 7. ***Describe distributed system.***

Answer:

- In distributed systems, more than one computer systems are connected together to form a computer network.
- A significant amount of computation can be carried out on the workstation itself.
- An example of distributed systems is the client-server systems, where the workstation is known as the client and the computer system is known as the server.
- The server stores the data commonly used by its connected clients as well as common programs for the users.

- The computation on the data can either be carried out on the server and the result returned to the client or it can be carried out on the workstation by using the data and program sent by the server on request.

Question 8. ***Why is system analysis necessary?***

Answer:

- System analysis refers to analyzing how the system works and what its needs are.
- It takes place when new systems are being built or existing ones are changed.
- System analysis is necessary because it identifies what is possible and how the new system will work.
- This includes gathering the necessary data and developing models for the new system.
- Its crucial role is in defining user requirements, which is a statement of what the users of the system need from the system.

Question 9. ***What qualities should a system analyst have? How would you acquire these qualities?***

Answer:

- System analysts are people who analyze the way the existing system works and find out what its problems are.
- In order to perform the task of system analysis effectively, system analysts should possess a number of qualities. Such qualities are:

i. Analytical skills

Analysts should be able to analyze the system properly. For this

- Analysts should be able to develop a proper system thinking
- They should have adequate organizational knowledge
- They should be able to identify problems in the existing system
- They should be able to analyze the problems and propose ways for solving them

ii. **Technical skills**

In order to develop computer-based information systems, analysts should have the technical knowledge about computers, data networks, database management systems, operating systems, etc.

iii. **Management skills**

System analysts are almost always members of project teams and are frequently asked to lead teams. So, they need to have management skills to lead teams properly. For this they should be able to perform the following management tasks effectively:

- Resource management
- Project management
- Risk management
- Change management

iv. **Interpersonal skills**

System analysts need to work with all types of people during analysis. They must interact with users of the system too find out what they need in the new system. Similarly, they need to interact the management of the organization. For proper interaction with the various kinds of people, system analysts should have the following interpersonal skills:

- Communication skills
- Skill to work alone or with a team
- Skill to facilitate groups
- Skill to manage expectations

Some of the ways to acquire these qualities are:

- Reading trade publications
- Joining professional societies
- Attending classes
- Attending professional conferences
- Participating in electronic bulletin boards, news groups, etc.

- Taking every opportunity to practicing speaking for developing communicational kills, this involves activities such as speaking to civic organizations.
- Taking classes on business and technical writing from colleges and professional organizations.

Question 10. ***What are the characteristics of personal system?***

Answer:

Personal systems are simple computer-based systems that support one person only to keep track of his records. The characteristics of a personal system are:

i. **Components:** It has components, or subsystems, made up of people, process and data. A personal system has only one computer where all the data and programs are stored, and only one person is supported.

ii. **Inter-related components:** The various components are inter-related, that is, the function of one subsystem is somehow related the functions of the others.

iii. **A purpose:** The system, like all other information systems, has a goal or purpose that defines exactly what the system is supposed to do.

iv. **A boundary:** It has system boundary, which defines the set of components that can be changed during system design.

v. **An environment:** It has a system environment, which defines anything outside the system boundary.

vi. **Interfaces:** It has interfaces for allowing the users interact with the system.

vii. **Input:** it tales certain inputs for carrying out the necessary computations and data manipulations.

viii. **Output:** it gives certain outputs depending upon the input provided and data-manipulations carried out.

ix. **Constraints:** it has certain constraints within which it has to work..

x. **Feedback:** Feedback is used to monitor the current system and compare it to the system goal, to see if any variations exist. If yes, the variations are used to ensure that the system meets its goals.

The various processes involved in building information systems

There are mainly three processes involved in the development of information systems. They are:

i. **Development process**
   - Development process is a set of steps used to build a system.
   - During development, first a concept is built about what is to be developed.
   - Then a detailed requirements specification is prepared and the system is developed so that it meets the requirements specified in the requirements specification.
   - There are various methods and tools used in the development process.

ii. **Management process**
   - It includes the tasks required to manage a development process.
   - It is mainly concerned with organizing the work, ensuring that adequate resources are made available and monitoring the process of the work.
   - Organizing people involves selecting proper analysts, programmer and computer operators.
   - Monitoring the development work involves testing the new system against the users' needs.

iii. **Supporting process**

It is a process to provide facilities needed by development teams. It includes

- Ensuring that the equipments necessary for developing the system is provided
- Facilitating teamwork and communication to ensure that all team members are aware of each other's activity so as to avoid overlap and unnecessary delays and to ensure that everyone is working towards the same goal.
- Keeping track of design documents, so that the team members always have the latest documents.

2.SYSTEM COORDINATION

The changing organization

- There are two ways of viewing the organization **:**
    - One is a traditional hierarchical structure.
    - The other is a flatter structure with people working in task-oriented teams.
- Most organizations are usually a mix of the two with a trend toward the flatter structures.
- This trend places more emphasis on teamwork, with people working in task teams toward well-defined objectives.
- People in teams may include those employed by the organization and those outside the organization.
- Another emphasis is on the processes actually followed by people in organizations.

2.1. Hierarchical Structure

- An early accepted view of organizations, first proposed by Anthony (1965), sees organizations in three levels shown in figure 1- the strategic level, the management level and the operational level.
- People at each management level determine the tasks needed to be carried out at the next level and delegate these tasks to lower levels.

    - The people at the strategic or top-management level decide on the broad objectives for an organization, e.g., the kind of product, marketing strategy, etc.
    - The management level must acquire and arrange the resources to meet the goals, and define the detailed tasks to be carried out at the operational level. These resources are the people, machines, buildings and computers needed to accomplish the goals.
    - People at the operational level then carry out the detailed tasks.

♠

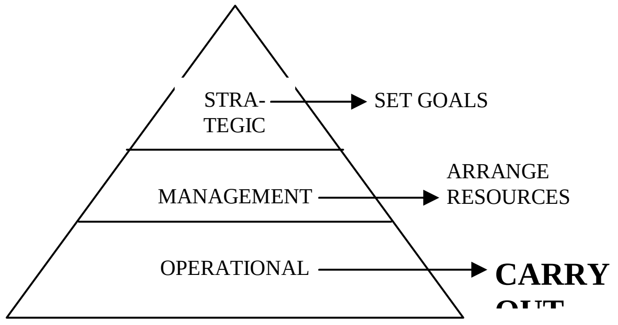


Fig. 1 **Organizational Levels**

Coordination in the hierarchical structure

- In hierarchical organizations, coordination between the people at the operational level is through the hierarchy.
- Communication paths are such that any requests or difficulties at one operational group are reported to management who then may coordinate with management of other operational groups to resolve any problems.
- Such coordination through the management structure is necessary to ensure that management is informed of any changes in resource requirements and that all levels of management are aware of changes at any point of the operational structure.
- This kind of structure has always assumed a relatively stable environment simply because a change to such hierarchical structures often requires major organizational changes.

2.2. Flatter Structure

- This view of organizations has been elaborated by Drucker (1998), and it suggests that flatter structures are needed because organizations are becoming information-rich.
- This implies that most information resides at the operational level of the enterprise and work must be organized to make best use of this information.
- Flatter structures thus enable information that exists at these levels to be coordinated, which leads to better decisions.

- The reasons for the need of flatter structure can be listed out as follows:
    - Customer preferences as well as economic factors continually change, and hence organizations must continually change to meet rapidly changing demands.
    - Products often require inputs of many skills and must often be customized to particular customer needs, which requires organizations that can quickly bring together people who have such skills and that can make changes quickly at the operational level by rearranging both resources and the tasks that people do.
    - Such rearrangement of activities is often difficult if it is to proceed through a number of hierarchical organizational levels. So, there has been a tendency to reduce the number of levels and to encourage change by supporting coordination at the operational levels

- Flatter structure of organization comprises of workgroups concerned with specific and often limited tasks.
- Such workgroups are usually empowered to make decisions on the use of resources without reference to management, whose main goal in this kind of organization is to provide support to the groups rather than to direct them.
- A workgroup or task team can be made up of people from a number of organizational units.
- Such workgroups do things like jointly preparing a document.
- Each workgroup may require people with different skills to match the objective of the workgroup.

Coordination in the flatter structure

- Coordination between people at the operational level in flatter structure is not through the management but direct between workgroups.
- Hence information in the operational level is well coordinated which leads to better decisions.

- Close collaboration in workgroups requires individuals to have a wide knowledge of the organization.
- In hierarchical organizations, each individual is concerned with their individual function, be it inventory control or financial management.
- In flatter structures, each individual must also have some understanding of the functions carried out by other groups with which the individual coordinates, and of how to coordinate with these groups because one individual may interact with individuals in many other groups.

The role of an individual in flatter structure

- The flattening of organizational structures changes the way in which people work.
- It adds responsibilities at the operational level, placing more emphasis on control through coordination at that level rather than on direction from the management level.
- The way in which individuals work and interact with each other is also different.
- One individual may interact with individuals in many other groups; hence each individual must also have some understanding of the functions carried out by other groups with which the individual coordinates, and of how to coordinate with these groups.
- Management in the flatter structures facilitates rather than directs activities.

Types of groups

- Open and closed groups
  - Open groups allow members to be freely added or deleted from the group.
  - Closed groups do not allow members to be freely added or deleted from the group.
- Loosely and tightly coupled groups
  - Loosely coupled groups allow members to act independently of each other.
  - Tightly coupled groups impose restrictions on interactions.

Information exchange and personal relationships

One of the most fundamental tools for people is to exchange messages and information. It is important because it allows people to develop a perception of what is going on and what possible problems can affect their decisions.

Question 1. ***What are the different kinds of communication found in an organization?***
Answer:
The different kinds of communication found in an organization are:

1. **The World Wide Web and intranets:**
   - The World Wide Web (WWW) uses a standard format to store and transmit information from one site to another using the Internet.
   - An organization develops a WWW site, which has the required information about the organization. WWW sites can be used for many purposes such as to publicize an organization's activities.
   - Networks within organizations are called local area networks, where they support one organizational unit.
   - *Intranets* support communication across the whole organization.
   - Intranets use the same technology as the Internet, but access is restricted to people within the organization.

2. **Formal meetings:**
   - Formal meetings or committees are one of the most common group interactions found in organizations.
   - They are most often face-to-face at the same location.
   - Computer support raises the possibility of holding a synchronous meeting with people at different locations.
   - Also, there can be mixed electronic and face-to-face components to the meeting.
   - Information can be collected and displayed with participants at a distance, whereas actual discussion takes place face-to-face.

3. **<u>Decision support system:</u>**
    - ❖ Decision support systems assist groups to make complex decisions.
    - ❖ Some decisions require an optimization algorithm, while many decision support systems are experimental in nature, where the user tries different inputs to see their effects.
    - ❖ A third kind of decision is one of policy nature where one develops alternate positions and then justifies them by argumentation.
    - ❖ Decision support usually assumes asynchronous interaction, although there can be some advantage in synchronous discussion to resolve conflicts.

Question 2. ***What do you understand by decision support system?***

Answer:

- ❖ Decision support systems assist groups to make complex decisions, that is, they support decision-making.
- ❖ They are one of the communication tools used in an organization.
- ❖ There are many ways to make decisions.
    - → Some decisions require an optimization algorithm.
    - → Many decision support systems are experimental in nature, where the user tries different inputs to see their effects. The response is used to try new inputs, and the process continues until a satisfactory result is obtained. Most decision support systems are based on a model that is continually refined. The user inputs some possibilities into the model and evaluates them. Then other possibilities may be tried.
    - → Some decisions are of policy nature where one develops alternate positions and then justifies them by argumentation, involving defining an objective, defining alternate solutions and making arguments for and against solutions.
- ❖ Decision support usually assumes asynchronous interaction, although there can be some advantage in synchronous discussion to resolve conflicts.

Question 3. ***What are the different types of communication work found in an organization?***

Answer:

The different types of communication work found in an organization are:

i. **<u>Planned Work</u>**

- ❖ In planned work it is possible to predefine the tasks to be done and the sequence of doing them.
- ❖ Everyone can then be assigned a task and told ahead of time what is expected of them.
- ❖ Thus, in planned work, process is predefined as a sequence of steps, and one or more different persons carry out each step.
- ❖ The persons may be at different locations, and may carry out their tasks at different times.
- ❖ Messages can be passed between the persons as a form, word of mouth, etc., or coordination can be achieved through shared files where a person records each action on a file. Such records are then used to inform others to carry out some action.
- ❖ Systems that support planned work are often called workflow processes because there is a defined flow of information and defined actions to be taken at different points of this flow.
- ❖ Transaction Processing System is an example of systems that support planned work.

ii. **<u>Situated work</u>**

- ❖ In situated work the next task is determined from the current situation, that is, people carry out tasks as the need arises.
- ❖ It is often unstructured and requires closer coordination than structured or preplanned work because situations change rapidly requiring team members to quickly adapt to the change and to coordinate their activities at a much more detailed level.
- ❖ This closer coordination requires more face-to-face interaction.

- An example of situated work is the design of an artifact, where different people work on different parts of the artifact (e.g., a joint report, an engineering design, etc.). Though each part of the artifact must fit precisely with the other parts, the final detailed artifact structure as well as the design process cannot be precisely predefined because an outcome at one point of the design may require new and unpredicted work at other points. The process is made up of a variety of tasks, some totally creative (e.g., brainstorming for ideas for the design) and some routine (e.g., evaluation of ideas). Usually, the creative tasks are performed with close and spontaneous interaction between many designers involving face-to-face coordination.

Question 4. ***Describe what you understand by workflows.***

Answer:

- Workflow refers to a predefined flow of information and defined actions to be taken at different points of this flow.
- Systems involving workflow are called workflow processes. They support planned work as they have a predefined set of actions.
- Computers support workflow processes usually using transaction-processing systems.

Question 5. ***Why is it preferable to synthesize group support systems from existing companies rather than building them from scratch?***

Answer:

- It is preferable to synthesize group support systems from existing companies rather than building them from scratch because writing each group system is not economical.
- Group support systems might include decision-making modules, brainstorming modules, procedural modules, etc. these generic modules when put together form the complete group support system.
- Because so many modules need to work together, such systems are complex, and building them is uneconomical in terms of money and time both.

- Hence, group support systems are synthesized using systems from existing companies, by performing some tailoring on them, that is, tools that can be customized to particular needs are used.
- This is called reuse and it is important in group-situations, which are naturally dynamic.

Transaction Processing System

- ***Transaction*** means an interaction with the database.
- Such interactions occur continually in a workflow, as people have to manipulate data at defined workflow stages.
- Computer programs can be used to allow people to access the database, make any necessary changes to it, and use them to initiate a further transaction.
- Such a computer system that manages transactions is called a ***Transaction Processing System.***
- The ***transaction supporting system*** provides responses to the user as the transaction progresses through the system.
- Any errors and inconsistencies, as well as the result of the final updated database, are reported. Responses can be provided in a number of ways, depending upon the transaction system mode.

Transactions may be input in two modes:

- → In the on-line mode, transactions are input into the system as soon as they arise, e.g., bank transactions.
- → In the batch mode, transactions are collected into batches, which may be held for a while and input into the computer later.

The transaction processing system supports planned work.

3. CONCEPT FORMATION

Question 1. W*hy is identifying problem important?*

Answer:

- Identifying problem is important because solving the wrong problem causes wastage of time and resources.
- Identifying problem, that is, finding out what problem we are going to tackle is one of the most important tasks in systems work, that is, in systems analysis and development.
- Systems analysis and design is done to improve systems. Hence, we must identify which system we are going to analyze and how we might improve it.
- If solution a problem that does not exist in the system is developed then it is a mere wastage of time and other resources, because the solution is of no value to anyone.
- Thus identifying problem is important so that the right solution is developed and the system is improved as a consequence of implementing the solution.

Question 2. ***How would you go about identifying problem?***

Answer:

- The first work to be done during systems analysis and design is to identify what problem we are going to solve.
- What has to be done is to define problems in the system and propose ways to overcome them, that is, propose ways to improve the system.
- Problems can be identified in many different ways, some of which are informal, such as listening to what people by saying.
- In any case, we often compare what is happening now to what we think should be happening.
- We get ideas about what should be happening both internally and externally.

  - **<u>Identifying problems using internal considerations</u>**

    While setting goals, organizational constraints are to be considered. Similarly, deficiencies in the existing system are to be identified. It then becomes the project goal to remove such deficiencies. Deficiencies are often found in the course of interviews or by examining documents about system performance. Such deficiencies can be

- Missing functions
- Unsatisfactory performance
- Excessively costly operations

While identifying deficiencies and setting goals, competitor-performance and technical developments outside an organization are important factors to be taken into consideration.

♠ **<u>Identifying problems using external considerations</u>**

We can compare our operations against some accepted benchmark or by looking at what our competitors are doing. External factors might also include changes in government policy, client preferences, etc.

Some of the ways of finding problems externally are:

- Using normative models, which describe an accepted or conventional way of doing something
- Using historical models of the ways in which organizations develop. This is particularly useful in information systems design because of the development of technology.
- Comparing our activities against a competitor activities
- Analyzing changes to government policy and community attitude

These external conditions can be used to identify differences between the way things are done in our organization and the accepted way of doing things outside. For example, the areas where others use computers effectively can be observed for proper implementation of computers in our own organization. Changes to government rules are also important. Changes to tax policy, for example, will often require changes to accounting systems.

Question 3. *How do you justify your solutions? Explain the activities to give the solution.*

Answer:

- Solutions given to any problem must not be unrealizable ideals that cannot be actually implemented.
- They must be developed within the practical bounds of the organization.
- Once conceptual solutions have been found these solutions should be justified.
- Justification that the solution is worthwhile in terms useful to the business is made to people including the users and management.
- Comparing the solutions with one another and selecting the one that best fits in the system justifies solutions.

The activities to give the solution are:

- **Generation of broad alternative solutions:**
  - Broad alternative solutions give indication of what the new system should look like. Analysts must be creative and imaginative in generating new ideas.
  - The broad solutions should provide enough information to make reasonable estimates about project cost and give users an indication of how the new system will fit into the organization, but they do not need to explain the detailed system operation.
  - The organization has certain constraints on the amount of available funds and personnel skills, and on working and accepted standards.
  - Proposed solutions should not exceed such funding limits or ignore some critical system operation or data needed.

- **Evaluation of the proposal:**

Once proposals are generated, they are evaluated. Three things must be done to establish feasibility. They are:

- ✓ *Technical feasibility*:

  This evaluation determines whether the technology needed for the proposed system is available or not and how it can be integrated within the system.

- ✓ *Operational feasibility*:

  Operational feasibility determines how the proposed system will fit in with the current operations and what, if any, job restructuring and retaining may be needed to implement the system. The evaluation also determines the general attributes and skills of existing personnel and whether any such restructuring of jobs will be acceptable to the current users.

- ✓ *Economic feasibility:*

  This evaluation looks at the financial aspects of the project. It determines whether the project's goals can be achieved within the resource limits allocated to it. It also determines whether it is worthwhile to proceed with the project at all, or whether the benefits obtained from the new system are not worth the costs.

Evaluation of proposals given as broad alternative solutions results in the generation of one solution that is found to be the most satisfactory, and hence, accepted.

Question 4. ***How do you generate broad alternate solutions?***

Answer:

- ◈ Broad alternate solutions give an indication of what the new system should look like.
- ◈ Generation of broad alternate solutions requires creativity and imagination to generate new ideas, that is, to think up new ways of doing things.

- In order to generate broad alternate solutions, first of all, the problem should be properly identified and defined.
- These define any system deficiencies that must be addressed in the solution, and in turn define the project goals.
- Then the creativity of the analysts is used to generate probable solutions to the problems.
- There is no need to go into the detailed system operation.
- The solution should provide enough information to make reasonable estimates about project cost and give users an indication of how the new system will fit into the organization.
- Only broad descriptions need to be given here. Detail report layouts, interfaces or handling error conditions should not be specified.
- While generating broad alternate solution, it should be kept in mind that the proposed alternatives should not be such that cannot be supported in the organization.
- These means that the proposed solutions should not exceed funding limits or ignore some critical system operation or data needed.

Question 5. ***Why ore constraints important when alternative solutions to a problem are proposed?***

Answer:

- Every organization has certain constraints on the amounts of available funds and personnel skills.
- Similarly, the organization has certain standards to be maintained within its periphery.
- The working environment in organizations might differ depending upon the nature of organization.
- Similarly, the organization should abide by the national and international business policies as well as government policies.
- The alternative solutions proposed for a problem should be such that all the above-mentioned standards and policies are abided by.
- Similarly, the available fund and personnel skills must also be considered.

Hence, constraints are important when alternate solutions to a problem are proposed so that the proposed alternatives can be supported in the organization.

Question 6. ***How do you prepare a statement of user's requirements?***

Answer:

- ❖ A statement of user's requirements is a proposal prepared after broad solutions have been generated and feasibility analysis performed to select an alternative.
- ❖ In order to prepare a statement of user's requirements, first of all the feasibility analysis and selection of alternative should be complete.
- ❖ Then a precise proposal, which is the required statement of user's requirements, is prepared, generally by including the following sections:
  - ♠ A statement that defines the business problem being solved
  - ♠ The chosen solution, explaining why it was chosen and briefly indicating the other alternatives
  - ♠ A description of how the new system will work and its impact on external clients and internal users
  - ♠ Justification for choosing the preferred alternative and its economic, technical and operational advantages
  - ♠ What various people in the organization will have to do to implement the solution
  - ♠ The effect of the solution on the way people work, including any new skills needed by people and the way these skills can be leant

Components of a broad solution

A broad solution includes the information needs to estimate the cost of project and to determine how the system will be used. It need not be a detailed description of system. Broad solutions generally include:

- ♠ Any additional equipment that will have to be purchased for the project, in order to estimate some of the direct costs of the project

- Any computer networking requirements, to estimate costs of communications equipment
- Any new computer systems that will have to be developed, to estimate the development costs
- What is to be done by the computer and what will remain manual
- The information that will be made available by the system
- The services that will be provided to the customers and specially any important improvements
- Any computer interfaces provided to computer users
- Rough ideas of processes that will be followed using the new system

Risk analysis

- Risk analysis centers on identifying those aspects of a project where there is the largest uncertainty about getting a successful outcome.
- Once areas of high risk are identified they are given special attention.
- This may be to either take an alternative course of action or to manage it in special ways.
- Areas of risk may arise due to lack of technical or economical efficiency.

Economics feasibility

- Economic feasibility concerns returns from investments in a project.
- It determines whether the project's goals can be achieved, within the resource limits allocated to it.
- It also determines whether it is worthwhile to invest the money in the proposed project or whether something else should be done with it.
- To carry out an economic feasibility study, the money values of any purchases or activities needed to implement the project are placed against any benefits that will accrue from the new system created by the project. Such calculations are described as cost-benefit analysis.

Cost- benefit analysis

It consists of two steps

- ♠ Producing the estimates of costs and benefits
- ♠ Determining whether the project is worthwhile

Producing costs and benefits

It produces a list of what is required to implement the system and a list of the new system's benefits.

Costs can be of two types:

- ✓ Tangible cost: It includes
  - Equipment costs
  - Personnel costs
  - Material costs
  - Conversion costs
  - Training costs
  - Other costs, e.g., consultants' cost, management overhead, travel budgets, etc.
- ✓ Intangible cost

Benefits can be of three types:

- ✓ Tangible benefits, e.g., reduced production cost, reduced processing cost, etc.
- ✓ Less tangible benefits, e.g., the possibility of increased sales, possibility of wider market, etc.
- ✓ Intangible benefits, e.g., benefit of better decision-making, benefit of maintaining a good business image, etc.

The sum value of costs of items needed to implement the system is known as the cost of the system.

The sum value of the savings made is known as the benefits of the new system.

Determining whether a project is worth while

The costs and benefits are used to determine whether a project is economically feasible.

There are two ways to do this:

- ✓ ***Pay back method***

  It defines the time required to recover the money spent on a project by summing up the net benefit (difference between the cost and benefit) for each year.

- ✓ ***Present value method***

  First, the project benefits are estimated for each year from the time of system implementation. Then, the present values of these savings are computed depending on the interest rate. If the project cost exceeds the present value, then the project is not worthwhile.

Payback Method : It is a method to determine the time required to recover the money spent on a project.

Example:

Original investment (cost) =Rs. 12,000

	Return Benefits	
Year	Project X	Project Y
1	3,000	3,000
2	3,000	3,500
3	3,000	4,000
4	3,000	4,500
5	3,000	5,000

Solution:

	Return Benefits		Cumulative benefits	
Year	Project X	Project Y	Project X	Project Y
1	3,000	3,000	3,000	3,000
2	3,000	3,500	6,000	6,500
3	3,000	4,000	9,000	10,500
4	3,000	4,500	12,000 🡐	15,000
5	3,000	5,000	15,000	20,000

Therefore, the cumulative benefit of project X at year 4 is equal to the cost (i.e. Rs.12,000) and that of project Y lies between year 3 & year 4. So, the payback period is 4 years for the project X and for project Y is calculated below:

Payback period= (Amt not covered in lower yr / Difference in cumulative amts. of HY & LY)+LY

= (1500 / 4500) + 3 = 3.33 yr.

<u>Present Value Method</u> : The Idea of present value method is to determine how much money it is worthwhile investing now in order to receive a given return in some years' time . The answer depends on the interest rate used in the evaluation calculations.

Example :

Let us suppose, we have :: interest rate =10% , cost Rs. 3000

Year	Benefit	Discount Factor $[1/(1+r)^1]$	Present value of benifit
1	600	0.909	600 x 0.909 = 545.4
2	900	0.082	900 x0.082 = 743.4
3	1,500	0.751	1,500 x 0.751 = 1126.5
4	1,500	0.683	1,500 x 0.683 = 1024.5
5	1,800	0.621	1,800 x 0.621 = 1117.8
Total	6,300	3.046	3,557.6

Since the present value (PV) of benefits is more than the cost (i.e. Rs. 3000), the project is worthwhile.

4.REQUIREMENTS ANALYSIS

Question 1. ***Why are unambiguous terms needed in different worlds?***

Answer:

- One of the most important factors in building correct system is to first clearly define what the system must do.
- For this a broad conceptual solution must be agreed upon. Then a detailed analysis of user requirements must be done, and a system specification must be developed.
- This step is needed to develop a good understanding of the system and its problems.
- This identification of detailed user requirements is very important so that no malfunctions exist and hence there are fewer chances of severe system failures.
- For identifying detailed user requirements, the analysts must discuss with users what they require of the system, that is, communication is very important.
- While identifying user requirements, it is necessary to reach upon an agreement to specify what we are to do.
- Ambiguities can easily arise during discussions between the various people involved in analysis because of the different jargon used by different people-in particular, the users, who often speak in terms common to their domain, and computer analysts, who may use computer terms.
- Because of such ambiguities, the correct requirements may not be identified since the analysts may not correctly understand the terms used by the users due to their ambiguous meanings.
- So, to ensure that no incorrect interpretation of user requirements is made and that ultimately a correct system is developed, unambiguous terms are needed in different worlds.

Question 2. ***What are the different worlds to be considered during requirement analysis?***

Answer:

The different worlds to be considered during requirement analysis are:

- ♠ **Usage World**

  People in the usage world are the users of the system. Such users use terms common to their work and describe their work in terms of scenarios, that is, by giving examples of things that happen in their world. Requirements are first identified from information collected in the usage world. One way to describe the usage world is to use rich pictures.

- ♠ **Subject World**

  The subject world focuses on business issues. Its goal is to identify the crucial measures of business success and propose ways to achieve them. The important factor in subject world modeling is to eliminate the jargon found in the usage world and to clearly define the terms that describe the user world in unambiguous, well-defined terms rather than in terms of specific scenarios.

- ♠ **System World**

  The system world represents the subject terms in abstract system terms that are useful to the developers but can still be understood by the users. This area is often termed conceptual modeling in computer system design. It describes the system in terms that are useful to computer system designers in developing a computer system specification. Data flow diagrams, ER diagrams, process descriptions, etc. are used to describe the system world.

- ♠ **Development World**

  Computers system developers talk about computer systems using terms particular to the development world, such as operating systems, databases and so on. In order to bring the usage world and development world together, it is necessary to translate from the language in one world to that of another in an unambiguous way. This involves using a number of interim steps or other languages, which are the subject world and system world.

Question 3. W*hy are scenarios important?*

Answer:

- Scenarios are examples of things that happen in the usage world. People in the usage world use scenarios to describe their work.
- Scenarios are important because they support models such as the rich picture by illustrating system dynamics using typical sequences of actions in the user world.
- A scenario may be a specific instance that describes not only the process followed but also the problems found and the ways in which they were overcome. A scenario may also be a general scenario that covers specific instances.
- Scenarios are important also because they are very helpful in describing requirements in object-oriented design, where general scenarios are known as use cases.
- Scenarios can be expanded to indicate the kind of support to be provided for the interactions.

Question 4. *What is the role of requirement analysts?*

Answer:

- In order to build correct systems, a proper understanding of what the system must do is required.
- For this, after agreeing upon a broad conceptual solution, a detailed analysis of user requirements must be done, and a system specification must be developed.
- This step is needed to develop a good understanding of the system and its problems.
- The role of requirement analysts is to perform this detailed system analysis and to produce an analysis model, which clearly describes how the system works now, and a requirements model of what the new system must do.
- In order to perform their role, system analysts need to discuss with users about what they require from the system.

- A repeated visit with the users validates the initial requirements model developed by the analyst.
- In order to define user requirements, analysts must have an understanding of how the system works and what its problems are.
- Analysts should not make their own assumptions while analyzing the requirements.

Question 5. ***Where would participation be the best way of gathering requirements?***

Answer:

- Participation would be the best way of gathering information in studying the way that groups of people work.
- Their goal is to study the dynamic social situations that occur in such environments. It is usual here to identify communities or workers and to analyze their interactions.
- Participation becomes especially important in such group situations because while gathering information by participation the analyst actually participates in the group activities by becoming a team member and assisting other team members. Hence, the analyst gets a clear picture of how the group actually functions.

Question 6. ***What do you understand by the term storyboard?***

Answer:

- ***Storyboard*** is a term used in prototyping process during requirements collection.
- In story boarding, a series of small prototypes are tied together, so that the user can see how the whole system works.
- A storyboard is often developed as a sequence of screens with which the user can experiment and make comments for further improvements.
- The kind of input can be shown on the requisition screen and the results can be illustrated in the approval screen.
- Initially screen inputs need not result in any actual computations or storage in the database but simply moved from one screen to another.
- As the prototype develops, we may add to these programs a database or some checking. Thus the prototype may gradually evolve into a working system.

Collection Methods

Collection methods refer to the methods for collecting information to develop an understanding of how the system works and what its requirements are. There are four main ways of collecting information. They are:

- **Asking questions**

  This includes interviewing people in the system; performing surveys and making the users fill up questionnaires, or using electronic means such as e-mail or a discussion database.

- **Observational studies**

  This includes ethnography and participation within the user environment. Direct observation of the working environment is done, rather than gathering information through an informal description through interviews.
- **Prototyping**

  Prototyping is primarily an experimental method in which users can experiment with a rough system and make comments in usage model terms about its suitability for the workplace. Prototyping can be done either for the requirements or for the interface.
- **Formal sessions**

  This includes structured workshops, group discussions and facilitated teams.

Ethnography

- It is used to collect information by observation.
- The main characteristics of ethnographic studies are:

  - Analysts observe or possibly even participate in users' activities
  - Interviews are conducted at the site of the work, possibly as informal discussion rather that formal interviews
  - There is emphasis on examining the interaction between users

- ♠ There is emphasis on tracing communication links
- ♠ There is detailed analysis of artifacts.

- ❖ The ethnographic approach has a number of advantages. Because interviewing is carried out in the actual work site, the system is directly observed as it actually works. The users are not disturbed in their activities, and information is gathered directly and not from an informal description obtained through interviews.

Ethnography can be done by direct participation or by observation.

- ♠ **<u>Analysis by participation</u>**
  In this the analyst becomes a member of a team assisting other team members, that is, the analyst actually participates in users' activities. It is especially important in studying the way that groups of people work

- ♠ **<u>Analysis by observation</u>**
  The goal here is to observe what people do in an unobtrusive way. The best way to do this is by video recording. It is important in video recording to ensure that the presence of the video camera itself does not alter behavior while at the same time to collect sufficient in-depth information to make useful observations.

Some analysis techniques

Ethnography includes a number of techniques. Some of these are:

- ♠ **Analyzing people's roles**
  This identifies how a particular person feels about their work and the kinds of problems they encounter.

- ♠ **Analyzing interaction**
  Interaction analysis defines how users work together in groups.

- **Analyzing location**

  A study is made of what happens at a particular place over a period of time. Often the study produces a set of snapshots of activities during that period.

- **Analyzing artifacts**

  The emphasis on artifact analysis is how it fits in to the flow of work rather than on artifact structure itself.

- **Task analysis**

  The analyst studies the processes within a system and the role of individual users. Emphasis is on the information needed by the user, what the user does with the information and where it is obtained.

Prototyping

- It is a method used to test or illustrate an idea and build a system in an explorative way.
- It is particularly useful when a totally new system is proposed.
- Prototyping is primarily an experimental method.
- In prototyping, a rough system is built, and users can experiment with it and make comments in usage model terms about its suitability for the workplace.
- Their reactions are obtained and used to define requirements in an iterative way. Prototypes can be used to describe an interface or a process.

Interface prototyping

- In this way of prototyping, screens that illustrate what user will have to do in the new system are developed.
- These can then be used to describe to user the kind of information that would be made available to them from the system and how they can work with this information.
- The actual system is not written. Instead, stored example screens can be illustrated to a user.

Prototyping process

Prototypes can be used to describe a process that involves a number of users. The term storyboarding is often used to describe this approach. In story boarding, a series of small prototypes are tied together, so that the user can see how the whole system works. A storyboard is often developed as a sequence of screens with which the user can experiment and make comments for further improvements.

5.ANALYSIS PROCESS

Question 1. ***What do you mean by model in system analysis? Define the following models:***

(a) Balancing Model (b) Essential Model

(c) Implementation Model (d) Behavioral Model

Answer:

Model in system analysis is the pictorial representation of the system. Examples of models are Data Flow Diagram (DFD), Entity Relationship (E-R) Diagram, etc.

(a) Balancing Model

Balancing is the synchronizing process of a DFD at different levels so as to make consistency between the vertical levels and to ensure completeness of the DFD. It is also used for quality assurance for DFD.

In balancing a DFD, the inputs to a process and outputs from it are kept intact, that is, the neighboring processes, data flows and storages are supposed to roll over to the exploded lower level diagram, when it is leveled for exploded into a number of sub processes.

(b) Essential Model

All the logical models (e.g., logical DFD), which illustrate the essence of the system, are called essential models. The essential or logical model describes what the system is and what is supposed to do, but not how it does it, that is, it is independent of any implementation issues.

(c) Implementation Model

The implementation model, also called the technical model, describes what the system is and what it does, as well as how the system is physically and technically implemented. They are dependent upon the limitations of the technology complemented. All the physical models, such as physical DFD, are called implementation models.

(d) Behavioral Model

From structured analysis perspective, activities and states of an entity are explained under entity behavioral model. Behavioral model also comes under object-oriented analysis where object behavior is explained.

Entity behavior modeling defines events and enquiries and their interaction with the logical data model and the conceptual model processes they trigger.

In this model three activities are involved:

i. Event identification
ii. Enquiry identification
iii. Entity life history analysis

Events are generally classified as Business Events (e.g., job application), which are triggered by external business functions and do not affect the data of computer, and System Events (e.g., receipt, notification, etc.), which follow business events and trigger the conceptual model which updates the data.

Question 2. ***What are the designing strategies in system design?***

Answer:

The designing strategies in system design are:

i. <u>Modern structure design</u>

It is a process-oriented technique for breaking up a large program into hierarchical modules that result in a computer program that is easier to implement and maintain. It is also called top-down program design. The tool for structured design is structure chart.

The drawbacks of structured design are:

- Because of evolving Event-Driven and Object-Oriented techniques, now-a-days structured design is not used effectively. It is more popular in mainframe-based applications.
- It does not help input-output design, database or file structure design.

ii. **<u>Information engineering</u>**

It is a data-centered technique for designing the system. This strategy uses the other techniques like modern structured Design, prototyping and Object-oriented analyses and Design.

iii. **<u>JAD (Joint Application Design)</u>**

Joint Application Design (JAD) is a structured process in which users, managers and analysts work together for several days in a series of intensive meetings to specify or review system requirements. Because of bringing the people directly affected by the system in one place and time, time and organizational resources are better managed. Also, group members develop a shared understanding of what the system is supposed to do.

iv. **<u>RAD (Rapid Application Design)</u>**

It is a merger of various other technologies from structure design. The fundamental principle of any RAD methodology is to delay producing detailed system design documents until after user requirements are clear. The prototype serves as the working description of needs. RAD methodologies emphasize on gaining user acceptance of the human-system interface and developing core capabilities as quickly as possible, sacrificing computer efficiency for gains in human efficiency in rapidly building and rebuilding working systems.

v. **<u>Object-oriented Design</u>**

In object-oriented design, the processes, data and flows are combined into one entity called object. It helps in easy conversion from analysis to design models and supports multimedia.

Question 3. ***What do you know about automated tools?***

Answer:

Automated tolls are used to make the information systems development process easier by automating the process of developing models, generating codes, etc.

Computer-aid software engineering (CASE) refers to automated software tolls used by system analysts to develop information systems. It aids the system analysts with tools so that higher quality systems are constructed on time and within budget, maintained economically, and changed rapidly.

The objectives of using CASE automated tolls are:

- Improving the quality of the systems developed
- Increasing the speed with which systems are designed and developed.
- Improving the testing process through the use of automated checking.
- Improving the integration of development activities via common methodologies.
- Improving the quality and completeness of documentation.
- Helping standardize the development process.
- Improving the management of the project.
- Simplifying program maintenance.
- Promoting reusability of modules and documentation.
- Improving software portability across environments.

Reverse engineering and reengineering are two categories of CASE.

Reverse engineering: Automated tolls that read program source code as input and create graphical and textual representations of program design-level information such as program control structures, data structures, logical flow, and data flow.

Reengineering: Automated tools that read program source code as input, perform an analysis of the program’s data and logic, and automatically, or interactively with a systems analyst, alter an existing system in an effort to improve its quality or performance.

CASE tools are used to support a wide variety of **SDLC** activities. **CASE** tools can be used to help in the project identification and selection, project initiation and planning, analysis, and design phases **(upper CASE)** and / or in the implementation and maintenance phases (**lower CASE**) of the **SDLC**. A third category of **CASE**, cross life cycle **CASE**, is tools used to support activities that occur across multiple phases of the **SDLC**.

The general types of CASE tools are:

- Diagramming tools
- Computer display and report generators
- Analysis tools
- A central repository
- Documentation generators
- Code generators

6.DEVELOPMENT PROCESS

Question 1. ***Why is a life cycle needed for the development of information system?***

Answer:

Life cycle refers to the set of steps that starts with a set of user requirements and produces a system that satisfies these requirements. Hence, an information system cannot be developed without a life cycle. The steps involved in the development of an information system form a cycle. This is because once the information system has been built, implemented and tested; new requirements may arise over time that may require minor or major changes to be made in the system. In such situations, all the steps needed to develop a system need to be carried out again. Thus, the development steps continue occurring as long as the system exits. Hence, life cycle is needed for the development of information system so that changes can be made after the system has been implemented.

Question 2. ***What is the difference between highly structured teams and adaptive teams?***

Answer:

The differences between highly structured teams and adaptive teams are:

i. Structured teams are most common in planed work whereas adaptive teams (that include open, synchronous and random teams) can be for planned work (open and synchronous teams) or for situated work (random teams).

ii. Structured or closed teams work towards well-defined requirements with the work subdivided into well- defined tasks whereas in adaptive teams, the tasks may not be all predefined or assigned to particular users. Moreover, in random teams, even the goal is not pre-specified but evolves as the project proceeds.

iii. Structured teams can be effective in projects that include many repetitive and similar tasks whereas adaptive teams are effective in projects such as decision support systems, or complex systems where tasks are not predefined.

iv. In structured teams the roles of team members remain more or less the same till the completion of the project whereas in adaptive teams, roles assigned to the team members may change as circumstance changes.

v. In structured teams, the documentation is structured, that is, the team members need to produce specific documents at the completion of their tasks whereas in adaptive teams, the team members may be added or removed as per the need and structured documentation is not possible.

Question 3. ***Describe some roles that you could expect to find in a team.***

Answer:

Role refers to the responsibility undertaken by a person. The roles of a different people are clearly defined in structured teams while in open, synchronous and random team structures, the roles of different people may change as the development process proceeds. Assignment of roles to different team members according to their capabilities and expertise is a critical factor in system development. Some of the roles that can be found in a team are:

i. **Team leader:** The leader in random teams does not usually assign tasks to other members, but facilitates the work of the rest of the team by arranging meeting, removing obstacles to progress, and recording progress towards achieving a goal.

ii. **Implementer:** He develops the programs.

iii. **Technical reviewer:** He validates outputs against requirements.

iv. **Scribe:** He keeps records of meeting.

v. **Specialist consultant**: He provides general consultative advice or support on computing tools and techniques as needed.

vi. **Chief programmer**: In chief programmer teams, a chief programmer writes the most difficult system programs.

vii. **Program librarian**: He maintains up- to date program documentation.

viii. **Manager**: He does the administrative tasks.

Question 4. ***Define the phases used in the liner cycle.***

Answer:

The phases used in linear cycle (waterfall cycle) are:

i. **Phase1-Problem definition**

In provides a broad statement of user requirements and thus sets the directions for the whole project, that is, it sets the project goal. It also sets the project bounds (that is, what parts of the system can be changed by the project and what parts are to remain the same. It also sets the resource limits for the project.

Phase output

- User requirements of the new system.
- Project goal, its bounds and resource limits.

ii. **<u>Phase 2-Developing the system specification</u>**

It involves the following activities

- Producing a detailed analysis model
- Producing a requirements model
- Producing the design models
- Producing a high-level description of computer system requirements using systems terms.

Phase output

- Analysis model
- Revisions to project goals and cost-benefit estimates
- Requirements model
- Design model
- Broad system specification
- Project development plan
- Test plan

iii. **<u>Phase 3-System design</u>**

It produces a design specification for the new system.

It usually proceeds in two steps:

- Phase 3A-Broad design

  It identifies the main architecture of the proposed system and verifies it against the proposed system model and validates it against user requirements.

  The models produced in the system specification are converted into computer systems.

  The network configuration, including the size of the computer and the software needed to put the system together is defined.

- Phase 3B-Detailed design

  During this phase the database and program modules are designed and detailed user procedures are documented. The interactions between the users and computers are also defined.

Phase output

- An implementation model for the new system: It includes
  - ✓ Proposed equipment configuration
  - ✓ Specifications for the database and computer programs

- – Detailed user procedures: It includes
  - ✓ Input forms
  - ✓ Interactions between users and computer
- – User manual

iv. **<u>Phase 4-System development</u>**

It consists of two smaller phases. They are:

- ▪ Development: During development
  - – Individual system components are built and tested
  - – Database is initialized with data
- ▪ Implementation: During implementation components built during development are put into operational use.

Phase output

-Working system

-Complete documentation.

After the above-mentioned four phases have been completed, post- implementation review and maintenance are carried out.

Question 5. W*hy is it difficult to build decision support systems using a linear cycle?*

Answer:

In linear cycle, the whole system is brought into service after the implementation phase. Linear cycle is best suited to problems that are well understood and highly structured and when accurate predictions about the system behavior can be made in the early design stages. But decision support systems have a degree of uncertainty because it is not clear whether a computer can actually be used to solve the problem. While developing decision support systems, the interactions used to support the decision-making cannot be accurately predicted and an experimental approach is needed.

Hence, because in a linear cycle, the interactions and steps need to be pre-specified, which is not possible with decision support systems, it is difficult to build decision support systems using a linear cycle. An experimental cycle such as the evolutionary cycle would be better to build decision support systems.

Question 6. ***Explain how the linear cycle meets top-down problem solving.***

Answer:

Linear cycle is a development process that centers on planned work. Here, development activities are grouped into a sequence of consecutive phases

- Problem definition
- Developing the system specification
- System design
- System development

After the completion of the fourth phase, post-implementation review and maintenance are done. Testing proceeds in parallel with the major phases. In the liner cycle, the above-mentioned phases must come in the specified order. A phase cannot begin until the proceeding phase is not complete, because the output of a phase is used as input for the next phase.

In top-down problem solving, successive phases elaborate the system in increasing detail; with each phase defining a partial solution and then calling for a more detailed evaluation in the next phase, that is, phases follow one another in a sequence. Hence, linear cycle meets top-down problem solving, since in a linear cycle, the phases occur one after the other in a strict sequence, and it is not possible to go back to a previous phase after another phase has begun.

Question 7. ***What is problem- solving cycle? How do you explain evolutionary development in problem solving cycle?***

Answer:

Problem-solving cycle is a set of steps that starts with a set of user requirements and produces a system that satisfies these requirements. It is also know system life cycle. It includes developing an understanding of the system, creating models, making decisions on what is to be done and planning the work.

Evolutionary development develops systems in an experimental way. The evolutionary design cannot be precisely specified. It develops the whole project as a number of stages, with the outcomes of one stage serving to identify the conceptual solutions for the next stage.

In evolutionary design, a system part is developed and more about the problem is learnt from the operation of the part. Each step adds a new capability to the system, and experience gained with a system is used to define the requirements for the next step. The next step extends system capability a little bit further and the process continues until no further improvement appears possible or worthwhile.

As seen above, the evolutionary development can be used to solve problems that are unstructured and harder to understand and that require more experimentation in the early stages. Evolutionary problem solving is a gradual problem solving cycle where transfer of ownership and conversion becomes a gradual process. One kind of problem that is best solved by evolutionary design is decision support systems.

7.STRUCTURED SYSTEM DESIGN

Question 1. ***What is system design? How does structure chart play the role in system design?***

Answer:

- System design is that phase of system development that produces design specification for the new system.
- It serves the objective of specifying modules that satisfy a variety of good design criteria. Such designs result in programs that are easy to develop and later to change.
- While progressing from system specification to program development, the DFD processes to be automated are selected and divided into subsystems, with each subsystem containing a number of logically connected processes.
- These subsystems are then converted to program modules, which are shown on a structure chart.
- The program modules are grouped into load modules during implementation.
- Hence, we see that structure charts are used to represent the program modules during system design.
- Structure charts also show the connections between modules. Thus, structure charts play an important role in system design by specifying the various program modules and the connections between them.

Question 2. ***Describe module coupling.***

Answer:

- Module coupling describes the nature, direction and quantity of parameters passed between modules, that is, they measure the quality of connections between modules in the structure chart.
- Structure charts with low module coupling cause greater independence between modules and easier maintenance.
- There are mainly four kinds of module coupling. They are:

i. <u>Content coupling</u>

- Two modules are content coupled if one module makes a direct reference to the contents of another module, thus allowing the calling module to modify a program statement in the called module.

- It also allows one module to refer to an internally defined data element of another module to branch into another module.

ii. **Common-environment coupling**

- Two modules are common-environment coupled if they refer to the same data structure or data element in a common environment (e.g., shared files).
- Because of this modules that appear unrelated in a structure chart are coupled through their use of common data.

iii. **Control coupling**

- Two modules are control coupled if one module passes a control element such as flags, switches, etc. to the other module.
- This control element affects the processing in the receiving module and violates the principle of information hiding.
- It implies that that calling module must know the method of operation of the called module and any changes made to the called module may require changes in the calling module.

iv. **Data coupling**

Two modules are data coupled if they are not content coupled, common-environment coupled or data coupled but only pass data elements as parameters.

The most desirable form of coupling is data coupling.

Question 3. ***What is module strength?***

Answer:

Module strength, also known as module cohesion, describes how system functions are coed into modules. A structure chart with a high strength has modules that represent well-defined system functions, one by each module. It is desirable to have structure charts with high strength. There are generally six levels of module strength. They are:

i. **Coincidental strength**

Coincidental strength exists if there is no meaningful relationship between the parts in a module. It often occurs when existing code is modularized. Often such modules are not related to well defined system functions, but result from techniques that have been used to write them.

ii. **<u>Logical strength</u>**

Logical module occurs when all elements in al module perform similar tasks, such as modules that include all editing. Considerable duplication can exist in the logical strength level. For example in the module to perform editing on a transaction, if there are many date items in a transaction, then separate code would be written to check that each date is a valid date.

iii. **<u>Temporal strength</u>**

In temporal strength all functions related to time are grouped into one module. It has some undesirable features as far as change is concerned.

iv. **<u>Procedural strength</u>**

Procedural strength often results when a flowchart, which represents one well-defined system function, is divided into a number of sections and each section is represented by one module, thus forming a number of modules for a single function.

v. **<u>Communicational strength</u>**

Communicational strength occurs when processes that communicate with each other, (such as reading a file, processing it, and writing the output back to the file) are included in the same module. Because of communicational strength, interdependence among modules is increased.

vi. **<u>Functional strength</u>**

A module that has functional strength carries out one well-defined function. This module does not have the properties of coincidental, logical, temporal, procedural or communicational strength.

Question 4. ***How do you convert DFD to structure chart?***

Answer:

While converting from DFD to structure chart, each DFD process is converted to one structure chart module and these modules are connected in a way consistent with DFD data flows. Conversion from DFD to structure chart involves two design techniques:

i. **<u>Transform analysis</u>**

- ❖ Transform analysis searches the DFD for a process that can be converted to a transform center in a structure chart.

- ❖ It looks for a central process, together with well-defined input and output streams.
- ❖ Then a MAIN module and a module for the central process are created, and a call is made from the MAIN module tothe central processing module.
- ❖ The processes that provide input to the central process are converted to structure chart modules and they provide input to the central process module through the MAIN module.
- ❖ Finally, the processes that take the outputs from the central process are converted to output modules and they obtain outputs from the central process through the MAIN module.

ii. <u>Transaction analysis</u>

- ❖ Transaction analysis identifies those parts of DFD that can be converted to transaction-centered structure charts.
- ❖ In the DFD we look for an input stream that is split up into a number of input streams by a process and this process is converted to the MAIN module.
- ❖ The transaction processes that take one of the split inputs each are converted to structure chart modules that are called by the module MAIN.
- ❖ A decision is made on which of the transaction modules is to be called, and hence a decision symbol is placed where the MAIN module calls the transaction modules.

Question 5. ***What is module specification?***

Answer:

- ❖ Module specification refers to that specification produced during program design that can be directly converted to program code.
- ❖ While implementing the system, the system model, specified as a structure chart, has to be converted into a set of program modules.
- ❖ To do this, first of all a detailed module specification must be prepared. Only then can the program code be developed.
- ❖ While preparing the module specification, the following steps are followed:

  - Developing detailed process specifications for the processes: These process specifications are written in structured English and begin to approximate program code. They include all the conditions that can arise in the process and how they are treated.
  - Developing structure charts

- Producing detailed module design
- Packaging into load modules

Evaluation criteria for structure charts

The evaluation criteria for structure charts are:

i. **Module coupling**

ii. **Module cohesion (module strength)**

iii. **Span of control**

   Span of control comprises the number of immediate subordinates of a module. Ideally, the span of control should not exceed seven.

iv. **Fan-in**

   Fan-in is the number of modules that call a particular module. Ideally, structure charts should have a high fan-in.

v. **Scope of control**

   The scope of control comprises all the subordinates of a module. It includes the immediate subordinates of the module, their immediate subordinates, and so on.

vi. **Scope of effect**

   The scope of effect of a decision consists of all modules whose processing is conditional on the outcome of the decision. In a good design, the effects of a decision should be confined to as few modules as possible. It is better if only immediate subordinates fall within the scope of effect of decision.

8.OBJECT MODELING

Question 1. ***What do you understand by the term object-oriented paradigm as compared to object-oriented languages?***

Answer:

Object oriented paradigm implies a way of thinking. It introduces the idea of encapsulation and constructing objects independently of each other, with all functionality encapsulated within the object. It emphasizes on.

- Combining processes, dada and flows into one modeling paradigm, thus allowing objects to be modeled as independent entities that can be flexibly combined into cooperating systems.
- Easy conversion from analysis to design models, through the use of similar terms.
- Supporting multimedia information and not only record structures.

In object-oriented paradigm data and processes can be modeled a single system model. Also, because of greater autonomy and coordination between objects, the client- server concept is supported.

Object-oriented languages are software products that support objects at the implementation level. Thus, object-oriented languages are based on the object-oriented paradigm. Programming using such languages uses objects and their features such as encapsulation and offers benefits such as reuse of objects, which is a feature of the object oriented paradigm.

Question 2. ***What are the advantages of combining all modeling components into one object?***

Answer:

Modeling components include processes, data and flows. Combining all modeling components into one object means combining processes, data and flows into one entity called object. This is also known as encapsulation. The advantages of combining all modeling components into one object (encapsulation) are:

i. The entire system can be clearly understood because of independent entities (objects) with their own local goals, exchanging messages between themselves to achieve a global goal of the large system.
ii. Separate models need not be maintained for data and process; rather everything can be modeled in one system model. So, links between data and process models need not be developed and validated.

iii. There is an easy conversion from analysis to design models through the use of similar terms.
iv. A system model developed during analysis can be directly converted to implementation model using object-oriented implementation.
v. It is easier to change systems, as only one model has to be changed.
vi. Objects can be designed independently of each other with usually only their interfaces specified.
vii. Greater autonomy between the objects supports the idea of independent and asynchronous objects cooperating with each other, often in a client-server relationship. Thus, we can replace a server by another server without changing the client, or add another client that uses the same server.
viii. As objects are autonomous, they are easily plugged into other systems. Thus the idea of reuse is supported.
ix. Because of reuse quick changes to functions or interactions between people is supported.
x. It supports multimedia information and not only record structures.

Question 3. ***What are some common object features?***

Answer:

Object features refer to the characteristics of the class to which the object belong. The number of features of an object can be as many as we like. The common object features are:

i. Properties, which refer to the data included within the object.
ii. Methods, which are processes that act on the data or properties.
iii. Object states that can initiate methods.
iv. Constraints maintained between object properties.
v. Checks on pre- and post- conditions to be satisfied on messages and replies.
vi. Triggers that activate messages for the given data conditions.

Question 4. ***How do you distinguish between classes and objects?***

Answer:

Classes and objects can be distinguished on the basis of the following points:

- ✓ A class is an object that describes a set of objects with the same features while an object is a separate entity in itself and not a set of other objects.
- ✓ Class serves as a base for the creation of objects while objects are instances of a class.
- ✓ Objects are instances of class in the same way as each entity is an instance of an entity set.
- ✓ Classes have properties and methods of their own, while objects inherit their methods from their class, that is, methods are generally stored in the class object rather than being duplicated in each instance of that class.

- ✓ For example, if PROJECTS is a class then each project is an instance of that class, that is, an object.

Question 5. ***Describe modeling behavior in brief.***

Answer:

Modeling behavior starts with use cases and reduces them to object classes through event flow and event trace diagrams.

- → **Use Case**
  - o Use case is a script that describes typical ways that a system is or will be used.
  - o A **use** case is general in the sense that it does not describe instances of how a system is used, but generalizes a number of instances into a general script.
  - o A **use** case class covers a large number of typical scenarios.
  - o A system is usually described by a number of use cases, which together become the use case model.

- → **Object states**
  - o Object states are commonly used at both the subject and system levels.
  - o They describe how objects and entities change as a system evolves.
  - o The way the objects progress is referred to by their states.
  - o For examples, the statement that ‘an order has been initiated’ implies that the order is in the ‘initiated’ state, and the statement that ‘the order has been approved’ implies that it is in the approved state.

- → **State diagrams**
  - o The movement of an object from state to state is called state transition.
  - o A state transition diagram represents the object states, and the transitions between them. A circle represents each state, and lines between the states represent the transitions between the states.
  - o The state name is placed inside the circle and the transition name is placed inside the circle and the transition name is placed on the transition between the circles.
  - o Transition names are usually names of processes that cause the transition. They can also be the outcome of a process, or a combination of the process and the outcome of the process.

- State transition diagrams, also called state diagrams, are useful later in design because they indicate system transactions.
- Each transition in the diagram becomes a transaction that operates on the entity.
- State transition diagrams are also used to verify DFDs.
- We nay find a transition that does not appear as a process in the DFD. We would then return to the DFD and add the process to it.

→ **Event trace diagrams**

- Event trace diagram is a diagram showing dynamic relationships between objects.
- It is also known as scenario diagram or interaction diagram. A use case usually follows a set of steps that define actions by objects.
- Each step involves one object passing information to another object. This information exchange is shown in event trace diagrams.
- In event trace diagrams, vertical lines labeled by the object name show the main objects in the scenario.
- Horizontal lines between the vertical lines define events in the use case.
- These events usually result in the exchange of information between the objects.
- These lines closely follow the sequence portrayed by the use case.
- A complete event trace diagram also contains all the error conditions.

→ **Event flow diagram**

- The event flow diagram groups all the flows in and out of objects in different event trace diagrams into one diagram.
- This diagram can then be used to identify the methods to be included in the object.
- The event flow diagram shows objects as rectangular boxes, with all the events from all the event trace diagrams grouped by input and output.

Question 6. ***What do you understand by inheritance and what is multiple inheritance?***

Answer:

- Inheritance is the use of the features of an object by another object.
- An object can inherit features from another object.
- It can also have some additional features or, if needed, replace some of the object features.
- If an object inherits properties from more than one object, then it is called multiple inheritance.

- For example, in the figure below, DISTANT-SALES inherit the properties of SALES as well as properties of DELIVERIES.

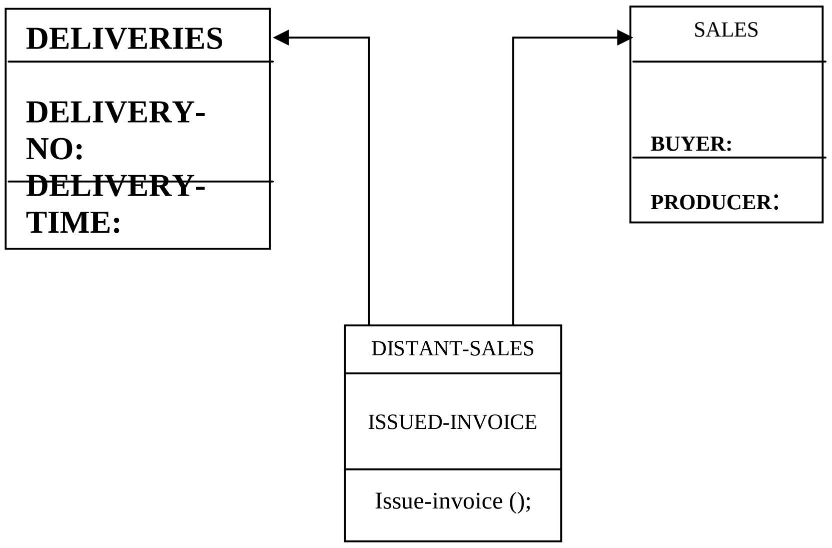


Figure: <u>Multiple Inheritance</u>

Question 7. ***What is the difference between a use case and a scenario?***

Answer:

- Use case is a script that describes typical ways that a system is or will be used.
- A use case is general in the sense that it does not describe instances of how a system is used, but generalizes a number of instances into a general script.
- Scenarios, on the other hand, are particular examples or instances of how the system is used.
- For example, 'Mary, the salesgirl, computes the total price of items and sells them' is a scenario. If such a sales process is generalized to show how a sale is made, instead of specifying a particular instance, then it is called a use case.
- A use case class covers a large number of typical scenarios.

Question 8. ***What do you mean by associations and containment?***

Answer:

- An object model is usually made up of many objects related to each other. Such relationships are shown by references between the objects.
- A reference to an object is usually made through an object property.
- The references used to show the relationships between different objects are associations.
- Thus, the figure below shows an association from projects to managers and persons assigned to the project.

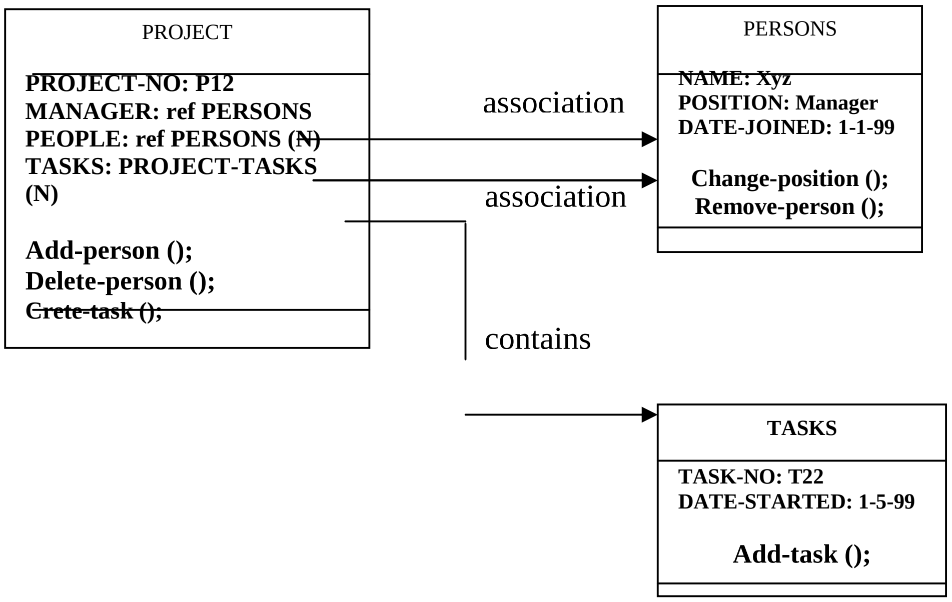


Figure: <u>Object relationship showing associations and containment</u>

If a relationship between any two objects, A and B is such that instances of B make up instance of A, then A is said to contain B. For example, in the above figure, a project is made of a no of tasks, hence PROJECT contains TASKS, that is, the relationship between PROJECTS and TASKS is containment.

Object Representation Methods

There are three representations generally used for representing objects at the analysis level. They are:

i. **Grady Booch Representation**

- Here, cloud-shaped objects, with attributes and methods listed in the object, represent object classes.
- The method is distinguished by the fact that it has a set of parentheses to represent methods.
- Lines are used to show the associations between objects. A line with a solid dot shows a 'has' association or containment.
- An arrow shows an inheritance association, and a line with an empty dot shows a using association.
- Cardinality is shown on the lines, using 2, N or M as in E-R diagrams.

ii. **Henderson-Sellers Representation**

Here, features are shown in different boxes and the arrows show the associations between the object classes. Arrows from the subclass to the owner class show inheritance. An arrow from the owner to the component shows aggregation.

iii. **Jacobson Representation**

This representation method concentrates on the objects without their methods. It shows the links between the classes and the class attribute types. Links to objects also show cardinality. Methods for the objects are derived from use cases.

9.DESIGNING THE NEW SYSTEM

Question 1. ***What are the roles of objectives in system design?***

Answer:

Objectives refer to the detail description of the statement of user requirements. Such objectives can be specified in terms of improvements to the organization's processes and functions and what is to be done to realize these improvements. Design of the new system begins by elaborating the statement of requirements in terms of more detailed objectives. Only when specific objectives are available is it possible to design to develop the correct system. Hence, it is important to state the objectives in a way that is useful to design so that the proper system can be built. Once precise objectives are set, the designers get precise goals to move towards.

Objectives can be of different types, some of which are:

- Functional objectives, which state new or amended functional requirements
- Process improvements, which include changes to
  - The way data is accessed
  - Sequence in which things are done
  - Process steps
  - Input and output methods
- Operational objectives, which specify the performance standards to be attained by the new system.
- Personal and job satisfaction needs.

All these different kinds of objectives need to be precisely defined for the design of the system. The various objectives specify what ideas are to be implemented during the system design. For example, personal and job satisfaction objectives may call for changes to the user interface to the computer.

Question 2. ***Describe the four problem-solving steps suggested by DeMarco. Do you think they are natural to the way analysts proceed in analysis?***

Answer:

DeMarco, in 1978, has proposed a method for creating a new logical model from the logical model of the existing system during problem solving. This method contains four steps:

i. **Determine processes affected by objectives**

All the processes that are affected by the objectives are identified.

ii. **<u>Create a domain of change</u>**

All the affected processes identified in the above step are included in the domain of change, which is that part of the logical DFD of the existing system that will be changed in the new system. The affected processes may be adjacent to each other or they may be made up of disconnected or disjointed DFDs. Designers may define alternative domains of change and one of these must be selected during design.

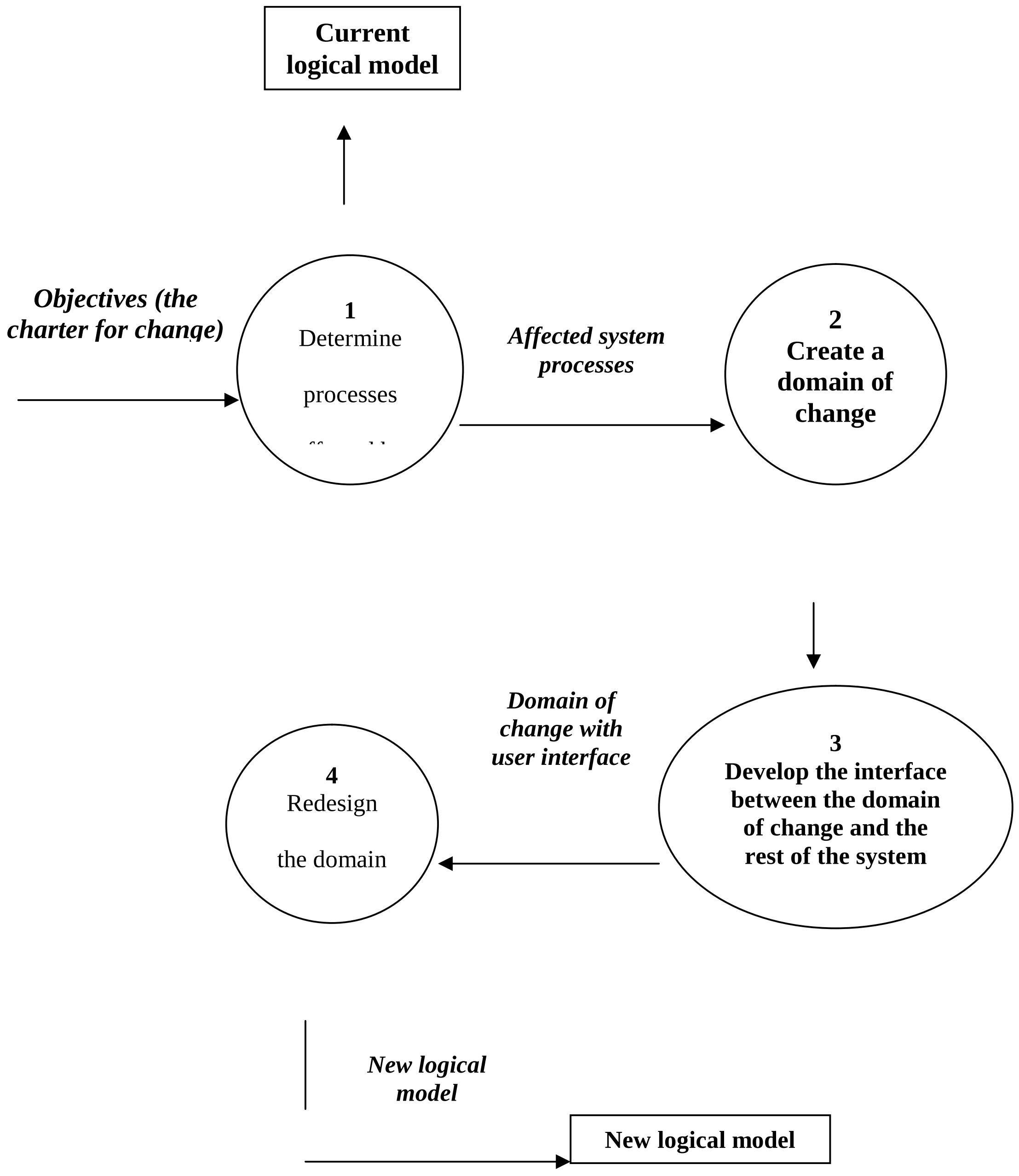
iii. **<u>Develop the interface between the domain of change and the rest of the system</u>**

The interface between the processes included in the domain of change and the rest of the system are developed.

iv. **<u>Redesign the domain of change</u>**

The domain’s processes, data flows and data stores are designed after the domain of change is selected. Redesigning the domain of change proceeds in the following steps:

- Add data stores by identifying the data needed inside the domain of change.
- Define processes, data flows and data stores by taking each input into the domain of change and defining processes that the input data flows through before it is stored or used to produce an output.
- Add processes that transform data by defining processes that use data in the data store and defining how the processes use the data.
- Add processes to create outputs by identifying the data stores used to produce the outputs and defining the processes that the data goes through before being output.


Current logical model
Objectives (the charter for change)
1 Determine processes
Affected system processes
2 Create a domain of change
Domain of change with user interface
3 Develop the interface between the domain of change and the rest of the system
4 Redesign the domain
New logical model
New logical model

Steps for developing the new logical model (by DeMarco)

Redesigning the domain of change, the fourth problem-solving step suggested by DeMarco, is not natural to way analysts proceed in analysis of a large system though it work quite fine for small systems. Because of first having to identify data stores, then processes and data flows, this process cannot be followed for large systems. Instead, a top-down approach, where the top-level processes are defined first is preferable.

Question 3. ***What is the domain of change and how would you create it?***

Answer:

Domain of change is that part of the logical **DFD** of the existing system that will be changed in the new system. It includes all the processes that are affected by the system objectives, together with the interface between these processes and the rest of the system. The affected processes may be adjacent to each other or they may be made up of disconnected or disjointed **DFDs**. Designers may define alternative domains of change and one of these must be selected during design.

In order to create the domain of change, first of all the processes that are affected by the objectives for the new system are identified. Then they are included in the domain of change.

For redesigning the domain of change, the following steps are used:

- Add data stores by identifying the data needed inside the domain of change.
- Define processes, data flows and data stores by taking each input into the domain of change and defining processes that the input data flows through before it is stored or used to produce an output.
- Add processes that transform data by defining processes that use data in the data store and defining how the processes use the data.
- Add processes to create outputs by identifying the data stores used to produce the outputs and defining the processes that the data goes through before being output.

An alternative to a complete redesign of the domain of change is to make changes to the individual components of the domain of change, that is, to redesign by parts. Such amendments can be

- Adding a new system process
- Creating a new data
- Changing the sequence of operations on information
- Eliminating redundant or unnecessary processes
- Combining two or more processes
- Adding new data and changing processes to use this data

Question 4. ***How would you convert physical model to logical one? Give the illustration.***

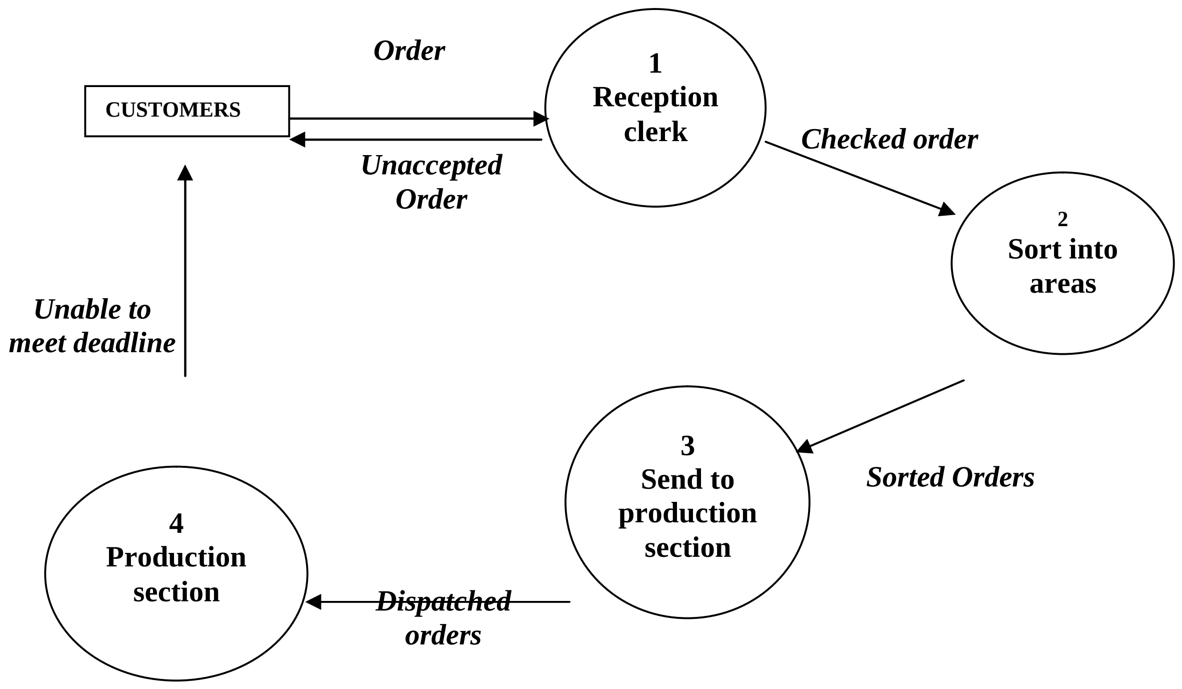
Answer:

In order to convert physical model to logical one, the following steps are followed:

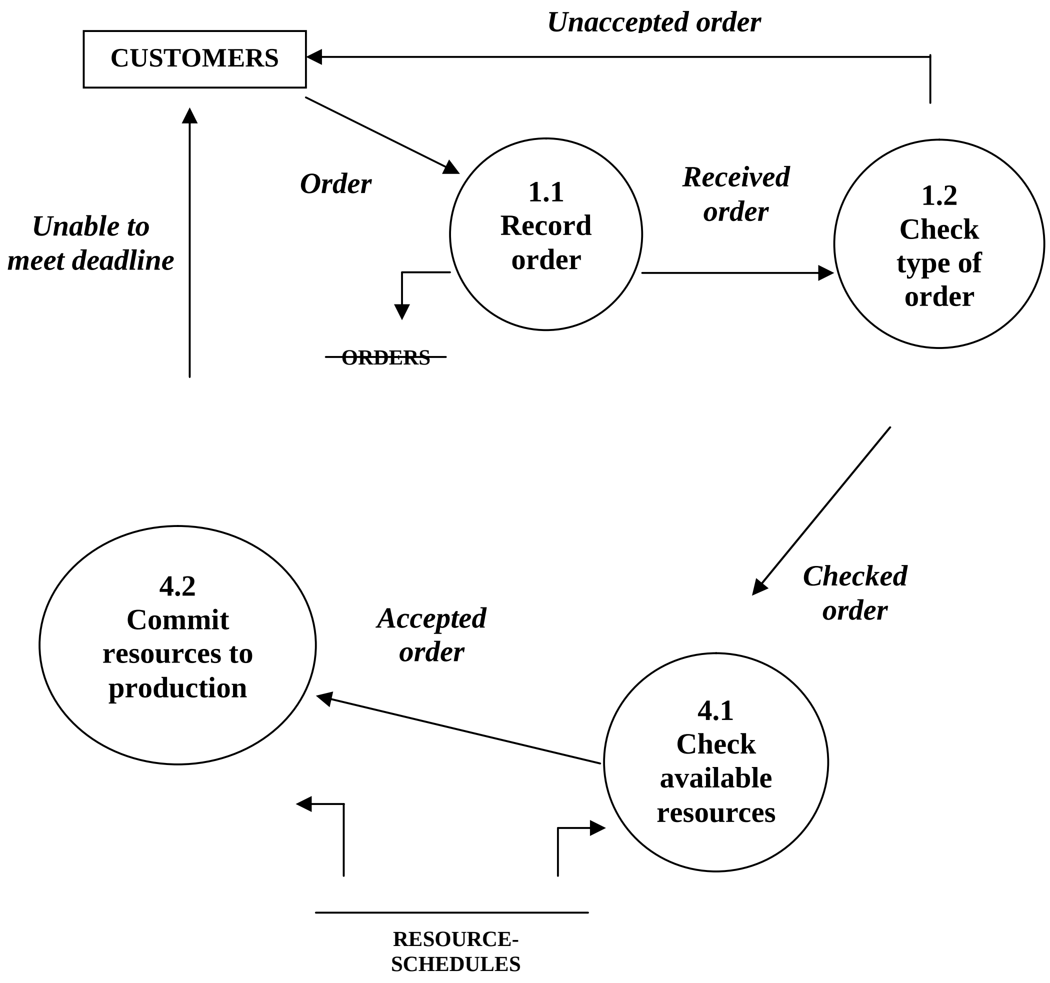
i. The processes that refer to physical activities only and do not transform information are removed.
ii. The remaining processes that are physical but transform data are expanded into their logical functions by first finding out what the physical process does and then replacing it by a leveled DFD of logical functions that represent the physical object's logical activities, or what the object does.
iii. All physical processes can be expanded this way and their expansion combined into a lower-level logical DFD.
iv. Any common or similar functions in this lower-level DFD are combined, and these combined higher-level processes become the higher-level logical DFD.

Thus conversion from physical model to logical model is done.

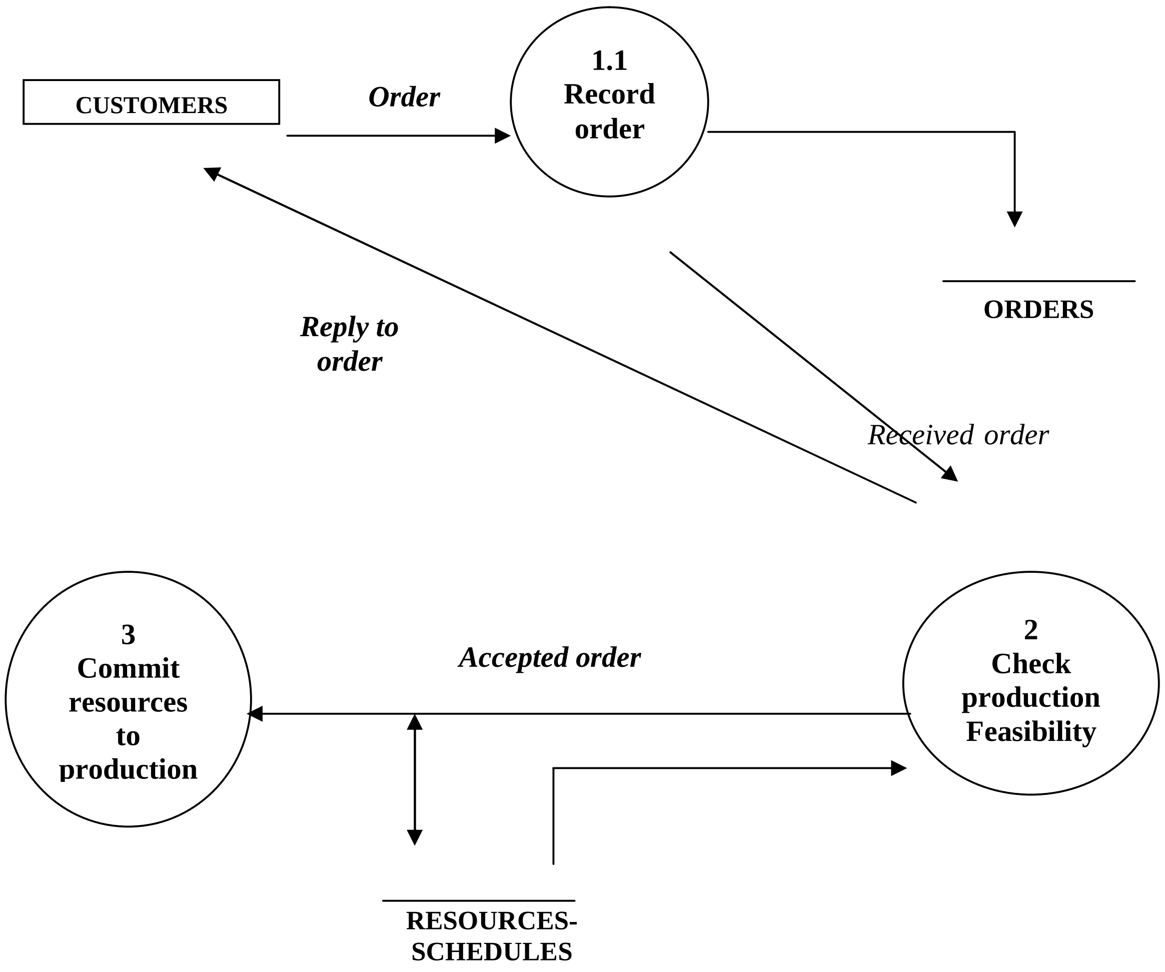
An illustration of conversion from physical model to logical model:



Physical Model



<u>Expanded Processes</u>



Recombined Logical Processes

10.IMPLEMENTAION

Question 1. ***What do you mean by planning in system implementation?***

Answer:

The purpose of implementation is to build a properly working system and to install it in the organization, replacing the old system and work methods as well as finalizing all system and user documentation, thoroughly training users and others to effectively use the new system, and preparing support systems to assist users as they encounter difficulties. Because implementation is such an important phase of system development, all the activities that are to be done during this phase are to be planned. Planning refers to discussing and making decisions on how the system is to be implemented, that is, how the users of the system are to be trained, what tests are to be carried out, how the system is to be installed, etc. Planning gives analysts and programmers an opportunity to think through all the potential problem areas, list these areas, and develop ways to test for problems. Planning for testing and implementation begins with the beginning of the system analysis.

Training plan involves deciding on who should be trained, what should the duration of training be, etc. testing plan involves deciding on what tests are to be carried out. Test plans can be:

- ♠ Master test plan, developed during analysis.
- ♠ Unit test plan, developed during design.
- ♠ Integration Test plan, developed during design.
- ♠ System Test plan, developed during design.

During implementation, these various plans are put into effect and the actual testing is performed.

Question 2. ***What is the objective of implementation? Define the stages in implementation.***

Answer:

The objectives of implementation are:

i. Converting system design specification into working and reliable software and hardware.
ii. Documenting the work that has been accomplished.
iii. Providing help for current and future users and system caretakers.

iv. Testing the software that has been developed.

The stages in implementation are:

i. Coding (Writing computer software): It means actually writing code or monitoring coding done by programmers to ensure that programs meet design specifications.
ii. Testing software: It involves using test data and scenarios to verify that each component and the whole system work under normal and abnormal circumstances.
iii. Installation (Converting from the ld to the new system): It includes installing the new system in organizational sites as well as dealing with personal and organizational resistance to the change that the new system causes.
iv. Documenting the system: It includes reviewing all project dictionary or CASE repository entries for completeness as well as finalizing all user documentation, such as user guides, reference cards and tutorials.
v. Training users and others: It may include a variety of human and computer-assisted sessions as well as tools to explain the purpose and use of the system.
vi. Designing support procedures: It ensures that users can obtain the assistance they need as questions and problems arise.

Question 3. ***How many tests are to be prepared in test preparation?***

Answer:

Seven types of tests are to be prepared in test preparation. All of these tests are either static or dynamic and manual or automated.

In static testing the code being tested is not executed while in dynamic testing the code being tested is executed. In automated testing, the computer conducts the test while in manual testing, people perform the test.

The seven types of test with their categorization are shown below.

	Manual	Automated
Static	Inspection	Syntax checking

Dynamic	Walkthrough Desk checking	Unit test Integration test System test

Table: **A categorization of test types**

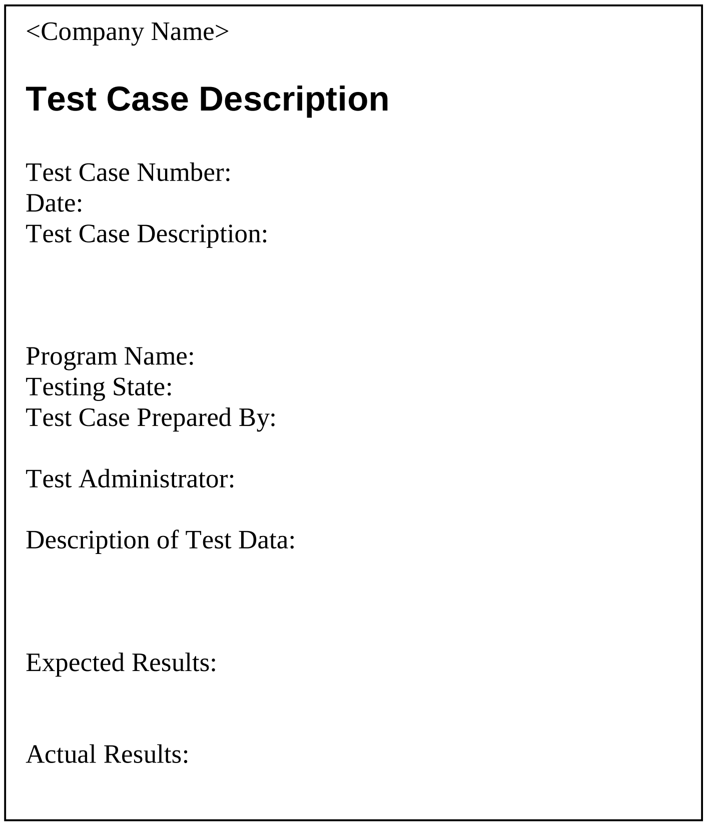
i. Inspection: A testing technique in which participants examine program code for predictable language-specific errors.
ii. Walkthrough: In walkthrough, the correctness f the models produced is checked and the errors detected are notified for amendments.
iii. Desk checking: A testing technique in which the reviewer sequentially executes the program code manually.
iv. Syntax checking: Typically a compiler does Syntax checking. Errors in syntax are uncovered but the code is not executed.
v. Unit test: Each module is tested alone in an attempt to discover any errors in its code.
vi. Integration test: It implies the process of bringing together all of the modules that a program comprises of for testing purpose. Modules are typically integrated in a top-down, incremental fashion.
vii. System test: It implies the bringing together of all the programs that a system comprises of for testing purpose. Programs are typically integrated in a top-down, incremental fashion.

Question 4. ***How do you define test preparation or testing process?***

Answer:

Test preparation or testing process involves the preparation of test cases with proper documentation. Attention must be paid to different aspects of a system, such as response time, response to boundary data, response to no input, response to heavy volumes of input, etc. anything that could go wrong needs to be tested as long as resources permit. At least the most frequently used parts of the system must be tested.

A test case, which is prepared during the testing process, is a specific scenario of transactions, queries, or navigation paths that represent a typical, critical or abnormal use of the system. A test case should be repeatable, so that it can be rerun as new versions of the software are tested. A test case description form and a test case results form are as shown below.



<Company Name>

Test Case Description

Test Case Number:
Date:
Test Case Description:

Program Name:
Testing State:
Test Case Prepared By:

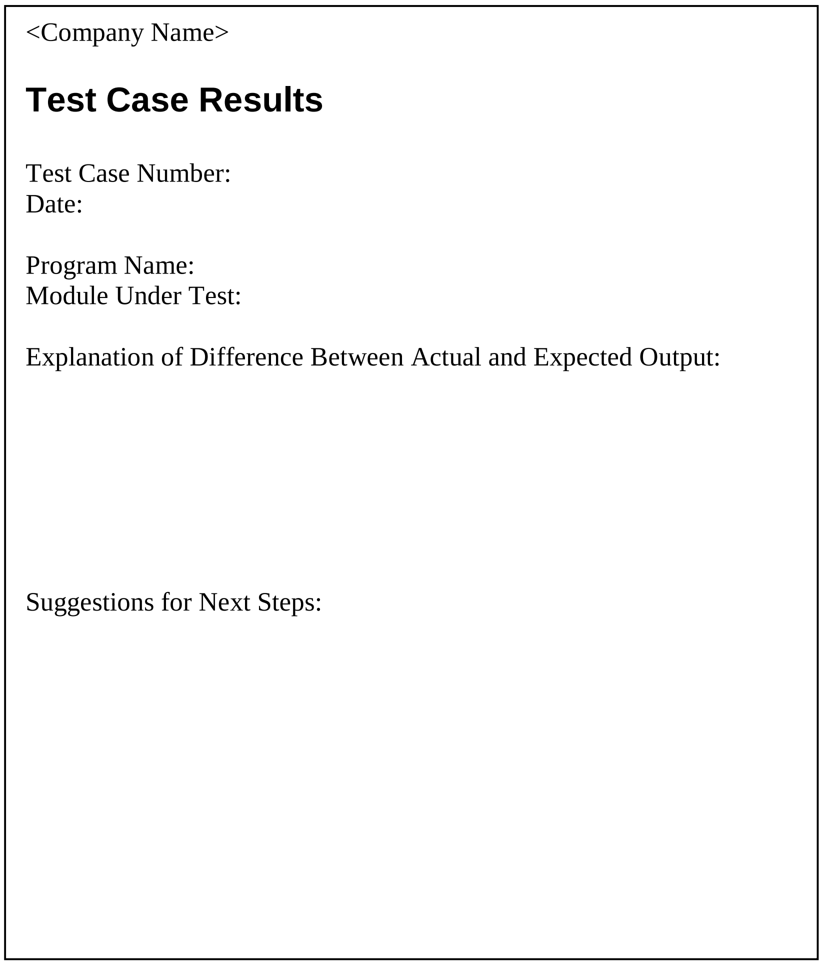
Test Administrator:

Description of Test Data:

Expected Results:

Actual Results:

Figure: **Test Case Description Form**


<Company Name>

Test Case Results

Test Case Number:
Date:

Program Name:
Module Under Test:

Explanation of Difference Between Actual and Expected Output:

Suggestions for Next Steps:

Figure: **Test Case Results Form**

Question 5. ***What do you mean by alpha and beta testing?***

Answer:

Alpha and beta testing are the tests included in acceptance testing. Acceptance testing is the process whereby actual users test a completed information system, the end result of which is the users' acceptance of it. A complete acceptance testing includes alpha testing, beta testing and a system audit.

Alpha testing: It is the user testing of a completed information system using simulated data. During alpha testing, the entire system is implemented in a test environment to discover whether or not the system is overtly destructive to itself or to the rest of the environment. The types of test performed during alpha testing include:

- ♠ Recovery testing, which forces the software or environment to fail in order to verify that recovery is properly performed.
- ♠ Security testing, which verifies that protection mechanisms built into the system will protect it from improper penetration.
- ♠ Stress testing, which tries to break the system to see what happens in conditions such as when a record is written to the database with incomplete information.
- ♠ Performance testing, which determines how the system performs on the range of possible environments in which it may be used, such as in different hardware configurations.

Beta testing: It is the user testing of a completed information system using real data in the real user environment. In beta testing a subset of the intended users run the system in their own environments using their own data. The intent of the beta test is to determine whether the software, documentation, technical support, and training activities wok as intended.

INSTALLATION

The process of moving from the current information system to the new one is called installation. Four different approaches to installation have emerged. The approach an organization decides to use will depend on the scope and complexity of the change associated with the new system and the organization's risk aversion. The four approaches to installation are:

i. Direct installation

   In direct installation, the old system is turned off and the new system is turned on. Any errors resulting from the new system will have a direct impact on the users and on how they do their jobs, as well as to the organization's performance.

ii. Parallel installation

   Under parallel installation, the old system continues to run alongside the new system until users and management are satisfied that the new system is effectively performing its duties and the old system can then be turned off. Since the old and new systems run parallel to each other, all the works need to be done twice, and hence this installation approach is expensive.

iii. Single location installation

It is also known as location and pilot installation. It involves changing from the current to the new system in only one place or in a series of separate sites over time. The main advantage of single location installation is that it limits potential damage and potential cost by limiting the effects to a single site. Once management has decided that installation in one site has become successful, the new system may be deployed in the rest of the organization, possibly continuing with single location installation. In this type of installation, if different locations require sharing of data, extra programs will need to be written to synchronize the current and new systems.

iv. Phased installation

Phased installation, also called staged installation, is an incremental approach. Under phased installation, the new system is brought on-line in functional components; different parts of the old and new systems are used in cooperation until the whole new system is installed. For a phased installation, the new and old systems must be able to coexist and probably share data. Thus, bridge programs connecting new and old databases and programs often must be built.

11.QUALITY ASSURANCE

Question 1. ***Why is it necessary for quality assurance checks to be separated from system development?***

Answer:

- **Quality assurance** means carrying out timely checks to ensure that the system being developed meets the quality standards specified. It is necessary to carry out quality assurance checks to make sure that there are no errors in the system and that the system developed meets the original user requirements.

- **Development process** must include checks throughout the process to ensure that the correct system is being developed. Because quality assurance is an important activity for the development of proper systems, it becomes necessary to carry the quality assurance checks with much emphasis. Because of this, quality assurance program has its own methods and processes. Thus it becomes necessary for quality assurance checks to be separated from system development so that separate methods can be used for them as required.

Question 2. ***Why is there a difference between the checks carried out to detect errors and those to evaluate some of the more qualitative system features?***

Answer:

- There is a difference between the checks carried out to detect errors and those to evaluate some of the more qualitative system features because detection of errors and evaluation of qualitative system features are both important activities of quality assurance, yet they differ in their nature and tools used.
- Evaluation of qualitative system features is done to see that the system being developed meets the user's needs.
- Detection of errors is used to see that no errors, such as the acceptance of alphabets for numbers, occur in the system.

- Because these two types of checks perform two different tasks of quality assurance, they are different.

Question 3. ***What is inspection?***

Answer:

- Inspection is an examination of a product to assure quality.
- It involves the producer, whose product is being inspected, the inspector, who evaluates the product, the moderator who controls the inspection process, and a reader who may guide inspectors through the product.

According to Fagan, inspections can be carried out in five steps:

i. Overview, where the producers of the work explain their work to inspectors.
ii. Preparation, where the inspectors prepare the work and the associated documentation for inspection.
iii. Inspection, which is a meeting moderated by the moderator and guided by a reader who goes through the work with the inspectors.
iv. Rework, which is any work required by the producers to correct any deficiencies.
v. Follow-up, where a check is made to ensure that any deficiencies have been corrected.

- Inspections are formal and have a report that must be acted on. It is also important that any recommendations made during inspections be acted upon and followed to ensure that any deficiencies are corrected.

Question 4. ***What are the roles under walkthrough?***

Answer:

The size of the walkthrough team depends upon the system for which the walkthrough is being done and upon the skills and review experiences of the potential participants. However, the common roles under walkthrough are:

- **<u>The walkthrough leader</u>**

- The walkthrough leader should ensure a good walkthrough and report the reasons why a good walkthrough was not achieved.
- The walkthrough leader selects the walkthrough team members and sets the meeting time, place and length of the walkthrough.
- During the walkthrough, the leader must make sure that the meeting keeps to the relevant topics and that there is an agreement on the outcome of the walkthrough.
- After the walkthrough, the leader sees that accurate reports are produced promptly and checks that the producer has a reasonable basis for clearing up any issues requiring attention.

- **<u>The walkthrough secretary</u>**
  - The function of the walkthrough secretary is to record the results of the walkthrough.
  - The secretary should identify all the walkthrough team members as well as collect all the materials necessary for keeping accurate records of the walkthrough.
  - During the walkthrough, the secretary must record all issues accurately and state each outcome explicitly, unambiguously and neutrally.
  - After the walkthrough, the secretary prepares all reports promptly, gets them signed by all participants and distributes them to all the relevant people.

- **<u>The walkthrough reader (producer)</u>**
  - The producer's job is to describe the product under review.
  - For structured systems analysis, this is usually a DFD together with any process descriptions, data flow and data models.
  - The producer should go through the documentation and bring out any points that caused difficulty or uncertainty during the development of the documentation.

- **<u>The walkthrough participants</u>**

The members of the walkthrough team except the leader, secretary and reader are the participants. They should be prepared for the walkthrough and take a neutral and constructive stand on all issues raised in the walkthrough.

Question 5. ***How is walkthrough carried out?***

Answer:

- For carrying walkthrough, first of all the system model for which the walkthrough is to be carried out should be prepared.
- Then walkthrough is carried out. During walkthrough, the walkthrough team first reads the walkthrough model, and then notes the omissions, ambiguities and and inaccuracies.
- Two outcomes are possible from the walkthrough.
    - One is that no errors are found in the model and it is accepted. Then review documents are prepared for a subsequent review.
    - The other outcome is where errors are detected in the model. In that case an action list is prepared and the model is amended and later submitted for another walkthrough.

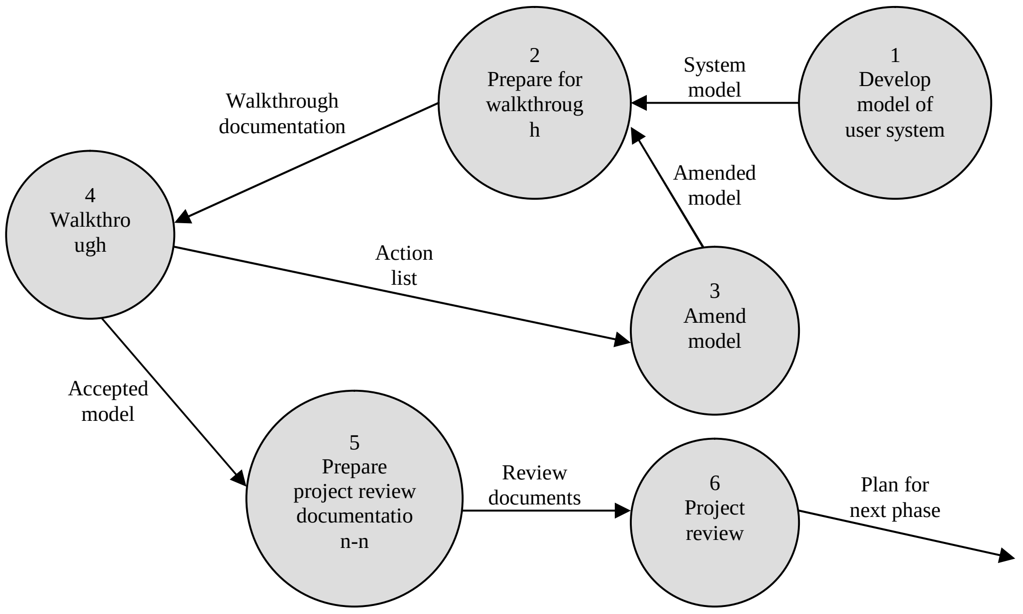


Figure: **<u>Walkthrough and review</u>**

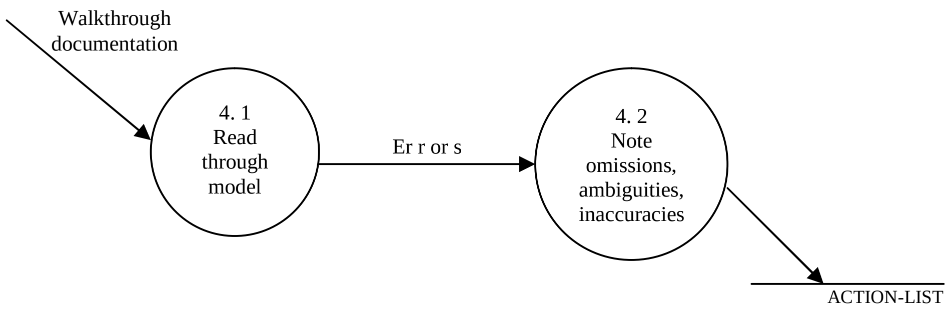

Walkthrough documentation
4. 1 Read through model
Er r or s
4. 2 Note omissions, ambiguities, inaccuracies
ACTION-LIST

Figure: **Walkthrough**

Question 6. ***What is walk through and when is it carried out?***

Answer:

- ***Walkthrough*** is a procedure that is commonly used in quality assurance to check the correctness of models produced by structured systems analysis.
- Walkthrough allocates specific tasks to various members of the walkthrough team and requires documentation to be produced during and after the walkthrough.

In walkthrough, it is checked that the model

- Meets system objectives
- Is a correct representation of the system
- Has no omissions or ambiguities
- Will do the job it is supposed to do
- Is easy to understand

- No actual design or system alteration takes place during the walkthrough; problems are only noted for further action.
- The responsibility of following up these problems is assigned to the walkthrough team members.
- The problems are documented in an action list, which also specifies which team members are to be responsible for following up these problems.
- Walkthrough can take place throughout system development. In structure systems analysis, they begin when the physical and logical models of the existing system have been completed. Some of the phases in which walkthrough can be carried out are:
    - When the existing system models have been prepared (the existing system models, physical and logical, are checked)
    - When the new system models have been developed (the new system models, logical and physical, are checked)

- There mat be more than one walkthrough in each project phase, and there are no set times for doing them.
- Walkthrough should be carried out whenever all works on a model are completed and it becomes necessary to verify that the model correctly represents the system.