Uttar Pradesh Rajarshi Tandon
open University

MCA-E4/
PGDCA-E4

Master in Computer Application

BLOCK

1

INTRODUCTION

Course Design Committee

Course Preparation Committee

Dr. Saurav Pal — **Author**
Associate Professor
Department of MCA, VBS Purvanchal University
Jaunpur-222001, Uttar Pradesh

Prof. R. R. Tiwari — **Editor**
University of Allahabad
Allahabad, Uttar Pradesh

Dr. Ashutosh Gupta
Director (In-charge), School of Computer and Information Science,
UPRTOU, Allahabad

Ms. Marisha (Coordinator)
Assistant Professor, School of Sciences,
UPRTOU, Allahabad

Printed and Published by Prof. (Dr.) Girija Shankar Shukla Registrar, **Uttar Pradesh Rajarshi Tandon open University, 2017.**
Printed By : Chandrakal Universal Pvt. Ltd. 42/7 Jawahar Lal Neharu Road Allahabad

UNIT - I

SYSTEM CONCEPT – AN INTRODUCTION

Unit Structure

1.1. INTRODUCTION

System Analysis and Design refers to the process of examining a situation with the intent of improving it through better procedures and methods. System analysis and design relates to shaping organizations, improving performance and achieving objectives for profitability and growth. The emphasis is on systems in action, the relationships among subsystems and their contribution to meeting a common goal. Looking at a system and determining how adequately it functions, the changes to be made and the quality of the output are parts of system analysis.

Organizations are complex systems that consist of interrelated and interlocking subsystems. Changes in one part of the system have both anticipated and unanticipated consequences in other parts of the system. The systems approval is a way of thinking about the analysis and design of computer based applications. It provides a framework for visualizing the organizational and environmental factors that operate on a system. When a computer is introduced into an organization, various functions' and dysfunction's operate on the user as well as on the organization. Among the positive consequences are improved performance and a feeling of achievement with quality information. Among the

unanticipated consequences might be a possible threat to employee's job, a decreased morale of personnel due to back of involvement and a feeling of intimidation by users due to computer illiteracy. The analyst's role is to remove such fears and make the system a success.

1.2 OBJECTIVES

After studying this Unit, you should be able to:

- Define system, systems study, systems analysis and systems approach.
- The primary characteristics of a system and the need for the system approach.
- How the various elements of a system work together.
- Various types of systems and behavior of the systems.
- The factoring of system into subsystem.

1.3 SYSTEM CONCEPTS AND THE NEED FOR SYSTEM APPROACH

The term "System" is derived from the Greek word systema. It means an organized relationship among functional units or components. We can define a System as a combination of resources or functional units working together to accomplish a given task.

The term "working together" in system definition is very important as all the components are interrelated and interdependent and cannot exist independently. As the definition says, these components interact with each other to accomplish a given task, which is actually the objective of the system.

The components that comprise a system may be the various inputs required by the system, the outcomes or the outputs of the system, the resources required to make the system functional etc.

System concept includes the following characteristics that are present in all systems.

a) Environment and boundaries

Systems considered as a complex system of interconnected parts. System defining its boundary; this means choosing which entities are inside the system and which are outside – part of the environment. We then make simplified representations (models) of the system in order to understand it and to predict or impact its future behavior. These models may define the structure and/or the behavior of the system.

b) Natural and human-made systems

There are natural and human-made (designed) systems. Natural systems may not have an apparent objective but their outputs can be interpreted as purposes. Human-made systems are made with purposes that are achieved by the delivery of outputs. Their parts must be related; they must be "designed to work as a coherent entity" – else they would be two or more distinct systems.

c) Theoretical framework

An open system exchanges matter and energy with its surroundings. Most systems are open systems; like a car, coffeemaker, or computer. A closed system exchanges energy, but not matter, with its environment; like Earth. An isolated system exchanges neither matter nor energy with its environment. A theoretical example of such system is the Universe.

d) Process and transformation process

A system can also be viewed as a bounded transformation process, that is, a process or collection of processes that transforms inputs into outputs. Inputs are consumed; outputs are produced. The concept of input and output here is very broad. E.g., an output of a passenger ship is the movement of people from departure to destination.

e) Subsystem

A *subsystem* is a set of elements, which is a system itself, and a component of a larger system.

f) System model

A system comprises multiple views. For the man-made systems it may be such views as planning, requirement (analysis), design, implementation, deployment, structure, behavior, input data, and output data views. A system model is required to describe and represent all these multiple views.

g) System architecture

A system architecture, using one single integrated model for the description of multiple views such as planning, requirement (analysis), design, implementation, deployment, structure, behavior, input data, and output data views, is a kind of system model.

1.4 DEFINITION OF SYSTEM AND SYSTEM ANALYSIS

A collection of components that work together to realize some objectives forms a system. Basically there are three major components in every system, namely input, processing and output.

Fig 1.1: Basic System Components

In a system the different components are connected with each other and they are interdependent. For example, human body represents a complete natural system. The objective of the system demands that some output is produced as a result of processing the suitable inputs. A well-designed system also includes an additional element referred to as 'control' that provides a feedback to achieve desired objectives of the system.

Systems analysis is a process of collecting factual data, understand the processes involved, identifying problems and recommending feasible suggestions for improving the system functioning. This involves studying the business processes, gathering operational data, understand the information flow, finding out bottlenecks and evolving solutions for overcoming the weaknesses of the system so as to achieve the organizational goals. System Analysis also includes subdividing of complex process involving the entire system, identification of data store and manual processes.

The major objectives of systems analysis are to find answers for each business process: What is being done How is it being done, who is doing it, When is he doing it, Why is it being done and How can it be improved? It is more of a thinking process and involves the creative skills of the System Analyst. It attempts to give birth to a new efficient system that satisfies the current needs of the user and has scope for future growth within the organizational constraints. The result of this process is a

logical system design. Systems analysis is an iterative process that continues until a preferred and acceptable solution emerges.

1.5 FACTORING INTO SUBSYSTEMS:

A complex system is difficult to comprehenced when considered as a whole. Therefore the system is decomposed or factored into subsystems. A subsystem is a part of a larger system. Each system is composed of subsystems, which in turn are made up of subsystems, each sub-system being delineated by its baunderies.

The interconnections and interactions between the subsystems are termed as interfaces. Interfaces occurs at the boundary and take the form of inputs and outputs.

The bounderies and interfaces are defined, so thar the sum of the subsystems constitutes the entire system. This process of decomposition is continued with subsystems divided into smaller subsystems until the smallest subsystem are of manageable size. The subsystems resulting from this process generally form hierarchical structures.

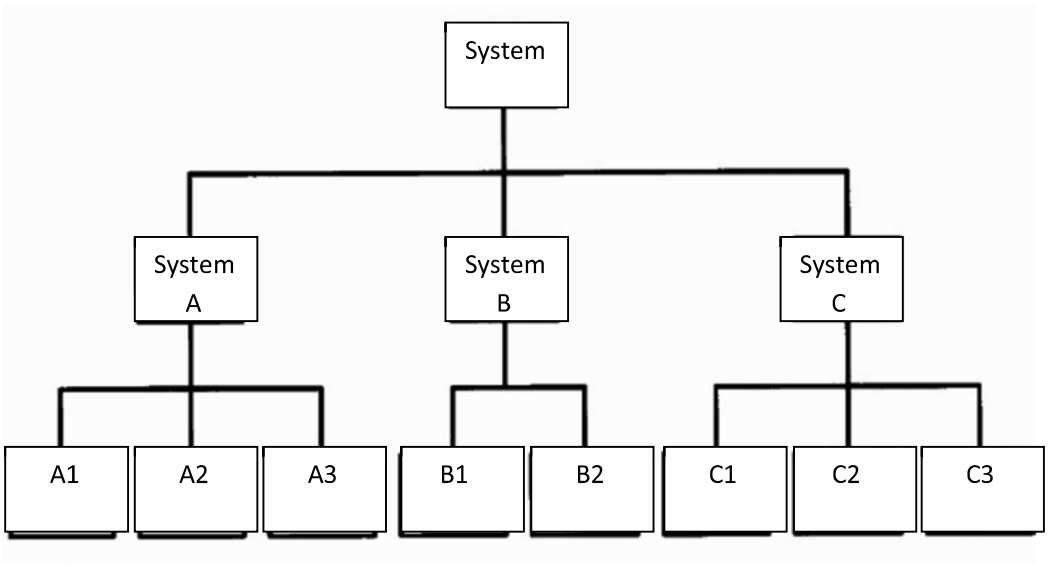


Fig 1.2: System and subsystems

Decomposistion into subsystems is used to analyze an existing system and to design and implement a new system. In both cases, the investigator or designer how to factor, i.e. where to draw the boundersies. The decisions will depend on the objectives of the decomposition and also on individual differences among designers, the latest should be minimized.

Black Box System:

The transformation process in certain sub-system, especially at the lowest level may not be defined. However, the inputs and outputs are known. Such a sub-system is called a black box system.

Fig 1.3: Black Box System

1.6. CHARACTERISTICS OF A SYSTEM

There are five types of characteristics for a system. They are

- Organization
- Interaction
- Interdependence
- Integration
- A central objective

a) Organization:

Organization implies structure and order. It can also be defined as the arrangement of components that helps to achieve objectives.

For eg: - in the design of a business system, the hierarchical relationships starting with the president on top and leading towards the workers represents the organization structure. So this gives the authority structure and specifies the formal flow of communication.

Likewise a computer system is designed around an input device, a central processing unit, an output device and one or more storage units.

b) Interaction:

Interaction refers to the manner in which each component functions with other components of the system.ie, there should be an interrelationship between each components of a system.

For eg: - in an organization there should be interaction between purchase department and production department, same way advertising with sales, payroll with personnel.

In computer system, the central processing unit must interact with the input device to solve a problem. In turn the main memory holds programs and data that the arithmetic unit uses for computation.

c) Interdependence:

This is one of the important characteristics of a system. Interdependence means the parts or the components of an organization or computer system depend on one another. Each component or parts should depend on other components of an organization. One component or subsystem depends on the input of another subsystem for proper functioning, ie, the output of one subsystem is required input for another subsystem. For example: - A decision to computerize an application is initiated by the user, analyzed and designed by the analyst, programmed and tested by the computer operator. In the below figure: none of these persons can perform properly without the required input from others in the computer center subsystem.

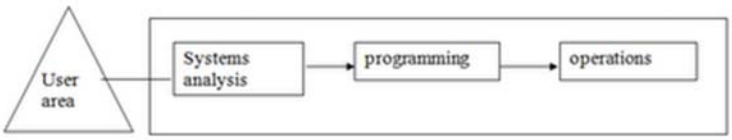


Fig 1.4: Task independence in a computer based subsystem

d) Integration:

Integration refers to the holism of systems. Synthesis follows analysis to achieve the central objective of the organization. It is concerned with how a system is tied together. It is more than sharing a physical part or location. It means that parts of the system work together within the system even though each part performs a unique function.

e) Central Objective:

The last characteristic of a system is its central objective. Objectives may be real or stated. The important point is that users must know the central objective of a computer application early in the analysis for a successful design and conversion.

1.7 INTRODUCTION TO BASIC ELEMENTS OF SYSTEM

Following are considered as the elements of a system in terms of Information systems: –

- Outputs and Inputs
- Processor
- Control
- Environment
- Feedback
- Boundaries and interface

a) Outputs and Inputs

A major objective of a system is to produce an output that has value to its user. Whatever the nature of the output (goods, services, or information), it must be in line with the expectations of the intended user. Inputs are the elements (material, human resources, and information) that enter the system for processing. Output is the outcome of processing. A system feeds on input to produce output in much the same way that a business brings in human, financial, and material resources to produce goods and services. It is important to point out here that determining the output is a first step in specifying the nature, amount, and regularity of the input needed to operate a system. For example, in systems analysis, the first concern is to determine the user's requirements of a proposed computer system – that is, specification of the output that the computer is expected to provide for meeting user requirements.

b) Processor

The processor is the element of a system that involves the actual transformation of input into output. It is the operational component of a system. Processors may modify the input totally or partially, depending on the specifications of the output. This means that as the output specifications change so does the processing. In some cases, input is also modified to enable the processor to handle the transformation.

c) Control

The control element guides the system. It is the decision – making subsystem that controls the pattern of activities governing input, processing, and output. In an organizational context, management as a decision – making body controls the inflow, handling and outflow of activities that affect the welfare of the business. In a computer system, the operating system and accompanying software influence the behavior of the system. Output specifications determine what and how much input is needed to keep the system in balance. In systems analysis, knowing the attitudes of the individual who controls the area for which a computer is being considered can make a difference between the success and failure of the installation. Management support is required for securing control and supporting the objective of the proposed change.

d) Environment

The environment is the "suprasystem" within which an organization operates. It is the source of external elements that impinge on the system. In fact, it often determines how a system must function. For example, the organization's environment, consisting of vendors, competitors, and others, may provide constraints and, consequently, influence the actual performance of the business.

e) Feedback

Control in a dynamic system is achieved by feedback. Feedback measures output against a standard in some form of cybernetic procedure that includes communication and control. Output information is fed back to the input and / or to management (Controller) for deliberation. After the output is compared against performance standards, changes can result in the input or processing and consequently, the output. Feedback may be positive or negative, routing or informational. Positive feedback reinforces the performance of the system. It is routine in nature. Negative feedback generally provides the controller with information for action. In systems analysis, feedback is important in different ways. During analysis, the user may be told that the problems in a given application verify the initial concerns and justify the need for change. Another form of feedback comes after the system is implemented. The user informs the analyst about the performance of the new installation. This feedback often results in enhancements to meet the user's requirements.

f) Boundaries and interface

A system should be defined by its boundaries – the limits that identify its components, processes and interrelationship when it interfaces with another system. For example, a teller system in a commercial bank is restricted to the deposits, withdrawals and related activities of customers checking and savings accounts. It may exclude mortgage foreclosures, trust activities, and the like.

Each system has boundaries that determine its sphere of influence and control. For example, in an integrated banking – wide computer system design, a customer who has a mortgage and a checking account with the same bank may write a check through the "teller system" to pay the premium that is later processed by the "mortgage loan system." Recently, system design has been successful in allowing the automatic transfer of funds form a bank account to pay bills and other obligations to creditors, regardless of distance or location. This means that in systems analysis, knowledge of the boundaries of a given system is crucial in determining the nature of its interface with other systems for successful design.

Check Your Progress 1

1. Define System.

   _____

   _____

   _____

2. What are the characteristics of a system?

   _____

   _____

   _____

3. What are the six basic elements of a system?

   _____

   _____

   _____

1.8 DIFFERENT TYPES AND BEHAVIOR OF THE SYSTEM

Systems are classified in different ways:

- Physical or abstract systems.
- Open or closed systems.
- 'Man-made' information systems.
- Formal information systems.
- Informal information systems.
- Computer-based information systems.
- Information system.

1.8.1 Physical or Abstract System

Physical systems are tangible entities that we can feel and touch. These may be static or dynamic in nature. For example, take a computer center. Desks and chairs are the static parts, which assist in the working of the center. Static parts don't change. The dynamic systems are constantly changing. Computer systems are dynamic system. Programs, data, and applications can change according to the user's needs.

Abstract systems are conceptual. These are not physical entities. They may be formulas, representation or model of a real system.

1.8.2 Open vs Closed System

Systems interact with their environment to achieve their targets. Things that are not part of the system are environmental elements for the system. Depending upon the interaction with the environment, systems can be divided into two categories, open and closed.

Open systems: Systems that interact with their environment. Practically most of the systems are open systems. An open system has many interfaces with its environment. It can also adapt to changing environmental conditions. It can receive inputs from, and delivers output to the outside of system. An information system is an example of this category.

Closed systems: Systems that don't interact with their environment. Closed systems exist in concept only.

1.8.3 Open Loop System vs Close Loop System

An **Open-loop system**, also referred to as non-feedback system, is a type of continuous system in which the output has no influence or effect on the action of the input signal. In other words, in an open-loop system the output is neither measured nor "fed back" for comparison with the input. Therefore, an open-loop system is expected to faithfully follow its input command or set point regardless of the final result.

Also, an open-loop system has no knowledge of the output condition so cannot self-correct any errors it could make when the preset value drifts, even if this results in large deviations from the preset value.

Another disadvantage of open-loop systems is that they are poorly equipped to handle disturbances or changes in the conditions which may reduce its ability to complete the desired task. For example, the dryer door opens and heat is lost. The timing controller continues regardless for the full 30 minutes but the clothes are not heated or dried at the end of the drying process. This is because there is no information fed back to maintain a constant temperature.

One way in which we can accurately Control the Process is by monitoring its output and "feeding" some of it back to compare the actual output with the desired output so as to reduce the error and if disturbed, bring the output of the system back to the original or desired response. The measure of the output is called the "feedback signal" and the type of system which uses feedback signals to control itself is called a Close-loop System.

A **Closed-loop System**, also known as a feedback system is a system which uses the concept of an open loop system as its forward path but has one or more feedback loops (hence its name) or paths between its output and its input. The reference to "feedback", simply means that some portion of the output is returned "back" to the input to form part of the systems excitation.

Closed-loop systems are designed to automatically achieve and maintain the desired output condition by comparing it with the actual condition. It does this by generating an error signal which is the difference between the output and the reference input. In other words, a "closed-loop system" is a fully automatic system in which its control action being dependent on the output in some way.

So for example, consider our electric clothes dryer from the previous open-loop tutorial. Suppose we used a sensor or transducer (input device) to monitor the temperature or dryness of the clothes and fed the signal back to the controller as shown below.

1.8.4 Static vs Dynamic Systems

A dynamic system is a system that is constantly changing, like the human body system. A static system is a system in where there is no change, like the solar system.

Dynamic systems tend to become static or attain a state of equilibrium. Example if a car is assumed to be a dynamic system, then it requires fuel to continue moving forward or else it would come to a stop and become static. Simply static systems are memory less systems example a circuit of battery and resister. Static systems' output only depends on the present value of input. Whereas dynamic systems have memory capability for example RLC circuit in which capacitor may have some initial value or flip-flops. Dynamic Systems' output depends upon on future and past values.

1.8.5 Dedicated vs General Purpose Systems

General purpose system can perform different types of tasks. A personal computer is a good example of general purpose computers.

Dedicated (special purpose) system is built to handle a specific task, it can perform that task only and no other. Thus we can find a variety of dedicated systems performing a wide variety of tasks For example an payroll management system, ATM.

1.8.6 On-Line vs Off-Line System

When a system is in the online mode of operation, it is performing operational functions. It is interfacing with other components, peripherals, display systems, and communication systems to perform many tasks. The type of task the system will use online will depend on the platform of the system (tactical, tactical support, and non-tactical). A system may perform the following types of operations in the online mode:

- Operational
- Maintenance

In the offline mode of operation, a system is limited to performing maintenance. The system can be either powered or unpowered depending on the maintenance you are performing. The system is limited to interfacing with only a single system, such as a display system or a peripheral system, to perform controlled tests or a diagnostic to test itself. In this mode some systems have the capability to not only operate in the run mode but other detailed steps such as instruction mode and sequence mode. These modes are quite useful for troubleshooting malfunctions that can’t be isolated using diagnostics or self-tests.

1.8.7 Real Time vs Non Real Time Systems

Real-time systems are classified as hard or soft real-time systems. Hard real-time systems have very strict time constraints, in which missing the specified deadline is unacceptable. The system must be designed to guarantee all time constraints. Every resource management system such as the scheduler, input–output (I/O) manager, and communications, must work in the correct order to meet the specified time constraints.

Military applications and space missions are typical instances of hard real-time systems. Some applications with real-time requirements include telecom switching, car navigation, the medical instruments with the critical time constraints, rocket and satellite control, aircraft control and navigation, industrial automation and control, and robotics.

Soft real-time systems also have time constraints; however, missing some deadline may not lead to catastrophic failure of the system. Thus, soft real-time systems are similar to hard real-time systems in their infrastructure requirements, but it is not necessary that every time constraint be met. In other words, some time constraints are not strict, but they are nonetheless important. A soft real time system is not equivalent to non-real-time system, because the goal of the system is still to meet as many deadlines as possible.

A system in which there is no deadline defined is called non-real time system. Non real time system cannot guarantee response time.

1.8.8 Man made Information System

The main purpose of information systems is to manage data for a particular organization. Maintaining files, producing information and reports are few functions. An information system produces customized information depending upon the needs of the organization. These are usually formal, informal, and computer based.

1.8.9 Formal Information Systems

It deals with the flow of information from top management to lower management. Information flows in the form of memos, instructions, etc. But feedback can be given from lower authorities to top management.

1.8.10 Informal Information systems

Informal systems are employee based. These are made to solve the day to day work related problems. Computer-Based Information Systems: This class of systems depends on the use of computer for managing business applications.

1.8.11 Computer Based Information System

A system of one or more computers and associated software with common storage called system. A computer is a programmable machine that receives input, stores and manipulates data, and provides output in a useful format.

The computer elements described thus far are known as "hardware." A computer system has three parts: the hardware, the software, and the people who make it work.

1.8.12 Information System:

An information system (IS) is any combination of information technology and people's activities using that technology to support operations, management, and decision-making. Information system deals with data of the organizations. The purposes of Information system are to process input, maintain data, produce reports, handle queries, handle on line transactions, generate reports, and other output. These maintain huge databases, handle hundreds of queries etc. The transformation of data into information is primary function of information system.

Information systems differ in their business needs. Also depending upon different levels in organization information systems differ. Three major information systems are

- Transaction processing systems
- Management information systems
- Decision support systems

Fig 1.5 - Relation of information systems to levels of organization

Figure shows relation of information system to the levels of organization. The information needs are different at different organizational levels. Accordingly the information can be categorized as: strategic information, managerial information and operational information.

Strategic information is the information needed by top most management for decision making. For example the trends in revenues earned by the organization are required by the top management for setting the policies of the organization. This information is not required by the lower levels in the organization. The information systems that provide these kinds of information are known as Decision Support Systems.

The second category of information required by the middle management is known as managerial information. The information required at this level is used for making short term decisions and plans for the organization. Information like sales analysis for the past quarter or yearly production details etc. fall under this category. Management information system (MIS) caters to such information needs of the organization. Due to its capabilities to fulfill the managerial information needs of the organization, Management Information Systems have become a necessity for all big organizations. And due to its vastness, most of the big organizations have separate MIS departments to look into the related issues and proper functioning of the system.

The third category of information is relating to the daily or short term information needs of the organization such as attendance records of the employees. This kind of information is required at the operational level for carrying out the day-to-day operational activities. Due to its capabilities to provide information for processing transaction of the organization, the information system is known as Transaction Processing System or Data Processing System. Some examples of information provided by such systems are processing of orders, posting of entries in bank, evaluating overdue purchaser orders etc.

a) Transaction Processing Systems

TPS processes business transaction of the organization. Transaction can be any activity of the organization. Transactions differ from organization to organization. For example, take a railway reservation system. Booking, cancelling, etc are all transactions.

Any query made to it is a transaction. However, there are some transactions, which are common to almost all organizations. Like employee new employee, maintaining their leave status, maintaining employee's accounts, etc. This provides high speed and accurate processing of record keeping of basic operational processes. These include calculation, storage and retrieval.

Transaction processing systems provide speed and accuracy, and can be programmed to follow routines functions of the organization.

b) Management Information Systems

These systems assist lower management in problem solving and making decisions. They use the results of transaction processing and some other information also. It is a set of information processing functions. It should handle queries as quickly as they arrive. An important element of MIS is database.

A database is a non-redundant collection of interrelated data items that can be processed through application programs and available to many users.

c) Decision Support Systems:

These systems assist higher management to make long term decisions. These type of systems handle unstructured or semi structured decisions. A decision is considered unstructured if there are no clear procedures for making the decision and if not all the factors to be considered in the decision can be readily identified in advance.

Gorry and Morton Coined the term decision support system (DSS). The origin of the term is simple:

- Decision – emphasizes decision making in problem situations, not information processing, retrieval, or reporting.
- Support – requires computer-aided decision situations with enough "structure" to permit computer support.
- System – accentuates the integrated nature of problem solving, suggesting a combined "man", machine, and decision environment.

These are not of recurring nature. Some recur infrequently or occur only once. A decision support system must very flexible. The user should be able to produce customized reports by giving particular data and format specific to particular situations.

Check Your Progress 2

1. Differentiate between Physical and Abstract system.

_____

_____

_____

2. What is a computer based information system?

   _____

   _____

   _____

3. Write the name of different information system.

   _____

   _____

   _____

1.9 SUMMARY

- A system is orderly grouping of interdependent components linked together according to a plan to achieve a specific objective. Its main characteristic are organization, interaction, interdependence, integration and a central objective.
- To construct a system, system analyst must consider its elements- input and output, processors, control, feedback, and environment.
- Systems can be decomposed into smaller units called subsystems.
- Systems are classified as physical or abstract, open or closed, and man-made information systems. A system may be schematic, static or dynamic. An information system is an open system that allows inputs and facilitates interaction with the user. The main characteristic of an open system are input from outside, processing, output, and operation in cycles through feedback.
- Three level of information in organization that requires a special type of information system. Strategic information system for long range planning policies and upper management. Managerial information system helps middle management and department heads in policy implementation and control. Operational information system helps the daily information needed to operate the business.
- Information systems are of many types. Management Information, transaction processing, and decision support systems are all information systems.
- Transaction processing system assist in processing day to day activities of the organization
- Future emphasizes on the decision support system not on information processing, it requires a computer aided environment and accentuates a combined man and machine and decision environment.

1.10. KEYWORDS

System: A system is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective.

Characteristics of a system: Organization, interaction, interdependence, integration and a central objective.

Elements of a system: Input, processor(s), control, output, feedback, environment, boundaries and interface.

Types of system:

- Physical or abstract system

  Physical Systems are tangible entities that may be static or dynamic in operation. Abstract Systems are conceptual or non-physical entities.

- Open or closed system.

  An open system has many interfaces with its environment. It permits interaction across its boundary. It receives inputs from and delivers outputs to the outside.

  A closed system is isolated from environmental influences.

1.11 MODEL ANSWERS

Check Your Progress 1

1 A System means an organized relationship among functioning units or components. It is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective.

2 There are five types of characteristics for a system. They are:
- Organization
- Interaction
- Interdependence
- Integration
- A central objective

3 The elements of a system in terms of Information systems are:
- Outputs and Inputs
- Processor
- Control
- Environment
- Feedback
- Boundaries and interface

Check Your Progress 2

1. Physical systems are tangible entities that may be static or dynamic in nature. Physical entities can be seen and counted.

   Abstract systems are conceptual or nonphysical entities.

2. System analysts develop several different types of information systems which depend mainly on the computers for handling business applications. This class of systems is known as Computer Based Information Systems.

3. Three major information systems are

   - Transaction processing systems
   - Management information systems
   - Decision support systems

1.12. TERMINAL QUESTIONS

1. Define the term 'System'.
2. What are the various elements of system?

3. Identify two systems in your surroundings.
4. What is man-made information system?
5. Explain the features of a system.
6. Elaborate the different types of systems.
7. Differentiate between
   a) Open and closed system
   b) Physical and abstract
8. Difference between Formal and Informal System.
9. Define system. List various types of system. Also explain various characteristics of system.
10. What are the typical components of a system?
11. What is a boundary?
12. What is importance of feedback? What types of feedback you know?
13. Give full form of TPS. Give two major characteristics of it.
14. List all categories of information systems. Differentiate TPS and DSS.
15. What is total information system?
16. Explain: "Information system is an essence of a business".
17. What do you mean by Black Box system?
18. Main aim of an information system is to process _____.
19. Transaction processing, _____ , and decision support system are three types of information system.
20. State true or false
    a) Decision support system is for middle level management.
    b) Closed systems don't interact with their environment.
    c) Transaction processing system handles day-to-day operations of the organization.
    d) Management information system deals with strategic information of organization.

UNIT - II

SYSTEM DEVELOPMENT LIFE CYCLE

Unit Structure

2.1. INTRODUCTION

System life cycle is an organizational process of developing and maintaining systems. It helps in establishing a system project plan, because it gives overall list of processes and sub-processes required for developing a system.

System development life cycle means combination of various activities. In other words we can say that various activities put together are referred as system development life cycle. In the System Analysis and Design terminology, the system development life cycle also means software development life cycle.

2.2 OBJECTIVE

After studying this Unit, you should be able to:

- How to build the computer based information system?
- What are the different steps in system development life cycle?
- The components of a feasibility analysis.
- The factors considered in system design.
- How systems are implemented?
- How system maintenance can be done?

2.3 SOURCE AND INSPIRATION OF A NEW SYSTEM DEVELOPMENT

Systems are created to solve problems. So, before a system is created, there must be a problem. Once the problem is defined, a system is developed to solve it. The investigation will result in finding out the best course of action:

a) Whether to leave things as they are,
b) Upgrade the current system, or
c) Develop a new computerized system.

If it is decided that a new system is to be developed, the next phase is analysis. Analysis involves carrying out a detailed study of the present system, leading to the specifications of a new system. As the weaknesses of the current system are identified, the user will specify what s/he wants out of the new system. Here, the aim target of analysis is a specification of what the new system is to accomplish.

Based on the user requirements, the new system must be designed. The features of the new system are specified and the costs of implementing them and the benefits of the system are estimated. Then, input, output and processing specifications are drawn up in detail. When the design is completed, implementation begins. Implementation involves the actual programming and testing of the new system.

The system is then installed (put in place ready to be used) and is ready to go into operation. When it is operating, the system may be monitored to determine if it is performing up to expectations and if not, modifications (changes) will be made. As time goes by, the system may have to be changed because of changing requirements, like more customers, more employees or even new technology.

Eventually, the useful life of the system comes to an end because changes are too expensive or the equipment used is outdated. It becomes necessary and economical to replace the old system with a new one. Thus the life cycle of a system ranges from problem definition to death.

2.4 SYSTEM LIFE CYCLE

The system life cycle is the name given to a various tasks which have to be carried out when a new information technology system is being created.

The System Life Cycle could be organized as follows:

- Recognition of need
- Feasibility study
- Analysis
- System Design
- Development
- Testing
- Implementation
- Maintenance

These stages form a cycle because after a period of time the system will need modifying or replacing and the process has to be repeated.

2.4.1. Recognition of need

This is the first phase or you can say as foremost phase in any System Development Life Cycle (SDLC). During this phase, the idea of developing an automated system is born and studies of current problem and objective, benefit and scope is undertaken as the saying goes "YOU CAN NOT DECIDE HOW TO GO UNLESS YOU HAVE DECIDED WHERE TO GO" unless the end objective is clearly started, We'll not to be able to know of the system achieved the objectives or it failed. The main questions in this phase answered are:

- What are the problems?
- What are the end objectives?
- What benefits are expected?
- What are the areas to be covered?

2.4.2. Feasibility Study

The feasibility study or simply saying initiation of any system is the second stage under System Development Life Cycle (SDLC) during this phase, the system analyst interacts with the end user. This interaction is to collect information about current system, Drawback of that system and suggestion for its improvement. The main questions answered in this phase are:

- What are possible solutions?
- What are possible alternatives?
- What benefits can be expected?
- What is time frame for development?
- What are resources requirements?

From the above question answered regarding resources requirement and time frames are more important. These are normally represented in form of cost benefit analysis or economical feasibility. However the three are:

- **Technical Feasibility**

  The system was developed using HTML, DHTML, JavaScript, VBScript and ASP (Active Server pages).

  - The site was developed with window as operating system.
  - The site is interactive i.e. user friendly, thus viewing information and the related features is easy.

  Easy retrieval and access of data is provided.

- **Economic Feasibility**

  Economic feasibility is a cost benefit keeping in view that the site is economically feasible. System is economically feasible due to following points: -

  Benefits in reducing the cost are in the form of staff cut off.

  The cost incurred to implemented the system are the payment of the data entry operator, a little maintenance required for the hardware and software from time to time consistency in efficiency

- **Operational Feasibility**

  Operational feasibility is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented.

  Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. Operational feasibility reviews the willingness of the organization to support the proposed system. This is probably the most difficult of the feasibilities to gauge. In order to determine this feasibility, it is important to understand the management commitment to the proposed project. If the request was initiated by management, it is likely that there is management support and the system will be accepted and used. However, it is also important that the employee base will be accepting of the change.

2.4.3. Analysis

The most important aspect of this phase is a thorough understanding of user requirement. If this phase is not properly completed then all probability the system will not satisfy the end user. A key question is "What must be done to solve the problem". One aspect of the analysis is defining the boundaries of the system and determining whether or not the candidate's system should consider other related system. There are many tools that are audible to analyze i.e. interview, questionnaires and review of written document and fact analysis. The environment in which the analysis is carried out plays an important role i.e. how the system analysis deals with the staff organization. Major questions under consideration during analysis are:

- What is being done?
- How is it being done?
- Does a problem exist?
- If a problem exists, how severe is it?
- How frequently does it occur?
- What is the underlining cause for the problem?

2.4.4. Design

Once analysis is completed, the analyst has the firm understanding of "What is to be done". The next step is to be deciding, "How the problem might be solved" Thus is the system design, we move from logical to physical aspect of the Life Cycle. The information collected during the detailed analysis phase is used and models are prepared. Alternatives to achieve objectives are considered.

However some of the considerations are:

- What data will be accepted and how it is captured?
- What output will be generated and in what format?
- What data will be stored, how, where and how much time will be stored?
- What control, validation and checks will be provided?
- What are exact requirement in term of computer H/W, Manpower and other infrastructure?

In nutshell, we can say that this phase produces a complete picture of the system.

2.4.5. Development

The fifth phase of SDLC is the development phase. This is the phase of actual activity of creating the new automated system. The development phase consists of coding of computer programs compilation and testing of the programs, link testing and integrating of software.

The development phase consists of many activities and these are:

- Coding programs to meet system specification
- Testing and debugging the programs
- Link testing separate program into the system and testing system performance as a whole
- Run the system with the trial data/test data
- Prepare System documentation

After a system has been developed, it is very important to check if it fulfills the customer requirements. For this purpose, testing of the system is done. For testing the systems, various test cases are prepared. A test case is a certain made up situation on which system is exposed so as to find the behavior of system in that type of real situation. These test cases require data. The data can be also made up artificial data or the real data provided by the user.

There are various types of tests which are used to test the system. These include unit, integration, and acceptance testing.

The smallest unit of software design is module. In unit testing these modules are tested. Since the modules are very small even individual programmer can test them. Once the individual modules are tested, these are integrated to build the complete system. But testing individual module doesn't guarantee if the system will work properly when these units are integrated. Acceptance testing ensures that the system meets all the requirements. If it fulfills the needs then the system is accepted by the customer and put into use.

Once the development phase is over, the system is ready for handing over to the user.

2.4.6. Implementation

During the implementation phase the developed and thoroughly test software is implemented for the user. During the phase the user is in driver's seat. It is preliminary concerned with the user training site preparation and file conversation. During the final testing checks the readiness and accuracy of the system to access or update or retrieve data from new files. Once the program becomes available test data are read into computer and proceed against file produced for testing.

All system are designed in a sense to substitute existing systems, so a major issue is the strategy to be used for replacing the old system with a new one, this is called implementation. The possible strategies include:

- Parallel implementation: In this both new and old systems are operated until the new system is sufficiently proven.
- Pilot operation: In this the new system is operated in a limited capacity until it is proven, then the old system is phased out as the new one is phased in.
- Cold turkey (the big bang) in this the new system is moved in one fell swoop and the old one is moved out.

During the implementation, system must be installed, tested and fine-tuned. The first strategy seems the safest; if the new system fails there is still the old one. But it is also the most expensive since two complete systems must be operated simultaneously. But being a pilot operation, however, it may not be representative of the full system operation. Often, only after the full system has been phased in certain critical problems become apparent. The last strategy is the fastest and potentially least costly, but it is also the most risky.

Once the system has been implemented, it should perform successfully in the user's environment.

2.4.7. Maintenance

When the implementation phase is completed and the user staff is adjusted to the change accredit by the system, the evaluation and maintenance begins like any system there is an aging process that request periodical of hardware and software. If the new information is inconsistent with the design specification, then changes have to be made. Hardware also requires periodical maintenance to keep in tune with the design specification. The importance is focused on to the bringing of the new system to standard; this is the final step in the System Development Life Cycle (SDLC).

The maintenance phase of the software life cycle is the time period in which the software requirement are changed hence the software up gradation is required or software gives some errors so the testing is again required. Typically the development cycle span is much less than the span of the maintenance phase.

Maintenance covers a wide range of activities including correcting coding and design errors, updating documentation and test data and upgrading user support. Many activities, classified as maintenance, are actually enhancements. Maintenance means restoring something to its original condition. In contrast, enhancement means adding, modifying or redeveloping the code to support changes in the specification.

Maintenance can be classified as:

- Corrective Maintenance: means repairing processing or performance failures or making changes because of previously uncorrected problems.
- Adaptive Maintenance: means changing the program functions and also it may involve moving the software to different machines.
- Perceptive Maintenance: means enhancing the performance or modifying the programs the programs to respond to users changing needs.

Software enhancement may involve providing new functional capabilities, improving user display and modes of interaction, upgrading external documents or the performance characteristics of the system.

Therefore these seven stages make System Development Life Cycle (SDLC). All the stages are equally important for the success of a system. Hence development life cycle is a building block of any system. If these stages are thoroughly done then the system becomes successful.

Check Your Progress 1

1. What are the activities which complete the system development life cycle?

   _____

   _____

   _____

2. What is feasibility study?

   _____

   _____

   _____

3. List the implementation strategies.

   _____

   _____

   _____

4. Why is maintenance of a system necessary?

   _____

   _____

   _____

5. What is the various classification of maintenance?

   _____

   _____

   _____

2.5 SUMMARY

- The system's development life cycle consists of many phases. These include Analysis
  - Recognition of the need
  - Feasibility Study
  - Analysis
  - Design
  - Implementation of system
  - Maintenance
- System analysis and design are keyed to the system development life cycle (SDLC).
- Analysis is a detailed study of the various operation performed by a system.
- System design refers to the technical specifications that will be applied in implementing the system.
- Implementation is concerned with creation of the system.
- After implementation, maintenance begins includes enhancements, modifications, or any changes from the original specifications.

2.6 KEYWORDS

- SDLC
- Analysis
- Design
- Feasibility study
- Implementation
- Corrective Maintenance
- Adaptive Maintenance
- Perceptive Maintenance

2.7 MODEL ANSWERS

Check Your Progress 1

1. The System Life Cycle contains the following phases to complete the development of a system
    - Recognition of need
    - Feasibility study
    - Analysis
    - System Design
    - Development
    - Testing
    - Implementation
    - Maintenance
2. The feasibility study is the primary stage under System Development Life Cycle (SDLC) during this phase; the system analyst interacts with the end user. This interaction is to collect information about current system, Drawback of that system and suggestion for its improvement.
3. The implementation strategies include:
    - Parallel implementation: In this both new and old systems are operated until the new system is sufficiently proven.
    - Pilot operation: In this the new system is operated in a limited capacity until it is proven, then the old system is phased out as the new one is phased in.
    - Cold turkey (the big bang) in this the new system is moved in one fell swoop and the old one is moved out.
4. Maintenance of a system is necessary to eliminate errors in the working system during its working life and to take the system to all variations within the working environment.
5. Maintenance can be classified as:
    - Corrective Maintenance: means repairing processing or performance failures or making changes because of previously uncorrected problems.
    - Adaptive Maintenance: means changing the program functions and also it may involve moving the software to different machines.
    - Perceptive Maintenance: means enhancing the performance or modifying the programs the programs to respond to users changing needs.

2.8 TERMINAL QUESTIONS

1. Define systems analysis.
2. Define systems design.
3. Give full form of SDLC and describe its phases in detail.
4. What is SDLC? Discuss different level of skills required at different stages of software development process.
5. What is the difference between analysis and design? Explain.
6. How would an analysis determine the users' needs for a system? Explain.
7. Distinguish between recognition of need and feasibility study.
8. What is testing?

UNIT - III

LIFE CYCLE MODELS

Unit Structure

3.1 INTRODUCTION

A methodology is a formalized approach of developing a system. SDLC provides the guidelines for the steps or activities that need to be performed to design and develop a systems; it does not prescribe the method of performing the phases or activities.

A methodology provides a sequential approach of traversing the activities to the task of designing and developing a system and as such there can be multiple methodologies of developing a system another methodology is iterative.

3.2 OBJECTIVE

After studying this Unit, you should be able to:

- What are the various approaches of system development?
- What is linear approach for system development?
- What is iterative approach for system development?
- How Rapid Application Development model works?
- What are the advantages and disadvantages of various models?
- The use of agile methodology.

3.3 APPROACHES TO SYSTEM DEVELOPMENT

We have studied the various stages that are involved in the development of systems. There are system development models, which follow these stages. There is Linear traditional model also called waterfall model. Along with this there are many other approaches to system development.

There are

- Waterfall Model – Linear approach

- Prototyping Model – Iterative approach
- Iterative Model – Combination of linear and Iterative approach
- Spiral Model- Combination of linear and Iterative approach
- Rapid Application Development (RAD) - Iterative approach
- Agile Methodologies

3.4 WATERFALL MODEL

Waterfall model is a systematic and linear approach towards system development. This model follows the stages, which we have studied in previous section. Each stage begins or originates only after the previous stage has finished. There are different levels, corresponding to which each stage starts. Probably due to different leveling of stages, this model is called Waterfall model.

The pure waterfall lifecycle consists of several non-overlapping stages, as shown in the following figure. The model begins with establishing system requirements and continues with system design, implementation, testing, deployment of system and maintenance. The waterfall model serves as a baseline for many other lifecycle models. The stages of Waterfall Models are:

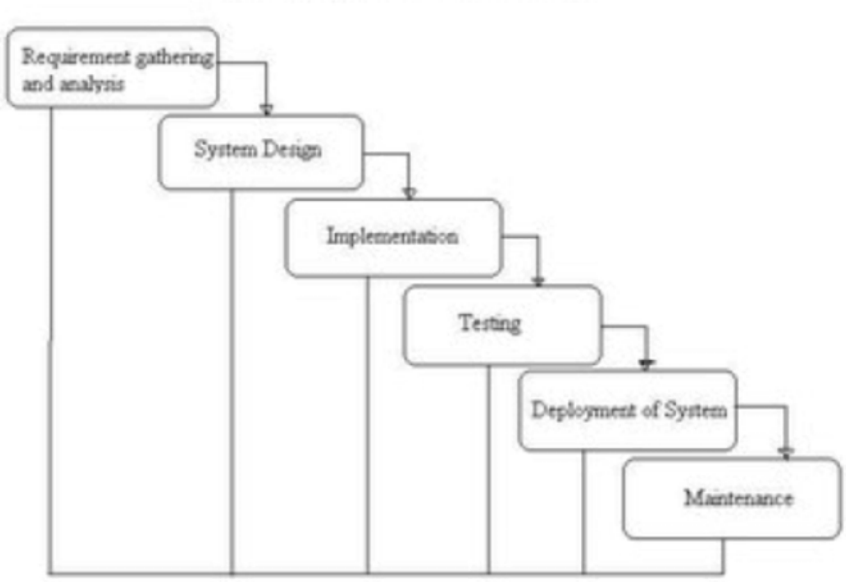


Fig 3.1: Waterfall Model

- **Requirement Gathering and Analysis:**

  All possible requirements of the system to be developed are captured in this phase. Requirements are set of functionalities and constraints that the end-user (who will be using the system) expects from the system. The requirements are gathered from the end-user by consultation, these requirements are analyzed for their validity and the possibility of incorporating the requirements in the system to be development is also studied. Finally, a Requirement Specification document is created which serves the purpose of guideline for the next phase of the model.

- **System Design:**

  Before a starting for actual implementation, it is highly important to understand what we are going to create and what it should look like?

  The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The system design specifications serve as input for the next phase of the model.

- **Implementation:**

  On receiving system design documents, the works divided in modules/units and actual implementation is started. The system is first developed in small units, which are integrated in the next phase. Each unit is developed and tested for its functionality; this is referred to as Unit Testing. Unit testing mainly verifies if the modules/units meet their specifications.

- **Testing:**

  As specified above, the system is first divided in units which are developed and tested for their functionalities. These units are integrated into a complete system during Integration phase and tested to check if all modules/units coordinate between each other and the system as a whole behaves as per the specifications.

- **Deployment of System:**

  Once the functional and non functional testing is done, the product is deployed in the customer environment or released into the market.

- **Maintenance:**

  This phase of "The Waterfall Model" is virtually never ending phase (Very long). Generally, problems with the system developed (which are not found during the development life cycle) come up after its practical use starts, so the issues related to the system are solved after deployment of the system. Not all the problems come in picture directly but they arise time to time and needs to be solved; hence this process is referred as Maintenance.

In each stage, documents that explain the objectives and describe the requirements for that phase are created. At the end of each stage, a review to determine whether the system can proceed to the next stage is held. Your prototyping can also be incorporated into any stage from the architectural design and after.

Many people believe that this model cannot be applied to all situations. For example, with the waterfall model, the requirements must be stated before beginning the design, and the complete design must be stated before starting implementation. There is no overlap between stages. In real-world development, however, one can discover issues during the design or implementation stages that point out errors or gaps in the requirements.

The waterfall method does not prohibit returning to an earlier phase, for example, returning from the design phase to the requirements phase. However, this involves costly rework. Each completed phase requires formal review and extensive documentation development. Thus, oversights made in the requirements phase are expensive to correct later.

Because the actual development comes late in the process, one does not see results for a long time. This delay can be disconcerting to management and customers. Many people also think that the

amount of documentation is excessive and inflexible. Although the waterfall model has its weaknesses, it is instructive because it emphasizes important stages of system development. Even if one does not apply this model, he must consider each of these stages and its relationship to his own system.

Advantages:

1. Easy to understand and implement.
2. Widely used and known (in theory!).
3. Reinforces good habits: define-before- design, design-before-code.
4. Identifies deliverables and milestones.
5. Document driven, URD, SRD… etc. Published documentation standards, e.g. PSS-05.
6. Works well on mature products and weak teams.

Disadvantages:

1. Idealized, doesn't match reality well.
2. Doesn't reflect iterative nature of exploratory development.
3. Unrealistic to expect accurate requirements so early in system.
4. Software is delivered late in system, delays discovery of serious errors.
5. Difficult to integrate risk management.
6. Difficult and expensive to make changes to documents, "swimming upstream".
7. Significant administrative overhead, costly for small teams and systems.

3.5 PROTOTYPING MODEL

After waterfall model, lets discuss what is prototyping model in System Development is. Here, a prototype is made first and based on it final product is developed. A prototype is a model which is not based on strict planning, but is an early approximation of the final product or system. A prototype acts as a sample to test the process. From this sample we learn and try to build a better final product. Please note that this prototype may or may not be completely different from the final system we are trying to develop.

- **Need of Prototyping Model**

  This type of System Development Method is employed when it is very difficult to obtain exact requirements from the customer (unlike waterfall model, where requirements are clear). While making the model, user keeps giving feedbacks from time to time and based on it, a prototype is made. Completely built sample model is shown to user and based on his feedback; the SRS (System Requirements Specifications) document is prepared. After completion of this, a more accurate SRS is prepared, and now development work can start using Waterfall Model.

  Now let's discuss the disadvantages and advantages of the Prototype model in system Development Method.

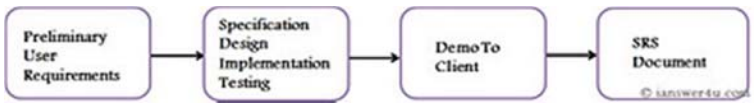


Fig 3.2: Prototyping Process Model

Advantages of Prototyping Model

1. When prototype is shown to the user, he gets a proper clarity and 'feel' of the functionality of the software and he can suggest changes and modifications.
2. This type of approach of developing the system is used for non-IT-literate people. They usually are not good at specifying their requirements, nor can tell properly about what they expect from the system.
3. When client is not confident about the developer's capabilities, he asks for a small prototype to be built. Based on this model, he judges capabilities of developer.
4. Sometimes it helps to demonstrate the concept to prospective investors to get funding for system.
5. It reduces risk of failure, as potential risks can be identified early and mitigation steps can be taken.
6. Iteration between development team and client provides a very good and conductive environment during system.
7. Time required to complete the system after getting final the SRS reduces, since the developer has a better idea about how he should approach the system.

Disadvantages of Prototyping Model:

1. Prototyping is usually done at the cost of the developer. So it should be done using minimal resources. It can be done using Rapid Application Development (RAD) tools. Please note sometimes the start-up cost of building the development team, focused on making prototype, is high.
2. Once we get proper requirements from client after showing prototype model, it may be of no use. That is why, sometimes we refer to the prototype as "Throw-away" prototype.
3. It is a slow process.
4. Too much involvement of client is not always preferred by the developer.
5. Too many changes can disturb the rhythm of the development team.

Check Your Progress 1

1. Write the phases of waterfall model.

   _____

   _____

   _____

2. List the advantages and limitations of waterfall model.

   _____

   _____

   _____

3. What is the need of prototype model?

   _____

   _____

   _____

3.6 ITERATIVE MODEL

Iterative process starts with a simple implementation of a subset of the system requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).

Iterative and Incremental development is a combination of both iterative design or iterative method and incremental build model for development. "During system development, more than one iteration of the system development cycle may be in progress at the same time." and "This process may be described as an "evolutionary acquisition" or "incremental build" approach."
In incremental model the whole requirement is divided into various builds. During each iteration, the development module goes through the requirements, design, implementation and testing phases. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is ready as per the requirement.

The key to successful use of an iterative system development lifecycle is rigorous validation of requirements, and verification & testing of each version of the system against those requirements within each cycle of the model. As the system evolves through successive cycles, tests have to be repeated and extended to verify each version of the system.

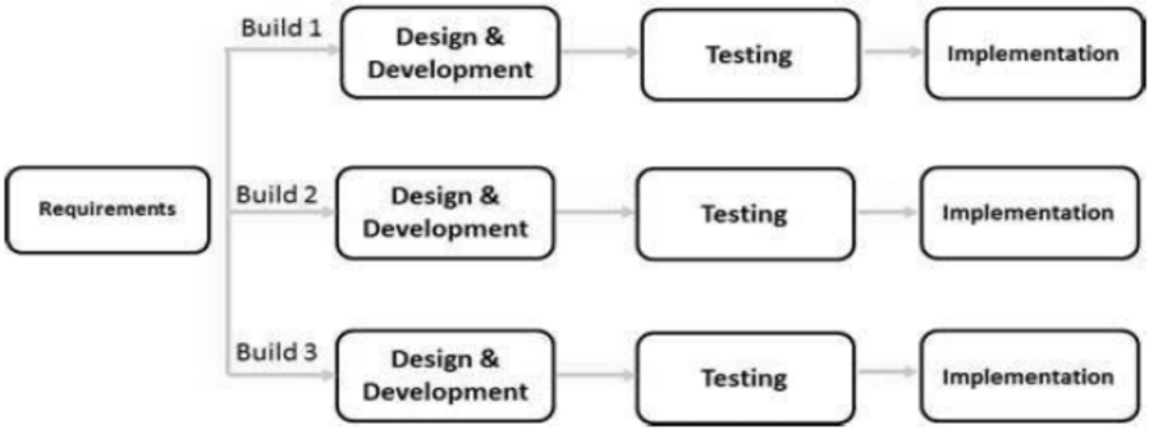


Fig 3.3: Iterative Model

Advantages

- Some working functionality can be developed quickly and early in the life cycle.
- Results are obtained early and periodically.
- Parallel development can be planned.
- Less costly to change the scope/requirements.
- Testing and debugging during smaller iteration is easy.
- Risks are identified and resolved during iteration; and each iteration is an easily managed milestone.
- Easier to manage risk - High risk part is done first.
- With every increment operational product is delivered.

- Issues, challenges & risks identified from each increment can be utilized/applied to the next increment.
- Better suited for large and mission-critical projects.
- During life cycle system is produced early which facilitates customer evaluation and feedback.

Disadvantages

- Although cost of change is lesser but it is not very suitable for changing requirements.
- More management attention is required.
- System architecture or design issues may arise because not all requirements are gathered in the beginning of the entire life cycle.
- Defining increments may require definition of the complete system.
- End of project may not be known which is a risk.
- Highly skilled resources are required for risk analysis.
- Project's progress is highly dependent upon the risk analysis phase.

3.7 SPIRAL MODEL

The spiral model has four phases. A system project repeatedly passes through these phases in iterations called Spirals.

- **Identification**

  This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.

  This also includes understanding the system requirements by continuous communication between the customer and the system analyst. At the end of the spiral the product is deployed in the identified market.

- **Design**

  Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and final design in the subsequent spirals.

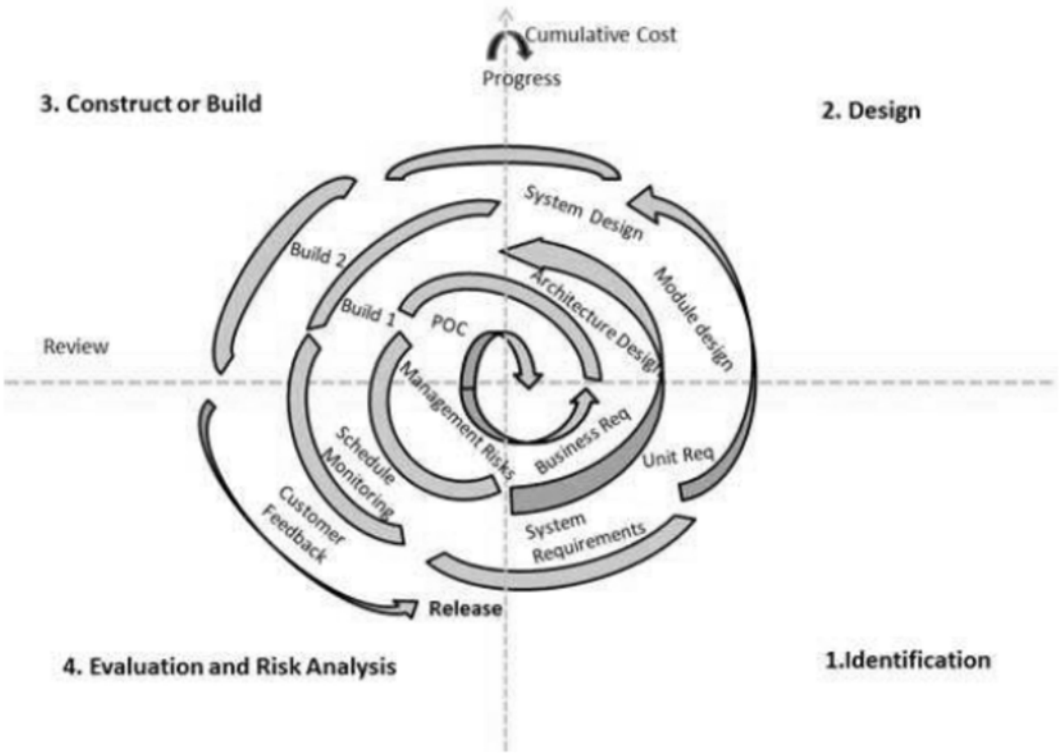


Fig 3.4: Spiral Model

- **Construct or Build**
  Construct phase refers to production of the actual system product at every spiral. In the baseline spiral when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.
  Then in the subsequent spirals with higher clarity on requirements and design details a working model of the system called build is produced with a version number. These builds are sent to customer for feedback.
- **Evaluation and Risk Analysis**
  Risk Analysis includes identifying, estimating, and monitoring technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the system and provides feedback.

  Based on the customer evaluation, system development process enters into the next iteration and subsequently follows the linear approach to implement the feedback suggested by the customer. The process of iterations along the spiral continues throughout the life of the system.

Advantages

- Changing requirements can be accommodated.
- Allows for extensive use of prototypes
- Requirements can be captured more accurately.
- Users see the system early.

- Development can be divided into smaller parts and more risky parts can be developed earlier which helps better risk management.

Disadvantages

- End of project may not be known early.
- Not suitable for small or low risk projects and could be expensive for small projects.
- Large number of intermediate stages requires excessive documentation.
- Spiral may go indefinitely.

3.8 RAD MODEL

Rapid application development (RAD) is a system development methodology that uses minimal planning in favor of rapid prototyping. A prototype is a working model that is functionally equivalent to a component of the product. In RAD model the functional modules are developed in parallel as prototypes and are integrated to make the complete product for faster product delivery.

Since there is no detailed preplanning, it makes it easier to incorporate the changes within the development process. RAD projects follow iterative and incremental model and have small teams comprising of developers, domain experts, customer representatives and other IT resources working progressively on their component or prototype. The most important aspect for this model to be successful is to make sure that the prototypes developed are reusable.

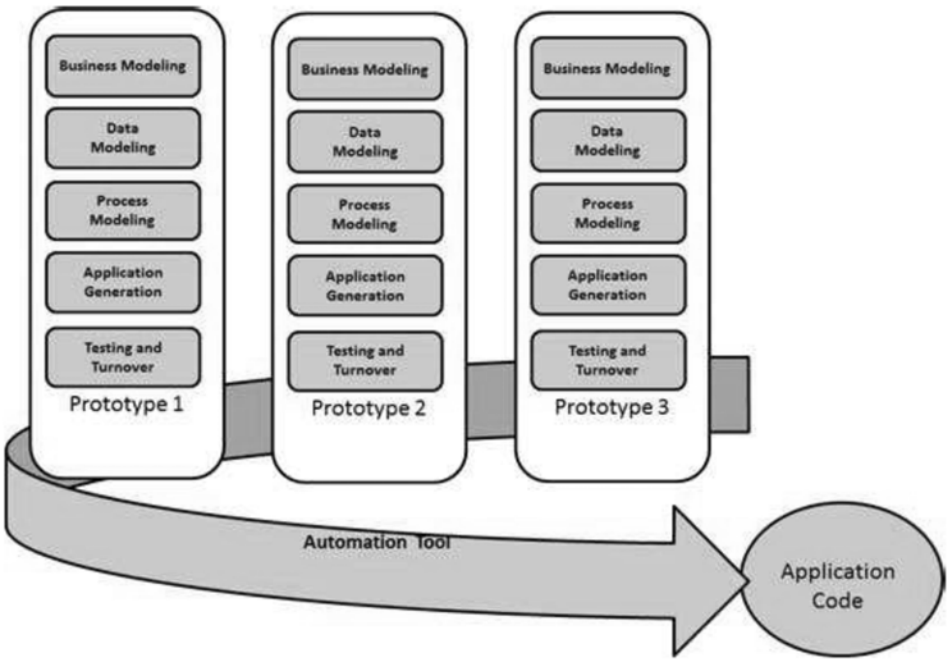


Fig 3.5: RAD Model

RAD model distributes the analysis, design, build, and test phases into a series of short, iterative development cycles. Following are the phases of RAD Model:

- **Business Modeling:**
  The business model for the product under development is designed in terms of flow of information and the distribution of information between various business channels. A complete business analysis is performed to find the vital information for business, how it can

be obtained, how and when is the information processed and what are the factors driving successful flow of information.

- **Data Modeling:**
  The information gathered in the Business Modeling phase is reviewed and analyzed to form sets of data objects vital for the business. The attributes of all data sets is identified and defined. The relation between these data objects are established and defined in detail in relevance to the business model.
- **Process Modeling:**
  The data object sets defined in the Data Modeling phase are converted to establish the business information flow needed to achieve specific business objectives as per the business model. The process model for any changes or enhancements to the data object sets is defined in this phase. Process descriptions for adding , deleting, retrieving or modifying a data object are given.
- **Application Generation:**
  The actual system is built and coding is done by using automation tools to convert process and data models into actual prototypes.
- **Testing and Turnover:**
  The overall testing time is reduced in RAD model as the prototypes are independently tested during every iteration. However the data flow and the interfaces between all the components need to be thoroughly tested with complete test coverage. Since most of the programming components have already been tested, it reduces the risk of any major issues.

Advantages

- Changing requirements can be accommodated.
- Iteration time can be short with use of powerful RAD tools.
- Productivity with fewer people in short time.
- Increases reusability of components

Disadvantages

- Dependency on technically strong team members for identifying business requirements.
- Only system that can be modularized can be built using RAD
- Requires highly skilled developers/designers.
- High dependency on modeling skills
- Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.

3.9 AGILE MODEL

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided to time boxes (small time frames) to deliver specific features for a release. Iterative approach is taken and working system build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.

Agile thought process had started early in the system development and started becoming popular with time due to its flexibility and adaptability. The most popular agile methods include Rational Unified Process (1994), Scrum (1995), Crystal Clear, Extreme Programming (1996), Adaptive Software Development, Feature Driven Development, and Dynamic Systems Development Method

(DSDM) (1995). These are now collectively referred to as agile methodologies, after the Agile Manifesto was published in 2001.
Following are the Agile Manifesto principles:

- **Individuals and interactions** – in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
- **Working software –** Demo working system is considered the best means of communication with the customer to understand their requirement, instead of just depending on documentation.
- **Customer collaboration** – As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.
- **Responding to change** – agile development is focused on quick responses to change and continuous development.

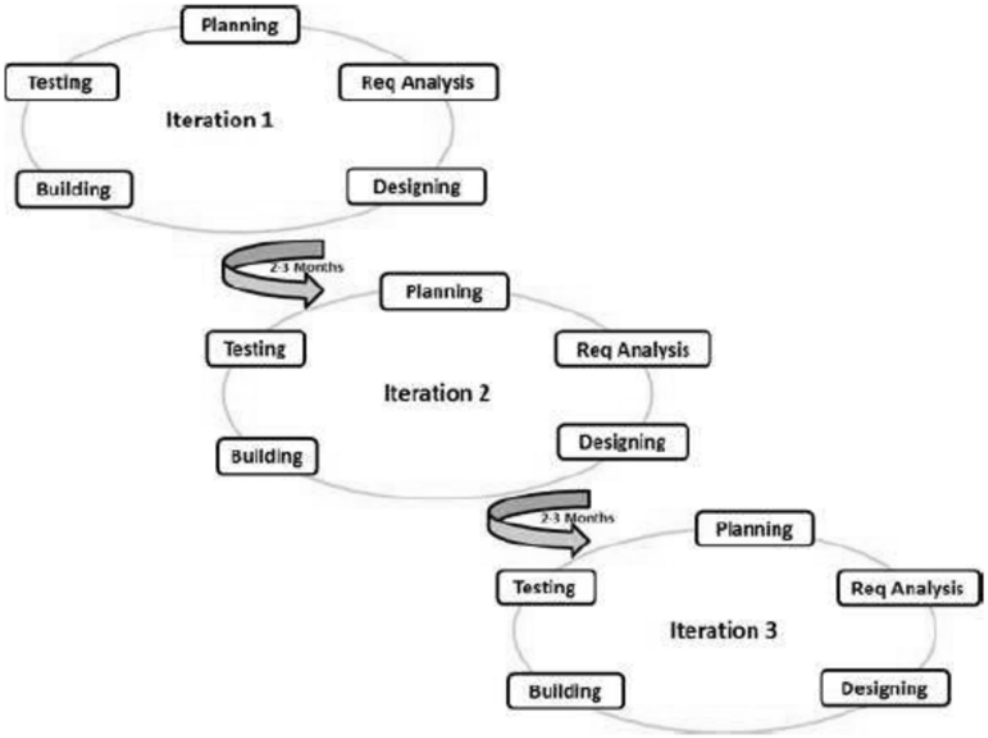


Fig 3.6: Agile Model

Advantages

- Is a very realistic approach to software development
- Promotes teamwork and cross training.
- Functionality can be developed rapidly and demonstrated.
- Resource requirements are minimum.
- Suitable for fixed or changing requirements
- Delivers early partial working solutions.

Disadvantages

- Not suitable for handling complex dependencies.
- More risk of sustainability, maintainability and extensibility.

- An overall plan, an agile leader and agile PM practice is a must without which it will not work.
- Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.

Check Your Progress 2

1. Write the phases of spiral model.

   _____

   _____

   _____

2. Write the phases of RAD model.

   _____

   _____

   _____

3. Explain the Agile Manifesto principles.

   _____

   _____

   _____

3.10 SUMMARY

- Various life cycle models are available in the industry.
  - Waterfall Model
  - Prototyping Model
  - Iterative Model
  - Spiral Model
  - Rapid Application Development (RAD)
  - Agile Methodologies
- Each model has its own limitations and advantages. These should be taken into consideration while deciding for the methodology.
- Waterfall is traditional SDLC model and is of sequential type. Sequential means that the next phase can start only after the completion of first phase. Such models are suitable for projects with very clear product requirements and where the requirements will not change dynamically during the course of project completion.
- Iterative and Spiral models are more accommodative in terms of change and are suitable for projects where the requirements are not so well defined, or the market requirements change quite frequently.
- Agile is the most popular model used in the industry. Agile introduces the concept of fast delivery to customers using prototype approach. Agile divides the project into small iterations with specific deliverable features. Customer interaction is the backbone of Agile methodology, and open communication with minimum documentation are the typical features of Agile development environment.
- RAD (Rapid Application Development) is modern techniques to understand the requirements in a better way early in the project cycle. These techniques work on the concept of providing a working model to the customer and stockholders to give the look and feel and collect the feedback. This feedback is used in an organized manner to improve the product.

3.11 KEYWORDS

- SDLC
- Prototype Model
- RAD
- Agile Model
- Agile Manifesto

3.12 MODEL ANSWERS

Check Your Progress 1

1. The Various phases of waterfall model are
    - Requirement Gathering and Analysis:
    - System Design:
    - Implementation:
    - Testing:
    - Deployment of System:
    - Maintenance:
2. Advantages and limitation of waterfall model

Advantages:

1. Easy to understand and implement.
2. Widely used and known (in theory!).
3. Reinforces good habits: define-before- design, design-before-code.
4. Identifies deliverables and milestones.
5. Document driven, URD, SRD… etc. Published documentation standards, e.g. PSS-05.
6. Works well on mature products and weak teams.

Disadvantages:

1. Idealized, doesn't match reality well.
2. Doesn't reflect iterative nature of exploratory development.
3. Unrealistic to expect accurate requirements so early in system.
4. Software is delivered late in system, delays discovery of serious errors.
5. Difficult to integrate risk management.
6. Difficult and expensive to make changes to documents, "swimming upstream".
7. Significant administrative overhead, costly for small teams and systems.

3. Prototype of System Development Method is employed when it is very difficult to obtain exact requirements from the customer (unlike waterfall model, where requirements are clear). While making the model, user keeps giving feedbacks from time to time and based on it, a prototype is made.

Check Your Progress 2

1. The Various phases of spiral model are
    - Design
    - Identification
    - Construct or Build
    - Evaluation and Risk Analysis

2. The Various phases of RAD model are
    - Business Modeling:
    - Data Modeling:
    - Process Modeling:
    - Application Generation:
    - Testing and Turnover:
3. The Agile Manifesto principles are
    - **Individuals and interactions** – in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
    - **Working software –** Demo working system is considered the best means of communication with the customer to understand their requirement, instead of just depending on documentation.
    - **Customer collaboration** – As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.
    - **Responding to change** – agile development is focused on quick responses to change and continuous development.

3.13 TERMINAL QUESTIONS

1. Define prototype. List any three purposes of it.
2. What do you mean by waterfall model? Explain.
3. Define prototype. Draw steps of prototype. Give any two situations (candidate applications) where prototyping should be used
4. List any three advantages of prototype.
5. What are the development difficulties that prototype face?
6. What is a Prototype of system? What are the advantages of it and what steps required to build a Prototype?
7. What is agile model? What is its use?
8. Write short note on RAD?
9. Describe the term prototype and outline advantages and disadvantages of prototyping.
10. Differentiate between spiral and iterative model.
11. What are the advantages and disadvantages of spiral model?
12. Describe the term RAD and outline advantages and disadvantages of RAD.
13. What are the various methodologies for system development?
14. List some advantages of iterative model.
15. Explain the key features of the followings:
    a. Waterfall Model
    b. Spiral Model

UNIT - IV

THE ROLE OF THE SYSTEM ANALYST

Unit Structure

4.1. INTRODUCTION

A system analyst studies the problems and need of an organization to determine how people, methods, and computer technology can best accomplish improvements for the business.

When computer technology is used, the analyst is responsible for the efficient capture of data from its business source, the flow of that data to the computer, the processing and storage of that data by the computer, and flow of useful and timely information back to business users.

4.2 OBJECTIVE

After studying this Unit, you should be able to:

- The need for systems analysis for business in the development of new systems.
- The role of the systems analyst in system development.
- What tasks to do for system analysis?
- The tasks of system analyst.
- What are the required skills for system analysts?
- The multifaceted role of the analyst.

4.3 SYSTEM ANALYSIS AND DESIGN

System development can generally be thought of having two major components: systems analysis and systems design. In System Analysis more emphasis is given to understanding the details of an existing system or a proposed one and then deciding whether the proposed system is desirable or not and whether the existing system needs improvements. Thus, system analysis is the process of

investigating a system, identifying problems, and using the information to recommend improvements to the system.

Systems analysis will identify

1. outputs and processing needed.
2. data required to provide this processing and output.
3. role of people in the process.
4. security aspects to ensure the efficient continuation of the business.
5. costs of providing the system.

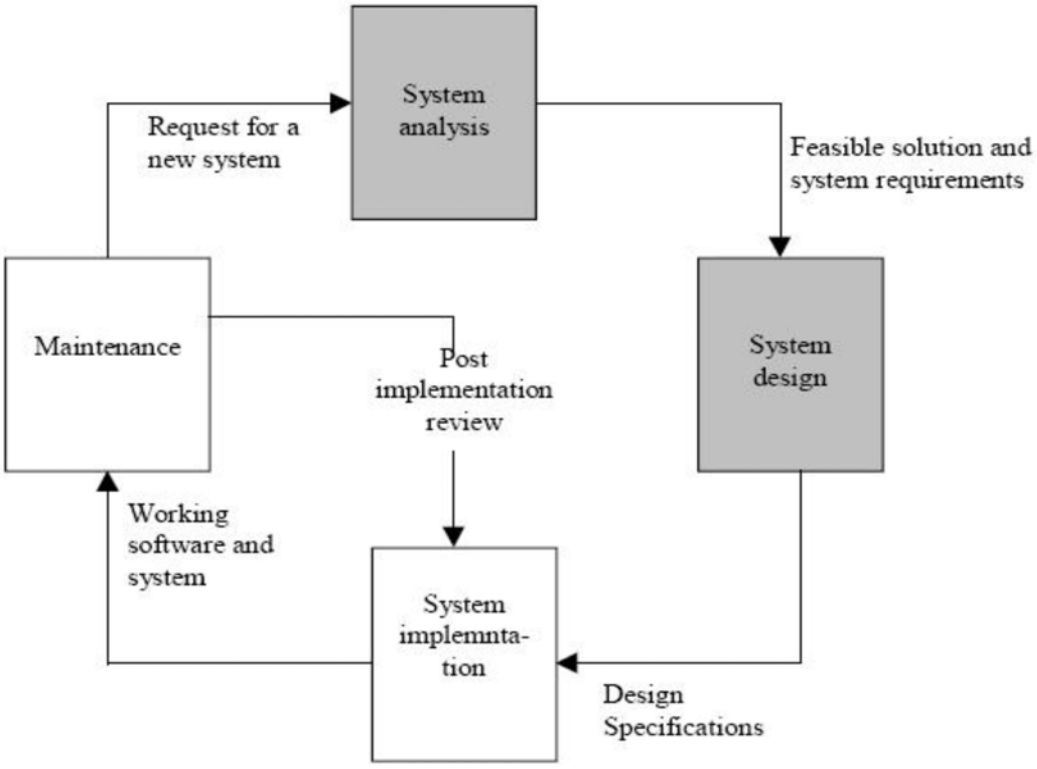


Fig 4.1: Stages in building an improved system

Fig shows the various stages involved in building an improved system. System design is the process of planning a new business system or one to replace or complement an existing system.

Analysis specifies what the system should do. Design states how to accomplish the objective. After the proposed system is analyzed and designed, the actual implementation of the system occurs. After implementation, working system is available and it requires timely maintenance. See figure.

4.4 ROLE OF SYSTEM ANALYST

The system analyst is the person (or persons) who guides through the development of an information system. In performing these tasks the analyst must always match the information system objectives with the goals of the organization.

Role of System Analyst differs from organization to organization. Most common responsibilities of System Analyst are following.

4.4.1 System analysis

It includes system's study in order to get facts about business activity. It is about getting information and determining requirements. Here the responsibility includes only requirement determination, not the design of the system.

4.4.2 System analysis and design

Here apart from the analysis work, Analyst is also responsible for the designing of the new system/application.

4.4.3 Systems analysis, design, and programming

Here Analyst is also required to perform as a programmer, where he actually writes the code to implement the design of the proposed application.

Due to the various responsibilities that a system analyst requires to handle, he has to be multifaceted person with varied skills required at various stages of the life cycle. In addition to the technical knowhow of the information system development a system analyst should also have the following knowledge.

- **Business knowledge:** As the analyst might have to develop any kind of a business system, he should be familiar with the general functioning of all kind of businesses.
- **Interpersonal skills:** Such skills are required at various stages of development process for interacting with the users and extracting the requirements out of them
- **Problem solving skills:** A system analyst should have enough problem solving skills for defining the alternate solutions to the system and also for the problems occurring at the various stages of the development process

4.5 SKILL REQUIRED FOR SYSTEM ANALYST

A systems analyst should have the following background and knowledge:

- **Knowledge of SDLC**: Essentially, the systems analyst performs systems analysis, systems design, systems implementation, and systems support for computer-based business applications.
- **Technical Knowledge**: An analyst does more than analyze and design systems. The analyst must be experienced in working with computers and must exploit the latest technological innovations.
- **Programming Knowledge**: He or she must have enough computer experience to program, to understand the capabilities of computers, to glean information requirements from users, and to communicate what is needed to programmers.
- **Problem Solver**: The analyst is a problem solver. He or she must have the capability to take a large business problem, break that problem down into its component parts, analyze various aspects of the problem, and then assemble a system to solve the problem. Analysts must be able to creatively define alternative solutions to problems and needs.
- **Knowledge of Tools and Techniques**: The analyst plays many roles, sometimes balancing several at the same time. He or she is a person who views the analysis of problems as a challenge and who enjoys devising workable solutions. When necessary, the analyst must be able to systematically tackle the situation at hand through skillful application of tools, techniques, and experience.
- **Good Communicator**: The analyst must be a good communicator – capable of relating meaningfully to other people over extended periods of time. He or she must be able to work

with people of all descriptions. The analyst must be willing and able to deal with his or her business clients (end-users), business management, programmers, information system management, auditors, and information system salespeople.

- **Teamwork:** System analysts work in teams, and so the ability to function as part of a term, to cooperate and compromise, is critical for the success of most projects.
- **Business Knowledge**: The system analyst need to have some general business knowledge, so that he or she can communicate with business experts to gain knowledge of problems and needs.

In many organizations, a systems analyst's role may not be well defined. A computer programmer may perform some or all of the responsibilities of an analyst. In some organization, a system analyst works as a programmer.

The most common title is programmer/analyst, whose job includes the responsibilities of both the computer programmer and the systems analyst. Other titles include systems engineer, systems designer, operations specialist, information specialist, data analyst, and business analyst.

Check Your Progress 1

1. What is System Analyst? Explain the role of it.

   _____

   _____

   _____

2. What knowledge should a system analyst have?

   _____

   _____

   _____

3. List five important skills required for a system analyst.

   _____

   _____

   _____

4.6 TASK OF SYSTEM ANALYST

The primary objective of any system analyst is to identify the need of the organization by acquiring information by various means and methods. Information acquired by the analyst can be either computer based or manual. Collection of information is the vital step as indirectly all the major decisions taken in the organizations are influenced. The system analyst has to coordinate with the system users, computer programmers, manager and number of people who are related with the use of system. Following are the tasks performed by the system analyst:

- **Defining Requirement:** The basic step for any system analyst is to understand the requirements of the users. This is achieved by various fact finding techniques like interviewing, observation, questionnaire etc. The information should be collected in such a

way that it will be useful to develop such a system which can provide additional features to the users apart from the desired.

- **Prioritizing Requirements**: Number of users uses the system in the organization. Each one has a different requirement and retrieves different information. Due to certain limitations in computing capacity it may not be possible to satisfy the needs of all the users. Even if the computer capacity is good enough is it necessary to take some tasks and update the tasks as per the changing requirements. Hence it is important to create list of priorities according to users requirements. The best way to overcome the above limitations is to have a common formal or informal discussion with the users of the system. This helps the system analyst to arrive at a better conclusion.
- **Gathering Facts, data and opinions of Users:** After determining the necessary needs and collecting useful information the analyst starts the development of the system with active cooperation from the users of the system. Time to time, the users update the analyst with the necessary information for developing the system. The analyst while developing the system continuously consults the users and acquires their views and opinions.
- **Evaluation and Analysis:** As the analyst maintains continuous he constantly changes and modifies the system to make it better and more user friendly for the users.
- **Solving Problems:** The analyst must provide alternate solutions to the management and should a in dept study of the system to avoid future problems. The analyst should provide with some flexible alternatives to the management which will help the manager to pick the system which provides the best solution.
- **Drawing Specifications:** The analyst must draw certain specifications which will be useful for the manager. The analyst should lay the specification which can be easily understood by the manager and they should be purely non-technical. The specifications must be in detailed and in well presented form.

4.7 ATTRIBUTES OF SYSTEM ANALYST

A System Analyst analyzes the organization and design of businesses, government departments, and non-profit organizations; they also assess business models and their integration with technology.

There are at least four tiers of business analysis:

1. **Planning Strategically** - The analysis of the organization business strategic needs
2. **Operating/Business model analysis** - the definition and analysis of the organization's policies and market business approaches
3. **Process definition and design** - the business process modeling (often developed through process modeling and design)
4. **IT/Technical business analysis** - the interpretation of business rules and requirements for technical systems (generally IT).

Within the systems development life cycle domain (SDLC), the business analyst typically performs a liaison function between the business side of an enterprise and the providers of services to the enterprise. A Common alternative role in the IT sector is business analyst, systems analyst, and functional analyst, although some organizations may differentiate between these titles and corresponding responsibilities.

4.8 THE MULTIFACETED ROLE OF THE ANALYST

Among the roles an analyst performs are change agent, monitor, architect, psychologist, salesperson, motivator and politician.

- **Change Agent**

  The analyst may be viewed as an agent of change. A candidate system is designed to introduce change and reorientation in how the user organization handles information or makes decisions. It is important, that the user accept change. Analyst can secure user acceptance is through user participation during design and implementation.

  In the role of a change agent, the systems analyst may select various styles to introduce change to the user organization. The styles range from that of persuader (the mildest form of intervention) to imposer (the most severe intervention). In between there are the catalyst and the confronter roles. When the user appears to have a tolerance for change the persuader or catalyst style is appropriate. On the other hand, when drastic changes are required, it may be necessary to adopt the confronter or even the imposer style. No matter what style is used, the goal is same: to achieve acceptance of the candidate system with a minimum of resistance.

- **Investigator and monitor**

  In defining a problem, the analyst will collect and put together all the information to determine why the present system does not work well and what changes will correct the problem. This work is similar to that of an investigator- extracting the real problems from existing systems and creating information structures that uncover previously unknown trends that may have a direct impact on the organization.

  Related to the role of investigator is that of monitor. To undertake and successfully complete a project, the analyst must monitor programs in relation to time, cost, and quantity of these resources, time is the most important. If time "gets away", the project suffers from increased costs and wasted human resources. Implementation will also get delayed.

- **Architect**

  As architect an analyst must create detailed physical design of candidate system. He aids users in formalizing abstract ideas and provides details to build the end product-the candidate system.

- **Psychologist**

  The analyst plays the role of a psychologist in the way he reaches people interprets their thoughts, assesses their behavior, and draws conclusions from these interactions. Understanding interfunctional relationships is important. It must be aware of people's feelings and be prepared to get around things in a graceful way. The art of listening is important in evaluating responses and feedback.

- **Salesperson**
  Selling change can be crucial as initiating change. Selling the system actually takes place at each step in the system life cycle. Sales skills and persuasiveness are crucial to the success of the system.
- **Motivator**

  A candidate system must be well designed and acceptable to the user. The analyst role as a motivator becomes obvious during the first few weeks after implementation and during times when turn over results in new people being trained to work with the candidate system. The amount of dedication it takes to motivate users often taxes the analyst's abilities to maintain the pace.

- **Politician**

  In implementing a candidate system, the analyst tries to appease all parties involves. Diplomacy and finesse in dealing with people can improve acceptance of the system. In as much as a politician must have the support of his or her constituency, so is the analyst's goal to have the support of the users staff. He or she represents their thinking and tries to achieve their goals through computerization.

In summary these multiple roles require analysts to be orderly, approach a problem in a logical, methodical way and pay attention to details.

Check Your Progress 2

1. Which is in your opinion the most difficult job of a system analyst?

   _____

   _____

   _____

2. List the multifaceted role of system analyst.

   _____

   _____

   _____

3. List three important tasks of a system analyst.

   _____

   _____

   _____

4.9 SUMMARY

- System analysis and design refers to the application of systems approach to problem solving.
- A System Analyst needs variety of skills. As organizational change agents, all analysts need to have general skills, such as technical, business, analytical, interpersonal, management, and ethical.
- System analyst should have
  - Business knowledge
  - Interpersonal skills
  - Problem solving skills
- Skills required for a system analyst
  - Knowledge of SDLC
  - Technical Knowledge
  - Programming Knowledge
  - Problem Solver
  - Knowledge of Tools and Techniques
- There are many tasks of systems analyst should have. Three of them are following
  - Defining Requirement

- Gathering Facts, data and opinions of Users
- Evaluation and Analysis

- The multifaceted roles an analyst performs are
  - Change agent,
  - Monitor,
  - Architect,
  - Psychologist,
  - Salesperson,
  - Motivator,
  - Politician.

4.10 KEYWORDS

- System Analysis
- System Design
- System Analyst
- Interpersonal Skill
- Teamwork
- Technical writer

4.11 MODEL ANSWERS

Check Your Progress 1

1. The system analyst is the person (or persons) who guides through the development of an information system. In performing these tasks the analyst must always match the information system objectives with the goals of the organization. Role of System Analyst differs from organization to organization. Most common responsibilities of System Analyst are following
   - System Analysis
   - System analysis and design
   - System analysis, design and programming.
2. System analyst should have
   - Business knowledge
   - Interpersonal skills
   - Problem solving skills
3. Skills required for a system analyst
   - Knowledge of SDLC
   - Technical Knowledge
   - Programming Knowledge
   - Problem Solver
   - Knowledge of Tools and Techniques

Check Your Progress 2

1. The most difficult job of a system analyst is Problem defining, as some business problems are quite difficult to define and no problem can be solved until it is precisely defined.
2. Among the roles an analyst performs are
    - Change agent,
    - Monitor,
    - Architect,
    - Psychologist,
    - Salesperson,
    - Motivator,
    - Politician.
3. There are many tasks of systems analyst should have. Three of them are following
    - Defining Requirement
    - Gathering Facts, data and opinions of Users
    - Evaluation and Analysis

4.12 TERMINAL QUESTIONS

1. Define systems analyst. Also list duties of systems analyst.
2. Define role of Systems Analyst in a small firm.
3. What is system analysis and design?
4. What are the roles of system analyst?
5. Make a list of traits that a system analyst should have.
6. Will the responsibility of a system analyst vary according to:
   (a) Organization size (for example small or large business)?
   (b) Type of organization (business, government agency, non-profit organization)?
7. Explain the task of a system analyst.
8. What are the multifaceted roles of system analyst?
9. Are excellent programmers necessarily excellent system analysts? Justify your answer.
10. Why should a systems analyst be able to communicate well?
11. Write the different techniques which can be used by the systems analyst to obtain a detailed understanding of the various business processes.

Uttar Pradesh Rajarshi Tandon
open University

MCA-E4/
PGDCA-E4

Master in Computer Application

BLOCK

2

SYSTEM ANALYSIS

Course Design Committee

Course Preparation Committee

Dr. Saurav Pal — **Author**
Associate Professor
Department of MCA, VBS Purvanchal University
Jaunpur-222001, Uttar Pradesh

Prof. R. R. Tiwari — **Editor**
University of Allahabad
Allahabad, Uttar Pradesh

Dr. Ashutosh Gupta
Director (In-charge), School of Computer and Information Science,
UPRTOU, Allahabad

Ms. Marisha (Coordinator)
Assistant Professor, School of Sciences,
UPRTOU, Allahabad

Printed and Published by Pro. (Dr.) Girija Shankar Shukla Registrar, **Uttar Pradesh Rajarshi Tandon open University, 2017.**
Printed By : Chandrakal Universal Pvt. Ltd. 42/7 Jawahar Lal Neharu Road Allahabad

UNIT - I

SYSTEM PLANNING

Unit Structure

1.1. INTRODUCTION

What can the analyst do to ensure the success of a system? First, a plan must be devised, detailing the procedure, some methodology, activities, resources, costs, and timetable for completing the system. Second, in larger projects, a project team must be formed of analysts, programmers, a system consultant, and user representatives. Shared knowledge, interaction, and the coordination realized through team effort can be extremely effective in contrast with individual analysts doing the same work. Finally, the project should be divided into manageable modules to reflect the phases of system development – analysis, design, and implementation.

1.2 OBJECTIVE

After studying this Unit, you should be able to:

- Why planning is important in system analysis?
- What are the planning dimensions?
- What are the user's requirements?
- What is fact analysis?
- How prototyping is used in determining information requirements?

1.3 DIMENSIONS OF PLANNING

The following conditions dictate present business strategies:

- High interest rates make it more important that business realizes a good return on investment
- Inflation puts pressure on profit when it occurs.
- The growing trend toward guaranteed employment suggests that costs are becoming fixed and the commitment to business expansion may not be easily changed.
- Resource shortages impede expansion.
- Regulatory constraints slow entry into the market.
- Increased productivity paves the way for expansion.

Information systems embedded in an organization provide users with the opportunity to add value to products and business operations at lower costs as shown in figure. Therefore they must be carefully planned.

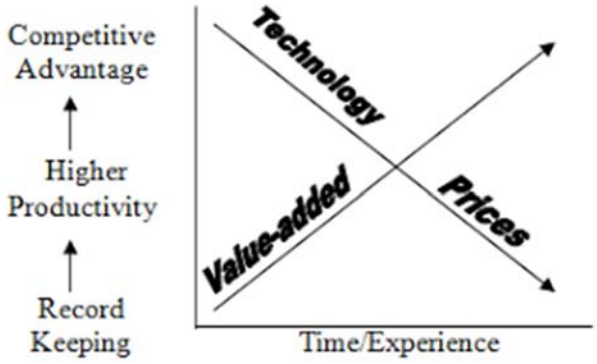


Fig 1.1: Technology Cost/Benefit Curve

1.3.1 Strategic MIS Planning

Planning for information system development must be done within the framework of the organization's overall MIS plan. It may be viewed from two dimensions:

- The time horizon dimension specifies whether it is short range, medium range or long range plan.
- The focus dimension tells whether the primary concern is strategic, managerial or operational Strategic (MIS) planning is an orderly approach that determines the basic objectives, the strategies and policies needed to achieve the objectives, and the plans to implement the strategies.

The first task in strategic planning is to set the MIS objectives and the results expected. The objectives must be set in such a way that they meet the organization's needs. Once the objectives are set, MIS policies are defined as a guideline to carry out the plan. These MIS policies are in turn translated into long-range, medium-range and short-range plans for implementation.

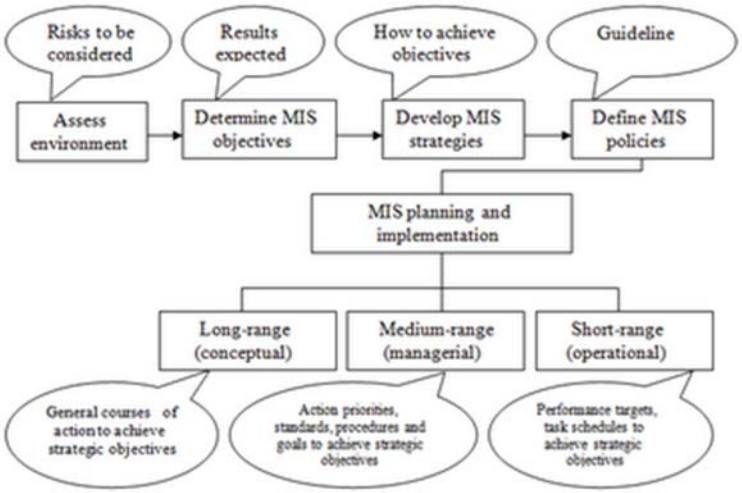


Fig 1.2: MIS Strategic Planning

In determining the MIS strategic plan, the following are to be taken into consideration

- The MIS objectives and strategies that can be derived from the corporate strategic plan.
- The person who will review and approve the plan.
- The time taken to complete the plan and the contents of the plan.
- The highlights or focus of the plan (computer security, new application development, new technology)

In most cases, the answers depend on the structure and complexity of the MIS organization, the level of computerization in the firm, the hit rate of the MIS division and the influence of MIS in getting projects approved by top management.

1.3.2 Managerial and Operational MIS Planning

Managerial MIS planning combines strategic with operational plans. It is a process in which specific functional plans are related to a specific number of years. These plans show how strategies are to be carried out to achieve long-range plans. The next step is to find short-range plans that are used for carrying out the day-to-day activities of the system. They are programmed plans requiring a year's commitment.

The MIS operating plan requires more user involvement to define the system requirements. System development must support organizational MIS objectives as laid out in the corporate plan. System development must also identify and select applications that are the organization's priorities. Bowman, Davis and Wetherbe have described this link as a three-stage model consisting of the following

- **Strategic system planning** – establishing relationships between the organization plan and the plan for a candidate system.
- **Information requirements analysis** – identifying organization requirements to direct the specific application of system development projects.

- **Resource allocation** – determining hardware, software, telecommunications, facilities, personnel and financial resources to execute the development of the system.

Thus planning for system development activities is a major aspect. Broad corporate strategic objectives should be the basis for system development objectives, which specify the goals in the form of specific action plans. Formalizing the planning process makes it easier to reorient and gain the support of upper, middle and operating management for candidate systems. The following figure shows a top-down approach to planning, the relationship between the corporate strategic plan and the goals and activities of the system development function.

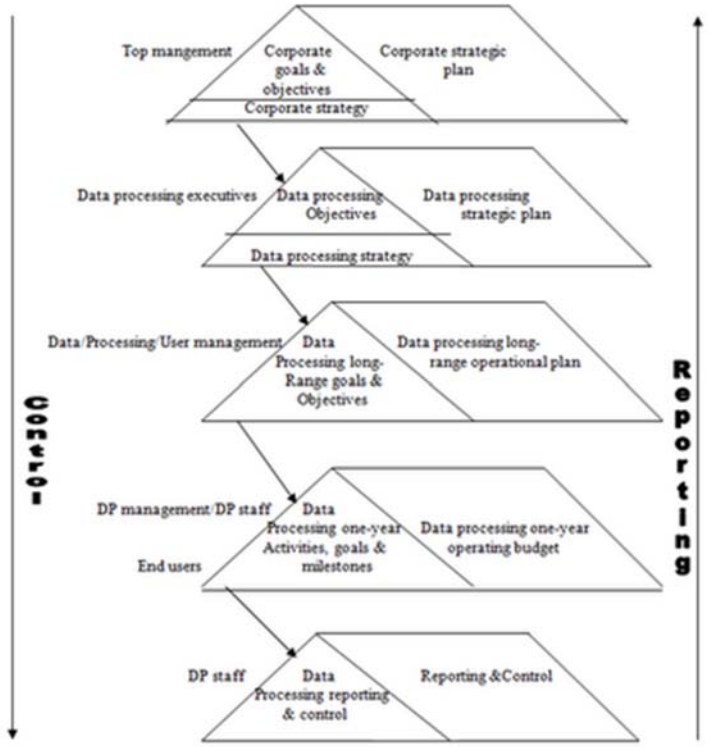


Fig 1.3: Top-down approach for system planning

1.4. INITIAL INVESTIGATION

The identification of a need is the first step in the system development life cycle. This is a user's request to change, improve, or enhance an existing system. Because there is likely to be a stream of such requests, standard procedures must be established to deal with them. The initial investigation is one way of handling such request. The objective is to determine whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system or existing one.

The user's request form specifies the following:

1. User-assigned title of work requested
2. Nature of work requested

3. The date the request was submitted.
4. The date the job should be completed
5. Job objectives – purpose of job requested
6. Expected benefits to be derived from proposed change
7. Input/output description – quantity of inputs and outputs of proposed change
8. Requester's signature, title, department and phone number
9. Signature, title, department and phone number of person approving the request.

The user request identifies the need for change and authorizes the initial investigation. It may undergo several modifications before it becomes a written commitment. Once the request is approved, the following activities are carried out.

- Background investigation
- Fact-finding and analysis
- Presentation of results – called project proposal.

1.5 NEEDS IDENTIFICATION

The success of a system depends largely on how accurately a problem is defined, thoroughly investigated and properly carried out through the choice of solution. Needs identification and analysis are concerned with what the user needs rather than what she wants. The analyst starts to think of the solution for the problem only after it has been identified, defined and evaluated. This helps the user and the analyst understand the real problem rather than its symptoms.

The user or the analyst may identify the need for a candidate system or enhancements in the existing system. The user initiates an investigation by filling out a request form for information. The request contains the objectives and expected benefits. (See figure – A sample user's request form). Thus for the design of a successful system, a clear knowledge of the system is essential. This information is given out in needs identification.

<table>
<tr><td>Job Title:</td><td colspan="2">Nature of job:
(new or revision of existing one)</td><td>Request date</td><td>Date of completion</td></tr>
<tr><td colspan="5">Job Objective:</td></tr>
<tr><td colspan="5">Expected Benefits:</td></tr>
<tr><td colspan="2">Output Specifications</td><td colspan="3">Input Specifications</td></tr>
<tr><td colspan="2">Report title :
Quantity :
Frequency :
Remarks :</td><td colspan="3">Document title :
Quantity :
Frequency :
Remarks :</td></tr>
<tr><td>Name of the Requester:</td><td>Signature:</td><td>Title:</td><td>Department:</td><td>Phone:</td></tr>
<tr><td>Approved By:</td><td>Signature:</td><td>Title:</td><td>Dept/Div:</td><td>Phone:</td></tr>
<tr><td colspan="5">For MIS Dept only</td></tr>
<tr><td colspan="2">Job No:</td><td colspan="3">Status:</td></tr>
<tr><td colspan="2">Acceptance authorized by:</td><td>Title:</td><td>Phone:</td><td>Remarks:</td></tr>
</table>

Fig 1.4: User request form

Check Your Progress 1

1. What do you mean by strategic MIS planning?

   _____

   _____

   _____

2. List the contents of user's request form.

   _____

   _____

   _____

3. Once the request is approved, List the activities which are carried out.

   _____

   _____

   _____

1.6 DETERMINING THE REQUIREMENTS

Shared, complete and accurate information requirements are essential in building computer-based information systems. Determining the information each user needs is a particularly difficult task. The information analyst determines the needs of the user and the information flow that will satisfy those needs. The usual approach is to ask the user what information is currently available and what information is required. Interaction between the analyst and the user usually leads to an agreement about what information will be provided by the candidate system. There are several difficulties in determining the user's requirements

- Changing system requirements. The user requirements must also be modified to accommodate the changes.
- Identification of requirements is difficult, except for experienced for users.
- Full user involvement is difficult to obtain.
- The pattern of interaction between the users and the analysts in designing the information requirements is complex.

Users and analysts do not share a common orientation towards problem definition. For the analyst, the problem definition should be in terms of input, output, processes etc. But the user requires only a qualitative definition. Based on these views there are different types of methods for the users to explain their requirements to the analyst.

- **Kitchen sink**: In this strategy the user throws everything into the requirement definition - overstatement of needs such as an overabundance of reports, exception processing and the like. This approach usually reflects the user's lack of experience in the area.

- **The smoking strategy:** This strategy sets up a smoke screen by requesting several system features when only one or two are needed. The extra requests are used as bargaining power. This strategy usually reflects the user's experience in knowing what he/she wants. Requests have to be reduced such that they are manageable, realistic and achievable.
- **The same thing strategy:** This strategy indicates the user's laziness, lack of knowledge, or both." Give me the same thing but in a better format through the computer" is a typical statement. The analyst has very little chance of succeeding because the user is not capable of identifying the problems in detail.

  Humans have problems in specifying the information requirements. Asking the user the requirements does not often yield the correct answer because of the following human limitations.
- **Humans as information processors:** The human brain has both high capacity, long-term memory and limited capacity, short-term memory. Short-term memory affects information requirements, because the user who is interviewed has a limited number requirements that he/she thinks important. This limits processing responses. The user may think some information as important and record them in the long-term memory. These information are recalled only at the time of the interview.
- **Human bias in data selection and use:** Humans are generally biased in their selection and use of data. Users are more influenced by the current events. Thus, information that was recently discovered tends to have a greater weight than a need experienced in the past. This is called the regency effect. In some other cases, users tend to use only the information that is available in the form in which it is displayed. Thus in both cases the requirements provided by the user are biased or based upon the information that is currently available. Human problem solving behavior: Humans have a limited capacity for rational thinking. They must simplify the information to deal with it. Called as the concept of bounded rationality, it means that rationality for determining information requirements is bounded by a simplified model that may not reflect the real situation. This kind of behavior is often seen in the analysts. A successful analyst uses a general model to search for information requirements by taking into consideration the organizational and the policy issues. On the other hand, the poor analyst fails to consider these issues and concentrates on the immediate requirements.

1.7 STRATEGIES FOR DETERMINING INFORMATION REQUIREMENTS

There are three general approaches for getting information regarding the user's requirements. They are

- Asking
- Getting information from the existing information system
- Prototyping.

1.7.1 Asking

This strategy obtains information from users by simply asking them about the requirements. It assumes a stable system where users are well informed and can overcome biases in defining their problem. There are three key asking methods.

1. **Questions:** Questions may be open-ended or closed. An open-ended question allows the respondent to formulate a response. It is used when feelings or opinions are important. A closed question requests one answer from a specific set of responses. It is used when factual responses are known.
2. **Brainstorming:** Brainstorming is a technique used for generating new ideas and obtaining general information requirements. This method is appropriate for getting non-conventional solutions to problems. A guided approach to brainstorming asks each participant to define ideal solutions and then select the best one. It works well for users who have sound system knowledge but have the difficulty of accepting new ideas.
3. **Group consensus:** This method asks participants for their expectations regarding specific variables. Each participant fills out a questionnaire. The results are summarized and given to participants along with a follow-up questionnaire. Participants are invited to change their responses. The results are again summarized and given back to the participants. This debate by questionnaire continues until participants responses have converged enough. This method is advantageous than brainstorming because the participants are not subjected to psychological pressure.

1.7.2 Getting Information From Existing Information System

There are two methods in extracting information from an already existing system

1. **Data Analysis approach**
   - Determining information from an existing application is called the data analysis approach.
   - It simply asks the user what information is currently received and what other information is required.
   - It depends on the user for getting accurate information.
   - The analyst examines all reports, discusses each piece of information with the user, and determines unfulfilled information needs by interviewing the user.
   - The analyst is primarily involved in improving the existing flow of data to the user.
   - The data analysis method is ideal for making structured decisions, although it requires that users articulate their information requirements.
   - A major drawback is a lack of established rules for obtaining and validating information needs that are not linked to organizational objectives.
2. **Decision Analysis**
   - This method breaks down a problem into parts, which allows the user to focus separately on the critical issues.
   - It also determines policy and organizational objectives relevant to complete each major decision.
   - The analyst and the user then refine the decision process and the information requirements for a final statement of information requirements.
   - In this method information needs are clearly linked to decision and organizational objectives.
   - It is useful for unstructured decisions and information tailored to the user's decision-making style.
   - The major drawback is that information requirements may change when the user is promoted or replaced
   -

1.7.3 Prototyping

The third strategy for determining user information requirements is used when the user cannot establish information needs accurately before the information system is built. The reason could be the lack of an existing model on which to decide requirements or a difficulty in visualizing candidate system. In this case the user needs to consider real life systems from which adjustments can be made. This iterative approach first set up the initial requirements and builds a system to meet these requirements. As users gain experience, they request additional requirements or modifications and the process continues. Prototyping is suitable for environments where it is difficult to formulate a concrete model for defining information requirements. Prototyping strategy is appropriate for determining high uncertainty information requirement.

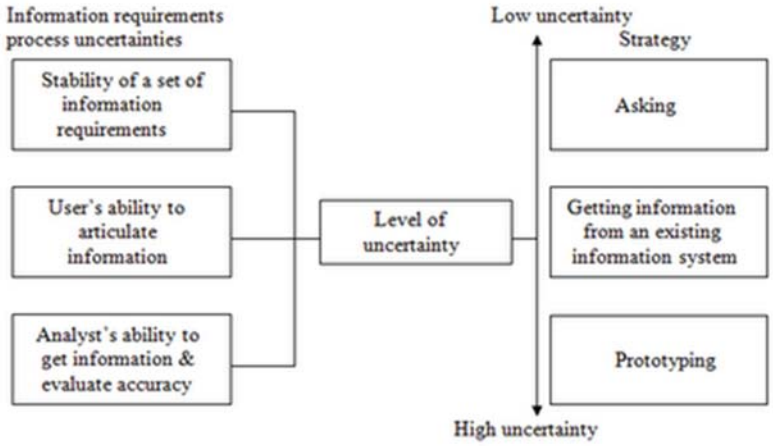


Fig 1.5: Strategies for determining requirements

1.8 FACT ANALYSIS

As data are collected, they must be organized and evaluated and conclusions drawn for preparing a report to the user for final review and approval. Some of the tools available for data organization and analysis are input/output analysis, decision tables, and structure charts.

1.8.1 Input/Output Analysis

Input/output analysis identifies the elements that are related to the inputs and output of a given system. Flow charts and data flow diagram are excellent tools for input/output analysis.

1.8.2 Flow Chart

A flow chart uses words to indicate a sequence of event. The words are enclosed in symbols linked by flow lines. Example of flow chart:

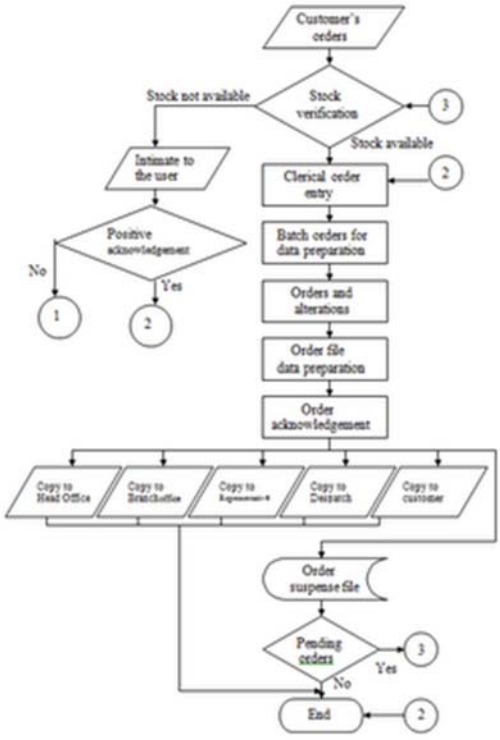


Fig 1.6: Example of Flow chart

Initially the customer orders are received. Once orders are received, stock in the warehouse is verified. If the amount of stock available is sufficient for the delivery to the customer, then the order details are recorded. Batch orders are then prepared for a day. Any alterations in the orders are also noted. The data file is prepared accordingly. Once the date of delivery is confirmed a copy of the acknowledgement is sent to the head office, branch office, the sales representative, dispatch and to the customer. Any pending orders are noted in the suspense file. If there are orders pending the whole process are repeated again else the process is ended. On the other hand if there is no sufficient stock, the information is passed on to the customer. If the customer sends a positive acknowledgement, the process continues else it ends.

Merits of a flowchart:

- They show logical interrelations clearly
- They are easy to follow
- They allow tracing of actions based on conditions

Demerits of flowchart:

- It is difficult to trace back from actions to the conditions
- Requirement to maintain a consistent level of detail to avoid confusion

1.8.3 Data flow Diagram:

Data flow diagram is graphical representation of flow of data in an information system. It is capable of depicting incoming data flow, outgoing data flow and stored data. The DFD does not mention anything about how data flows through the system.

There is a prominent difference between DFD and Flowchart. The flowchart depicts flow of control in program modules. DFDs depict flow of data in the system at various levels. DFD does not contain any control or branch elements.

- Types of DFD

  Data Flow Diagrams are either Logical or Physical.

  - **Logical DFD** - This type of DFD concentrates on the system process and flow of data in the system. For example in a Banking software system, how data is moved between different entities.
  - **Physical DFD** - This type of DFD shows how the data flow is actually implemented in the system. It is more specific and close to the implementation.

- DFD Components

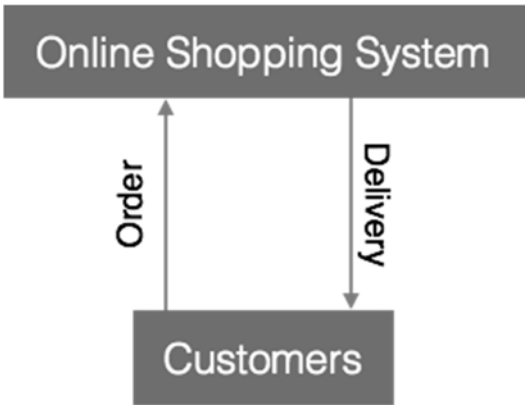
  DFD can represent Source, destination, storage and flow of data using the following set of components -

Fig. 1.7: DFD Symbols

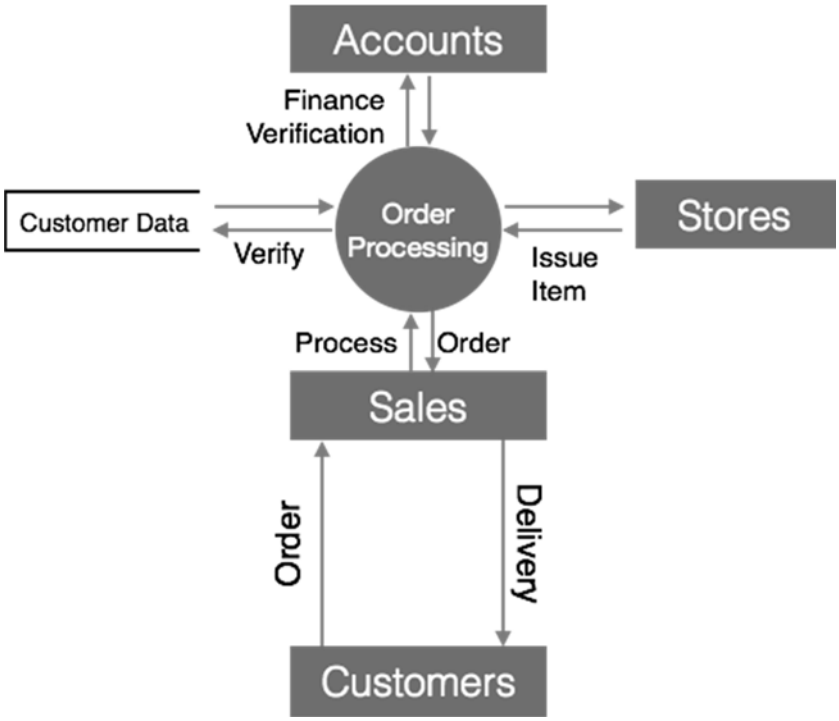
  - **Entities** - Entities are source and destination of information data. Entities are represented by rectangles with their respective names.
  - **Process** - Activities and action taken on the data are represented by Circle or Round-edged rectangles.
  - **Data Storage** - There are two variants of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

- Levels of DFD

  - **Level 0** - Highest abstraction level DFD is known as Level 0 DFD, which depicts the entire information system as one diagram concealing all the underlying details. Level 0 DFDs are also known as context level DFDs.



  - **Level 1** - The Level 0 DFD is broken down into more specific, Level 1 DFD. Level 1 DFD depicts basic modules in the system and flow of data among various modules. Level 1 DFD also mentions basic processes and sources of information.



  - **Level 2** - At this level, DFD shows how data flows inside the modules mentioned in Level 1.

Higher level DFDs can be transformed into more specific lower level DFDs with deeper level of understanding unless the desired level of specification is achieved.

1.8.4 Decision Tables

A decision table is a table of contingencies for defining a problem and the actions to be taken. It is single representation of the relationships between conditions and actions. A decision table consists of two parts: stub and entry. The stub part is divided into an upper quadrant called the condition stub and a lower quadrant called the action stub. The entry part is also divided into an upper quadrant, called the condition entry and a lower quadrant called the action entry.

For example

If the deductible has been met, the amount to be reimbursed depends on whether or not the doctor or hospital is a preferred provider. For preferred providers, doctor's office visits are reimbursed at 65% and hospital visits are reimbursed at 95%. For other providers, reimbursement is at 50% for doctor's office visits or 80% for hospital visits. No charges are reimbursed to the patient until the deductible has been met.

The four elements and their definitions are summarized in Figure 1.8.

Conditions	1	2	3	4	5	6	7	8
1. Deductible met	Y	Y	Y	Y	N	N	N	N
2. Preferred provider	Y	Y	N	N	Y	Y	N	N
3. Doctor's visit	Y	N	Y	N	Y	N	Y	N
Actions								
1. Reimburse 65%	X							
2. Reimburse 95%		X						
3. Reimburse 50%			X					
4. Reimburse 80%				X				
5. No reimbursement					X	X	X	X

Fig 1.8: Decision table

1.8.5 Structure Chart

A structure ch art is a working tool and an excellent way to keep track of the data collected for a system. There are several variations of a structure chart. Initially the analyst starts with a single input/processing/output (IPO) chart, locates the module associated with the IPO on the hierarchy chart, and identifies the data elements along the line linking the module to a higher level.

The basic building blocks which are used to design structure charts are the following:

- **Rectangular boxes**: Represents a module.
- **Module invocation arrows**: Control is passed from one module to another module in the direction of the connecting arrow.
- **Data flow arrows**: Arrows are annotated with data name; named data passes from one module to another module in the direction of the arrow.
- **Library modules**: Represented by a rectangle with double edges.
- **Selection:** Represented by a diamond symbol.
- **Repetition:** Represented by a loop around the control flow arrow.

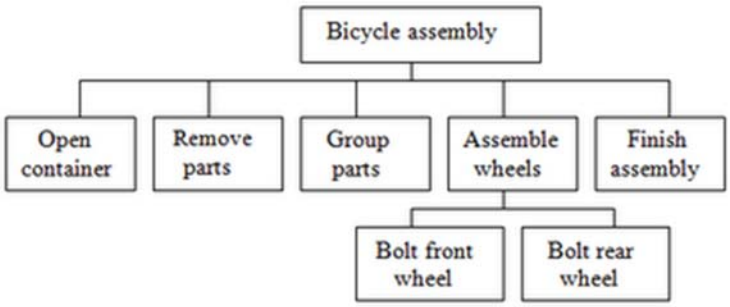


Fig 1.9: Structure Chart

Structure Chart vs. Flow Chart

We are all familiar with the flow chart representation of a program. Flow chart is a convenient technique to represent the flow of control in a program. A structure chart differs from a flow chart in three principal ways:

- It is usually difficult to identify the different modules of the system from its flow chart representation.
- Data interchange among different modules is not represented in a flow chart.
- Sequential ordering of tasks inherent in a flow chart is suppressed in a structure chart.

Check Your Progress 2

1. What are the different types of methods for the users to explain their requirements to the analyst?

   _____

   _____

   _____

2. List different asking methods.

_____

_____

_____

3. List the names of tools for fact finding.

_____

_____

_____

1.9 SUMMARY

- To ensure the success of the system, careful and often extensive planning is required.
- Planning for information systems has a time horizon and a focus dimension.
- The initial investigation has the objective of determining the validity of the user's request for a system and whether a feasibility study should be conducted.
- There are three strategies for electing information regarding the user's requirements.
    - Asking
    - Getting information from the existing information system
    - Prototyping.
- The DFD shows the flow of data, the process and the areas where they are stored.

1.10 KEYWORDS

- System Planning
- Initial Investigation
- Kitchen Sink Strategy
- Prototyping
- DFD
- Decision Table
- Structure Chart
- Flow Chart

1.11 MODEL ANSWERS

Check Your Progress 1

1. The first task in strategic planning is to set the MIS objectives and the results expected. The objectives must be set in such a way that they meet the organization's needs. Once the objectives are set, MIS policies are defined as a guideline to carry out the plan.
2. The user's request form specifies the following:
    - User-assigned title of work requested
    - Nature of work requested
    - The date the request was submitted.
    - The date the job should be completed
    - Job objectives – purpose of job requested
    - Expected benefits to be derived from proposed change

- Input/output description – quantity of inputs and outputs of proposed change
- Requester's signature, title, department and phone number
- Signature, title, department and phone number of person approving the request.

3. Once the request is approved, the following activities are carried out
   - Background investigation
   - Fact-finding and analysis
   - Presentation of results – called project proposal.

Check Your Progress 2

1. The different types of methods for the users to explain their requirements to the analyst.
   - Kitchen sink
   - The smoking strategy
   - The same thing strategy
   - Humans as information processors
   - Human bias in data selection and use
2. There are three key asking methods.
   - Questions
   - Brainstorming
   - Group consensus
3. The tools for data organization and analysis are
   - Flow Chart
   - Input/output analysis
   - Decision tables
   - Structure charts.

1.12 TERMINAL QUESTIONS

1. Why it is critical to manage system development? Explain.
2. Differentiate between managerial and operational MIS planning.
3. What important information does the user's request form provide? Why it is so important in initial investigation? Explain.
4. What do you mean by user requirements? Explain.
5. What are data flow diagram? Differentiate between DFD and structure chart.
6. Give structure of decision table. Explain it with one example.
7. What is DFD? Why is it used?
8. Define prototype. List purposes of it.
9. Explain Structure Chart.
10. Discuss the Fact Finding techniques.

UNIT - II

INFORMATION GATHERING

Unit Structure

2.1. INTRODUCTION

The key part of feasibility analysis is Information Gathering about the present system. Improper and wrong gathering of information may lead the system to the failure. The mistaken gathering of data at the initial level affects the system life cycle at every phase and finally causes the system failure. There are many tools and techniques that help to collect the correct and efficient data that help to develop the system which satisfy the needs of customer.

2.2 OBJECTIVE

After studying this Unit, you should be able to:

- The need for information gathering.
- Which kind of information needed?
- What are the sources of information?
- What are the different tools used for information gathering?
- What is on-site observation?
- How to arrange an interview?
- The types of interview.
- What are the guidelines for successful interview?
- The types of questionnaires.

2.3 KINDS OF INFORMATION REQUIRED

Information gathering is an art and science of gathering information regarding present system so that designing a new system will be easy as well as free from errors and upto the customer requirement. The

approach and manner in which information is gathered require persons with sensitivity, common sense and knowledge of what and when to gather and the channels used to secure information. Before one determines where to go for information or what tools to use, the first requirement is to figure out what information to gather. Much of the information we need to analyze relates to the organization in general, the user staff, and the workflow.

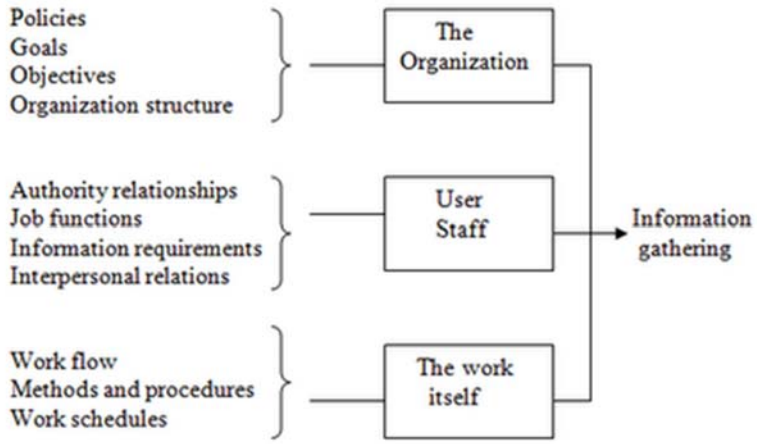


Fig 2.1: Categories of Information

2.3.1 Information of the Firm

Information about the organization's policies, goals, objectives, and structure explains the kind of environment that promotes the introduction of computer-based systems. Company policies are guidelines that determine the conduct of business. Policies are translated into rules and procedures for achieving goals. A statement of goals describes management's commitment to objectives and the direction system development will follow. Objectives are milestones of accomplishments toward achieving goals. Information from manuals, pamphlets, annual reports etc help the analyst to get an idea of the goals of the Organization.

After policies and goals are set, a firm is organized to meet these goals. The organization structure indicates management directions and orientation. The organization chart represents an achievement-oriented structure. It helps us understand the general climate in which candidate systems will be considered. In gathering information about the firm, the analyst should watch for the correspondence between what the organization claims to achieve goals and actual operations. Policies, goals, objectives and structure are important elements for analysis.

2.3.2 Information about the User Staff

Another kind of information for analysis is knowledge about the people who run the present system, their job functions and information requirements, the relationships of their jobs to the existing system and the interpersonal network that holds the user group together. The main focus is on the roles of the people, authority relationships, job status and functions, information requirements and inter personnel relations. Information of this kind highlights the organization chart and establishes a basis for determining the importance of the existing system for the organization. Thus the major focus is to find out the expectations of the people before going in for the design of the candidate system.

2.3.3 Information about the Work Flow

The workflow focuses on what happens to the data through various points in a system. This can be shown by a data flow diagram or a system flow chart.

A data flow diagram represents the information generated at each processing point in the system and the direction it takes from source to destination.

A system flowchart describes the physical system. The information available from such charts explains the procedures used for performing tasks and work schedules.

2.4 INFORMATION SOURCES

There are two main sources of information gathering:

- Primary internal sources
  - o Financial reports.
  - o Personal staff.
  - o Professional staff, EDP
  - o System documentation or manuals.
  - o The user or user staff.
  - o Reports and transaction documents.
- Primary external sources
  - o Vendors.
  - o Government documents.
  - o Newspapers and professional journals.

2.5 INFORMATION GATHERING TOOLS

Searching for information can be gathered by interviewing top-level management, middle level management and operational staff. Besides Interviews group discussions also help the analyst to gather information. It is not possible to obtain all information in a single interview; more than one interview is thus required. There are various information-gathering tools. Each tool has a special function depending on the information needed. The various information gathering tools are shown in fig.

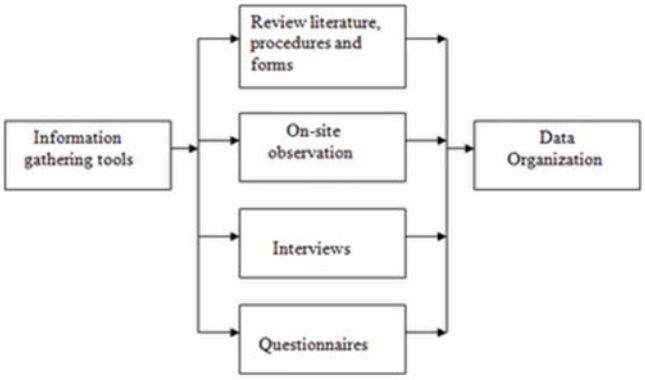


Fig 2.2: Information Gathering Methods

2.5.1 Review of Literature, Procedures and Forms

When available, all documentation (in the form of forms, records, reports, manuals, etc) is organized and evaluated. The procedures manuals contain the requirements of the system. This helps in determining to what extent the goals are met by the present system. Day to day problems may have forced changes that are not reflected in the manual. Regarding existing forms, the analyst needs to find out how they are filled out, how useful they are to the user, what changes need to be made and how easy they are to read.

Very few system problems are unique. The increasing number of software packages suggests that problem solutions are becoming standardized. Therefore a search of the literature through professional references and procedures manuals, textbooks, company studies, government publications or consultant studies may prove invaluable.

Disadvantages:

- The primary drawback of this search is time.
- Sometimes it will be difficult to get certain reports.
- Publications may be expensive and the information may be out dated due to a time lag in publication.

Procedures manuals and forms are useful sources for the analyst. They describe the format and functions of the present system. Most of the manuals include system requirements, which help to determine how well various objectives are met. Up-to-date manuals save hours of information–gathering time.

Printed forms are widely used for capturing and providing information. The objective is to understand how forms are used. The following questions may be useful:

- Who uses the forms? How important are they to the user?
- Do the forms include all the necessary information? What items should be added or deleted?
- How many departments receive the existing forms?
- How readable and easy to follow is the form?
- How does the information in the form help other users make better decisions?

2.5.2 On-Site Observations

An information gathering method used by the systems analyst is on-site or direct observation. It is the process of recognizing and noting people, objects and occurrences to obtain information. The major objective of on-site observation is to get as close as possible to the "real" system being studied. For this reason it is important that the analyst is knowledgeable about the general makeup and activities of the system. The analyst's role is that of an information seeker. As an observer, the analyst follows a set of rules.

- While making observations he/she is more likely to listen than talk and has to listen with interest when information is passed on.
- The analyst has to observe the physical layout of the current system, the location and movement of the people and the workflow.
- The analyst has to be alert to the behavior of the user staff and the people to whom they come into contact. A change in behavior provides a clue to an experienced analyst. The clue can be used to identify the problem.

The following questions can serve as a guide for on-site observations

- What kind of system is it? What does it do?

- Who runs the system? Who are the important people in it?
- What is the history of the system? How did it get to its present stage of development?
- Apart from its formal function, what kind of system is it in comparison with other systems in the organization?

Difficulties in on-site observations

- On-site observation is the most difficult fact-finding technique. It requires intrusion into the user's area and can cause adverse reaction by the user's staff if not handled properly.
- If on-site observation is to be done properly in a complex situation, it can be time-consuming.
- Proper sampling procedures must be used to identify the stability of the behavior being observed. Otherwise inferences drawn from these samples will be inaccurate and unreliable.
- Attitudes and motivations of subjects cannot be easily observed.

As an observer, the analyst follows a set of rules. While making observations, he /she is more likely to listen than talk and to listen with a sympathetic and genuine interest when information is conveyed. Four alternative observation methods are considered

1. **Natural or contrived.** A natural observation occurs in a setting such as the employee's place of work, the observer in a place like a laboratory sets up a contrived observation.
2. **Obtrusive or unobtrusive.** An obtrusive observation takes place when the respondent knows he/she is being observed; an unobtrusive observation takes place in a contrived way such as behind a one-way mirror.
3. **Direct or indirect.** A direct observation takes place when the analyst actually observes the subject or the system at work. In an indirect observation, the analyst uses mechanical devices such as cameras and videotapes to capture information.
4. **Structured or unstructured.** In a structured observation, the observer looks for and records a specific action. Unstructured methods place the observer in a situation to observe whatever might be pertinent at the time.

Any of these methods may be used in information gathering. Natural, direct, obtrusive and unstructured observations are frequently used to get an overview of an operation. Electronic observation and monitoring methods are becoming widely used tools for information gathering.

2.5.3 Interviews

On-site observation is less effective for learning about people's perceptions, feelings and motivations. The alternative is the personal interview and the questionnaire. In both the methods heavy reliance is placed on the interviewees report for information about the job, the present system, or experience. The quality of the response is judged in terms of its reliability and validity.

Reliability means that the information gathered is dependable enough to be used for making decisions about the system being studied. Validity means that the questions to be asked are worded in such a way as to elicit (obtain) the intended information. So the reliability and validity of the data collected depends on the design of the interview or questionnaire and the manner in which each instrument is administered.

The interview is a face-to-face interpersonal role situation in which a person called the interviewer asks a person being interviewed questions designed to gather information about a problem area. The interview is the oldest and most often used device for gathering information in system work. It can be used for two main purposes

- As an exploratory device to identify relations or verify information and
- To capture information, as it exists.

In an interview, since the analyst (interviewer) and the person interviewed meet face to face, there is an opportunity for flexibility in collecting information. The analyst is also in a position to observe the subject. In contrast, the information obtained through a questionnaire is limited to the subject's written responses to predefined questions

The advantages of interview are

- Its flexibility makes the interview a superior technique for exploring areas where not much is known about what questions to ask or how to formulate questions.
- It offers a better opportunity than the questionnaire to evaluate the validity of the information gathered. The interviewer can observe not only what subjects say but also how they say it.
- It is an effective technique for eliciting information about complex subjects and for probing the sentiments underlying expressed opinions
- Many people enjoy being interviewed, regardless of the subject. They usually cooperate in a study when all they have to do is talk. In contrast the percentage of returns to a questionnaire is relatively low.

The disadvantages of an interview are

1. The major drawback of the interview is the long preparation time.
2. Interview also takes a lot of time to conduct, which means time and money.

In an interview, since the analyst and the person interviewed meet face to face, there is an opportunity for greater flexibility in eliciting information. The interviewer is also in a natural position to observe the subjects and the situation to which they are responding. In contrast the information obtained through a questionnaire is limited to the written responses of the subjects to predefined questions.

- **The art of interviewing:**

  Interviewing is an art. The analyst learns the art by experience. The interviewer's art consists of creating a permissive situation in which the answers offered are reliable. Respondent's opinions are offered with no fear of being criticized by others. Primary requirements for a successful interview are to create a friendly atmosphere and to put the respondent at ease. Then the interview proceeds with asking questions properly, obtaining reliable responses and recording them accurately and completely.

- **Arranging the interview:**

  The interview should be arranged so that the physical location, time of the interview and order of interviewing assure privacy and minimal interruption. A common area that is non-threatening to the respondent is chosen. Appointments should be made well in advance and a fixed time period adhered to as closely as possible. Interview schedules generally begin at the top of the organization structure and work down so as not to offend anyone.

- **Guides to a successful interview:**

  In an interview the following steps should be taken.

  1. Set the stage for the interview.
  2. Establish rapport: put the interviewee at ease.

3. Phrase questions clearly and succinctly
4. Be a good listener; avoid arguments.
5. Evaluate the outcome of the interview.

The interviews are of two types namely structured and unstructured.

- **Structured Interview**

  Structured interviews are those where the interviewee is asked a standard set of questions in a particular order. All interviewees are asked the same set of questions.

  The questions are further divided in two kinds of formats for conducting this type of interview. The first is the open-response format in which the respondent is free to answer in his own words. An example of open-response is "Why are you dissatisfied with the current leave processing method?" The other option is of closed-response format, which limits the respondents to opt their answer from a set of already prescribed choices. An example of such a question might be "Are you satisfied with the current leave processing methods?" or "Do you think that the manual leave processing procedure be changed with some automated procedure?"

- **Unstructured Interview**

  The unstructured interviews are undertaken in a question-and-answer format. This is of a much more flexible nature than the structured interview and can be very rightly used to gather general information about the system.

  Here the respondents are free to answer in their own words. In this way their views are not restricted.

  So the interviewer gets a bigger area to further explore the issues pertaining to a problem.

2.5.4 Questionnaires

Questionnaires are another way of information gathering where the potential users of the system are given questionnaires to be filled up and returned to the analyst.

Questionnaires are useful when the analyst need to gather information from a large number of people. It is not possible to interview each individual. Also if the time is very short, in that case also questionnaires are useful. If the anonymity of the respondent is guaranteed by the analyst then the respondent answers the questionnaires very honestly and critically.

The questionnaire may not yield the results from those respondents who are busy or who may not give it a reasonable priority.

The analyst should sensibly design and frame questionnaires with clarity of its objective so as to do justice to the cost incurred on their development and distribution. Just like the interviews and on the same lines questionnaires are of two types i.e. Open-Response Based and the Closed-Response Based.

- **Open-Response Based Questionnaires**

  The objective of open-response questionnaire is to gather information and data about the essential and critical design features of the system. The open-ended question requires no response direction or specific response.

This form is also used to learn about the feelings, opinions, and experiences of the respondents. This information helps in the making the system effective because the analyst can offer subsequent modifications as per the knowledge gained.

- **Closed-Response Based Questionnaires**

  The objective of closed-response questionnaire is to collect the factual information of the system. It gives an insight in how the people dealing with the system behave and how comfortable are they with it. In this case the respondents have to choose from a set of given responses. Thus the respondent can express their liking for the most favorable one from the possible alternatives.

  The closed questions can be of various types and the most common ones are listed below.

  1. Fill-in-the-blanks.
  2. Dichotomous i.e. Yes or No type.
  3. Ranking scale questions ask the respondents to rank a list of items in the order of importance or preference.
  4. Multiple-choice questions which offer respondents few fixed alternatives to choose from.
  5. Rating scale questions are an extension of the multiple-choice questions. Here the respondent is asked to rate certain alternatives on some given scale.

- **Open Response-Based Vs Closed Response-Based**

  The basic comparison between the two can be made on the grounds of the format used. Open form offers more flexibility and freedom to the respondents whereas the closed form is more specific in nature.

  Open-ended questionnaires are useful when it is required to explore certain situation. They also require a lot of time of the analyst for evaluation.

  Closed questionnaires are used when factual information is required. Closed questions are quick to analyze but typically most costly to prepare but they are more suitable to obtain factual and common information.

  It is the job of the analyst to decide which format should be employed and what exactly should be its objective. The care should be taken to ensure that all the parts of the form are easy to understand for the respondents so that they can answer with clarity

Check Your Progress 1

1. What are the primary internal sources of information?

   _____

   _____

   _____

2. List the names of information gathering tools.

   _____

   _____

   _____

3. What are the difficulties in on-site observations?

_____

_____

_____

4. Define Interviewing.

_____

_____

_____

5. What are the types of questionnaires?

_____

_____

_____

2.6 SUMMARY

- Information is gathered from sources within the organization and from the organizations environment.
- The different sources of information are Eternal source like vendor, government document and professional journals. The primary internal sources are reports, personnel, system documentation and users.
- Information gathering tools are very useful during the analysis phase. There are many information gathering techniques available like interviews, questionnaires, on site observation etc.
- The major objective of on-site observation is to get as close as possible to the "real" system being studied.
- The interview is a face-to-face interpersonal role situation in which a person called the interviewer asks a person being interviewed questions designed to gather information about a problem area.
- The interviews are of two types namely structured and unstructured.
- Questionnaires are another way of information gathering where the potential users of the system are given questionnaires to be filled up and returned to the analyst.
- Each of the technique has its own format, suitability, merits and demerits. The analyst has to make a judgment on which format to be used to collect what information and who should be the respondents.

2.7 KEYWORDS

- Interview
- On-site observation
- Natural observation
- Questionnaires
- Structured interview
- Unstructured interview
- Reliability

2.8 MODEL ANSWERS

Check Your Progress 1

1. The primary internal sources of information are
   - o Financial reports.
   - o Personal staff.
   - o Professional staff, EDP
   - o System documentation or manuals.
   - o The user or user staff.
   - o Reports and transaction documents.
2. There are many information gathering techniques available like
   - o Review of Literature, Procedures and Forms
   - o On-site observation
   - o Interviews
   - o questionnaires
3. The main difficulties in on-site observations are
   - o On-site observation is the most difficult fact-finding technique. It requires intrusion into the user's area and can cause adverse reaction by the user's staff if not handled properly.
   - o If on-site observation is to be done properly in a complex situation, it can be time-consuming.
   - o Proper sampling procedures must be used to identify the stability of the behavior being observed. Otherwise inferences drawn from these samples will be inaccurate and unreliable.
   - o Attitudes and motivations of subjects cannot be easily observed.
4. A data gathering or data verification approach, talking with people in an organized manner and with a purpose is called Interviewing.
5. Questionnaires are of two types
   - o Open-Response Based
   - o Closed-Response Based.

2.8TERMINAL QUESTIONS

1. What are the methods of gathering information
2. What do you mean by interview?
3. What type information is often best obtained through interview?
4. Which method does an analyst use to gather data?
5. What is information gathering?
6. Make a list of do's and don'ts that you should follow when interviewing a user.
7. List different types of questionnaire formats.
8. What are the differences between open and closed-form questionnaires? For what types of information is each form most useful?
9. Explain the differences between
   (a) structured and unstructured interviewing and
   (b) open-ended and closed questions. Give an example of each.
10. Summarize the advantages and limitations of interviews and questionnaires.

UNIT - III

TOOLS OF STRUCTURED ANALYSIS

Unit Structure

3.1 INTRODUCTION

System analysis is about understanding situations, not solving problems. Effective analysts therefore emphasize investigation and questioning to learn how a system currently operates and to identify the requirements users have for a new or modified one. Only after analysts fully understand the systems are they able to analyze it and assemble recommendations for systems design.

The system analyst needs something analogous to the architect's blueprint as a starting point for system design. It is a way to focus on functions rather than physical implementation. One such tool is structured analysis.

Tools of structured analysis overcome the drawbacks of traditional tools used for data gathering. Structured tools such as Data Flow Diagram, Data Dictionary, and Structure English provide alternative ways to design candidate system. Some real-life applications require the combination of both structured tools as well as traditional tools.

3.2 OBJECTIVE

After studying this Unit, you should be able to:

- What is structured analysis?
- What tools are used in structured analysis?
- How to construct a data flow diagram.
- What are the advantages and uses of a data dictionary and structured English.
- The elements and construction of decision trees and decision tables.

3.3 TOOLS OF STRUCTURED ANALYSIS

Structured analysis is a set of techniques and graphical tools that allow the analyst to develop a new kind of system specifications that are easily understandable to the user. Analysts work primarily with their wits, pencil, and paper. Most of them have no tools. The traditional approach focuses on cost/benefit and feasibility analysis, project management, hardware and software selection and personnel considerations. In contrast, structured analysis considers new goals and structured tools for analysis. The new goals specify the following:

- Use graphics wherever possible to help communicate better with the user.
- Differentiate between logical and physical systems.
- Build a logical system model to familiarize the user with system characteristics and interrelationships before implementation.

The structured tools focus on the listed earlier- essentially the date flow diagram data dictionary, structured English, decision trees, and decision tables. The objective is to build a new document, called system specifications. This document provides the basis for design and implementation.

3.4 THE DATA FLOW DIAGRAM (DFD)

The DFD was first developed by Larry Constantine as a way of expressing system requirements in a graphical from; this led to a modular design.

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. It describes the systems data and how the processes transform the data in a graphical manner. Data flow diagrams can be used to provide a clear representation of any business function. It starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. It uses a top-down approach to show all the levels of the functions of the system.

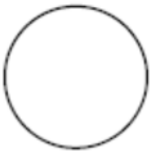
A graphical tool used to describe and analyze the movement of data through a system including the processes, stores of data and delays in the system. DFD are the central tool and the basis from which other components are developed. The transformation of data from input to output, through process, may be described logically and independently of the physical components are called logical DFD. In contrast, physical DFD show the actual implementation and the movement of data between people, departments and workstations.

Logical DFD can be completed using only four simple notations. The symbols are as follow

1. A **SQUARE** defines a source or destination of system data.

2. An **ARROW** identifies data flow or data in motion. It is a pipeline through which information flow.

3. A **CIRCLE** or a **BUBBLE** (Some people use an over bubble) represents a process transforms in coming data flow into outgoing data flow.



4. An **OPEN RECTANGLE** is a data store or data at rest or a temporary rest repository of data.

Each component in a DFD is labeled with a descriptive name. Process name are identified with a number. The number assigned to a specific process does not represent the sequence of process. It is strictly for identification and will take on added value when we study the components that make up a specific process.

3.4.1 Constructing a DFD

Several rules of thumb are used in drawing DFDs.

- Arrows should not cross each other.
- Squares, circles, and files must bear names.
- Decomposed data flows must be balanced (all data flows on the decomposed diagram must reflect flows in the original diagram).
- No two data flows, squares, or circles can have the same name.
- Draw all data flows around the outside of the diagram.
- Choose meaningful names for data flows, processes, and data storks. Use strong verbs followed by nouns.
- Control information such as record counts, passwords, and validation requirements are not pertinent to a data-flow diagram.
- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and from left to right. Data traditionally flow from the source (upper left corner) to the destination (lower right corner), although they may flow back to a source. One way to indicate this is to draw a long flow line back to the source. An alternative way is to repeat the source symbol as a destination. Since it is used more than one in the DFD, it is marked with a short diagonal in the lower right corner.

- When a process is exploded into lower- level details, they are numbered.
- The names of data stores, sources and destinations are written in capital letters. Process and data flow names have the first letter of each word capitalized.

For example DFD for payroll management system.

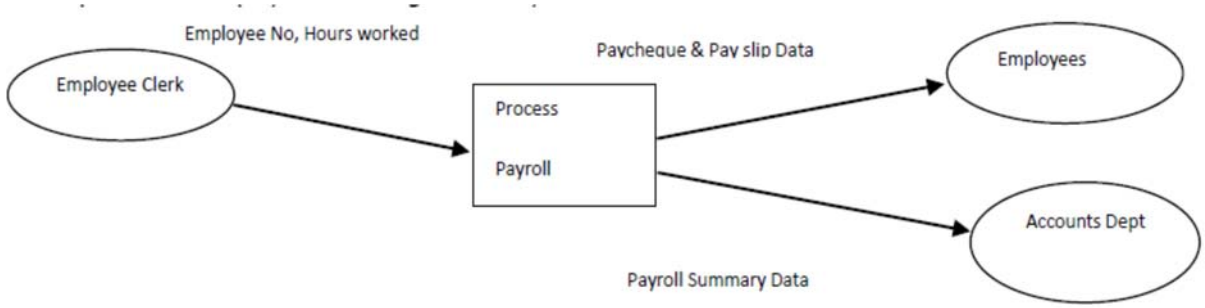


Fig 3.1: An example DFD

3.4.2 Type of DFD

There are two types of DFD:

1. **Physical DFD: –** The data flow diagrams which represent the model of the current system are known as physical DFD. These diagrams are drawn, when the analyst studies the current working system in detail.
2. **Logical DFD: –** The data flow diagrams which represent the model of the proposed system are known as the logical DFD. These diagrams are drawn from the Physical DFD.

Each of DFD can be further categorized in to different levels like zero level (context diagram), first level, second level and third level. Each higher level DFD is drawn by adding more details to each process or lower level.

3.4.3 Levels of DFD

- **Level 0 (Context diagram)** - Highest abstraction level DFD is known as Level 0 DFD, which depicts the entire information system as one diagram concealing all the underlying details. Level 0 DFDs are also known as context level DFDs.

  This DFD provides an overview of the data entering and leaving the system. It also shows the entities that are providing or receiving that data. These correspond usually to the people that are using the system we will develop. The context diagram helps to define our system boundary to show what is included in, and what is excluded from, our system. The diagram consists of a rectangle representing the system boundary, the external entities interacting with the system and the data which flows into and out of the system. Figure gives an example of a context diagram.

- **Level 1** - The Level 0 DFD is broken down into more specific, Level 1 DFD. Level 1 DFD depicts basic modules in the system and flow of data among various modules. Level 1 DFD also mentions basic processes and sources of information.

- **Level 2** - At this level, DFD shows how data flows inside the modules mentioned in Level 1.

3.5 DECISION TREES

Decision tree is a graphical representation that presents conditions and actions sequentially. It is a method of showing the relationship of each condition and its permissible actions. The route of the tree is starting point and it proceeds towards the various possible nodes.

The size of the tree will depend upon the number of conditions and actions. Each condition is expressed in two ways True/False or Yes/No.

Advantages of Decision tree

1. It expresses the logic of if then else in pictorial form.
2. It is useful to express the logic when a value is variable or an action is dependent on nested decision i.e. the outcome of another decision.
3. It helps the analyst to identify the actual decision to be made.
4. It is used to verify logic and problems that involve a few complex decision and limited number of actions.

Disadvantages of Decision tree

1. The lack of decision tree is that there is absence of information in its format to take what other combinations of conditions to test.
2. A large number of branches with many paths will confuse rather than help in analysis.

Example:

Draw a decision tree for policy followed by a company in giving discount to its customers as follows.

1. If transaction is on credit and customer's record is good the order will be accepted but do not give any discount.
1. If customer's record is not good do not accept any order.
2. If transaction is on cash and sells amount is more than 100 rupees discount will be given 20%.
3. If transaction is on cash and sells amount is between 50 & 100 rupees than discount will be given 10%.
4. If transaction is on cash and sells amount is less than 50 rupees, order is accepted but no discount will be made.

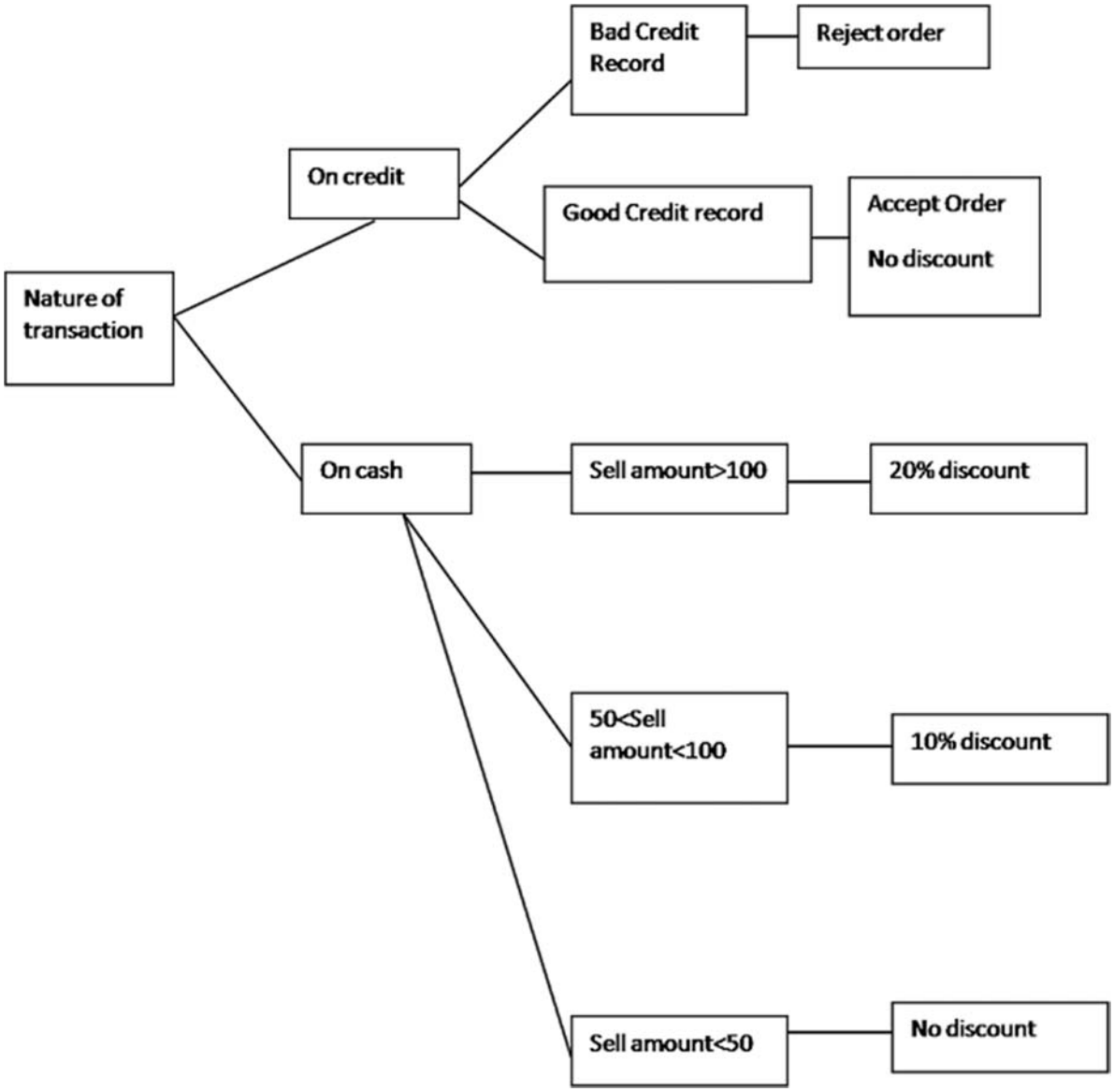


Fig 3.2: Decision tree

3.5.1 Decision Tree Characteristics

A decision tree is a diagram that presents conditions and actions sequentially and thus shows which conditions to consider first, which second, and so on. It is also a method of showing the relationship of each condition and its permissible actions. The diagram resembles branches on a tree, hence the name.

The root of the tree, on the left of the diagram, is the starting point of the decision sequence. The particular branch to be followed depends on the conditions that exist and the decision to be made. Progression from left to right along a particular branch is the result of making a series of decisions. Following each decision points is the next set of decision to be considered. The nodes of the tree thus represent conditions and indicate that a determination must be made about which condition exists before the next path can be chosen. The right side of the tree lists the actions to be taken depending on the sequence of conditions that is followed.

3.5.2 Using Decision Trees

Developing decision trees is beneficial to analysts in two ways. First of all, the need to describe conditions and actions forces analysts to formally identify the actual decision that must be made. It becomes difficult for them to overlook and integral step in the decision process, whether it depends on quantitative or non-quantitative variables. For instance, bookstores get a trade discount of 25%; for orders from libraries and individuals, 5% allowed on orders of 6-19 copies per book title; 10% on orders for 20-49 copies per book title; 15% on orders for 50 copies or more per book title.

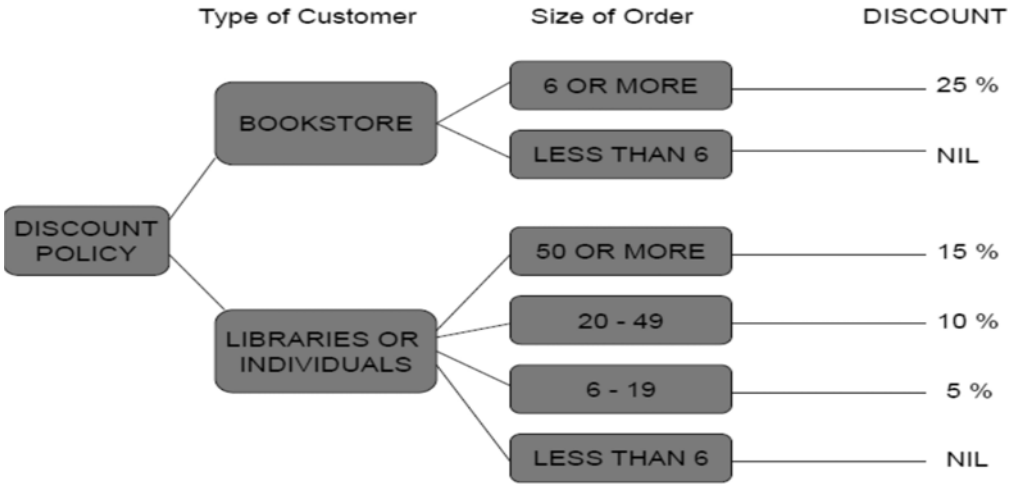


Fig 3.3: Representation of decision tree

3.5.3 Avoiding problems with Decision Trees

Decision trees may not always be the best tolls for decision analysis. A decision tree for a complex system with many sequences of steps and combinations of conditions will be unwieldy. A large number of branches with many paths through them will could rather than aid analysis. The analyst may not be able to determine which business policies or practices guide the formulation of specific decisions. Where these problems arise, decision tables should be considered.

3.6 STRUCTURED ENGLISH

Structured English is one more tool available to the analyst. It comes as an aid against the problems of ambiguous language in stating condition and actions in decisions and procedures. Here no trees or tables are employed; rather with narrative statements a procedure is described. Thus it does not show but states the decision rules. The analyst is first required to identify the conditions that occur in the process, subsequent decisions, which are to be made and the alternative actions to be taken.

Here the steps are clearly listed in the order in which they should be taken. There are no special symbols or formats involved unlike in the case of decision trees and tables, also the entire procedure can be stated quickly as only English like statements are used.

Structured English borrows heavily from structured programming as it uses logical construction and imperative statements designed to carry out instructions for actions. Using "IF", "THEN", "ELSE" and "So" statement decisions are made. In this structured description terms from the data dictionary are widely used which makes the description compact and straight.

3.6.1 Developing Structured Statements

Three basic types of statements are employed to describe the process.

1. **Sequence Structures** - A sequence structure is a single step or action included in a process. It is independent of the existence of any condition and when encountered it is always taken. Usually numerous such instructions are used together to describe a process.

2. **Decision Structures** - Here action sequences described are often included within decision structures that identify conditions. Therefore these structures occur when two or more actions can be taken as per the value of a specific condition. Once the condition is determined the actions are unconditional.

3. **Iteration Structures**- these are those structures, which are repeated, in routing operations such as DO WHILE statements.

The decision structure of example discussed in previous sections

IF order is from Bookstore

And IF order is for 6 copies or more per book title

THEN: Discount is 25%

ELSE (order is for fewer than 6 copies per book title)

SO: no discount is allowed

ELSE (order is from libraries or individuals)

ELSE (order is from libraries or individuals)

SO-IF order is for 50 copies or more per book title

Discount is 15%

ELSE IF order is for 20 to 49 copies per book title

Discount is 10%

ELSE IF order is for 6 to 19 copies per book title

Discount is 5%

ELSE (order is for less than 6 copies per book order)

SO: no discount is allowed

3.7 DECISION TABLES

A major drawback of a decision tree is the lack of information in its format to tell us what other combinations of conditions to test. This is where the decision table is useful. A decision table is a table of contingencies for defining a problem and the actions to be taken. It is single representation of the relationships between conditions and actions.

A decision table consists of two part: stub and entry. The stub part is divided into an upper quadrant called the condition stub and a lower quadrant called the action stub. The entry part is also divided into

an upper quadrant, called the condition entry and a lower quadrant called the action entry. The four elements and their definitions are summarized in Figure.

Elements	Location	Definition
Conditions Stub	Upper left quadrant	Sets forth in question form the condition that may exist.
Action Stub	Lower left quadrant	Outlines in narrative form the action to Be taken to meet such condition.
Condition entry	Upper right quadrant	Provides answers to questions asked in the condition stub quadrant.
Action entry	Lower right quadrant	Indicates the appropriate action resulting from the answers to the conditions in the condition entry quadrant

The decision table of example discussed in previous sections

Condition Stub		Condition Entry					
		1	2	3	4	5	6
	Customer is Bookstore	Y	Y	N	N	N	N
IF	Order size 6 copies or more ?	Y	N	N	N	N	N
(Condition)	Customer Librarian or Individual			Y	Y	Y	Y
	Order-size 50 copies or more ?			Y	N	N	N
	Order-size 20-49 copies ?				Y	N	N
	Order-size 6-19 copies ?					Y	N
Then	Allow 25% Discount	X					
(action)	Allow 15% Discount			X			
	Allow 10% Discount				X		
	Allow 5% Discount					X	
	No Discount allowed		X				X
	Action Stub	**Action Entry**					

3.8 DATA DICTIONARY

When the volume of data is very large, it is very much difficult for analyst to manage the data definitions. If the information system is very big, then more than one person are working on the same data, at that time, any data defined by any person, can be used by the other person, hence they need the definition or description of the data.

Data dictionaries are an integral component of structured analysis; it provides additional information about the system.

A data dictionary is a catalog – a repository – of the elements in a system. As the name suggest, these elements center around data and the way they are structured to meet user requirements and organization needs. It contains a list of all the elements composing the data flowing through a system. The major elements are data flows, data stores and processes. The data dictionary stores details and descriptions of these elements. If data dictionary is developed properly, then any data related questions – answer can be extracted from data dictionary.

It is developed during data flow analysis and assists the analysts. The stored details are used during system design.

3.8.1 Importance of Data dictionary:

- **To manage the detail in large system:** Large system has huge volumes of data flowing through them in the form of documents, reports and even conversations. The analyst should remember all the definition for letter use, the best organized and most effective analyst use automated data dictionary designed specifically for systems analysis and design.
- **To communicate a common meaning for all system elements:** Data dictionary assists in ensuring common meanings for system elements and activities. It records additional details about the data flow in a system so that all persons involved can quickly look up the description of data flows, data stores, or processes.
- **To document the features of the system:** It includes the parts or components and characteristics that distinguish each. Sometimes we also need to know under what circumstances each process is performed and how often the circumstance occurs. Once a feature have been articulated and recorded, all participants in the project will have a common source for information about the system.
- **To facilitate analysis of the details in order to evaluate characteristics and determination when system changes should be made:** It is used to determine whether new features are needed in a system or whether changes of any type are in order.
- **Locate errors and omissions:** It is also used to locate errors in the system descriptions. Conflicting data flow descriptions, processes that neither receive input nor generate output, data store that are never updated etc. indicate incomplete or incorrect analysis. Automatic data dictionary system have feature that will detect these difficulties to present in report.

3.8.2 What does a data dictionary record?

The dictionary contains two types of descriptions for the data flowing through the system (i.e. data elements, data structure)
The most fundamental data level is the data element. It consists of data name, data descriptions, aliases, length, data values etc. Data elements are building blocks for all other data in the system.

Data structure is a set of data items that are related to one another and that collectively describe components of the system.

In addition, the data dictionary also gives information about data element/data structure, process list, cross-reference checking, error detection.

Data dictionary are an essential aspect of data flow analysis and requirement determination. They should be used in conjunction. They should be used in conjunction with logic and process definitions.

3.9 PROS AND CONS OF EACH TOOL

Which tool is the best depends on a number of factors: the nature and complexity of the problem the number of actions resulting from the decision, and the ease of use. In reviewing the benefits and limitations of each tool, we come to the following conclusion:

- The primary strength of the DFD is its ability to represent data flows. It may be used at high or low level of analysis and provides good system documentation. However, the tool only weakly shows input and output detail. The user often finds it confusing initially.
- The data dictionary helps the analyst simplify the structure for meeting the data requirements of the system. It may be used at high or low levels of analysis, but it does not provide functional details, and it is not acceptable to many nontechnical users.
- Structured English is best used when the problem requires sequences of actions with decisions.
- Decision trees are sued to verify logic and in problems that involve a few complex decisions resulting in limited number of actions.
- Decision trees and decision tables are best suited for dealing with complex branching routines such as calculating discounts or sales commissions or inventory control procedures.

Given the pros and cons of structured tools, the analyst should be trained in the sue of various tools for analysis and design He/She should use decision table and structured English to get to the heart of complex problems. A decision table is perhaps the most useful tool for communicating problem details to the user.

The major contribution of structured analysis to the system development life cycle is producing a definable and measurable document – the structured specification. Other benefits include increased user involvement, improved communication between user and designer, reduction of total personnel time, and fewer "kinks" during detailed design and implementation. The only drawback is increased analyst and user time in the process. Overall the benefits outweigh the drawbacks, which make-structured analysis tools viable alternatives in system development.

Check Your Progress 1

6. What is DFD?

_____

_____

_____

7. What are the Tools for Structured Analysis?

_____

_____

_____

8. Why is a data dictionary necessary?

_____

_____

_____

9. Explain Structured English.

_____

3.10 SUMMARY

- Structured analysis is a set of techniques and graphical tools that allow the analyst to develop a new kind of system specifications that are easily understandable to the user.
- The DFD was first developed by Larry Constantine as a way of expressing system requirements in a graphical from; this led to a modular design.
- A data dictionary is a structured repository of data about data. It offers primary advantages of documentation and improving analyst/user communication by establishing consistent definitions of various elements, terms and procedures.
- A decision tree sketches the logical structure based on some criteria. It is easy to construct, read, and update. A decision tree is a diagram that presents conditions and actions sequentially and thus shows which conditions to consider first, which second, and so on.
- A decision table is a table of contingencies for defining a problem and the actions to be taken. It is single representation of the relationships between conditions and actions.
- Structured English is best used when the problem requires sequences of actions with decisions.

3.11 KEYWORDS

- Structured Analysis
- DFD
- Decision Tree
- Decision Table
- Data Dictionary
- Structured English

3.12 MODEL ANSWERS

Check Your Progress 1

4. A DFD has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design
5. The various tools for structured analysis are
    - o Data Flow Diagram (DFD)
    - o Data Dictionary
    - o Decision Table
    - o Decision Tree
    - o Structured English
6. Data Dictionary is necessary for

- To manage the detail in large system.
- To communicate a common meaning for all system elements.
- To document the features of the system.
- To facilitate analysis of the details in order to evaluate characteristics and determination when system changes should be made.
- Locate errors and omissions.

7. Structured English contains the steps that are clearly listed in the order in which they should be taken. There are no special symbols or formats involved unlike in the case of decision trees and tables, also the entire procedure can be stated quickly as only English like statements are used.

3.13 TERMINAL QUESTIONS

1. Give structure of decision table. Explain it with one example.
2. Convert the example of decision table discussed in class into structured English.
3. What is DFD? What is Context Level DFD? Explain with an example of a context level DFD.
4. What is FDD? Why is it used?
5. Explain Structure English.
6. Difference between Logical DFD and Physical DFD.
7. What is DFD? List out the Symbol used in DFD.
8. What are the main advantages of creating a data dictionary?
9. What data about a data element is stored in a data dictionary?
10. Write short note on the following.
    1. DFD
    2. Decision tree
    3. Data dictionary
11. Construct a data flow diagram (level 0 and 1) for a library management system?
12. Differentiate between a decision table and a decision tree. Prepare a sample data dictionary for an employee management system.
13. "A data dictionary is a structured repository of data about data." Discuss

UNIT - IV

FEASIBILITY STUDY

UNIT STRUCTURE

4.1 INTRODUCTION

A feasibility study is an evaluation of a proposal designed to determine the difficulty in carrying out a designated task. Generally, a feasibility study precedes technical development and project implementation. In other words, a feasibility study is an evaluation or analysis of the potential impact of a proposed project.

The objective of feasibility study is not to solve the problem but to acquire a sense of its scope. During the study the problem definition is crystallized and aspects of the problem to be included in the system are determined. Consequently, costs and benefits are estimated with greater accuracy at this stage.

The result of the feasibility study is a formal proposal. This is simply a report- a formal document detailing the nature and scope of the proposed solution. The proposal summarizes what is known and what is going to be done.

4.2 OBJECTIVE

After studying this Unit, you should be able to:

- What key considerations are involved in feasibility study?
- How the different types of feasibility will be done.

- What are cost and benefit categories?
- How to identify and classify cost and benefits?
- What are various evaluation methods for cost/benefit analysis?

4.3 FEASIBILITY STUDY

Feasibility is the measure of how beneficial the development of information system would be to an organization. Feasibility analysis is a management-oriented activity, by which alternative solutions are obtained and a specific solution is recommended for optimal performance.

The objective of a feasibility study is to provide the management with enough information to decide:

- Whether the project can be done?
- Whether the final product will benefit its intended users?
- What are the possible alternatives solutions?
- What are the costs and the benefits associated with each of the alternatives?
- Is there a preferred alternative?

After a feasibility study, management makes a go/no-go decision. During the feasibility study look at:

- Present system- users, policies, functions, objectives.
- Problems with the present system- inconsistencies, functional inadequacies,
- And performance issues.
    - Objectives and other requirements for the new system- what needs to be changed?
    - Constraints including nonfunctional requirements on the system
    - Possible alternatives
    - Advantages and disadvantages of the alternatives.

Feasibility analysis includes study of technical feasibility, economic feasibility and operational feasibility, social feasibility, management feasibility, legal feasibility, and time feasibility.

4.3.1 Technical Feasibility:

The assessment is based on an outline design of system requirements in terms of Input, Processes, Output, Fields, Programs, and Procedures. This can be quantified in terms of volumes of data, trends, frequency of updating, etc. in order to estimate whether the new system will perform adequately or not.

Technological feasibility is carried out to determine whether the company has the capability, in terms of software, hardware, personnel and expertise, to handle the completion of the project. Whether the project can be done with the available equipment, technology and personnel. Technical feasibility analysis questions whether the hardware, software, and other technologies needed for the project to exist in the organization or it has to be acquired or it needs to be developed?

4.3.2. Economic feasibility:

Economic analysis is the most frequently used method for evaluating the effectiveness of a new system. More commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. An entrepreneur must accurately weigh the cost versus benefits before taking an action.

Cost-based study: It is important to identify cost and benefit factors, which can be categorized as follows:

1. Development costs;
2. Operating costs.

This is an analysis of the costs to be incurred in the system and the benefits derivable out of the system.

Time-based study: This is an analysis of the time required to achieve a return on investments. The benefits derived from the system. The future value of a project is also a factor.

4.3.3 Operational Feasibility

Proposed projects are beneficial only if they can be turned into information systems that will meet the organization's operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to implementation? Here are questions that will help test the operational feasibility of project:

- Is there sufficient support for the project from management? From users? If the current system is well liked and used to the extent that persons will not be able to see reasons for a change, there may be resistance.
- Are current business methods acceptable to the users? If they are not, users may welcome a change that will bring about a more operational and useful system.
- Have the users been involved in the planning and development of the project? Early involvement reduces the chances of resistance to the system and change in general and increases the likelihood of successful projects.
- Will the proposed system cause harm? Will it produce poorer result in any respect or area? Will loss of control result in any area? Will accessibility of information be lost? Will individual performance be poorer after implementation than before?

Will customers be affected in an undesirable way? Will the system slow performance in any areas?

Issues that appear to be relatively minor in the beginning have ways of growing into major problems after implementation. Therefore, all operational aspects must be considered carefully.

4.3.4 Social feasibility

Social feasibility is a determination of whether a proposed project will be acceptable to the people or not. This determination typically examines the probability of the project being accepted by the group directly affected by the proposed system change.

4.3.5 Management feasibility

It is a determination of whether a proposed project will be acceptable to management. If management does not accept a project or gives a negligible support to it, the analyst will tend to view the project as a non-feasible one.

4.3.6 Legal feasibility:

Determines whether the proposed system conflicts with legal requirements, e.g. a data processing system must comply with the local Data Protection Acts.

4.3.7 Time feasibility

Time feasibility is a determination of whether a proposed project can be implemented fully within a stipulated time frame. If a project takes too much time it is likely to be rejected.

4.4 STEPS IN FEASIBILITY ANALYSIS

Feasibility analysis involves eight steps:

1. Form a project team and appoint project leader.
2. Prepare system flowcharts.
3. Enumerate potential candidate systems.
4. Describe and identify characteristics of candidate system.
5. Determine and evaluate performance and cost effectiveness of each candidate system.
6. Weight system performance and cost data.
7. Select the best candidate system.
8. Prepare and report final project directive to management

Now, we discuss each of them

1. **Form a project team and appoint a project leader**

   The first step involves forming a project team. The team consists of analysts and user staff. In many cases, an outside consultant and an information specialist join the team until the job is completed.

   Projects are planned to occupy a special time, ranging from several weeks to months. The senior systems analyst is generally appointed as project leader. The appointment is temporary, lasting as long as the project. Regular meetings take place to keep up the momentum and accomplish the mission – selection of the best candidate system. A record is kept of the progress made in each meeting.

2. **Prepare system flowcharts**

   The next step in feasibility study is to prepare generalized system flowcharts for the system. Information oriented charts and data flow diagrams prepared in the initial investigation are reviewed at this time. The chart brings up the importance of inputs; outputs and data flow among key points in the existing system.

3. **Enumerate potential candidate systems.**

   This step identifies the candidate systems that are capable of producing the outputs included in the generalized flowcharts. This requires a transformation from logical to physical system models. Another aspect of this step is consideration of the hardware that can handle the total system requirements

4. **Describe and identify characteristics of candidate system.**

   In this step, the analysis is mainly based on what each candidate system can and cannot do. For determining this, technical knowledge and expertise in the hardware/software area are critical.

5. **Determine and evaluate performance and cost effectiveness of each candidate system.**

   Here the analyst has to determine and evaluate the performance and cost of the candidate system.

Evaluation for both design and implementation is performed here. It includes user training, updating the physical facilities and documenting etc.

6. **Weight system performance and cost data**

   According to the performance and cost of the candidate system, some weight is given to each alternative of the system. Then the candidate system with the highest total score is selected.

   The procedure for weighting candidate system is

   - Assign a weighting factor to each evaluation criterion based on the criterion's effect on the success of the system.
   - Assign a quantitative rating to each criterion's qualitative rating.
   - Multiply the weight assigned to each category by the relative rating to determine the score.
   - Sum the score column for each candidate system.

7. **Select the best candidate system.**

   The system with the highest total score is judged as the best system. This assumes the weighting factors are fair and the rating of each evaluation criterion is accurate.

8. **Prepare and report final project directive to management**

   After feasibility study, a document called feasibility report is prepared and is directed to the management. The report is a formal document for management use; it should be brief, and sufficiently nontechnical to be understandable.

   The report contains the following sections.

   1. **Cover letter-**It represents the report and briefly indicates to management the nature, general findings and recommendations to be considered.
   2. **Table of contents-** specifies the location of the various parts of the reports.
   3. **Overview**- is a narrative explanation of the purpose and scope of the project, the reason for undertaking the feasibility study and the departments involved or affected by the candidate system.
   4. **Detailed findings**- Outline the methods used in the present system. The System's effectiveness and efficiency as well as operating costs are emphasized. This section also provides a description of the objectives and general procedures of the candidate system.
   5. **Economic justification**- details point by point cost comparisons and preliminary cost estimates for the development and operation of the candidate system.
   6. **Recommendations and conclusions-** suggest to management the most beneficial and cost effective system. They are written only as a recommendation, not a command. Following the recommendations, any conclusions from the study may be included.
   7. **Appendixes** -document all memos and data compiled during the investigation. They are placed at the end of the report for reference.

Oral presentation

The project leader or analyst is expected to give an oral presentation to the end user. The most critical requirements for the analyst who gives the oral presentation are

1. Communication skills and knowledge about the candidate system that can be translated into language understandable to the user,

2. The ability to answer questions, clarifies issues, maintain credibility, and pick up on any new ideas or suggestions.

4.4.1 Other Feasibility Factors

- **Market and real estate feasibility:**

  Market Feasibility Study typically involves testing geographic locations for a real estate development project, and usually involves parcels of real estate land. Developers often conduct market studies to determine the best location within a jurisdiction, and to test alternative land uses for given parcels. Jurisdictions often require developers to complete feasibility studies before they will approve a permit application for retail, commercial, industrial, manufacturing, housing, office or mixed-use project. Market Feasibility takes into account the importance of the business in the selected area.

- **Resource feasibility:**

  This involves questions such as how much time is available to build the new system, when it can be built, whether it interferes with normal business operations, type and amount of resources required, dependencies, etc. Contingency and mitigation plans should also be stated here.

- **Cultural feasibility**

  In this stage, the project's alternatives are evaluated for their impact on the local and general culture. For example, environmental factors need to be considered and these factors are to be well known. Further an enterprise's own culture can clash with the results of the project.

Check Your Progress 1

1. List all types of feasibility tests.

   _____

   _____

   _____

2. Name some points which are to be considered in the operational feasibility.

   _____

   _____

   _____

3. Write the eight steps of feasibility analysis.

   _____

   _____

   _____

4.5. COST AND BENEFIT ANALYSIS

Cost and benefit analysis is a procedure by which we can find out the estimate of gross benefit of a new system specification. Or in other words it is the method by which we find and determine the increased operating costs associated with gross benefits. To find out gross benefit we must first determine what kinds of cost and benefit should be taken in mind.

4.5.1 Cost:

In developing cost estimates for a system, we need to consider several cost elements. Among them is hardware, personnel, facility, operating and supply costs.

1. **Hardware costs** relate to the actual purchase or lease of the computer and peripherals (for example, printer, disk drive, tape unit). Determining the actual cost of hardware is generally more difficult when the system is shared by various users than for a dedicated stand- alone system. In some cases, the best way to control for this cost is to treat it as an operating cost.
2. **Personnel costs** include EDP staff salaries and benefits (health insurance, vacation time, sick pay, etc.) as well as pay for those involved in developing the system Costs incurred during development of a system are one – time costs and are labeled developmental costs. Once the system is installed, the costs of operating and maintaining the system become recurring costs.
3. **Facility costs** are expenses incurred in the preparation of the physical site where the application or the computer will be in operation. This includes wiring, flooring, acoustics, lighting and air conditioning. These costs are treated as onetime costs and are incorporated into the overall cost estimate of the candidate system.
4. **Ongoing support and growth costs** include all costs associated with the day-to-day operation of the system; the amount depends on the number of shifts, the nature of the applications, and the caliber of the operating staff. There are various ways of covering operating costs. One approach is to treat operating costs as overhead. Another approach is to charge each authorized user for the amount of processing they request from the system. The amount charged is based on computer time, staff time and volume of the output produced. In any case, some accounting is necessary to determine how operating costs should be handled.
5. **Supply costs** are variable costs that increase with increased use of paper, ribbons, disks, and the like. They should be estimated and included in the overall cost of the system.

4.5.2 Benefits

Since cost plays quite an important role in deciding the new system, it must be identified and estimated properly. Costs vary by type and consist of various distinct elements. Benefits are also of different type and can be grouped on the basis of advantages they provide to the management. The benefits of a project include four types:

1. Cost saving benefits.
2. Cost avoidance benefits.
3. Improved service level benefits.
4. Improved information benefits.

Cost-savings benefits lead to reductions in administrative and operational costs. A reduction in the size of the clerical staff used in the support of an administrative activity is an example of a cost-saving benefit.

Cost-avoidance benefits are those which eliminate future administrative and operational costs. No need to hire additional staff in future to handle an administrative activity is an example of a cost-avoidance benefit.

Improved-service-level benefits are those where the performance of a system is improved by a new computer-based method. Registering a student in ten minutes rather than two hour is an example of this third type of benefit.

Improved-information benefit is where computer based methods lead to better information for decision making. For example, a system that reports the most-improved ten customers, as measured by an increase in sales is an improved-information. This information makes it easier to provide better service to major customers.

4.5.3. Steps in cost benefits analysis

Cost/ benefit analysis is a procedure that gives a picture of the various costs, benefits and rules associated with a system. The determination of costs and benefits entails the following steps:

1. Identification of costs and benefits.
2. Classifications of costs and benefits.
3. Selection of evaluation method.
4. Determining the feasibility of the project.

1. Identification of costs and benefits

The analyst identifies those costs and benefits of the project that can be measured. For example, the costs of hardware, system software, stationery etc. and the savings from reduced costs can easily be identified and measured. The analyst also identifies those costs and benefits which cannot be measured. For example it is often difficult to measure the cost incurred in providing better customer service and improved company image during implementation of a new system.

2. Classification of costs and benefits

Costs and benefits can be classified into four major categories.

a) **Tangible costs and benefits:** Tangibility refers to the ease with which, costs or benefits can be measured. For example purchase of hardware or software is tangible costs. Benefits are more difficult to specify exactly than costs. For example suppliers can easily quote the cost of purchasing a terminal but it is difficult for them to tell specific benefits or financial advantages for using it in a system.

b) **Intangible costs and benefits:** Costs that are known to exist but whose financial value cannot be accurately measured are known as intangible costs. The estimate is only an approximation. For example how much moral of an employee has affected cannot be exactly measured in terms of financial values. In tangible benefit such as more satisfied customers or an improved corporate image because of using new system are not easily quantified.

c) **Direct or Indirect costs and benefits:** Direct costs are those which are directly associated with a system. They are applied directly to the operator. For example the purchase a CD for Rs 250/- is a direct cost. Direct benefit can also be specifically attributable to a given project. For example a new system that cans 30% more transaction per day is a direct benefit.

d) **Fixed or Variable costs and benefits:** Some costs and benefits remain constant regardless of how a system is used. Fixed costs are considered as sunk costs. Once encountered, they will not occur again. In contrast variable costs are incurred on a regular basis. They are

generally proportional to work volume and continue as long as the system is in operation. Same definition is for benefits.

3. Selection of Evaluation method

Evaluation method deals with the way of determining costs and benefits so as to determining the gross benefit of the system. A good evaluation method will help the analyst to find out the quality of the product. There can be various methods by which one can evaluate the cost and benefits.

a)Payback method.
b) Present value method.
c)Net benefit method.
d) Break even method.

4. Determining the feasibility of the project

It includes technical feasibility, economic feasibility, operational feasibility, legal feasibility, time feasibility etc

4.6 CONSTRUCTIVE COST MODEL (COCOMO):

Constructive Cost model was developed by Barry W Boehm in 1981. It is an algorithmic cost model. Algorithmic cost model is developed based on relating the current project to previous projects. It is based on historical information.

Cocomo is based on size of the project. The size of the project may vary depending upon the function points.

Boehm proposed three levels of the model: **basic, intermediate, detailed.**

Basic COCOMO

The basic COCOMO model is a single-valued, static model that computes system development effort (and cost) as a function of program size expressed in estimated thousand delivered source instructions (KDSI).

Intermediate COCOMO

The intermediate COCOMO model computes system development effort as a function of program size and a set of fifteen "cost drivers" that include subjective assessments of product, hardware, personnel, and project attributes.

Advanced COCOMO

The advanced or detailed COCOMO model incorporates all characteristics of the intermediate version with an assessment of the cost driver's impact on each step (analysis, design, etc.) of the system process.

COCOMO models depends on the two main equations

1. development effort : $MM = a * KDSI^b$

based on MM - man-month / person month / staff-month is one month of effort by one person. In COCOMO, there are 152 hours per Person month. According to organization this values may differ from the standard by 10% to 20%.

2. effort and development time (TDEV) : $TDEV = 2.5 * MM^c$

The coefficents a, b and c depend on the mode of the development. There are three modes of development:

Development Mode	Project Characteristics			
	Size	Innovation	Deadline/constraints	Dev. Environment
Organic	Small	Little	Not tight	Stable
Semi-detached	Medium	Medium	Medium	Medium
Embedded	Large	Greater	Tight	Complex hardware/ customer interfaces

Table 4.1: development modes

4.6.1 BASIC COCOMO

The basic COCOMO applies the parameterized equation without much detailed consideration of project characteristics.

$MM = a * KDSI^b$
$TDEV=2.5 * MM^c$

Basic COCOMO	a	b	c
Organic	2.4	1.05	0.38
Semi-detached	3.0	1.12	0.35
Embedded	3.6	1.20	0.32

Table 4.2: equations and parameters of the basic COCOMO

4.6.2 INTERMEDIATE COCOMO

The same basic equation for the model is used, but fifteen cost drivers are rated on a scale of 'very low' to 'very high' to calculate the specific effort multiplier and each of them returns an adjustment factor which multiplied yields in the total EAF (Effort Adjustment Factor). The adjustment factor is 1 for a cost driver that's judged as normal.

In addition to the EAF, the model parameter "a" is slightly different in Intermediate COCOMO from the basic model. The parameter "b" remains the same in both models.

$MM = a * KDSI^b$
$TDEV=2.5 * MM^c$

Intermediate COCOMO	a	b	c
Organic	3.2	1.05	0.38
Semi-detached	3.0	1.12	0.35
Embedded	2.8	1.20	0.32

Man Month Correction is $MM_{Korr} = EAF * MM_{nominal}$

Table 4.3: equations and parameters of the Intermediate COCOMO

4.6.3 ADVANCED, DETAILED COCOMO

The Advanced COCOMO model computes effort as a function of program size and a set of cost drivers weighted according to each phase of the software lifecycle. The Advanced model applies the Intermediate model at the component level, and then a phase-based approach is used to consolidate the estimate. The four phases used in the detailed COCOMO model are: requirements planning and product design (RPD), detailed design (DD), code and unit test (CUT), and integration and test (IT). Each cost driver is broken down by phases as in the example shown in Table 4.4.

Cost Driver	Rating	RPD	DD	CUT	IT
ACAP	Very Low	1.80	1.35	1.35	1.50
	Low	0.85	0.85	0.85	1.20
	Nominal	1.00	1.00	1.00	1.00
	High	0.75	0.90	0.90	0.85
	Very High	0.55	0.75	0.75	0.70

Table 4.4: Analyst capability effort multiplier for detailed COCOMO

Estimates for each module are combined into subsystems and eventually an overall project estimate. Using the detailed cost drivers, an estimate is determined for each phase of the lifecycle.

4.6.4 LIMITATIONS OF COCOMO

- COCOMO is used to estimate the cost and schedule of the project, starting from the design phase and till the end of integration phase. For the remaining phases a separate estimation model should be used.
- COCOMO is not a perfect realistic model. Assumptions made at the beginning may vary as time progresses in developing the project.
- When need arises to revise the cost of the project. A new estimate may show over budget or under budget for the project. This may lead to a partial development of the system, excluding certain requirements.
- COCOMO assumes that the requirements are constant throughout the development of the project; any changes in the requirements are not accommodated for calculation of cost of the project.

- There is not much difference between basic and intermediate COCOMO, except during the maintenance and development of the software project.
- COCOMO is not suitable for non-sequential ,rapid development, reengineering ,reuse cases models.

Check Your Progress 2

1. Name different types of costs.

   _____

   _____

   _____

2. Name different types of benefits.

   _____

   _____

   _____

3. Write various classifications of cost and benefit.

   _____

   _____

   _____

4.7 SUMMARY

- A feasibility study is conducted to select the best system that meets performance requirements.
- A system required performance is defined by statement of constraints, the identification of specific system objectives, and a description of outputs. The analyst is ready to evaluate the feasibility of the candidate systems to produce these outputs.
- Feasibility analysis is of seven type technical feasibility, economic feasibility, operational feasibility, social feasibility, management feasibility, legal feasibility, and time feasibility.
- Feasibility analysis involves eight steps:
    1. From a project team and appoint a project leader.
    2. Prepare system flowcharts.
    3. Enumerate potential candidate systems.
    4. Describe and identify characteristics of candidate systems.
    5. Determine and evaluate performance and cost effectiveness of each candidate system.
    6. Weight system performance and cost data.
    7. Select the best candidate system.
    8. Prepare and report final project directive to management.

- Data analysis is a prerequisite to cost/ benefit analysis. System investigation and data gathering lead to an assessment of current findings. From the analysis, the system design requirements are identified, and alternative system evaluated.
- In developing cost estimates for a system, we need to consider several cost elements. Among them is hardware, personnel, facility, operating and supply costs.
- Cost/ benefit analysis is a procedure that gives a picture of the various costs, benefits and rules associated with a system. The determination of costs and benefits entails the following steps:
    - Identify the costs and benefits pertaining to given project.
    - Categorize the various costs and benefits for analysis.
    - Select a method of evaluation.
    - Interpret the results of the analysis.
    - Take action.
- Cost and benefit determination is to categorize costs and benefits. They may be tangible or intangible, direct or indirect, fixed or variable.
- When all financial data have been identified and broken down into cost categories, the analyst must select a method of evaluation. Several evaluation methods are available, each with pros and cons. The common methods are:
    - Net benefit analysis
    - Present value analysis
    - Net present value
    - Payback analysis
    - Break-even analysis
    - Cash-flow analysis

4.8 KEYWORDS

- Feasibility Study
- Indirect Cost
- Intangible Cost
- Fixed cost
- Cost Benefit Analysis

4.9 MODEL ANSWERS

Check Your Progress 1

1. Feasibility analysis is of seven type as following
    1. Technical feasibility,
    2. Economic feasibility,
    3. Operational feasibility,
    4. Social feasibility,
    5. Management feasibility,
    6. Legal feasibility, and
    7. Time feasibility.

2. The main points to be considered in Operational feasibility are :
    a) What changes will be brought with the system?
    b) What organizational structures are disturbed?

c) What new skills will be required?
d) Do the existing staff members have these skills? If not, can they be trained in due course of time?

3. Feasibility analysis involves eight steps:
   1. From a project team and appoint a project leader.
   2. Prepare system flowcharts.
   3. Enumerate potential candidate systems.
   4. Describe and identify characteristics of candidate systems.
   5. Determine and evaluate performance and cost effectiveness of each candidate system.
   6. Weight system performance and cost data.
   7. Select the best candidate system.
   8. Prepare and report final project directive to management.

Check Your Progress 2

1. Various types of cast are:
   - Hardware costs
   - Personnel costs
   - Facility costs
   - Ongoing support and growth costs
   - Supply costs
2. The benefits of a project include four types:
   - Cost saving benefits.
   - Cost avoidance benefits.
   - Improved service level benefits.
   - Improved information benefits.
3. Costs and benefits can be classified into four major categories.
   - Tangible costs and benefits
   - Intangible costs and benefits
   - Direct or Indirect costs and benefits
   - Fixed or Variable costs and benefits

4.10 TERMINAL QUESTIONS

1. What is cost and benefit Analysis?
2. How are tangible costs different from direct costs?
3. What are the various cost benefit categories?
4. Why is it necessary to conduct cost/benefit analysis?
5. What do you understand by the term feasibility study of a solution?
6. Briefly discus the various steps in Feasibility Analysis.
7. Explain all the inter-related types of feasibility studies.
8. What is the importance of feasibility study?
9. What do you mean by operational feasibility?
10. Why is feasibility analysis necessary before designing a system?

APPENDIX – A

BIBLIOGRAPHY

- Systems Analysis and Design, Elias M. Awad, Galgotia
- Analysis and Design of Information Systems, James A Senn, McGraw Hill International
- Analysis and Design of Information Systems, V Rajaraman, PHI 2002
- Modern Systems Analysis and Design, Jeffrey A Hoffer, Joey F George, Joseph S. Valacich, Pearson Education
- Systems Analysis and Design, Kendall and Kendall, PHI
- Systems Analysis and Design, Igor Hawryszkiewycz, PHI
- Davenport, Thomas (1993), Process Innovation: Reengineering work through information technology, Harvard Business School Press, Boston
- Software Engineering by Roggers S. Pressmen
- http://en.wikipedia.org/wiki/Systems_Development_Life_Cycle

APPENDIX – B

GLOSSARY

Action Entry: The lower right quadrant of a decision table indicates the responses of the question entered in the condition entry.

Action Stub: Lower left quadrant of a decision table outlines in narrative form the conditions that may exist.

Analysis: Breaking a program into successively manageable parts for individual study.

Audit Trail: A feature of data processing system that allow for the study of data as processed from step to step, an auditor can then trace all transactions that affect an account.

Corrective Maintenance: Changes made to a system to repair flaws in its design, coding, or implementation.

Context Diagram: An overview of an organizational system that shows the system boundary, external entities that interact with the system and the major information flows between the entities and the system

Conversion: The organizational process of changing over from the current information system to a new one.

Cost/benefit analysis: the analysis to compare costs and benefits to see whether investing in the development of a new system will be beneficial.

Data: Data are raw facts that collected for the entity in question

Database Management Systems: Software that is used to create, maintain, and provide controlled access to user databases

Data Flow Diagrams (DFD): Graphical depiction of data processes, data flows and data stores in a business system

Data Store: Data at rest, which may take the form of many different physical representations

Decision table: a tabular representation of processing logic containing decision variables, decision variable values, and actions or formulas.

Decision tree: a graphical description of process logic that uses lines organized like branches of a tree.

Decision Support System (DSS): An interactive information system that supports the decision making process through the presentation of information designed specifically for decision maker's problem approach and application needs

Economic Feasibility: A process of identifying the financial benefits and costs associated with a development project.

Gantt chart: a bar chart that represents the tasks and activities of the project schedule.

Gap analysis: The process of discovering discrepancies between two or more sets of data flow diagrams or discrepancies within a single DFD

Implementation: The phase of system development where the information system is coded, tested, installed, and supported in the organization

Information: Information is processed data. Information attributes a meaning to the data.

Information system: a collection of interrelated components that collect, process, store, and provide as output the information needed to complete business tasks.

Management Information System (MIS): A computer based system composed of people, hardware, software and procedures that share a common database to help users interpret data and apply to business

Maintainability: The ease with which software can be understood, corrected, adapted, and enhanced.

Maintenance: Maintenance includes all the software engineering activities that occur following delivery of a software product to the customer.

Operational Feasibility: The process of assessing the degree to which a proposed system solves business problems or takes advantage of business opportunities.

Perfective Maintenance: Changes made to a system to add new features or to improve performance.

Preventive Maintenance: Changes made to a system to avoid possible future problems Prototyping: An iterative process of systems development in which requirements are converted to a working system which is continually revised through close work between an analyst and users.

Schema: The design of a database is called as schema

Structured analysis: is a technique that helps the developer define what the system needs to do (the processing requirements), what data the system needs to store and use (data requirements), what inputs and outputs are needed, and how the functions work together overall to accomplish tasks.

Structure chart: a graphical model showing the hierarchy of program modules produced by the structured design technique.

Structured English: a method of writing process specifications that combine structured programming techniques with narrative English.

Subsystem: a system that is part of a larger system.

Super-system: a larger system that contains other systems

System a collection of interrelated components that function together to achieve a common goal.

Systems Analysis: the process of understanding and specifying in detail what the information system should do.

System boundary: the separation between a system and its environment that inputs and outputs must cross.

Systems Design: the process of specifying in detail how the many components of the information system should be physically implemented.

Systems Analyst: The organizational role most responsible for the analysis and design of information systems

Systems Analysis and Design: The complex organizational process whereby computer-based information systems are developed and maintained.

Systems design: Phase of the SDLC in which the system chosen for development in systems analysis is first described independent of any computer platform (logical design) and is then transformed into technology-specific details (physical design) from which all programming and system construction can be accomplished

System Documentation: Detailed information about a system - design specifications, it's internal workings, and its functionality.

Transactions: Individual, simple events in the life of an organization that contain data about organizational activity.

Transaction Processing System (TPS) A computerized information system developed to process large amounts of data for routine business transactions such as payroll and inventory.

Waterfall method: a method of executing an SDLC where one phase leads (falls) to the next phase.

Uttar Pradesh Rajarshi Tandon
open University

MCA/
PGDCA-E4

Master in Computer Application

BLOCK

3

SYSTEM DESIGN

Course Design Committee

Dr. Ashutosh Gupta — **Chairman**
Director-In-charge,
School of Computer and Information Science, UPRTOU, Allahabad

Prof. R. S. Yadav — **Member**
Department of Computer Science and Engineering
MNNIT-Allahabad, Allahabad

Ms. Marisha — **Member**
Assistant Professor, Computer Science
School of Science, UPRTOU, Allahabad

Mr. Manoj Kumar Balwant — **Member**
Assistant Professor, Computer Science
School of Science, UPRTOU, Allahabad

Course Preparation Committee

Dr. Saurav Pal — **Author**
Associate Professor
Department of MCA, VBS Purvanchal University
Jaunpur-222001, Uttar Pradesh

Prof. R. R. Tiwari — **Editor**
University of Allahabad
Allahabad, Uttar Pradesh

Dr. Ashutosh Gupta
Director (In-charge), School of Computer and Information Science,
UPRTOU, Allahabad

Ms. Marisha (Coordinator)
Assistant Professor, School of Sciences,
UPRTOU, Allahabad

Printed and Published by Pro. (Dr.) Girija Shankar Shukla Registrar, **Uttar Pradesh Rajarshi Tandon open University, 2017.**
Printed By : Chandrakal Universal Pvt Ltd. 42/7 Jawahar Lal Neharu Road Allahabad

UNIT - I

SYSTEM DESIGN AND DESIGN METHODOLOGIES

UNIT STRUCTURE

1.1 INTRODUCTION

This process includes specifications for meeting the system requirements. The reports and the display outputs are identified. The display is sketched using available automated tools. It also involves description of data to be input, calculated or stored. Individual data items and calculation procedures are written in detail. The storage devices are selected. The procedures to produce outputs are written. The design specification document is created that depicts information using charts, tables and special symbols. Hence, at the end of the design phase, the analyst presents the programmer with a clear and complete outline of the system specifications.

1.2 OBJECTIVE

After studying this Unit, you should be able to:

- What is the process of system design?
- How logical design differs from physical design?
- What are design methodologies?
- How Structure chart is used in system design?
- Form driven methodologies.
- HIPO and IPO charts.
- How source documents are designed?

1.3 SYSTEMS DESIGN

System design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory

to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

The purpose of System Design is to create a technical solution that satisfies the functional requirements for the system. At this point in the project lifecycle there should be a Functional Specification, written primarily in business terminology, containing a complete description of the operational needs of the various organizational entities that will use the new system. The challenge is to translate all of this information into Technical Specifications that accurately describe the design of the system, and that can be used as input to System Construction. The Functional Specification produced during System Requirements Analysis is transformed into a physical architecture.

System components are distributed across the physical architecture, usable interfaces are designed and prototyped, and Technical Specifications are created for the Application Developers, enabling them to build and test the system.

Many organizations look at System Design primarily as the preparation of the system component specifications; however, constructing the various system components is only one of a set of major steps in successfully building a system. The preparation of the environment needed to build the system, the testing of the system, and the migration and preparation of the data that will ultimately be used by the system are equally important.

In addition to designing the technical solution, System Design is the time to initiate focused planning efforts for both the testing and data preparation activities.

1.3.1 Design Objectives

The primary objective of the design is to deliver the requirements as specified in the feasibility report. In general, the following design objectives should be kept in mind:

a. Practicality: The system must be stable and can be operated by people with average intelligence.
b. Efficiency: This involves accuracy, timeliness and comprehensiveness of the system output.
c. Cost: It is desirable to aim for a system with a minimum cost subject to the condition that it must satisfy all the requirements.
d. Flexibility: The system should me modifiable depending on the changing needs of the user. Such modifications should not entail extensive reconstructing or recreation of software. It should also be portable to different computer systems.
e. Security: This is very important aspect of the design and should cover areas of hardware reliability, fall back procedures, physical security of data and provision for detection of fraud and abuse.

System design involves fist logical design and then physical constriction of the system. The logical design describes the structure and characteristics of features, like the outputs, inputs, files, databases and procedures. The physical construction, which follows the logical design, produces actual program software, files and a working system.

1.3.2 Logical Design:

For a candidate system logical design describes:

- The inputs(source)
- Output (destination)

- Databases (data stores)
- Procedures (data flows) all in the format that meet the user's requirements

When Analysts prepare the logical system design, they specify the user needs at a level of the system and required data sources. The design covers the following:

1. Review the current physical system-its data flows, files content, volumes, frequencies etc.
2. Prepare output specifications-that is determines the format, content and frequency of reports, including terminal specifications and locations.
3. Prepare input specifications- format, content and most of the input functions. This includes determining the flow of documents from the input data source to the actual input decision.
4. Prepare edit security and control specifications.
5. Specifies the implementation plan.
6. Prepare a logical design walkthrough of the information flow, output, input controls and implementation plan.
7. Review the benefits costs, target dates and system constraints

1.3.3 Physical Design:

The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is inputted into a system, how it is verified/ authenticated, how it is processed, and how it is displayed as output.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc.

1. The physical design produce the working system by defining the design specifications that tell the programmer exactly what the candidate must do
2. For this Programmer writes the necessary programs or modifies the software package that accepts input from the user, performs the necessary calculations through the existing database.
3. Database produces the report on a hard copy or displays it on a screen and maintains an updated database at all time.

Physical system consists of the following:

- Design the physical System:
    a) Specify the input/output media.
    b) Design the data base and specify the backup procedures.
    c) Design physical information flow through the system and physical design walkthrough.
- Plan system Implementation:
    a) Prepare a conversion schedule and a target date.
    b) Determine training procedure, courses, and time table.
- Devise a test and implementation plan and specify any new Hardware/Software.
- Update benefits, costs conversion date, and system constraints (legal), financial, hardware, etc.

1.4. DESIGN METHODOLOGIES

In the last decade a growing moves to transform the art of system analysis and design in to an engineering type discipline. Therefore more clearly defined logical methods for developing a system that fulfill user requirements has led to new techniques and methodologies that fundamentally attempt to do the following

- Cut down cost overruns and delay.
- Improve productivity of analysts and programmer.
- Standardize the approach to analysis and design.
- Improve documentation.
- Improve communication among the users, analysts and programmers.
- Simplify design by segmentation.

1.4.1 Top Down Design

We know that a system is composed of more than one sub-systems and it contains a number of components. Further, these sub-systems and components may have their own set of sub-system and components and creates hierarchical structure in the system.

Top-down design takes the whole system as one entity and then decomposes it to achieve more than one sub-system or component based on some characteristics. Each sub-system or component is then treated as a system and decomposed further. This process keeps on running until the lowest level of system in the top-down hierarchy is achieved.

Top-down design starts with a generalized model of system and keeps on defining the more specific part of it. When all components are composed the whole system comes into existence.

Top-down design is more suitable when the solution needs to be designed from scratch and specific details are unknown.

1.4.2 Bottom-up Design

The bottom up design model starts with most specific and basic components. It proceeds with composing higher level of components by using basic or lower level components. It keeps creating higher level components until the desired system is not evolved as one single component. With each higher level, the amount of abstraction is increased.

Bottom-up strategy is more suitable when a system needs to be created from some existing system, where the basic primitives can be used in the newer system.

Both, top-down and bottom-up approaches are not practical individually. Instead, a good combination of both is used.

1.4.3 Structure Chart

A hierarchical diagram showing the relationships between the modules of a computer program. A module is the basic component of a structure chart and is used to identify a function. Modules are relatively simple and independent components. Higher-level modules are “control” modules that control the flow of execution. Lower level modules are “worker bee” modules and contain the program logic to actually perform the functions.

The vertical lines connecting the modules indicate the calling structure from the high-level modules to the lower-level modules. The little arrows next to the lines show the data that is passed between modules and represent the inputs and outputs of each module. At the structure chart level, we are not concerned with what is happening inside the module yet. We only want to know that somehow it does the function indicated by its name using the input data and producing the output data. A program call is when one module invokes a lower-level module to perform a needed service or calculation. Program call: The transfer of control from a module to a subordinate module to perform a requested service. The arrows with the open circle, called data couples, represent data being passed into and out of the module. A data couple can be an individual data item (e.g., a flag or a customer account number) or a higher-level data structure (e.g., an array, record, or other data structure). The arrow with the darkened circle is a "flag." A flag is purely internal information that is used between modules to indicate some result. Data couples: The individual data items that are passed between modules in a program call.

A basic idea of structured programming is that each module only has to do a very specific function. The module at the very top of the tree is the "boss" module. Its functions will be to call the modules on the next tier, pass information to them, and receive information back. The function of each middle-level module is to control the processing of the modules below it. Each has control logic and any error-handling logic that is not handled by the lower-level module. The modules at the extremities, or the leaves, contain the actual algorithms to carry out the functions of the program.

Structure charts are developed to design a hierarchy of modules for a program. A structure chart is in the form of a tree with a root module and branches. A subtree is simply a branch that has been separated from the overall tree. When the subtree is placed back in the larger tree, the root of the subtree becomes just another branch in the overall tree.

- **Overview:**

A structure chart is a top-down modular design tool, constructed of squares representing the different modules in the system, and lines that connect them. The lines represent the connection and or ownership between activities and sub activities as they are used in organization charts.

In structured analysis structure charts, according to Wolber (2009), are used to specify the high-level design, or architecture, of a computer program. As a design tool, they said the programmer in dividing and conquering a large software problem, that is, recursively breaking a problem down into parts that are small enough to be understood by a human brain. The process is called top-down design, or functional decomposition. Programmers use a structure chart to build a program in a manner similar to how an architect uses a blueprint to build a house. In the design stage, the chart is drawn and used as a way for the client and the various software designers to communicate. During the actual building of the program (implementation), the chart is continually referred to as the master-plan".

A structure chart is also used to diagram associated elements that comprise a run stream or thread. It is often developed as a hierarchical diagram, but other representations are allowable. The representation must describe the breakdown of the configuration system into subsystems and the lowest manageable level. An accurate and complete structure chart is the key to the determination of the configuration items, and a visual representation of the configuration system and the internal interfaces among its CIs. During the configuration control process, the structure chart is used to identify CIs and their associated artifacts that a proposed change may impact.

- **Applications of Structure Chart**

  Use a Structure Chart to illustrate the high level overview of software structure. Structure Charts do not show module internals. Use a method, such as Pseudo code or Structured English, to show the detailed internals of modules.

- **Advantages of structure chart**
  - Representing Sequence, Repetition, and Condition on a Structure Chart
  - Modules on a Structure Chart
  - Interrelationships among Modules
  - **Information Transfers**
  - **Reducing Clutter on a Structure Chart.**

Example: The example Structure Chart illustrates the structure of the modules to **process a customer order.**

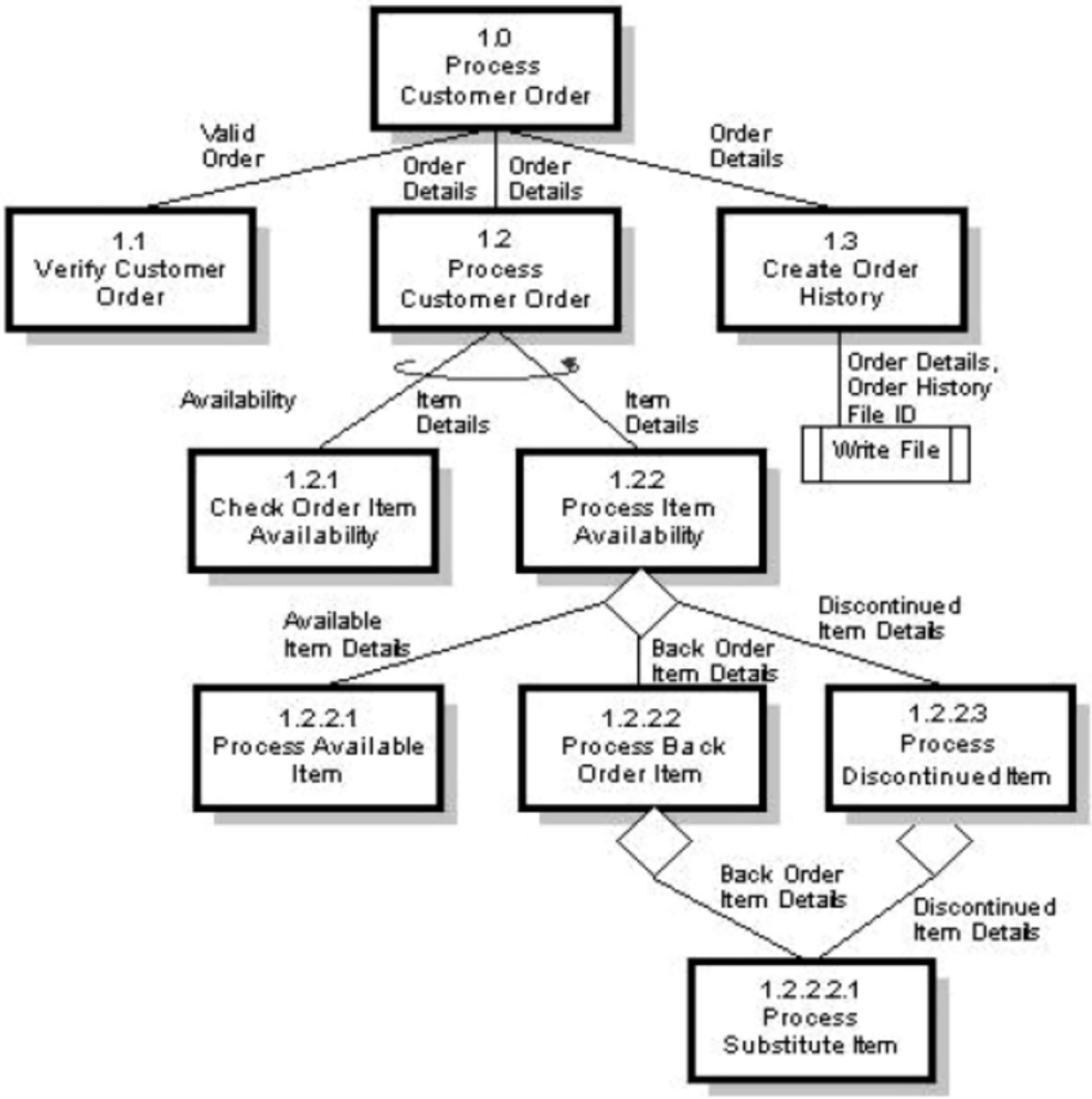


Fig 1.1: Structure chart

1.4.4 Forms Driven Methodology – The IPO Charts

HIPO charts show relationships between modules. It describes the data input and output from the processes and defines the data flow. It provides a structure by which the functions of a system can be understood. It also provides a visual description of input to be used and output to be produced for each level of the diagram. It makes the transformation from input to output data visible.
There are two parts to a HIPO chart, a hierarchy chart and an IPO chart.
The **hierarchy chart** is useful for showing hierarchy of procedures within a program. Hierarchy charts are also called structure charts, top-down charts, or VTOC (Visual Table of Contents) charts. All these names refer to planning diagrams that are similar to a company's organization chart. Hierarchy charts depict the organization of a program but omit the specific processing logic. They describe what each part, or module, of the program does and how the modules relate to each other.

The **IPO** chart describes the system in terms of its inputs, outputs and the processes that are performed on the inputs to transform them into outputs. It provides the following:

- The Input section that contains the data items used by the process steps.

- The Output section that contains the data items created by the process steps.
- Process section that contains numbered steps that describes the functions to be performed. Arrows connect them to the output steps and the input/output data items.

The IPO chart is in the form of a table with three columns, one for each of Input, Output and Process. The flow between screens is indicated by the use of arrows.

- **Use of HIPO chart?**

  The HIPO (Hierarchy plus Input-Process-Output) technique is a tool for planning and/or documenting a computer program. A HIPO model consists of a hierarchy chart that graphically represents the program's control structure and a set of IPO (Input- Process-Output) charts that describe the inputs to, the outputs from, and the functions (or processes) performed by each module on the hierarchy chart.

- **Advantages of HIPO Chart:**

  Using the HIPO technique, designers can evaluate and refine a program's design, and correct flaws prior to implementation. Given the graphic nature of HIPO, users and managers can easily follow a program's structure. The hierarchy chart serves as a useful planning and visualization document for managing the program development process. The IPO charts define for the programmer each module's inputs, outputs, and algorithms.

- **Limitation of HIPO Chart:**
  - HIPO provides valuable long-term documentation. However, the "text plus flowchart" nature of the IPO charts makes them difficult to maintain, so the documentation often does not represent the current state of the program.

- By its very nature, the HIPO technique is best used to plan and/or document a hierarchically structured program.

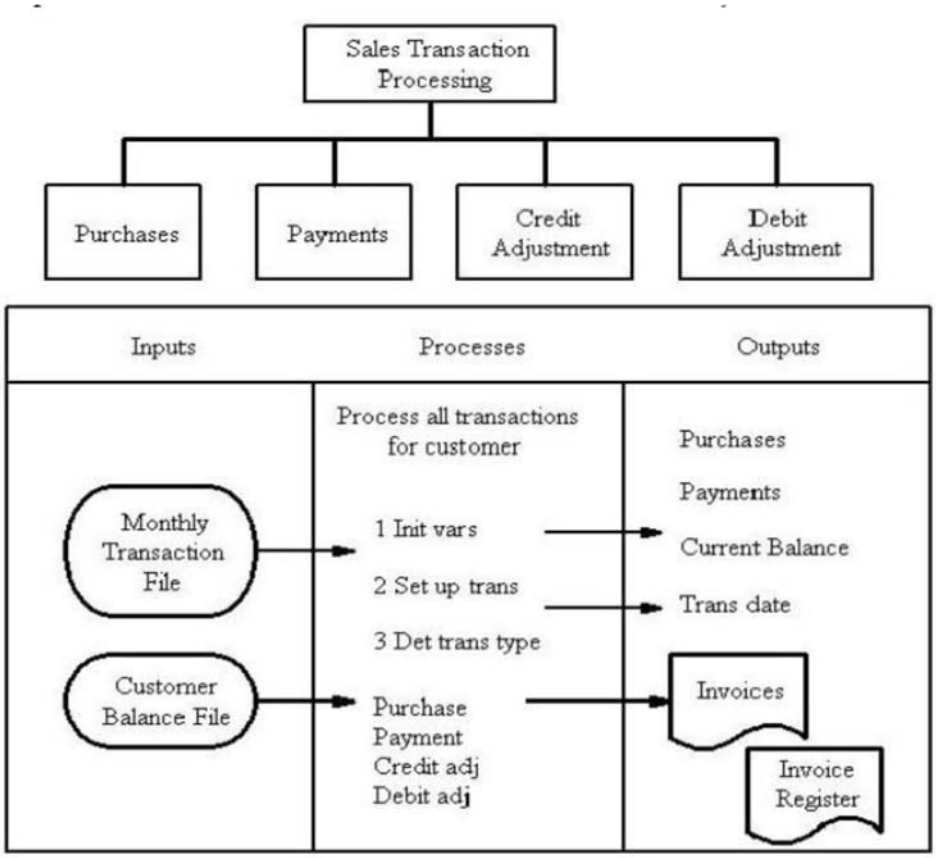


Fig 1.2: HIPO chart

Check Your Progress 1

1. Name the five objective of system design.

_____

_____

_____

2. List the two types of system design.

_____

_____

_____

3. Write the advantages of structure chart.

_____

_____

_____

4. What are the limitations of a HIPO chart?

_____

_____

_____

1.5 SUMMARY

- System design is a transition from user document to database personnel. It goes through logical and physical design.
- The design objectives are:
    - Practicality
    - Efficiency
    - Cost:
    - Flexibility
    - Security
- Logical design describes:
    - The inputs(source)
    - Output (destination)
    - Databases (data stores)
    - Procedures (data flows)
- The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is inputted into a system, how it is verified/ authenticated, how it is processed, and how it is displayed as output.
- One way of developing an input/process/output chart for modules is with HIPO chart. It consists of the hierarchy chart and the input/process/output (IPO) chart, thus capturing the essence of top-down decomposition.

1.6 KEYWORDS

- Logical Design
- Physical Design
- Structure Chart
- System Design
- Modules
- Top-down Design
- HIPO
- IPO Chart

1.7 MODEL ANSWERS

Check Your Progress 1

1. The main objectives of system design are
    - Practicality: The system must be stable and can be operated by people with average intelligence.
    - Efficiency: This involves accuracy, timeliness and comprehensiveness of the system output.
    - Cost: It is desirable to aim for a system with a minimum cost subject to the condition that it must satisfy all the requirements.
    - Flexibility: The system should me modifiable depending on the changing needs of the user. Such modifications should not entail extensive reconstructing or recreation of software. It should also be portable to different computer systems.
    - Security: This is very important aspect of the design and should cover areas of hardware reliability, fall back procedures, physical security of data and provision for detection of fraud and abuse.
2. Two types of system designs are
    - Logical Design
    - Physical design
3. The main advantages of system chart are
    - Representing Sequence, Repetition, and Condition on a Structure Chart
    - Modules on a Structure Chart
    - Interrelationships among Modules
    - Information Transfers
    - Reducing Clutter on a Structure Chart.
4. Few limitations of HIPO chare are
    - HIPO provides valuable long-term documentation. However, the "text plus flowchart" nature of the IPO charts makes them difficult to maintain, so the documentation often does not represent the current state of the program.
    - By its very nature, the HIPO technique is best used to plan and/or document a hierarchically structured program.

1.8 TERMINAL QUESTIONS

1. Explain HIPO diagram with example.
2. Write the Advantages of HIOP chart.
3. Differentiate between
    a) Logical and Physical Design

b) HIPO and IPO

4. What design methodologies are used in system design? Explain.
5. Explain objectives of design.
6. What are the items one has to design during systems design?
7. What is the outcome of design phase?
8. What are the various classical design activities?
9. Explain the principle of system Design.
10. What do you mean by structure chart?
11. List limitations of structure chart.
12. What are the advantages of structure chart?
13. Explain structure chart with example.
14. What are the uses of IPO chart?
15. List limitations of HIPO.

UNIT - II

INPUT/OUTPUT AND FORM DESIGN

UNIT STRUCTURE

2.1 INTRODUCTION

Output is what the customer is buying when he or she pay for a development of system. Inputs, databases, and processes are present to provide output.

A data input specification is a detailed description of the individual fields (data elements) on an input document together with their characteristics. In this chapter we will learn about Input design, Output design and Form design.

2.2 OBJECTIVE

After studying this Unit, you should be able to:

- What are the objectives of input design?
- What is the process of input design?
- What is screen design?
- How input design differs from output design?
- The classes of forms and how they are designed?
- What are the requirements of form design?

2.3 INPUT DESIGN

Input design is the process of converting user-oriented inputs to a computer-based format. Input data are collected and organized into group of similar data. Once identified, appropriate input media are selected for processing.

Input design has six main objectives:

1. Select suitable input and data entry method.
2. Reduce input volume.
3. Design attractive data entry screens.
4. Use validation checks to reduce input errors.
5. Design required source documents.
6. Develop effective input controls.

There are two main data entry methods; batch and online input.

- **Batch input**
  - Data entry is performed on a specified time schedule
  - Collection (batch) of data is input at one time
- **Online input**
  - Data is validated and available immediately.
  - Source data automation.
  - Combines online data entry with online data capture.
  - Uses magnetic data strips and swipe scanners.
  - Common examples: ATMS, point-of-sale terminals, bar code readers, patient ID bracelets, libraries.

Online data entry has become a requirement in many dynamic business operations. An example is the online package tracking service offered by courier and other package delivery services: as the shipment moves through the carrier's delivery system, information about the shipment's progress can be accessed almost immediately online by the customer. Source data automation is important, and examples might include optical scanners that grade standardized test forms and magnetic ink character recognition devices that read the numbers on checks.

Because most data problems occur during data entry, a reduction in input volume will reduce the number of errors that must be located and corrected. The four guidelines for reducing input volume are:

- Input necessary data only.
- Do not input data that can be retrieved from system files or calculated from other data.
- Do not input constant data.
- Use codes.

2.4 SCREEN DESIGN

When designing data entry screens, form filling is the most effective method of online data entry since it resembles filling a paper-based form on the screen. Input data can be grouped into three:

- Data that must be entered by the user
- Data generated by the system
- Data calculated by the system

Some screen design guidelines are:

1. Restrict user access to screen locations where data is entered
2. Provide a descriptive caption for every field
3. Display a sample format if a user must enter values in a specific format

4. Require an ending keystroke in every field
5. Do not require leading zeros for numeric fields
6. Do not require trailing zeros
7. Display default values
8. Use default values for constant entries
9. Display a list of acceptable values for fields
10. Provide a way to leave the data entry screen without entering the current record
11. Provide the opportunity to confirm to confirm the accuracy of input data
12. Provide for movement among fields in a standard order or any chosen order
13. Design the screen form layout to match that of the source documents
14. Allow users to add, change, delete, and view records
15. Provide for users to search for specific information

Fewer input errors mean better data quality, and elimination of these errors is performed by validation checks. There are eight types of major data validation checks:

1. Sequence checks
2. Existence checks
3. Data type checks
4. Range checks
5. Reasonableness checks
6. Validity checks
7. Combination checks
8. Batch controls

Designing source documents is an important step in input design, since some of the entered data come from these source documents or forms. They are characterized by:

- Paper-based form design
- Request and collect input data
- Can trigger or authorize input actions
- Provide a record of the original transaction

Form layout guidelines are:

- Allow sufficient space
- Offer clear instructions
- Provide logical organization
- Use captions effectively

There are several different ways of handling captions. All are acceptable techniques, but that below-the-line and below-the-box captions can be troublesome if a typewriter is used to fill out the form. A source document is typically divided into several zones such as:

- Heading zone
- Control zone
- Instruction zone
- Body zone
- Totals zone
- Authorization zone

Information on a form or screen should flow left to right and top to bottom.

An information system cannot be successful without strong, effective input control measures that ensure correct, complete, accurate and secure data. The systems analyst is responsible for designing input control features as part of the overall system design. Examples of these measures are:

- Effective source document design
- Data validity checks
- Batch input controls
- Log files for rejected records
- Audit trails
- Data security measures, including encryption
- Password and sign-on procedures
- Records retention policies

2.5 OUTPUT DESIGN

Output is the most important task of any system. These guidelines apply for the most part to both paper and screen outputs. Output design is often discussed before other feature of design because, from the customer's point of view, the output is the system. Output is what the customer is buying when he or she pay for a development of project. Inputs, databases, and processes are present to provide output. Problems often associated with business information output are information hold-up, information (data) overload, paper domination, extreme distribution, and no tailoring.

For example:

Mainframe printers: high volume, high speed, located in the data centre Remote site printers: medium speed, close to end user.

COM is **Computer Output Microfilm.** It is more compressed than traditional output and may be produced as fast as non-impact printer output.

- Turnaround documents trim down the cost of internal information processing by reducing both data entry and associated errors.
- Periodic reports have set frequencies such as daily or weekly; ad hoc reports are produced at irregular intervals.
- Detail and summary reports differ in the former support day-to-day operation of the business while the latter include statistics and ratios used by managers to consider the health of operations.
- Page breaks and control breaks allow for abstract totals on key fields. Report requirements documents include general report information and field specifications; print layout sheets present a picture of what the report will actually look like.
- Page decoupling is the separation of pages into cohesive groups.

Two ways to create output for strategic purposes are

1. Make it compatible with processes outside the immediate scope of the system
2. Turn action documents into turnaround documents.

People often receive reports they do not require because the number of reports received is perceived as a measure of power.

Fields on a report should be selected carefully to provide organized reports, facilitate 80-column remote printing, and reduce information (data) overload.
The types of fields which should be considered for business output are: key fields for access to information, fields for control breaks, fields that change, and exception fields.
Output may be designed to aid future change by stressing formless reports, defining field size for future growth, making field constants into variables, and leaving room on review reports for added ratios and statistics.
Output can now be more easily tailored to the needs of individual users because inquiry-based systems allow users themselves to generate ad hoc reports. An output intermediary can restrict access to key information and avoid illegal access. An information clearinghouse (or information centre) is a service centre that provides consultation, assistance, and documentation to encourage end-user development and use of applications. The specifications essential to describe the output of a system are: data flow diagrams, data flow specifications, data structure specifications, and data element specifications.

- Output Documents
- Printed Reports
- External Reports: for use or distribution outside the organization; often on pre-printed forms.
- Internal Reports: for use within the organization; not as "pretty", stock paper, green bar, etc.
- Periodic Reports: produced with a set frequency (daily, weekly, monthly, every fifth Tuesday, etc.)
- Ad-Hoc (On Demand) Reports: unbalanced interval; produced upon user demand.
- Detail Reports: one line per transaction.
- Review Reports: an overview.
- Exception Reports: only shows errors, problems, out-of range values, or unexpected conditions or events.

2.6 FORM DESIGN

People read from forms, write on forms and spend hours handling forms and filing forms. The data the forms carry come from people and the informational output goes to people. Form is a tool with a message. It is the physical carrier of data-of information. It is an either an authority for action or a request for action.

2.6.1 Classification of forms

A printed form is generally classified by what it does in the system. There are three primary classifications

1. **Action:** This type of form requests the user to do something.

   Example: purchase orders.

2. **Memory:** This form is a record of historical data that remains in a file, is used for reference, and serves as control on key details.

   Example: Inventory records, purchase records

3. **Report:** This form guides supervisors and other administrators in the activities. It provides data on a project or a job.

   Example: profit and loss statements, sales analysis report

2.6.2 Requirements of form design

Form design follows analyzing forms. Since the purpose of a form is to communicate effectively through forms design, there are several major requirements.

1. **Identification and wording**: The form title must clearly identify its purpose. Columns and rows should be labeled to avoid confusion. The form should also be identified by firm name or code number to make it easy to reorder.
2. **Maximum readability and use:** The form must be easy to use and fill out. It should be legible, intelligible and uncomplicated. Ample writing space must be provided for inserting data.
3. **Physical factors:** The forms composition, color, layout and paper stock should lend themselves to easy reading. Pages should be numbered when multipage reports are being generated for the user.
4. **Order of data items:** The data requested should reflect a logical sequence. Related data should be in adjacent positions. Data copied from source documents should be in the same sequence on both forms.
5. **Ease of data entry:** If used for data entry, the form should have field positions indicated under each column of data and should have some indication of where decimal points are.
6. **Size and arrangement**: The form must be easily stored and filed. It should provide for signatures. Important items must be in a prominent action on the form.
7. **Use of instructions**: The instructions that accompany a form should clearly show how it is used and handled.
8. **Efficiency considerations:** The form must be cost effective. This means eliminating unnecessary data and facilitating reading lines across the form.
9. **Type of report**: Forms design should also consider whether the content is executive summary, intermediate managerial information, or supporting data. The user requirements for each type determine the final form design.

2.6.3 Carbon paper as form copier

Carbon paper is one way of duplicating information in a form. There are two types of carbon, classified by the action they encounter

1. **Glide action** carbon is inserted between a set of forms. It allows the glide action of the pencil to transfer dye to the surface of the sheet beneath.
2. **Hammer action** carbon is used in typewriters and line printers of computers. The hammer action of the keys transfers the carbon coating to the sheet beneath.

Various methods of transferring impressions between copies are as follows

1. **One time carbon:** It is made of inexpensive Kraftex paper. It is interleaved between two sheets in the form. It is used once and then thrown away. It is the most cost-effective for multipart forms.
2. **Carbon backed paper:** The back of each form copy is coated with carbon, which transfers data to the copy beneath.
3. **NCR (No Carbon Required) paper:** The top sheet is chemically treated with invisible dye, that allows impressions to be transferred to the next lower copy. It is the cleanest and the costliest method. Erasing removes the coating permanently.

The readability of the carbon copies depends on the color and outline. In multiple copies, the copies below the original should be lighter in weight for easy transfer of the carbon. Generally,

one time carbon is preferred when a small number of copies are required. If carbon is unacceptable, NCR can be used.

Check Your Progress 1

1. Explain briefly the objectives of input designs.

_____

_____

_____

2. What are the two methods of data entry?

_____

_____

_____

3. Explain various data validation checks.

_____

_____

_____

4. List out three primary classification of forms.

_____

_____

_____

5. What are the requirements for form design?

_____

_____

_____

2.7 SUMMARY

- Input design is the process of converting user-oriented inputs to a computer-based format. Input data are collected and organized into group of similar data. Once identified, appropriate input media are selected for processing.

- Input design has six main objectives:
    - Select suitable input and data entry method.
    - Reduce input volume.
    - Design attractive data entry screens.
    - Use validation checks to reduce input errors.
    - Design required source documents.
    - Develop effective input controls.
- When designing data entry screens, form filling is the most effective method of online data entry since it resembles filling a paper-based form on the screen.
- Output is the most important task of any system. These guidelines apply for the most part to both paper and screen outputs. Output design is often discussed before other feature of design because, from the customer's point of view, the output is the system.
- Form is a tool with a message. It is the physical carrier of data-of information. It is an either an authority for action or a request for action.

2.8 KEYWORDS

- Form
- Form Design
- NCR paper
- Computer Output Microfilm (COM)
- Kraftex paper

2.9 MODEL ANSWERS

Check Your Progress 1

1. Input design has six main objectives:
    - Select suitable input and data entry method.
    - Reduce input volume.
    - Design attractive data entry screens.
    - Use validation checks to reduce input errors.
    - Design required source documents.
    - Develop effective input controls.
2. Two methods of data entry are:
    - Batch input
    - online input
3. There are three primary classifications
    - Action
    - Memory
    - Report

4. The purpose of a form is to communicate effectively through forms design, there are several major requirements.
    - Identification and wording:
    - Maximum readability and use
    - Physical factors
    - Order of data items
    - Ease of data entry
    - Size and arrangement
    - Use of instructions
    - Efficiency considerations
    - Type of report

2.10 TERMINAL QUESTIONS

1. What is the goal of input design?
2. What is unique about on-line data entry?
3. What is a form? Explain the characteristics of action, memory and report form.
4. Describe design of output.
5. Describe design of input.
6. In what respect is carbon paper a form copier? How is it classified?
7. What else one has to design besides input and output?
8. List some guidelines that might be useful while designing printed output.
9. Which types of data are not advisable to be considered as input?
10. What is the difference between on – line and off – line data entry?

UNIT - III

FILE ORGANIZATION

UNIT STRUCTURE

3.1 INTRODUCTION

Records are stored in files. The organization of file specifies how records are stored, located and retrieved. Here we are discussing three ways of records storing. Among them two are sequential and direct are available in most if the computer where index method is possible with special type of software.

3.2 OBJECTIVE

After studying this Unit, you should be able to:

- What are the alternative methods of file organizations?
- What is sequential organization of files?
- What is indexed sequential organization of files?
- Difference between sequential and indexed sequential organization of files.
- Inverted list method of file organization.
- Direct method of file organization.
- What is chaining?

3.3 BASIC FILES RELATED KEYWORDS

- **Byte**: - It is the smallest addressable unit in computer. A byte is a set of 8 bits and represents a character.
- **Element: -** It is a combination of one or more bytes. It is referred to as a field. A field is actually a physical space on tape or disk. A roll number, age, name of employee etc. are examples of it.
- **Record: -** The elements related to are combined into a record. An employee has a record with his name, designation, basic pay, allowances, deductions etc. as its fields. A record may have a unique key to identify a record e.g. employee number. Records are represented as logical & physical records. A logical record maintains a logical relationship among all the data items in the

record. It is the way the program or user sees the data. In contrast a physical record is the way data are recorded on a storage medium.

- **File: -** It is a collection of similar records. The records will have the same fields but different values in each record. The size of a file is limited by the size of memory available.
- **Database:** - It is a set of interrelated files. The files in combination tend to link to a common solution. For example, a student attendance file, a student result file, a student admission file, etc. are related to academic software pertaining to students.

3.4 FILE ORGANIZATION

A file is organized to ensure that records are available for processing. It should be designed in the line with the activity and volatility of the information and the nature of the storage media and devices. Other considerations are

1. Cost of file media (highest for disk, lowest for tape)
2. Inquiry requirements (real – time versus batch processing) and
3. File privacy, integrity, security, and confidentiality.

There are four methods of organizing files:

- Sequential
- Direct access
- Indexed – sequential
- Inverted list
- Random

Each method is explained below.

3.4.1. Sequential Organization

Sequential organization simply means storing and sorting in physical, contiguous blocks within files on tape or disk. Records are also in sequence within each block. To access a record, previous records within the block are scanned. Thus sequential record design is best suited for "get next" activities, reading one record after another without a search delay.

In a sequential organization, records can be added only at the end of the file. It is not possible to insert a record in the middle of the file, without rewriting the file. In a data base system, however, a record may be inserted anywhere in the file, which would automatically resequence the records following the inserted record. Another approach is to add all new records at the end of the file and later sort the file on a key (name, number, etc.). Obviously, in a 60,000- record file it is less time-consuming to insert the few records directly than to sort the entire file.

In a sequential file update, transaction records are in the same sequence as in the master file. Records from both files are matched, one record at a time, resulting in an updated master file. For example, the system changes the customer's city of residence as specified in the transaction file (on floppy disk) and corrects it in the master file. A "C" in the record number specifies "replace"; an "A," "add"; and a "D," "delete."

In a personal computer with two disk drives, the master file is loaded on a diskette into drive A, while the transaction file is loaded on another diskette into drive B.

Updating the master file transfers data from drive B to A, controlled by the software in memory.

3.4.2 Direct- Access organization

In direct – access file organization, records are placed randomly throughout the file. Records need not be in sequence because they are updated directly and rewritten back in the same location. New records are added at the end of the file or inserted in specific locations based on software commands. Some time Direct-Access organization is also called as Random File Organization.

Records are accessed by addresses that specify their disk locations. An address is required for location a record, for linking records, or for establishing relationships.

Addresses are of two types: absolute and relative. An absolute address represents the physical location of the record. It is usually stated in the format of sector/track/record number. For example, 3/14/6 means go to sector 3, track 14 of that sector, and the sixth record of the track. One problem with absolute addresses is that they become invalid when the file that contains the records is relocated on the disk. One way around this is to use pointers for the updated records.

A relative address gives a record location relative to the beginning of the file. There must be fixed-length records for reference. Another way of locating a record is by the number of bytes it is from the beginning of the file. Unlike relative addressing, if the file is move, pointers need not be updated, because the relative location of the record remains the same regardless of the file location.

Thus each file organization method has advantages and limitations. Many applications by their nature are best done sequentially. Payroll is a good example. The system goes through the employee list, extracts the information and prepares pay slips.

There are no lengthy random-access seeks. In contrast, real-time applications where response requirements are measured in seconds are candidates for random-access design. Systems for answering inquires, booking airlines or stadium seats, updating checking or savings accounts in a bank, or interacting with a terminal are examples for random access design.

3.4.3 Indexed- Sequential Organization

Like sequential organization, keyed sequential organization stores data in physically contiguous blocks. The difference is in the use of indexes to locate records. To understand this method, we need to distinguish among three areas in disk storage: prime area, overflow area and index area. The prime area contains file records stored by key or ID numbers. All records are initially stored in the prime area. The overflow area contains records added to the file that cannot be placed in logical sequence in the prime area. The index area is more like a data dictionary. It contains keys of records and their locations on the disk. A pointer associated with each key is an address that tells the system where to find a record.

Indexed-sequential organization reduces the magnitude of the sequential search and provides quick access for sequential and direct processing. The primary drawback is the extra storage space required for the index. It also takes longer to search the index for data access or retrieval.

- **Chaining**

  File organization requires that relationships be established among data items. It must show how characters form fields, fields form files, and files relate to one another.

  Establishing relationships is done through chaining or the use of pointers. The example on airline reservation file showed how pointers, link one record to another. Part number retrieves a record. A better way is to chain the records by linking a pointer to each. The pointer gives the address of the next part type of the same class. The search method applies similarly to other parts in the file.

3.4.4. Inverted List Organization

Like the indexed-sequential storage method, the inverted list organization maintains an index. The two methods differ, however, in the index level and record storage. The indexed- sequential method has a multiple index for a given key, whereas the inverted list method has a single index for each key type. In an inverted list, records are not necessarily stored in particular sequence. They are placed in the data storage area, but indexes are updated for the record keys and location.

It can be seen that inverted lists are best for applications that request specific data on multiple keys. They are ideal for static files because additions and deletions cause expensive pointer updating.

Example:
Data for the flight reservation system.

The flight number, description and the departure time are as given as keys. In the data location area, no particular sequence is followed. If a passenger needs information about the Houston flight, the agent requests the record with Houston flight. The DBMS carries a sequential search to find the required record. The output will then be that the flight number is 170 departing at 10.10 A.M and flight number 169 departing at 8.15 A.M.

If the passenger searches for information about a Houston flight that departs at 8.15, then the DBMS searches the table and retrievesR3 and R6. Then it checks the flight departure time and retrieves R6 standing for flight number 169.

Check Your Progress 1

1. What do you mean by file?

   _____

   _____

   _____

2. List the methods of file organization.

   _____

   _____

   _____

3. What do you mean by chaining?

   _____

   _____

   _____

4. What is the inverted list organization?

_____

_____

_____

5. Explain direct access file organizations.

_____

_____

_____

3.5 SUMMARY

- A file is organized to ensure that records are available for processing. It should be designed in the line with the activity and volatility of the information and the nature of the storage media and devices.
- There are four methods of organizing files:
    - Sequential organization simply means storing and sorting in physical, contiguous blocks within files on tape or disk according to key.
    - Indexed sequential organization stores records sequentially but uses an index to locate record. Records are related through chaining using pointers.
    - Inverted list organization uses an index for each key type. Records are not necessarily in particular sequences.
    - Direct access organization has records placed randomly throughout the file. Records are updated directly and independently of the other records.

3.6 KEYWORDS

- Sequential organization
- Indexed sequential organization
- Inverted list organization
- Direct access organization
- chaining

3.7 MODEL ANSWERS

Check Your Progress 1

1. File is a collection of similar records. The records will have the same fields but different values in each record. The size of a file is limited by the size of memory available.
2. There are four methods of organizing files:
    - Sequential organization
    - Indexed sequential organization
    - Inverted list organization
    - Direct access organization

3. File organization requires that relationships be established among data items. It must show how characters form fields, fields form files, and files relate to one another. Establishing relationships is done through chaining or the use of pointers.
4. The indexed- sequential method has a multiple index for a given key, whereas the inverted list method has a single index for each key type. In an inverted list, records are not necessarily stored in particular sequence.
5. In direct – access file organization, records are placed randomly throughout the file. Records need not be in sequence because they are updated directly and rewritten back in the same location. New records are added at the end of the file or inserted in specific locations based on software commands.

3.8 TERMINAL QUESTIONS

1. Define organization of sequential access, direct access, indexed sequential access, and inverted files.
2. What is chaining? How does it relate to indexed-sequential file organization?
3. Differentiate between indexed-sequential and inverted list.
4. What methods does the designer consider in file organization?
5. What is sequential organization of files?
6. What is indexed sequential organization of files?
7. Differentiate between sequential and indexed sequential organization of files.
8. Explain various file organization methods.
9. Differentiate between file and record.
10. What do you mean by database?

UNIT - IV

DATA BASE DESIGN

UNIT STRUCTURE

4.1 INTRODUCTION

Once the analyst has decided onto the basic processes and inputs and outputs of the system, he also has to decide upon the data to be maintained by the system and for the system. The data is maintained in the form of data stores, which actually comprise of databases.

Each database may further be composed of several files where the data is actually stored. The analyst, during the design of the system, decides onto the various file-relating issues before the actual development of the system starts.

4.2 OBJECTIVE

After studying this Unit, you should be able to:

- What Is a Database?
- Objectives of data base.
- Define a Database Management System
- Description of the Database Management Structure
- What are the benefits of Database Management System
- Describe the features and capabilities of a typical DBMS
- What are the functions of a DBMS?
- What is database normalization?
- How database administrator works?

4.3 DATA BASE

A *database* is a mechanism that is used to store information, or data. Information is something that we all use on a daily basis for a variety of reasons. With a database, users should be able to store data in an organized manner. Once the data is stored, it should be easy to retrieve information. Criteria can be used to retrieve information. The way the data is stored in the database determines how easy it is to search for information based on multiple criteria. Data should also be easy to add to the database, modify, and remove.

A *legacy database* is simply a database that is currently in use by a company. The term legacy implies that the database has been around for several years. The term legacy can also imply that the existing database is not up to date with current technology. When a company has determined to design a new database, the existing database is considered the legacy database.

Examples of databases with which we are all familiar include

- Personal address books
- Telephone books
- Card catalogs at libraries
- Online bookstores
- Personal finance software
- Road maps

Some of these databases are static, whereas others are dynamic.

For example, a road map is a static database (in an abstract sense) that contains information such as states, cities, roadways, directions, distance, and so forth. By looking at a map, you can quickly establish your destination as related to your current location. Once the road map is printed, it is distributed and used by travelers to navigate between destinations. The information does not change on a map. There is no way to change the information on a map without printing new maps and redistributing them. A telephone book is also a static database because residential and commercial information is listed for a particular year. As with a road map, telephone book entries cannot be changed once the book is printed (unless they are changed by hand).

A personal address book is a good example of a dynamic database that many of us use on a daily basis. The address book is dynamic in the sense that entries are changed in the book as friends and family move or change telephone numbers. New friends can be added and old friends can be removed, although this can be done by hand. An online bookstore is also a dynamic database because orders are constantly being placed for books. On a regular basis, new authors and titles are added, titles are removed, inventory is updated, and so forth.

4.3.1 Objectives of Data Base

A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make information access easy, quick, inexpensive and flexible for the user. In data base design, several specific objectives are considered:

2. **Controlled redundancy: -** Redundant data occupies space and, therefore, is wasteful. If versions of the same data are in different phases of updating, the system often gives

conflicting information. A unique aspect of data base design is storing data only once, which controls redundancy and improves system performance.

3. **Ease of learning and use**: - A major feature of a user- friendly database package is how easy it is to learn and use. Related to this point is that a database can be modified without interfering with established ways of using the data.
4. **Data independence: -** An important database objective is changing hardware and storage procedures or adding new data without having to rewrite application programs. The database should be "tunable" to improve performance without rewriting programs.
5. **More information at low cost:** - Using, storing and modifying data at low cost are important. Although hardware prices are falling, software and programming costs are on the rise. This means that programming and software enhancements should be kept simple and easy to update.
6. **Accuracy and integrity: -** The accuracy of a database ensures that data quality and content remain constant. Integrity controls detect data inaccuracies where they occur.
7. **Recovery from failure: -** With multi-user access to a database, the system must recover quickly after it is down with no loss of transactions. This objective also helps maintain data accuracy and integrity.
8. **Privacy and security: -** For data to remain private, security measures must be taken to prevent unauthorized access. Database security means that data are protected from various forms of destruction; users must be positively identified and their actions monitored.
9. **Performance: -** This objective emphasizes response time to inquiries suitable to the use of the data. How satisfactory the response time is depends on the nature of the user-data base dialogue. For example, inquiries regarding airline seat availability should be handled in a few seconds. On the other extreme, inquiries regarding the total sale of a product over the past two weeks may be handled satisfactorily in 50 seconds.

4.3.2 Data Base Management System

A database management system is a set of software programs that allows users to create, edit and update data in database files, and store and retrieve data from those database files. Data in a database can be added, deleted, changed, sorted or searched all using a DBMS. If you were an employee in a large organization, the information about you would likely be stored in different files that are linked together. One file about you would pertain to your skills and abilities, another file to your income tax status, another to your home and office address and telephone number, and another to your annual performance ratings. By cross-referencing these files, someone could change a person's address in one file and it would automatically be reflected in all the other files. DBMSs are commonly used to manage:

- Membership and subscription mailing lists
- Accounting and bookkeeping information
- The data obtained from scientific research
- Customer information
- Inventory information
- Personal records
- Library information

4.3.3 Advantages of a DBMS

- **Improved availability:** One of the principle advantages of a DBMS is that the same information can be made available to different users.
- **Minimized redundancy:** The data in a DBMS is more concise because, as a general rule, the information in it appears just once. This reduces data redundancy, or in other words, the need to repeat the same data over and over again. Minimizing redundancy can therefore significantly reduce the cost of storing information on hard drives and other storage devices. In contrast, data fields are commonly repeated in multiple files when a file management system is used.
- **Accuracy:** Accurate, consistent, and up-to-date data is a sign of data integrity. DBMSs foster data integrity because updates and changes to the data only have to be made in one place. The chances of making a mistake are higher if you are required to change the same data in several different places than if you only have to make the change in one place.
- **Program and file consistency:** Using a database management system, file formats and system programs are standardized. This makes the data files easier to maintain because the same rules and guidelines apply across all types of data. The level of consistency across files and programs also makes it easier to manage data when multiple programmers are involved.
- **User-friendly:** Data is easier to access and manipulate with a DBMS than without it. In most cases, DBMSs also reduce the reliance of individual users on computer specialists to meet their data needs.
- **Improved security:** As stated earlier, DBMSs allow multiple users to access the same data resources. This capability is generally viewed as a benefit, but there are potential risks for the organization. Some sources of information should be protected or secured and only viewed by select individuals. Through the use of passwords, database management systems can be used to restrict data access to only those who should see it.

4.3.4 Disadvantages of a DBMS

There are basically two major downsides to using DBMSs. One of these is cost, and the other the threat to data security.

- **Cost:** Implementing a DBMS system can be expensive and time-consuming, especially in large organizations. Training requirements alone can be quite costly.

- **Security:** Even with safeguards in place, it may be possible for some unauthorized users to access the database. In general, database access is an all or nothing proposition. Once an unauthorized user gets into the database, they have access to all the files, not just a few. Depending on the nature of the data involved, these breaches in security can also pose a threat to individual privacy. Steps should also be taken to regularly make backup copies of the database files and store them because of the possibility of fires and earthquakes that might destroy the system.

4.4 FUNCTIONS OF DBMS

The functions performed by a typical DBMS are the following:

- **Data Definition**

  The DBMS provides functions to define the structure of the data in the application. These include defining and modifying the record structure, the type and size of fields and the various constraints/conditions to be satisfied by the data in each field.

- **Data Manipulation**

  Once the data structure is defined, data needs to be inserted, modified or deleted. The functions which perform these operations are also part of the DBMS. These function can handle planned and unplanned data manipulation needs. Planned queries are those which form part of the application. Unplanned queries are ad-hoc queries which are performed on a need basis.

- **Data Security and Integrity**

  The DBMS contains functions which handle the security and integrity of data in the application. These can be easily invoked by the application and hence the application programmer need not code these functions in programs.

- **Data Recovery and Concurrency**

  Recovery of data after a system failure and concurrent access of records by multiple users are also handled by the DBMS.

- **Data Dictionary Maintenance**

  Maintaining the Data Dictionary which contains the data definition of the application is also one of the functions of a DBMS.

- **Performance**

  Optimizing the performance of the queries is one of the important functions of a DBMS. Hence the DBMS has a set of programs forming the Query Optimizer which evaluates the different implementations of a query and chooses the best among them.

Thus the DBMS provides an environment that is both convenient and efficient to use when there is a large volume of data and many transactions to be processed.

4.5 DBMS LANGUAGES

A DBMS must provide appropriate languages and interfaces for each category of users to express database queries and updates. Database Languages are used to create and maintain database on computer. There are large numbers of database languages like Oracle, MySQL, MS Access, dBase, FoxPro etc. SQL statements commonly used in Oracle and MS Access can be categorized as data definition language (DDL), data control language (DCL) and data manipulation language (DML).

- **Data Definition Language (DDL)**

  It is a language that allows the users to define data and their relationship to other types of data. It is mainly used to create files, databases, data dictionary and tables within databases.

  It is also used to specify the structure of each table, set of associated values with each attribute, integrity constraints, security and authorization information for each table and physical storage structure of each table on disk.

- **Data Manipulation Language (DML)**

  It is a language that provides a set of operations to support the basic data manipulation operations on the data held in the databases. It allows users to insert, update, delete and retrieve data from the database. The part of DML that involves data retrieval is called a query language.

- **Data Control Language (DCL)**

  DCL statements control access to data and the database using statements such as GRANT and REVOKE. A privilege can either be granted to a User with the help of GRANT statement. The privileges assigned can be SELECT, ALTER, DELETE, EXECUTE, INSERT, INDEX etc. In addition to granting of privileges, you can also revoke (taken back) it by using REVOKE command.

4.6 LOGICAL AND PHYSICAL VIEWS OF DATA

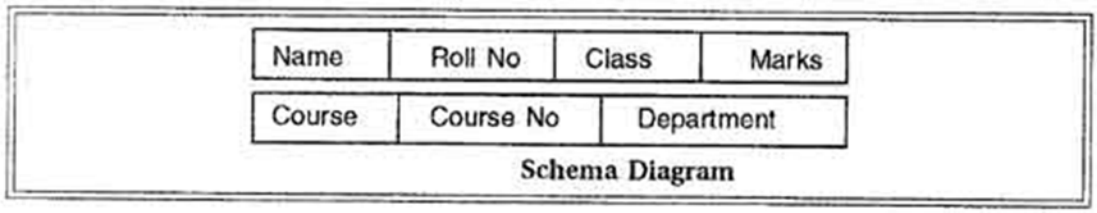
In data base design, several views of data must be considered along with the persons who use them. In addition to data structuring, where relationships are reflected between and within entities, we need to identify the application program's logical views of data within an overall logical data structure. The logical view is what the data look like, regardless of how they are stored. The physical view is the way data exist in physical storage. It deals with how data are stored, accessed, or related to other data in storage. Four views of data exist: three logical and one physical. The logical views are the user's view, the programmer's view and the overall logical view, called a schema.

4.7 SCHEMAS AND SUBSCHEMAS

The overall design of the database is called the database schema. The schema will remain the same while the values filled into it change from instant to instant. When the schema framework is filled in with data item values, it is referred as an instance of the schema. The data in the database at a particular moment of time is called a database state or snapshot, which is also called the current set of occurrences or instances in the database

In other words, "the description of a database is called the database schema, which is specified during database design and is not expected to change frequently". A displayed schema is called a schema diagram.

For Example: Student-Schema.



A subschema is a subset of the schema and inherits the same property that a schema has. The plan (or scheme) for a view is often called subschema. Subschema refers to an application programmer's (user's) view of the data item types and record types, which he or she uses. It gives the users a window through which he or she can view only that part of the database, which is of interest to him. Therefore, different application programs can have different view of data.

4.8 DATABASE NORMALIZATION

Normalization is the process of removing redundant data from your tables in order to improve storage efficiency, data integrity and scalability. This improvement is balanced against an increase in complexity and potential performance losses from the joining of the normalized tables at query-time. There are two goals of the normalization process: eliminating redundant data (for example, storing the same data in more than one table) and ensuring data dependencies make sense (only storing related data in a table). Both of these are worthy goals as they reduce the amount of space a database consumes and ensure that data is logically stored.

Normalization is the aim of well design Relational Database Management System (RDBMS). It is step by step set of rules by which data is put in its simplest forms. We normalize the relational database management system because of the following reasons:

- Minimize data redundancy i.e. no unnecessarily duplication of data.
- To make database structure flexible i.e. it should be possible to add new data values and rows without reorganizing the database structure.
- Data should be consistent throughout the database i.e. it should not suffer from following anomalies.
- **Insert Anomaly** - Due to lack of data i.e., all the data available for insertion such that null values in keys should be avoided. This kind of anomaly can seriously damage a database
- **Update Anomaly** - It is due to data redundancy i.e. multiple occurrences of same values in a column. This can lead to inefficiency.
- **Deletion Anomaly** - It leads to loss of data for rows that are not stored elsewhere. It could result in loss of vital data.
- Complex queries required by the user should be easy to handle.
- On decomposition of a relation into smaller relations with fewer attributes on normalization the resulting relations whenever joined must result in the same relation without any extra rows. The join operations can be performed in any order. This is known as Lossless Join decomposition.
- The resulting relations (tables) obtained on normalization should possess the properties such as each row must be identified by a unique key, no repeating groups, homogenous columns, each column is assigned a unique name etc.

4.8.1 Advantages of Normalization

The following are the advantages of the normalization.

- More efficient data structure.
- Avoid redundant fields or columns.
- More flexible data structure i.e. we should be able to add new rows and data values easily
- Better understanding of data.
- Ensures that distinct tables exist when necessary.

- Easier to maintain data structure i.e. it is easy to perform operations and complex queries can be easily handled.
- Minimizes data duplication.
- Close modeling of real world entities, processes and their relationships.

4.8.2 Disadvantages of Normalization

The following are disadvantages of normalization.

- You cannot start building the database before you know what the user needs.
- On Normalizing the relations to higher normal forms i.e. 4NF, 5NF the performance degrades.
- It is very time consuming and difficult process in normalizing relations of higher degree.
- Careless decomposition may leads to bad design of database which may leads to serious problems.

4.8.3 Different type of Normalization

There are six normal forms

- First Normal Form
- Second Normal Form
- Third Normal Form
- Boyce-Cod Normal Form
- Fourth Normal Form
- Fifth Normal Form
- Sixth or Domain-key Normal form

Normal form	Rule
• First Normal Form	An entity type is in 1NF when it contains no repeating groups of data.
• Second Normal Form	An entity type is in 2NF when it is in 1NF and when all of its non-key attributes are fully dependent on its primary key.
• Third Normal Form	An entity type is in 3NF when it is in 2NF and when all of its attributes are directly dependent on the primary key.
• Boyce-Codd Normal Form	An entity type is in BCNF if it is in 3rd NF and every determinant can be used as a Primary Key.
• Fourth Normal Form	An entity type is in the 4NF if it is in BCNF and has no attribute with multivalued dependencies.

• Fifth Normal Form	An entity type is said to be in the 5NF if and only if it is in 4NF and every join dependency in it is implied by the candidate keys.
• Sixth or Domain-key Normal form	DKNF requires that each key uniquely identifies each row in a table. By enforcing key and domain restrictions, the database is assured of being freed from modification anomalies.

4.9 ROLE OF A DATABASE ADMINISTRATOR

The Database Administrator (DBA) who is like the super-user of the system. The role of the DBA is very important and is defined by the following functions.

- **Defining the Schema**

  The DBA defines the schema which contains the structure of the data in the application. The DBA determines what data needs to be present in the system ad how this data has to be represented and organized.

- **Liaising with Users**

  The DBA needs to interact continuously with the users to understand the data in the system and its use.

- **Defining Security & Integrity Checks**

  The DBA finds about the access restrictions to be defined and defines security checks accordingly. Data Integrity checks are also defined by the DBA.

- **Defining Backup / Recovery Procedures**

  The DBA also defines procedures for backup and recovery. Defining backup procedures includes specifying what data is to backed up, the periodicity of taking backups and also the medium and storage place for the backup data.

- **Monitoring Performance**

  The DBA has to continuously monitor the performance of the queries and take measures to optimize all the queries in the application.

Check Your Progress 1

6. What do you mean DBMS?

_____

_____

_____

7. What do you mean by normalization?

_____

_____

_____

8. Explain various functions of DBMS.

_____

_____

_____

9. Differentiate between logical and physical view of data.

_____

_____

_____

10. What do you mean by schema?

_____

_____

_____

4.10 SUMMARY

- A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make information access easy, quick, inexpensive and flexible for the user.
- A DBMS is a complex set of software programs that controls the organization, storage, management, and retrieval of data in a database.
- There are basically two major downsides to using DBMSs. One of these is cost, and the other the threat to data security.
- The functions performed by a typical DBMS are the following:
    - Data Definition
    - Data Manipulation
    - Data Security and Integrity
    - Data Recovery and Concurrency
    - Data Dictionary Maintenance
    - Performance

- Database languages can be categorized as data definition language (DDL), data control language (DCL) and data manipulation language (DML).
- Normalization is the process of removing redundant data from your tables in order to improve storage efficiency, data integrity and scalability. This improvement is balanced against an increase in complexity and potential performance losses from the joining of the normalized tables at query-time.
- The role of the DBA is very important and is defined by the following functions.
    - Defining the Schema
    - Liaising with Users
    - Defining Security and Integrity Checks
    - Defining Backup / Recovery Procedures
    - Monitoring Performance

4.11 KEYWORDS

- Data Base
- Data Base management System
- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)
- Schema
- Subschema
- Logical View
- Physical View
- Data Base Administrator (DBA)
- Boyce-Codd Normal Form (BCNF)

4.12 MODEL ANSWERS

Check Your Progress 1

1. A database management system is a set of software programs that allows users to create, edit and update data in database files, and store and retrieve data from those database files. Data in a database can be added, deleted, changed, sorted or searched all using a DBMS.
2. Normalization is the process of removing redundant data from tables in order to improve storage efficiency, data integrity and scalability. This improvement is balanced against an increase in complexity and potential performance losses from the joining of the normalized tables at query-time.
3. DBMS performs several important functions:
    - Storing, retrieving, and updating data.
    - Creating program and data independence. Either one can be altered independently of the other.

- Enforcing procedures for data integrity. Data are immune from deliberate alteration because the programmer has no direct method of altering physical databases.
- Reducing data redundancy. Data are stored and maintained only once.
- Providing security facilities for defining users and enforcing authorization. Access is limited to authorized users by passwords or similar schemes.
- Reducing physical storage requirements by separating the logical and physical aspects of the database.

4. The logical view is what the data look like, regardless of how they are stored. The physical view is the way data exist in physical storage. It deals with how data are stored, accessed, or related to other data in storage.
6. The overall structure of a data base is called schema.

4.13 TERMINAL QUESTIONS

1. What are the steps used in the design on Database design.
2. What is DBMS?
3. What are the advantages of DBMS?
4. What is normalization?
5. What is DDL (Data Definition Language)?
6. What is database or database management systems (DBMS)?
7. What is normalization? What is different type of normalization?
8. What is the role of DBA?
9. What are the functions of a DBMS?
10. Differentiate between

    a) Logical and Physical view of data

    b) Schema and Subschema

Uttar Pradesh Rajarshi Tandon
open University

MCA-E4/
PGDCA-E4

Master in Computer Application

BLOCK

4

SYSTEM IMPLEMENTATION

Course Design Committee

Dr. Ashutosh Gupta — **Chairman**
Director-In-charge,
School of Computer and Information Science, UPRTOU, Allahabad

Prof. R. S. Yadav — **Member**
Department of Computer Science and Engineering
MNNIT-Allahabad, Allahabad

Ms. Marisha — **Member**
Assistant Professor, Computer Science
School of Science, UPRTOU, Allahabad

Mr. Manoj Kumar Balwant — **Member**
Assistant Professor, Computer Science
School of Science, UPRTOU, Allahabad

Course Preparation Committee

Dr. Saurav Pal — **Author**
Associate Professor
Department of MCA, VBS Purvanchal University
Jaunpur-222001, Uttar Pradesh

Prof. R. R. Tiwari — **Editor**
University of Allahabad
Allahabad, Uttar Pradesh

Dr. Ashutosh Gupta
Director (In-charge), School of Computer and Information Science,
UPRTOU, Allahabad

Ms. Marisha (Coordinator)
Assistant Professor, School of Sciences,
UPRTOU, Allahabad

Printed and Published by Pro. (Dr.) Girija Shankar Shukla Registrar, **Uttar Pradesh Rajarshi Tandon open University, 2017.**
Printed By : Chandrakal Universal Pvt. Ltd. 42/7 Jawahar Lal Neharu Road, Allahabad

UNIT - I

SYSTEM TESTING

UNIT STRUCTURE

1.1 INTRODUCTION

The last phases of system development life cycle are testing, maintenance and implementation. In this unit, we study these phases in details. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. A strategy must provide guidance for the practitioner and a set of milestones for the manager. Because the steps of the test strategy occur at a time when dead-line pressure begins to rise, progress must be measurable and problems must surface as early as possible.

1.2 OBJECTIVES

After studying this Unit, you should be able to:

- Why systems are tested?
- Need of testing.
- What are test plans?
- What steps are taken to test systems?
- Various special system tests.
- The goal of quality assurance in the system life cycle.

1.3 SYSTEM TESTING

Testing ensures that the system still meets the client needs (functional requirements). The intent of System Test is to find defects and correct them before implementation. There is no approach or method to guarantee a system completely free of defects. However, following a System Test approach will assist in mitigating risks and ensuring a successful system.

1.3.1. Need of Testing

A system testing is an expensive but critical process that can take as much as 50 percent of the budget for program development. The common view of testing held by users is that it is performed to prove that there are no errors in a program. However, this is virtually impossible, since analysts cannot prove that system is free and clear of errors.

Therefore, the most useful and practical approach is with the understanding that testing is the process of executing a program with explicit intention of finding errors that is, making the program fail. The tester, who may be an analyst, programmer, or specialist trained in software testing, is actually trying to make the program fail. A successful test, then, is one that finds an error.

Analysts know that an effective testing program does not guarantee systems reliability. Reliability is a design issue. Therefore, reliability must be designed into the system. Developers cannot test for it.

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved. Inadequate testing or non-testing leads to errors. This creates two problems

1. The time lag between the cause and the appearance of the problem, and
2. The effect of system errors on files and records within the system.

System testing is done to make the user known of the system before implementation.

The first test of a system is to see whether it produces the correct outputs. No other test can be more crucial. A variety of other tests are done.

1. **Online response** – Online systems must have a response time that will not cause a hardship to the user. One way to test this is to input many transactions during peak hours and time the response to each online function to establish a true performance level.
2. **Volume** - We create as many records as would normally be produced to verify that the hardware and software will function correctly. The user is asked to provide test data for volume testing.
3. **Stress testing** – The purpose of stress testing is to prove that the candidate system does not malfunction under peak loads. Unlike volume testing where time is not a factor, we subject the system to a high volume of data over a short period of time. This simulates an online environment where a high volume of activities occur.
4. **Recovery and security-** A forced system failure is induced to test a backup recovery procedure for file integrity. Inaccurate data are entered to see how the system responds in terms of error detection and protection. Related to file integrity is a test to demonstrate that data and programs are secure from unauthorized access.
5. **Usability documentation and procedure** – The usable test verifies the user-friendly nature of the system. This relates to normal operating and error-handling procedures.

1.3.2. Test Plan

The first step in system testing is to prepare a plan that will test all aspects of the system in a way that promotes its credibility among potential users.

A test plan consists of the following activities

- **Prepare test plan:**

  A test plan includes the following items

  - Outputs expected from the system
  - Criteria for evaluating outputs
  - A volume of test data
  - Procedure for using test data
  - Personnel and training requirements

- **Specify conditions for user acceptance testing**

  Planning for user acceptance testing calls for the analyst and the user to agree on the conditions for the test. Many of these conditions may be derived from the test plan. Others are an agreement on the test schedule, the test duration and the persons designated for the test. The start and termination dates for the test should also be specified in advance.

- **Prepare Test data for program testing**

  As each program is coded, test data are prepared and documented to ensure that all aspects of the program are properly tested. The data are filed for future reference.

- **Prepare Test data for transaction path testing:**

  This activity develops the data required for testing every condition and transaction to be introduced into the system. The path for each transaction from origin to destination is carefully tested for reliable results.

- **Plan user training:**
  User training is designed to prepare the user for testing and converting the system. User involvement and training take place parallel with programming for three reasons.
  - The system group has time for training.
  - Gives the systems group a clearer image of the user's interest in the new system.
  - A trained user participates more effectively in system testing

  For user training, preparation of a checklist is useful. This includes provisions for developing training materials like training manuals and other documents to complete training activity. Facility requirements and the necessary hardware are specified and documented. A common procedure is used to train the supervisors for the following reasons.

  - User supervisors are knowledgeable about the capabilities of their staff.
  - Staff members usually respond more favorably and accept instructions better from supervisors than from outsiders
  - Familiarity of users with their particular problems makes them better candidates for handling user training than the systems analyst. The analyst gets feedback to ensure that proper training is provided

- **Compile and assemble programs:**
  All programs have to be compiled/ assembled for testing. Before this, a complete program description should be available. It should include the purpose of the program, its use, the programmers who prepared it and the amount of computer time it takes to run it. Program and system flowcharts of the project should be available for reference. Three types of activities are done in this phase
  - **Desk checking** the source code uncovers programming errors.
  - **A run order schedule** specifies the transactions to test and the order in which they should be tested.
  - **A test scheme** specifies how program should be debugged. Consists of 2 approaches as follows
    - Bottom-up approach tests small-scale program modules which are linked to a higher level module and so on until the program is completed.
    - Top-down approach where the general program is tested first, followed by the addition of program modules one level at a time to the lowest level.
- **Prepare Job performance Aids:**
  In this activity the materials to be used by personnel to run the system are specified and scheduled. This includes a display of materials such as program codes, a list of input codes attached to the CRT terminal, and a posted instruction schedule to load the disk drive. These aids reduce the training and employ personnel at lower positions.
- **Prepare Operational Documents:**

  During the test plan stage, all operational documents are finalized. Also section on the experience, training and educational qualifications of personnel for the proper operation of the new system is also documented.

1.3.3. Levels of Test

Systems are not designed as entire systems nor are they tested as single systems. The analyst must perform both unit and integration testing.

- ## Unit Testing

  In unit testing the analyst tests the programs making up a system. (For this reason unit testing is sometimes called program testing.) The software units in a system are the modules and routines that are assembled and integrated to perform a specific function. In a large system, many modules at different levels are needed.

  Unit testing focuses first on the modules, independently of one another, to locate errors. This enables the tester to detect errors in coding and logic that are contained within that module alone. Those resulting from the interaction between modules are initially avoided.

  For example, a hotel information system consists of modules to handle reservations; guest check-in and checkout; restaurant, room service, and miscellaneous charges; convention activities; and accounts receivable billing. For each, it provides the ability to enter, change, or retrieve data and respond to inquiries or print reports. The test cases needed for unit testing should exercise each condition and option.

  For example, test cases are needed to determine how the system handles attempts to check-in guests who do and do not have reservations, as well as those instances involving changing the name on the reservation when a person other than the one listed arrives.

Also needed are test cases for the checkout situations of paying the exact amount of the bill, only part of the bill, and more than the amount shown. Even checking out without making any payment at all must be included in a test case.

If the module receives input or generates output, test cases are also needed to test the range of values expected, including both valid and invalid data. What will happen in the hotel checkout example if a guest wishes to make a payment of Rs. 1,00,000 for an upcoming convention? Are the payments and printing modules designed to handle this amount? Testing for this question quickly detects existing errors.

If the module is designed to perform iterations, with specific processes contained within a loop, it is advisable to execute each boundary condition: 0 iteration, 1iteration though the loop, and the maximum number of iterations through the loop. Of course, it is always important to examine the result of testing, but special attention should be given to these conditions. Analysts too often make the mistake of assuming that a case of 0 iteration will automatically be handled properly.

Unit testing can be performed from the bottom up, starting with the smallest and lowest – level modules and proceeding one at a time. For each module in bottom-up testing, a short program (called a driver program because it drives or runs the module) executes the module and provides the needed data, so that the module is asked to perform the way it will when embedded within the larger system. When bottom-level modules are tested, attention turns to those on the next level that use the lower – level ones. They are tested individually and then linked with the previously examined lower – level modules.

Top-down testing, as the name implies, begins with the upper – level modules. However, since the detailed activities usually performed in lower-level routines are not provided (because those routines are not being tested), stubs are written. A stub is a module shell that can be called by the upper – level module and that, when reached properly, will return a message to the calling module, indicating that proper interaction occurred. No attempt is made to verify the correctness of the lower-level module.

- ## Integration testing

Integration testing does not test the software per se but rather the integration of each module in the system. It also tests to find discrepancies between the system and its original objective, current specifications, and systems documentation. The primary concern is the compatibility of individual modules. Analysts are trying to find areas where modules have been designed with different specifications for data length, type, and data element name. For example, one module may expect the data item for customer identification number to be a numeric field, while other modules expect it to be a character data item. The system itself may not report this as an error, but the output may show unexpected results. If a record created and stored in one module, using the identification number as a numeric field, is later sought on retrieval with the expectation that it will be a character field, the field will not be recognized and the message “REQUESTED RECORD NOT FOUND” will be displayed.

Integration testing must also verify that file sizes are adequate and that indices have been built properly. Sorting and re-indexing procedures assumed to be present in lower-level modules must be tested at the systems level to see that they in fact exist and achieve the results modules expect.

- **Regression Testing**

  Each time a new module is added as part of integration testing, the software changes. New data flow paths are established, new I/O may occur, and new control logic is invoked. These changes may cause problems with functions that previously worked flawlessly. In the context of an integration test strategy, regression testing is the re-execution of some subset of tests that have already been conducted to ensure that changes have not propagated unintended side effects.

  In a broader context, successful tests (of any kind) result in the discovery of errors, and errors must be corrected. Whenever software is corrected, some aspect of the software configuration (the program, its documentation, or the data that support it) is changed. Regression testing is the activity that hclp5 to en5ure that changes (due to testing or for other reasons) do not introduce unintended behavior or additional errors.

  For instance, suppose you are going to add new functionality to your software, or you are going to modify a module to improve its response time. The changes, of course, may introduce errors into software that was previously correct. For example, suppose the program fragment

  ```
  x := c + 1 ;
  proc (z);
  c := x + 2; x:= 3;
  ```

  works properly. Now suppose that in a subsequent redesign it is transformed into

  ```
  proc(z);
  c := c + 3;
  x:= 3;
  ```

  in an attempt at program optimization. This may result in an error if procedure proc accesses variable x. Thus, we need to organize testing also with the purpose of verifying possible regressions of software during its life, i.e., degradations of correctness or other qualities due to later modifications. Properly designing and documenting test cases with the purpose of making tests repeatable, and using test generators, will help regression testing. Conversely, the use of interactive human input reduces repeatability and thus hampers regression testing.

  Finally, we must treat test cases in much the same way as software. It is clear that such factors as resolvability, reusability, and verifiability are just as important in test cases as they are in software. We must apply formality and rigor and all of our other principles in the development and management of test cases.

- **Soak Testing**

  Soak testing also known as Endurance Testing is performed to determine if the application under test can sustain the continuous loads. Soak testing is a type of performance testing, wherein software running is subjected to high load over a prolonged duration of time. Soak testing may go on for few days or even for few weeks.

  Soak testing is a type of testing that is conducted to find errors that result in deterioration of software performance with continued usage. Soak testing is extensively done for electronic devices, which are expected to run continuously for days or months or years without restarting or

rebooting. With growing web applications soak testing has gained significant importance as web application availability is critical for sustaining and success of business.

For example, running several times with huge transactions in an entire day (or night) greater than expected in a busy day, to identify and performance problems that appear after a large number of transactions has been executed.

This testing is also called Aging or Longevity Testing.

- **Gorilla Testing**

Objective of Gorilla Testing is to exercise one or few functionality thoroughly or exhaustively by having multiple people test the same functionality. Gorilla Testing is that in which tester and sometimes developer test one particular module heavily, means they test functionality of one particular module heavily.

Gorilla Testing is that which is always used to describe repetitive and boring (frustrating) Manual Testing process which a tester has already done a hundred times before. Due to this repetitive manual testing process Gorilla Testing is also known by the name Frustrating Testing.

Actually Gorilla Testing is used in Software Testing to check functionality of one particular module heavily. As you know Software Testing is a very wide concept so concept of Gorilla Testing is also wide in nature.

- **Alpha Testing**

This test is the first stage of testing and will be performed amongst the teams (developer and QA teams). Unit testing, integration testing and system testing when combined are known as alpha testing. During this phase, the following will be tested in the application:

  - Spelling Mistakes
  - Broken Links
  - Cloudy Directions

The Application will be tested on machines with the lowest specification to test loading times and any latency problems.

This is a formal testing performed by the end-users at development site. This is the previous stage to Beta Testing.

- **Beta Testing**

A formal testing conducted by end customers before releasing to market. A successful completion of Beta Testing confirms to customer acceptance of the software.

This test is performed after Alpha testing has been successfully performed. In beta testing a sample of the intended audience tests the application. Beta testing is also known as pre-release testing. Beta test versions of software are ideally distributed to a wide audience on the Web, partly to give the program a “real-world” test and partly to provide a preview of the next release. In this phase the audience will be testing the following:

  - Users will install, run the application and send their feedback to the project team.

- Typographical errors, confusing application flow, and even crashes.
- Getting the feedback, the project team can fix the problems before releasing the software to the actual users.
- The more issues you fix that solve real user problems, the higher the quality of your application will be.
- Having a higher-quality application when you release to the general public will increase customer satisfaction.

1.3.4 Special Systems Tests

There are other tests that are in special category, since they do not focus on the normal running of the system. Six tests are essential.

1. **Peak Load Testing**

   There are critical times in many systems, particularly online systems. For example, in a banking system, analysts want to know what will happen if all teller sign on at their terminals at the same time before the start of the business day. Will the system handle them one at a time without incident, will it attempt to handle all of the at once and be so confused that it "locks up" and must be restarted, or will terminal addresses be lost?

   The only sure way to find out is to test for it. The same situations can arise when tellers' sign out during lunch periods and at the end of the day, so testing is looking at real situations.

2. **Storage Testing**

   Analysts specify a capacity for the system when it is designed and constructed. Capacities are measured in terms of the number of records that a disk will handle or a file can contain. These capacities are linked to disk space and the size of indices, record keys, and so on. But they too must be tested. If the documentation for a new system to be run on a microcomputer claims that a disk file can store up to 10,000 records, each 393 bytes long, the claim must be verified before implementation.

   Storage testing often requires entering data until the capacity is reached. Comparing the actual and claimed capacities will verify the accuracy of the documentation on the one hand and allow a judgment about actual capacity at the same time. Many, many systems are never tested in this way. Users find out too late that claims made during installation are not true: there is not enough storage capacity for transactions and master file records.

3. **Performance Time Testing**

   When analysts are developing a design, their concerns are more on reports, inputs, files, and processing sequences than on performance time, although this changes with experience. During simple unit and integration testing, relatively small sets of data are used to find errors or cause failures. Therefore, users frequently find out how slow or fast the response time of the system is only after it has been installed and loaded up with data.

   That may be too late. Systems are rarely too fast for users. Performance time testing is conducted prior to implementation to determine how long it takes to receive a response to an inquiry, make a backup copy of a file, or send a transmission and receive a response. It also includes test runs to time indexing or resorting of large files of the size the system will have during a typical run or to prepare a report.

A system that runs well with only a handful of test transactions may be unacceptably slow when fully loaded. And the time to know about this is prior to implementation, when adjustments can be more easily made. Once files are fully loaded and the user is relying on the system for daily activities, it is difficult to pull it back and being large- scale changes. The user needs the system and the analyst will not want to risk the loss of live data.

4. **Recovery Testing**

   Analysts must always assume that the system will fail and data will be damaged or lost. Even though plans and procedures are written to cover these situations, they also must be tested. By creating a failure or data loss event where the users are forced to reload and recover from a backup copy, analysts can readily determine whether recovery procedures are adequate. The best – designed plans usually are adjusted or augmented after this test.

5. **Procedure Testing**

   Documentation and run manuals tell the user how to perform certain functions are tested quite easily by asking the user to follow them exactly through a series of events. It is surprising how not including instructions about when to depress the enter key, about removing diskettes before powering down, or what to do when the paper – out light on the printer lights up can raise questions.

   There is, of course, no substitute for a well – designed set of procedure manuals. Analysts concentrate on the major and critical details of a systems design and include them in the documentation. They also pay attention to the little details, when designing the system. But often descriptions of the details do not get into the documentation. This type of testing not only shows where they are needed but also where they are wrong, that is, where actions suggested in the documentation do not match those that must actually be taken to make the system.

6. **Human factors Testing**

   What do users do if, after submitting a transaction through a terminal, the screen goes blank while the data are being processed? They may not take the actions the analyst wants or expects, instead responding in unusual ways: they may depress the send key several times, turn the power switch on the terminal off and back on, unplug it and replug it, or beat on the terminal. Obliviously, they will do just about anything if the analyst has not given them some message on the screen to indicate that their request has been received, that it is being processed, and that there will be a short delay. This is what human factors testing is all about – finding answers to questions about how people will react to the system in ways not anticipated. And as a general rule, as strange as the above actions may sound, the people are right; they are taking actions that are normal under the circumstances.

It is the responsibility of the analyst to anticipate questions that will arise in the minds of the users as they interact with the system. If a screen will go blank during transaction processing, the analyst should make sure that it displays a message informing the user that processing is occurring. Even that is not enough if the delay will be more than a second or two. For processing that will take long periods, the analyst should have the screen give the user a message telling approximately how long it will take and providing an option to cancel the request. The user may decide to have that one – hour job run some other time when the system is not so busy.

If the system is going into a long sorting step, the analyst should keep the user informed about how much of the sort is completed. Users appreciate systems that display the numbers of records sorted or the percentages completed.

Also the analyst should be sure to watch how people enter data. Do they use different keystroke from those anticipated (such as the top row of numbers on the typewriter pad rather than those on the numeric keypad)? Are any keystrokes awkward and therefore error prone (for example, having to hold down the shift key with the little finger while depressing the + key with the index finger)?

How will the user of a system feel after working with the system for a lengthy period of time? Glare on the screen or simply too much detail on one display is physically and mentally irritating. Slight modifications in the display contents or the location of the equipment are important human factor concerns that dramatically affect the user and, therefore, the system over time.

These simple testing questions are of monumental importance and extremely helpful in finding flaws that can cause the system to fail. Some analysts will find these flaws the hard way – through bad experiences. It is difficult to forget the system that was damaged because a user banged on the terminal when data were submitted and accepted by the system without displaying a response. But, following the guidelines above, the analyst can avoid those situations.

Check Your Progress 1

1. What do you mean by testing?

   _____

   _____

   _____

2. What do you mean by test plan?

   _____

   _____

   _____

3. Describe briefly about unit testing.

   _____

   _____

   _____

4. What do you mean by various special system testing? Explain briefly.

   _____

   _____

   _____

1.4. QUALITY ASSURANCE

Quality assurance is the review of software products and related documentation for completeness, correctness, reliability, and maintainability. And, of course, it includes assurance that the system meets the specifications and the requirements for its intended use and performance.

1.4.1. Quality Factors Specifications

The goal of this stage is to define the factors that contribute to the quality of the candidate system. Several factors determine the quality of a system:

1. Correctness-the extent to which a program meets system specifications and user objectives.
2. Reliability-the degree to which the system performs its intended functions over a. time.
3. Efficiency-the amount of computer resources required by a program to perform a function.
4. Usability-the effort required to learn and operate a system.
5. Maintainability-the ease with which program errors are located and corrected.
6. Testability-the effort required to test a program to ensure its correct performance.
7. Portability-the ease of transporting a program from one hardware configuration to another.
8. Accuracy-the required precision in input editing, computations, and output.
9. Error tolerance-error detection and correction versus error avoidance.
10. Expandability-ease of adding or expanding the existing database.
11. Access control and audit-control of access to the system and extent to which that access can be audited.
12. Communicativeness-how descriptive or useful the inputs and out of the system are.

1.4.2 Software requirements Specifications:

The quality assurance goal of this stage is to generate the requirements document that provides the technical specifications for the design and development of the software. This document enhances the system's quality by improving communication between the system developer and the user and provides the proper information for accurate documentation.

1.4.3 Software Design Specifications:

The software design document defines the overall architecture of the software that provides the functions and features described in the software requirements document. It specifies how it will be done. The document describes the logical subsystems and their respective physical modules. It ensures that all conditions are covered.

1.4.4 Software Testing and Implementation:

The quality assurance goal of the testing phase is to ensure that completeness and accuracy of the system and minimize the retesting process. In the implementation phase, the goal is to provide a logical order for the creation of the modules and in turn the creation of the system.

1.4.5 Maintenance and Support:

This phase provides the necessary software adjustment for the system to continue to comply with the original specifications. The quality assurance goal is to develop a procedure for correcting errors and enhancing software. This procedure improves quality assurance by encouraging complete reporting and logging of problems, ensuring that reported problems are promptly forwarded to the appropriate group

for resolution and reducing redundant effort by making known problem reports available to any department that handles complaints.

1.4.6 Levels of Quality Assurance

Analysts use four levels of quality assurance: testing, verification, validation, and certification.

- **Testing**

  A system testing is an expensive but critical process that can take as much as 50 percent of the budget for program development. The common view of testing held by users is that it is performed to prove that there are no errors in a program. However, this is virtually impossible, since analysts cannot prove that software is free and clear of errors.

  Therefore, the most useful and practical approach is with the understanding that testing is the process of executing a program with explicit intention of finding errors that is, making the program fail. The tester, who may be an analyst, programmer, or specialist trained in software testing, is actually trying to make the program fail. A successful test, then, is one that finds an error.

  Analysts know that an effective testing program does not guarantee systems reliability. Reliability is a design issue. Therefore, reliability must be designed into the system. Developers cannot test for it.

- **Verification and validation**

  Like testing, verification is also intended to find errors. Executing a program in a simulated environment performs it. Validation refers to the process of using software in a live environment on order to find errors.

  When commercial systems are developed with the explicit intention of distributing them to dealers for sale or marketing them through company – owned field offices, they first go through verification, some-times called alpha testing. The feedback from the validation phase generally produces changes in the software to deal with errors and failures that are uncovered. Then a set of user sites is selected that puts the system into use on a live basis. These beta test sites use the system in day- to - day activities; they process live transactions and produce normal system output. The system in live in every sense of the word, except that the users are aware they are using a system that can fail. But the transactions that are entered and the persons using the system are real. Validation many continue for several months. During the course of validating the system, failure may occur and the software will be changed. Continued use may produce additional failures and the need for still more change.

- **Certification**

  Software certification is an endorsement of the correctness of the program, an issue that is rising in importance for information systems applications. There is an increasing dependence on the purchase or lease of commercial software rather than on its in-house development. However, before analysts are willing to approve the acquisition of a package, they often require certification of the software by the developer or an unbiased third party.

  For example, selected accounting firms are now certifying that a software package in fact does what the vendor claims it does and in a proper manner. To so certify the software, the agency

appoints a team of specialists who of specialists who carefully examine the documentation for the system to determine what the vendor claims the system does and how it is accomplished. Then they test the software against those claims.

If no serious discrepancies or failures are encountered, they will certify that the software does what the documentation claims. They do not, however, certify that the software is the right package for a certain organization. That responsibility remains with the organization and its team of analysts.

1.5. TRENDS IN TESTING

We can expect rapid growth in the development and use of automated tools and software aids for testing. One such tool is functional tester, which determines whether the hardware is operating up to a minimal standard. It is a computer program that controls the complete hardware configurations and verifies that it is functional. For example, it can test computer memory by performing read and write test, and it tests each peripheral device individually.

The functional tester is of greater value when minute hardware problems are disguised as software bugs. For example, hardware faults are usually repeatable whereas software bugs are generally erratic. Problems arise when the delicate interaction between hardware and software cause a hardware problem to appear as an erratic software bug. A function tester determines immediately that the problem is in the hardware. This saves considerable time during testing.

Another software aid is the debug monitor. It is computer program that regulates and modifies the applications software that is being tested. It can also control the execution of the functional test and automatically patches are modifying the application program being tested.

Check Your Progress 2

1. List out various factors which are responsible for quality of a system.

   _____

   _____

   _____

2. Explain briefly the different levels of quality assurance.

   _____

   _____

   _____

3. Describe verification and validation.

   _____

   _____

   _____

1.6. AUDIT TRAILS

An important function of system controls is providing for an audit trail. An audit trail is a routine designed to allow the analyst, user or auditor to verify a process or an area in the new system.

- **Definition of Audit trail**

  A feature of data processing systems that allows for the study of data as processed from step to step, an auditor may then trace all transactions that affect an account. In a manual system, the audit trail includes journals, ledgers and other documents used by auditor to trace transactions. In a computerized system, record content and format frequently make it difficult to trace a transaction completely. Some reasons are the following:

  1. Files stored on the tape or disk can be read only by a computer, which limits the auditing function. A data dump is possible, though, to compare the data against a data map.
  2. Direct data entry eliminates the physical documentation for an audit program.
  3. Data processing activities are difficult to observe, since they take place within the computer system.

  For the audit trail to show its impact a detailed file of the transactions need to be maintained. During evaluation of a system following steps should be considered.

  1. Define the control objectives as separate design and test requirements. Input preparation and transmission by the user are important control areas that are viewed with an emphasis on audit trails and adequate documentation during testing.
  2. Examine budget costs to see whether system testing is within the limits.
  3. Review specifications. The auditor should evaluate program acceptance test specifications and assist the programmer in developing test standards, levels of testing and actual test conditions.

  It is the auditor's responsibility to build controls into candidate systems to ensure reliability, integrity and confidence of the users at all levels. The auditor should be called in during design as well as testing so that suggestion can be considered before implementation. Including the auditor in the system development team makes it easy for monitoring testing procedures and considers the acceptance of new controls to replace those changed by the new design.

1.7 SUMMARY

- Testing is the practice of making objective judgments regarding the extent to which the system (device) meets, exceeds or fails to meet stated objectives.
- The philosophy behind testing is to find errors. There are two general plans for testing software: The strategies of code testing ad specification testing.
- Systems are not designed as entire systems nor are they tested as single systems. The analyst must perform both unit and integration testing. There are other tests that are in special category, since they do not focus on the normal running of the system. Six tests are essential.
  - Peak load testing,
  - Storage testing,
  - Performance time testing,
  - Recovery testing,
  - Procedure testing
  - Human factor testing.

- The system testing contains the following
    - Prepare test plan
    - Specify conditions for user acceptance testing
    - Prepare test data for program testing
    - Prepare test data for transaction path testing
    - Plan user training
    - Compile/ assemble programs.
    - Prepare job performance aids.
    - Prepare operational documents.
- Quality assurance is the review of software products and related documentation for completeness, correctness, reliability, and maintainability. And, of course, it includes assurance that the system meets the specifications and the requirements for its intended use and performance.
- Four levels of quality assurance: testing, verification, validation, and certification.
- An important function of system controls is providing for an audit trail. An audit trail is a routine designed to allow the analyst, user or auditor to verify a process or an area in the new system.

1.8 KEYWORDS

- System testing
- Unit testing
- Stress testing
- Audit trail
- Quality Assurance
- Validation
- Verification
- Audit Trail

1.9 MODEL ANSWERS

Check Your Progress 1

1. Testing is the practice of making objective judgments regarding the extent to which the system (device) meets, exceeds or fails to meet stated objectives
2. The first step in system testing is to prepare a plan that will test all aspects of the system in a way that promotes its credibility among potential users.
3. In unit testing, the analyst tests the programs making up a system. Unit testing gives stress on the modules independently of one another, to find errors. This helps the tester in detecting errors in coding and logic that are contained within that module alone. The errors resulting from the interaction between modules are initially avoided.
4. Various special system tests are:
    - Peak load test
    - Storage testing
    - Performance time testing
    - Recovery testing
    - Procedure testing
    - Human factors testing.

Check Your Progress 2

1. Various factors are as follows:
    - Correctness
    - Reliability
    - Efficiency
    - Usability
    - Maintainability
    - Testability
    - Portability
    - Accuracy
    - Error tolerance
    - Expandability
    - Access control and audit
    - Communicativeness
2. Different levels of quality assurance are:
    - Testing
    - Verification with validation
    - Certification
3. Verification is intended to find errors. Executing a program in a simulated environment performs it. Validation refers to the process of using software in a live environment on order to find errors.

1.10 TERMINAL QUESTIONS

1. What is System Testing?
2. Why do we test systems? How important is testing? Discuss.
3. Outline the various activities that represent a test plan.
4. What design specifications are considered in preparing a test plan?
5. What is the need of system testing? Explain any five testing techniques and their basic objectives.
6. Explain Level of Testing.
7. List and explain (each in one line) special system tests.
8. Define quality assurance.
9. How quality can be tested? List three approaches of quality assurance.
10. What level of quality assurance must a system meet? Explain.
11. What is Validation?
12. What do you mean by audit trail? Explain.

UNIT - II

IMPLEMENTATION AND PROJECT SCHEDULING

UNIT STRUCTURE

2.1 INTRODUCTION

Systems implementation is the construction of the new system and its delivery into 'production' or day-to-day operation.

The key to understanding the implementation phase is to realize that there is a lot more to be done than programming. During implementation you bring your process, data, and network models to life with technology. This requires programming, but it also requires database creation and population, and network installation and testing. You also need to make sure the people are taken care of with effective training and documentation. Finally, if you expect your development skills to improve over time, you need to conduct a review of the lessons learned.

Systems implementation involves installation and changeover from the previous system to the new one, including training users and making adjustments to the system. Many problems can arise at this stage. You have to be extremely careful in implementing new systems. First, users are probably nervous about the change already. If something goes wrong they may never trust the new system. Second, if major errors occur, you could lose important business data.

2.2 OBJECTIVES

After studying this Unit, you should be able to:

- The makeup of Post Implementation Review.
- What are administrative review plan?
- What are personnel requirement plan?
- How project scheduling done?
- How Gantt and PERT charts are used in project planning?

2.3 POST IMPLEMENTATION REVIEW

Every system requires periodic evaluation after implementation. A post implementation review measures the system's performance against predefined requirements. Unlike system testing, which determines where the system fails so that the necessary adjustments can be made, a post-implementation review determines how well the system continues to meet performances specifications. It is done after design and conversion are complete. It also provides information to determine whether major redesign is necessary.

2.3.1 Request for Review:

The initiating study begins with the review team, which gathers and reviews requests for evaluation. It also files discrepancy notices after the system has been accepted. Unexpected change in the system that affects the user or system performance is a primary factor that prompts system review. Once a request is filled, the user is asked how well the system is functioning. Suggestions regarding changes and improvements are also sought. This phase sets the stage for a formal post-implementation review.

2.3.2 A Review Plan:

This review team prepares a formal review plan around the objectives of the review, the type of evaluation to be carried out, and the time schedule required.

1. **Administrative plan:**

   The review group probes the effect of the operational system on the administrative procedures of the user. The following activities are reviewed:

   - **User objectives** – This is an extremely critical area since it is possible that over time either the system fails to meet the user's initial objectives or the user objectives change as a reflection of changes in the organizational objectives. We need to think in terms of problems and of further opportunities. The results of the evaluation are documented for future reference.
   - **Operating costs and benefits** – Under the administrative plan, the cost structures of the system is closely reviewed. This includes a review of all costs and savings, a review and update of the non-cost benefits of all the system and a current budget designed to manipulate the costs and savings of the system.

2. **Personnel Requirement Plan:**

   This plan evaluates all activities involving system personnel and staff as they directly deal with the system. The emphasis is on productivity and job satisfaction. After test plan is developed, the review group evaluates the following:

   - **Personnel performance** objectives compared with current performance levels Turnover, tardiness and absenteeism are also evaluated. The results are documented and made available to the maintenance group for follow-up.
   - **Training performance:** Through testing, interviews, and all other data gathering techniques, the review group attempts to answer questions about the adequacy of the training materials.

3. **Hardware Plan:**

   The hardware of the new system is also reviewed, including terminals, CRT screens, software programs, and the communication network. The primary target is a comparison of current

performance specifications. The outcome of the evaluation indicates any differences between expectations and realized results. It also points to any necessary modifications to be made.

4. **Documentation Review Plan:**

   The reason for developing a documentation review plan is to evaluate the accuracy and completeness of the documentation compiled to date and its conformity with pre-established documentation standards. Irregularities prompt action where changes in documentation would improve the format and content.

2.4 PROJECT SCHEDULING

The process of planning, designing and implementing computer system is called a project. It is directed by a project manager who uses available resources to produce systems for the organizations. It takes an effective manager to organize the available resources, schedule the events, establish standards and complete the project on time, within the budget and with successful results.

During Project Planning (Detail Level), an agreed-upon baseline was established for the Project Schedule. This schedule baseline will be used as a starting point against which performance on the project will be measured. It is one of many tools the Project Manager can use during Project Execution and Control to determine if the project is on track.

Project Team members use the communications mechanisms documented in the Communications Plan to provide feedback to the Project Manager on their progress. Generally team members document the time spent on tasks and provides estimates of the time required to complete them. The Manager uses this information to update the Project Schedule. In some areas there may be formal time tracking systems that are used to track project activity.

After updating the Project Schedule, the Project Manager must take the time to review the status of the project. Some questions that the Project Manager should be able to answer by examining the Project Schedule include:

- Is the project on track?
- Are there any issues that are becoming evident that need to be addressed now?
- Which tasks are taking more time than estimated? Less time?
- If a task is late, what is the effect on subsequent tasks?
- What is the next deliverable to be produced and when is it scheduled to be complete?
- What is the amount of effort expended so far and how much is remaining?
- Are any Project Team members over-allocated or under allocated?
- How much of the time allocated has been expended to date and what is the time required to complete the project?
- Most project scheduling tools provide the ability to produce reports to display a variety of useful information. It is recommended that the Project Manager experiment with all available reports to find those that are most useful for reporting information to the Project Team, customer, and Project Sponsor and / or Project Director.
- When updating the Project Schedule, it is very important that the Project Manager maintain the integrity of the current schedule. Each version of the schedule should be archived. By creating a new copy of the schedule whenever it is updated, the Project Manager will never lose the running history of the project and will also have a copy of every schedule for audit purposes.

Project scheduling is concerned with the techniques that can be employed to manage the activities that need to be undertaken during the development of a project. Scheduling is carried out in advance of the project commencing and involves:

- Identifying the tasks that need to be carried out;
- Estimating how long they will take;
- Allocating resources (mainly personnel);
- Scheduling when the tasks will occur.

Once the project is underway control needs to be exerted to ensure that the plan continues to represent the best prediction of what will occur in the future:

- Based on what occurs during the development;
- Often necessitates revision of the plan.

Effective project planning will help to ensure that the systems are delivered:

- within cost;
- within the time constraint;
- to a specific standard of quality.

Two project scheduling techniques will be presented, the Gantt chart and the PERT.

2.4.1 Gantt Charts

A Gantt chart provides a graphical illustration of a schedule that helps to plan, coordinate, and track specific tasks in a project and was developed in 1917 by Henry L. Gantt. Gantt charts are a project planning tool that can be used to represent the timing of tasks required to complete a project. Because Gantt charts are simple to understand and easy to construct, they are used by most project managers for all but the most complex projects.

In a Gantt chart, each task takes up on row. Dates run along the top in increments of days, weeks or months, depending on the total length of the project. The expected time for each task is represented by a horizontal bar whose left end marks the expected beginning of the task and whose right end marks the expected completion date. Tasks may run sequentially, in parallel or overlapping.

As the project progresses, the chart is updated by filling in the bars to a length proportional to the fraction of work that has been accomplished on the task. This way, one can get a quick reading of project progress by drawing a vertical line through the chart at the current date. Completed tasks lie to the left of the line and are completely filled in. Current tasks cross the line and are behind schedule if their filled-in section is to the left of the line and ahead of schedule if the filled-in section stops to the right of the line. Future tasks lie completely to the right of the line.

In constructing a Gantt chart, keep the tasks to a manageable number (no more than 15 or 20) so that the chart fits on a single page. More complex projects may require subordinate charts which detail the timing of all the subtasks which make up one of the main tasks. For team projects, it often helps to have an additional column containing numbers or initials which identify who on the team is responsible for the task.

Often the project has important events which you would like to appear on the project timeline, but which are not tasks. For example, you may wish to highlight when a prototype is complete or the date of a

design review. You enter these on a Gantt chart as "milestone" events and mark them with a special symbol, often an upside-down triangle. Example Gantt chart for boat making is given below.

A Gantt chart allows for

- Graphical representation of a schedule
- Clear and easy communication
- Resource allocation
- Tracking of the schedule
- Providing a history of the project

For Example we want to draw a Gantt chart to prepare a house according to following schedule

Sr. No	Work	Start Date	Duration
1.	Foundation	1-Jan	10
2.	Walls	12-Jan	7
3.	Roof	20-Jan	10
4.	Window and Doors	1-Jul	5
5.	Plumbing	7-Jul	3
6.	Electric	7-Jul	3
7.	Painting	11-Jul	2
8.	Flooring	13-Jul	2

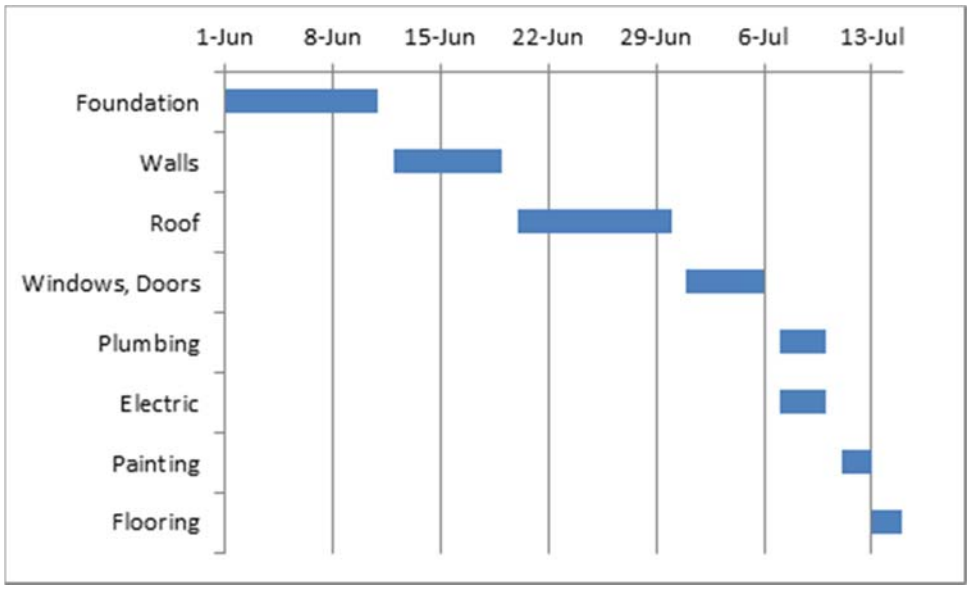


Fig 2.1: Gantt Chart example

The basic Gantt chart is an easy way to document schedules. It is a horizontal-bar schedule showing activity start, duration, and completion. It shows the connection between events and the calendar, and provides a graphical analog of the activity duration.
The Gantt schedule can illustrate the relationship between work activities having duration, events without duration that indicate a significant completion, and milestones that represent major achievements or decision points. Various annotations can be used to communicate the progress of the project effort compared to the baseline plan, as well to depict in a graphical way, areas where there are modified expectations from the baseline plan.

Once a Gantt schedule has been established for a project, progress should be periodically plotted against the baseline schedule. If different functional areas are involved in a project, each area may need its own detailed schedules to support the project master schedule. In such cases it is important that working schedules be linked to a common master schedule in a way that they can be easily updated. Each activity or event on the schedule should have a responsible individual assigned, so there is clear ownership and so schedule status can be updated.

2.4.2 PERT Chart

PERT is basically a method to analyze the tasks involved in completing a given project, especially the time needed to complete each task, and identifying the minimum time needed to complete the total project.

PERT charts are visualization tools commonly used by project managers to control and administer the tasks required to complete a project.

The Program Evaluation and Review Technique or Project Evaluation and Review Technique commonly abbreviated PERT is a model for project management to analyze and represent the tasks involved in completing a given project.

This model was invented by Booz Allen Hamilton, Inc. under contract to the United States Department of Defense's US Navy Special Projects Office in 1958 as part of the Polaris mobile submarine-launched ballistic missile project.

PERT was developed in the 1950's, primarily to simplify the planning and scheduling of large and complex projects. It was able to incorporate uncertainty by making it possible to schedule a project not knowing precisely the details and durations of all the activities. It is more of an event-oriented technique rather than start- and completion-oriented

The most famous part of PERT is the "PERT Networks", charts of timelines that interconnect. PERT is intended for very large-scale, one-time, complex,

The PERT chart provides a graphical display of Critical Path on a project. Most scheduling tools highlight the activities on the Critical Path. It is useful to include the following information in the activities on the PERT chart:

Duration, Float Start date, End date, Resources Float – The amount of surplus time and leeway allowed in scheduling tasks so that the network critical path is maintained on schedule.

For Example we want to draw a PERT chart for following activities

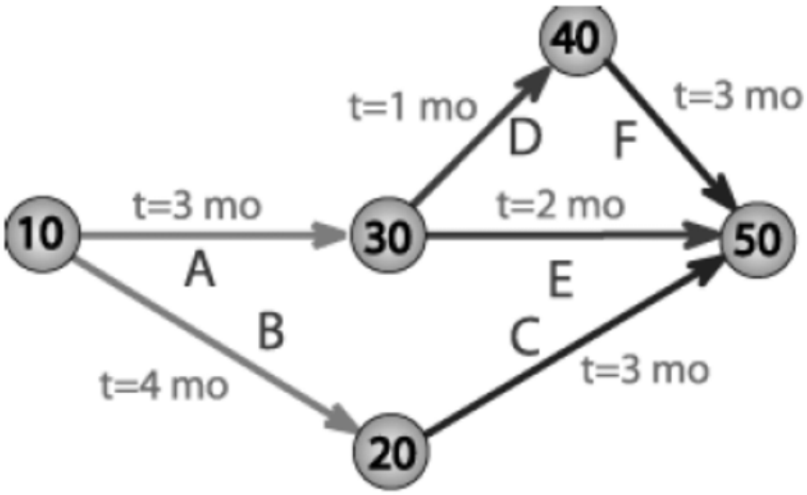


Fig 2.2: PERT Chart

Let's explain various conventions used in above figure 2.2.

- A PERT chart is a tool that facilitates decision making. The first draft of a PERT chart will number its events sequentially in 10s (10, 20, 30, etc.) to allow the later insertion of additional events.
- Two consecutive events in a PERT chart are linked by activities, which are conventionally represented as arrows in the diagram above.
- The events are presented in a logical sequence and no activity can commence until its immediately preceding event is completed.
- The planner decides which milestones should be PERT events and also decides their "proper" sequence.
- A PERT chart may have multiple pages with many sub-tasks.
- A PERT activity: is the actual performance of a task. It consumes time, it requires resources (such as labor, materials, space, machinery), and it can be understood as representing the time, effort, and resources required to move from one event to another. A PERT activity cannot be completed until the event preceding it has occurred.
- PERT event: is a point that marks the start or completion of one or more tasks. It consumes no time, and uses no resources. It marks the completion of one or more tasks, and is not "reached" until all of the activities leading to that event have been completed.
- Critical Path: the longest possible continuous pathway taken from the initial event to the terminal event. It determines the total calendar time required for the project; and, therefore, any time delays along the critical path will delay the reaching of the terminal event by at least the same amount.
- A predecessor event: an event (or events) that immediately precedes some other event without any other events intervening. It may be the consequence of more than one activity.
- A successor event: an event (or events) that immediately follows some other event without any other events intervening. It may be the consequence of more than one activity.
- Slack: the slack of an event is a measure of the excess time and resources available in achieving this event. Positive slack (+) would indicate ahead of schedule; negative slack would indicate behind schedule; and zero slack would indicate on schedule.

2.4.3 Difference between PERT and CPM Chart

Though **PERT and CPM** both are used for project management, there are differences between CPM and PERT. The relative table for PERT vs CPM is shown below.

CPM	PERT
• CPM uses activity oriented network.	• PERT uses event oriented Network.
• Durations of activity may be estimated with a fair degree of accuracy.	• Estimate of time for activities are not so accurate and definite.
• It is used extensively in construction projects.	• It is used mostly in research and development projects, particularly projects of non-repetitive nature.
• Deterministic concept is used.	• Probabilistic model concept is used.
• CPM can control both time and cost when planning.	• PERT is basically a tool for planning.
• In CPM, cost optimization is given prime importance. The time for the completion of the project depends upon cost optimization. The cost is not directly proportioned to time. Thus, cost is the controlling factor.	• In PERT, it is assumed that cost varies directly with time. Attention is therefore given to minimize the time so that minimum cost results. Thus in PERT, time is the controlling factor.

Check Your Progress 1

1. What do you mean by Gantt chart?

_____

_____

_____

2. Explain the importance of post implementation review.

_____

_____

_____

3. What is the use of PERT chart?

_____

_____

_____

2.5 SUMMARY

- The post-implementation review has the objective of evaluating the system in terms of how well performance meets stated objectives. The study begins with the review team, which gathers requests for evaluation. The team prepares a review plan around the type of evaluation to be done and the time frame for its compilation. The plan considers administrative, personnel and system performance and the changes that are likely to take place through maintenance.
- During Project Planning, an agreed-upon baseline was established for the Project Schedule. This schedule baseline will be used as a starting point against which performance on the project will be measured. It is one of many tools the Project Manager can use during Project Execution and Control to determine if the project is on track.
- Scheduling is an essential activity for the development of the software project.
- Scheduling can be resource scheduling, time scheduling and development scheduling.
- PERT charts and Gantt charts are primary scheduling techniques.
- Gantt chart is derived automatically from the PERT chart. Each kind of chart has its own place. Gantt charts helps in planning the utilization of resources, while the PERT chart is better for monitoring the timely progress of activities.
- Two planning tools are used in project scheduling.
    - Gantt Chart
    - Program Evaluation and Review Technique (PERT)

2.6 KEYWORDS

- Gantt Chart
- Milestone
- PERT
- Critical Path
- Implementation
- Post-Implementation Review

2.7 MODEL ANSWERS

Check Your Progress 1

1. A Gantt chart provides a graphical illustration of a schedule that helps to plan, coordinate, and track specific tasks in a project and was developed in 1917 by Henry L. Gantt. Gantt charts are a project planning tool that can be used to represent the timing of tasks required to complete a project.
2. A post implementation review measures the system's performance against predefined requirements. Unlike system testing, which determines where the system fails so that the necessary adjustments can be made, a post-implementation review determines how well the system continues to meet performances specifications.
3. PERT is basically a method to analyze the tasks involved in completing a given project, especially the time needed to complete each task, and identifying the minimum time needed to complete the total project

2.8 TERMINAL QUESTIONS

1. Briefly explain the procedure and makeup of the post-implementation review. Can one perform maintenance on a system without a post implementation review? Why?
2. What is post implementation review? How is it carried out?
3. Which tools can be used to carry out post implementation review? Also list some review methods.
4. Differentiate between Gantt chart and PERT chart.
5. What is Gantt chart?
6. What information does a PERT chart show?
7. Distinguish between System maintenance and enhancement.
8. Review the primary activities of a maintenance procedure
9. What is project scheduling? Explain different techniques for project scheduling.
10. What is PERT? Explain.
11. What are the various automated scheduling tools available?

UNIT - III

HARDWARE AND SOFTWARE SELECTION

UNIT STRUCTURE

3.1 INTRODUCTION

The design phase is also the time to begin selecting and acquiring the hardware and software that will be needed for the future system. In many cases, the new system will simply run on the existing equipment in the organization. Other times, however, new hardware and software (usually, for servers) must be purchased. The hardware and software specification is a document that describes what hardware and software are needed to support the application. There are several steps involved in creating the document.

First, you will need to define the software that will run on each component. This usually starts with the operating system (e.g., Windows, Linux) and includes any special purpose software on the client and servers (e.g., Oracle database). Here, you should consider any additional costs such as technical training, maintenance, extended warranties, and licensing agreements (e.g., a site license for a software package). Again, the needs that you list are influenced by decisions that are made in the other design phase activities.

3.2 OBJECTIVES

After studying this Unit, you should be able to:

- The phases in hardware/software selection.
- The financial considerations in selection.
- What are the major phases in selection?
- What is evaluation process?

3.3 SELECTION FOR HARDWARE AND SOFTWARE

The systems come with hardware, software and support. Today, selecting a system is a serious and time-consuming business.

There are several factors to consider prior to system selection:

1. Define the system capabilities that make sense for business. Computers have proven valuable to business in the following areas :
    - Cost reduction includes reduction of the inventory, savings on space and improved ability to predict business trends
    - Cost avoidance includes early detection of problems and ability to expand operations without adding clerical help.
    - Improved service emphasizes quick availability of information to customers, improved accuracy and fast turnaround
    - Improved profit reflects the bottom line of the business and its ability to keep receivables within reason.
2. Specify the magnitude of the problem, that is, clarify whether selections consist of a few peripherals or major decision concerning the mainframes
3. Assess the competence of the in-house staff. This involves determining the expertise needed in areas such as telecommunications and data base design. Acquiring a computer often results in securing temporary help for conversion. Planning for this help is extremely important.
4. Consider hardware and software as a package. This approach ensures compatibility. In fact, software should be considered first, because often the user secures the hardware and then wonders what software is available for it.
5. Develop a schedule for the selection process. Maintaining a schedule helps keeps the project under control
6. Provide user indoctrination. This is crucial, especially for first-time users. Selling the system to the user staff, providing adequate training and preparing an environment a conductive to implementation are pre- requisites for system acquisition.

3.3.1. Major phases in selection

Selecting hardware and software for implementing information system in an organization is a serious and time-consuming process that passes through several phases. The main steps of the selection process are listed below:

1. Requirements analysis
2. Preparation of tender specifications
3. Inviting tenders
4. Technical scrutiny and short listing
5. Detailed evaluation of short listed vendors
6. Negotiation and procurement decision
7. Delivery and installation
8. Post-installation review

- **Requirements analysis**

  The first step in selection understands the user's requirements within the framework of the organization's objectives and the environment in which the system is being installed. Consideration is given to the user's resources as well as to finances.

  In selecting software, the user must decide whether to develop it in house, hire a software company or contract programmer to create it, or simply acquire it from a software house. The choice is logically made after the user has clearly defined the requirements expected of the software. Therefore, requirements analysis sets the tone for software selection.

- **Preparation of tender specifications**

  After studying the feasibility and deciding upon the configuration, tender documents are prepared for the benefit of vendors to clarify the details of various specifications, as listed below.

  - Purchase procedure and schedule: it includes

    a) Date of tender submission
    b) Evaluation criteria
    c) Scope for negotiations, if any and
    d) Expected usage environment and load pattern

  - Equipment specification

    Detailed technical specifications of each item required for both mandatory and optional items.

  - Quotation format:

    a) Format for stating technical details and quoting prices
    b) Whether deviations from specifications should be specifically listed
    c) Prices and levies (duties, taxes etc.) could be quoted as lump sum or required separately.
    d) Required validity of the quotation.
    e) Earnest money deposit required, if any.

  - Proposed terms of contract

    a) Expected delivery schedule.
    b) Uptime warranties required
    c) Penalty clause, if any
    d) Payment terms (Whether advance payment acceptable)
    e) Arbitrary clauses
    f) Training needs.
    g) Post warranty maintenance terms expected.

  - Any additional information required.

- **Inviting tenders**

  After the preparation of tender specifications, tenders are invited. Invitation of tenders may depend upon the magnitude of purchase (estimate equipment cost). It may be through

  - Open tender (through newspaper advertisement)
  - Limited tender (queries sent to a few selected vendors)
  - Propriety purchase (applies mostly to upgrade requirements)
  - Direct purchase from market. (applies mostly to consumables)

- **Technical scrutiny and short listing**

  This step involves the following activities.

  - All tendered bids are opened on a pre-defined date and time.
  - Deviations from the specifications, if any, in each bid are noted.

- A comparative summary is prepared against the list of tendered technical features.

Additional factors to considered are:

- Financial health of the vendor
  (from balance sheets)
- Nature and extent of support
  (from information provided on number of support staff per installed site an cross-check with selected customers)
- Engineering quality of products
  (factory inspection of product facilities, QA procedures and R&D)

- Detailed evaluation of short listed vendors

  This step primarily involves getting any finer technical clarifications. Visits to customer sites and factory inspections may be planned. If any specific performance requirement is stipulated, the offered product is to be examined at this stage through suitable benchmark tests. For benchmark tests, standard benchmarks may be used as adequate performance indicators.

- **Negotiation and procurement decision**

  Because of the extensive competition, computer system vendors may offer significant concessions. Negotiations are held to maximize these concessions. However, price negotiations are often not permitted by some organizations.

  When price negotiations are permitted, the committee members should have a good knowledge of the prevailing market prices, current trends, and also the duty/tax structure.

  i) Computer magazines
  ii) Vendor directories.
  iii) Contact with other users
  iv) Past personal experience.

- **Delivery and installation**

  In this step, the vendor delivers the hardware/software to the buyer's organization, where it is matched with the specifications mentioned in the purchase order. If conforms to these specifications, the vendor installs the system in the premises of the organization.

- **Post- installation Review**

  Sometime after the package is installed, a system evaluation is made to determine how closely the new system conforms to plan. System specifications and user requirements are audited to pinpoint and correct any differences

3.3.2 Software selection

Software selection is a critical aspect for system development. There are 2 ways of acquiring the software.

- Custom -made
- Packages

Criteria for Software selection

- **Reliability –** It is the probability that the software will executed in a specific period of time without any failures. It is important to the professional user. It brings up the concept of modularity, or the ease which a package can be modified.
- **Functionality –** It is the definition of the facilities, performance and other factors that the user requires in the finished product.
- **Capacity –** Capacity refers to the capability of the software package to handle the users requirements for size of files, number of data elements, and reports. All limitations should be checked.
- **Flexibility –** It is a measure of effort required to modify an operational program. One feature of flexibility is adaptability.
- **Usability –** this criterion refers to the effort required to operate, prepare the input, and interpret the output of a program. Additional points considered here are portability and understandability. Portability refers to the ability of the software to be used. Understandability is the purpose of the product.
- **Security** – It is a measure of the likelihood that a system's user can accidentally or intentionally access or destroy unauthorized data.
- **Performance** – It is a measure of the capacity of the software package to do what it is expected to do. This criterion focuses on throughput or how effectively a package performs under peak load.
- **Serviceability** –This criteria focuses on documentation and vendor support.
- **Ownership** – Who owns the software, and to consider whether he has the right to access the software or he can sell or modify the software.
- **Minimal costs** – Cost is a major consideration in deciding between in-house and vendor software.

3.3.3 Evaluation process

There are three process for evaluating hardware and software.

1. **Benchmark programs:** It is a sample program for evaluating different computers and their software. It is necessary because computers often use the same instructions, words of memory or machine cycle to solve a problem. Benchmarking includes the following
    - Determination of the minimum hardware.
    - An acceptance test
    - Testing in an ideal environment to determine the timings and in the normal environment to determine its influence on other programs.
2. **Experience of other users:** Benchmarking only validates vendors' claims. Experience of other users with the same system software is essential.

3. **Product reference manuals:** These evaluate a system's capability. These reports elaborate on computer products, services and prices.

3.3.4 Evaluation of proposals:

After all proposals are evaluated, the final vendor is selected using any of the three methods

1. **Adhoc** refers to the user's inclination to favor one vendor over others.
2. **Scoring**. In this method the characteristics of each system are listed and score is given in relation to the maximum point rating. Then each proposal is rated according to its characteristics.
3. **Cost value approach.** In this method a dollar credit method is applied to the proposal that meets the user's desirable characteristics. This credit is subtracted from the vendor's quoted price. The proposal with the lowest price is selected.

3.3.5 Performance evaluation

Hardware selection requires an analysis on the following criteria

1. System availability
2. Compatibility
3. Cost
4. Performance
5. Uptime
6. Support
7. Usability

For the software evaluation, the following are considered

1. The programming language and its suitability to the applications.
2. Ease of installation and training.
3. Extent of enhancements to be made prior to installation.

In addition to hardware and software evaluation, the quality of the vendor's should be examined. Considerations to ensure vendor quality are as follows

1. Backup
2. Conversion
3. Maintenance
4. System development

Check Your Progress 3

1. What are the major phases in selection?

   _____

   _____

   _____

2. What are the criteria for software selection?

_____

_____

_____

3. Explain the process of evaluating hardware and software.

_____

_____

_____

4. List the methods for vendor selection.

_____

_____

_____

3.4 SUMMARY

- A major element in building systems is selecting compatible Hardware and software. The kind of hardware and peripherals required is to be determined. The suitable software has to be selected. Comparisons are often made among different computer systems on the basis of actual performance data.
- There are several thing to do before selection :
    - Define system capabilities that makes sense for the business
    - Specify the magnitude of the problem
    - Access the competence of in-house staff
    - Consider the hardware and software as a package
    - Develop the time farm for selection
    - Provide user indoctrination
- The selection process consists of several steps
    - Prepare the requirement analysis
    - Specify system specifications
    - Prepare a request for proposal
    - Rank vendor proposal
    - Decide best proposals or vendor
- The criteria for software selection are:
    - Reliability gives consist result
    - Functionality functions to standard
    - Capacity satisfy volume requirement
    - Flexibility adapts to changing needs
    - Usability is user friendly
    - Security
    - Performance
    - Serviceability
    - Ownership
    - Minimal cost

3.5 KEYWORDS

- Benchmark
- Request for Proposal (RFP)
- Tender
- Usability
- Uptime
- Compatibility
- Reliability
- Vendor

3.6 MODEL ANSWERS

Check Your Progress 3

1. Major phases in selection are
    - Requirements analysis
    - Preparation of tender specifications
    - Inviting tenders
    - Technical scrutiny and short listing
    - Detailed evaluation of short listed vendors
    - Negotiation and procurement decision
    - Delivery and installation
    - Post-installation review
2. The criteria for software selections are
    - Reliability gives consist result
    - Functionality functions to standard
    - Capacity satisfy volume requirement
    - Flexibility adapts to changing needs
    - Usability is user friendly
    - Security
    - Performance
    - Serviceability
    - Ownership
    - Minimal cost
3. There are three process for evaluating hardware and software.

1. **Benchmark programs:** It is a sample program for evaluating different computers and their software. It is necessary because computers often use the same instructions, words of memory or machine cycle to solve a problem. Benchmarking includes the following

    - Determination of the minimum hardware.
    - An acceptance test
    - Testing in an ideal environment to determine the timings and in the normal environment to determine its influence on other programs.

2. **Experience of other users:** Benchmarking only validates vendors' claims. Experience of other users with the same system software is essential.

3. **Product reference manuals:** These evaluate a system's capability. These reports elaborate on computer products, services and prices.

4. The sources available to check on vendors selection include the following
    - Users
    - Software houses
    - Trade associations
    - Universities
    - Publications/Journals
    - Vendor software lists
    - Vendor referral directories
    - Published directories
    - Consultants
    - Industry contacts

3.7 TERMINAL QUESTIONS

1. What is benchmarking?
2. What factors play a role in hardware selection?
3. Discuss the various methods of acquiring hardware.
4. What parameters are applied to select a vendor?
5. Both hardware and software are equally important in selection. Do you agree?
6. What are the major phases in selection?
7. What do you mean by performance evaluation?
8. What are the criteria for software selection?

UNIT - IV

SECURITY AND DISASTER RECOVERY

UNIT STRUCTURE

4.1 INTRODUCTION

Every system must provide in-built features for security and integrity of data. Without safeguards against unauthorized access fraud, fire and natural disaster, a system could be so vulnerable as to threaten the survival of the organization.

The strength behind system integrity and success is ethics and professional standards of behavior.

4.2 OBJECTIVES

After studying this Unit, you should be able to:

- The various threats to system security.
- How to do risk analysis and specify measures?
- The importance of disaster and recovery planning.
- The importance of ethics in system development.

4.3 SYSTEM SECURITY

System security refers to the technical innovations and procedures applied to the hardware and operating systems to protect against deliberations or accidental damage from a defined threat.

System integrity refers to the proper functioning of hardware and programs, appropriate physical security and safety against external threats such as eaves dropping and wiretapping.

Privacy defines the rights of the users or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair, or excessive dissemination of information about it.

Confidentiality is a special status given to sensitive information in a database to minimize the possible invasion of privacy.

An organization that depends heavily on the use of databases requires special controls to maintain viable information. These controls are classified into three general categories:

1. Physical security or protection from fire, flood, and other physical damage.
2. Database integrity through data validation techniques.
3. Control measures through passwords, encryption, and monitoring users on a regular basis.

4.3.1. Threats to System Security

A procedure for protecting systems makes sure that the facility is physically secure, provides a recovery/restart capability, and has access to backup files. The potential threats within a firm are:

1. Errors and omissions
2. Disgruntled and dishonest employees.
3. Fire.
4. Natural disasters.
5. External attack.

When huge quantities of information are stored in one database, sensitive data can easily be copied and stolen. Information can also be entered directly into a computer without any written record or proper authorization and can be changed without a trace. A dishonest programmer can bypass control and surreptitiously authorize his/her own transactions.

Dishonest employees have an easier time identifying the vulnerabilities of a software system than outside hackers because they have access to the system for a much longer time and can capitalize on its weakness.

Fire and other man-made disasters that deny the system power conditioning, or needed supplies can have a crippling effect. Proper planning for safeguards against such disasters is critical, especially in organizations that depend on centralized database systems. Natural disasters are floods, hurricanes, snowstorms, lightning and other calamities.

System reliability is also important in system security design. For example, a facility plagued by hardware outages, bug-ridden software, or a deficient communication network can cause chaos for the end user.

4.3.2. The Personal Computer and System Integrity:

It is easy to make changes in accounting systems that require rigid controls. There is also a tendency to put everything on the microcomputer with hardly a backup. A third problem is the lack of audit trails in most off-the-shelf software packages. It is difficult to reconstruct transactions for audit purposes. Finally, as more personal computers are linked to company mainframes so remote users can access the data, the potential increases for altering the data deliberately or by mistake. Many of today's operating systems contain no password.

4.3.3. Risk analysis:

The purpose of risk analysis is to determine the probability of problems occurring, the cost of each possible disaster, the areas of vulnerability, and the preventive measures to adopt as part of a security plan.

First, the designer lists the objectives of the system and evaluates them against the existing computer facility to determine the security requirements. The facility in turn, is evaluated against the potential hazards to determine the specific exposures. Security measures are then compared with specific exposures to pinpoint unacceptable exposures. The outcome is a draft specifying the preventive and recovery measures to be adopted for effective system security.

A special risk analysis matrix that specifies the risks, costs and effects, and probability of exposure helps the designer to determine the actions to be taken and how quickly they must be taken. The two key elements in risk analysis are the value or impact of a potential loss and the probability of loss. The goal is to identify the treat that results in the greatest monetary loss and provide protection to the appropriate degree.

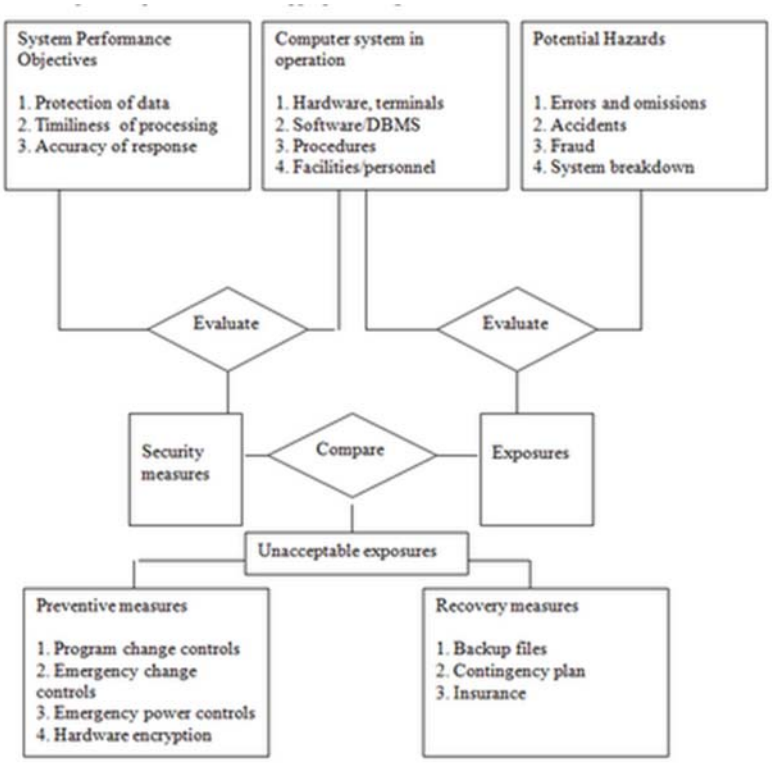


Fig 4.1: Risk Analysis

4.4. CONTROL MEASURES

After system Security risks have been evaluated, the next step is to select the measures that are internal and external to the facility. These measures are classified into four categories-

1. Identification
2. Access controls
3. Audit controls
4. System integrity

4.4.1. Identification:

There are three schemes for identifying persons to the computer. They are:

1. **Password:** A password is the most commonly used means for authenticating the identity to the people. Passwords should be hard to guess and easy to remember. They should not be recoverable. Experience has shown that many illicit entries to the system are due to written passwords. Another scheme under the "Something you know" category is the picture badge, which identifies the people who bring the work to the center. Although it positively identifies the carrier of the information, the badge does not verify that the person is authorized to submit a job or receive reports from the system.
2. **Something you are, such as fingerprints or voice prints.** Voice prints is a reliable method for verifying authorized users. The technique essentially analyzes a person's voice against prerecorded voice patterns of the same person. An exact match allows access to the system.
3. **Something you have, such as the credit card, key, or a special terminal.** Magnetic stripe credit card readers on terminals identify the operator to the system. The card along with a password gives added assurance of the authentication of the user.

4.4.2. Access Control:

Various steps have been taken to control the access to a computer facility. One way is to use an encoded card system with a log-keeping capability. The card serves as a key to unlock doors, including tape storage and other classified areas. The card is essentially a magnetic key and a "Keyport" is a lock. Inserting the card into the Lockport unlocks the door. A card that includes a photograph of the bearer may double as an employee ID badge.

Encryption:

An effective and practical way to safeguard data transmitted over the telephone lines is by encryption. Data are scrambled during transmission from one computer or terminal to the other. A plaintext message is transmitted over an unprotected communications channel. To prevent unauthorized acquisition of the message, it is enciphered with a reversible transformation to produce a cryptogram or ciphertext. When it arrives at an authorized receiver, it is decrypted back to the plaintext data form.

Most of today's encryption is based on the National Bureau of Standards encryption algorithm, known as the Data Encryption Standard (DES). It is a general technique developed in 1977 and used in many commercial network security systems. A system that assures that encryption and

decryption are done without human intervention is virtually secure from unauthorized access. Encryption devices for personal computers are available at the chip level and in the software.

4.4.3. Audit Controls:

Audit controls protect a system from external security breaches and internal fraud or embezzlement. The resources invested in audit controls, however should balance with the sensitivity of the data being manipulated. One problem with audit controls is that it is difficult to prove their worth until the system has been violated.

Programmers can pirate, modify and even sell software for potential gain. To audit the maintenance process properly, there must be an audit trail from the change requests to the production programs. Various audit software available to do the job properly. Generalized audit software helps the auditor examine files and databases for consistency, correctness and completeness. There are also programs to trace the flow of data through a program and the activity that they generate.

Neither the auditor nor the user can verify the system activities adequately, so the system must check itself. The internal controls required mean that programmers and analysts build controls into every system.

4.4.4. System Integrity:

The most costly software loss is the program error. It is possible to eliminate such error through proper testing routines. Parallel runs should be implemented whenever possible. Physical security provides safeguards against the destruction of the hardware, databases, and the documentation; fire, flood, theft, sabotage, and eavesdropping; and the loss of power through proper backup.

The proper use of the file library is another important feature. This involves adequate file backup and reliable personnel to handle the file documentation when needed. File backup means keeping duplicate copies of the master and other key files and storing them in suitable environmental conditions. For tape files, a common procedure is to save the old master file after each update. The most recently created file is called the son, the previous the father, and the one previous to the latter the grandfather. Since it is a costly procedure, the decision to proceed along these lines has to be balanced against potential loss if the files are destroyed.

Recovery/Restart Requirements:

Restoring a damaged database is done by roll forward or rollback procedure. Roll forward method involves updating a valid copy of the database with the necessary changes to produce a current version of the database.

Rollback method starts with the current invalid state and removes the records of the activity to produce the prior valid state of the database.

Backup is essential for recovery/restart procedure. If the database is physically damaged then it cannot be rolled back. Only roll forward can be done. For a sequential file a grandfather-father-son approach is followed. In a database environment, master files are not copied as they are updated.. Instead transactions are posted directly to the file which replaces the original data. So to recover documents in such type of storage, backup is required.

System failures and recovery

There are three types of failures

1. **Catastrophic failure** is one where part of the database is unreadable. To restore use roll forward method of recovery.

2. **Logical error** occurs when the activity of the database is interrupted with no chance to complete the current transactions. So when the system runs again, it is not sure if the changes have been updated or not. Data though available may be inaccurate. To restore the original contents, rollback method is used.

3. **Structural damage.** An example is a pointer incorrectly stored in a record that points to a unrelated or nonexistent data. If the problem cannot be corrected by software utility, then the database must be recovered to the most recent up-to-date point before the damage occurred.

4.5. DISASTER/RECOVERY PLANNING

Disaster/recovery planning is a means of addressing the concern for system availability by identifying potential exposure, prioritizing applications and designing safeguards and minimizes loss if a disaster occurs. There are several alternatives. They range from having an entire facility in one location with a complete redundancy of hardware to leasing a site with no computer but adequate electricity and air conditioning to support a computer facility on temporary basis. After an alternative has been determined a decision must be made about the applications to be processed. The hardware to process the applications and what should be relocated after the disaster. In disaster/recovery the management's role is to accept the plan select an alternative and recognize the benefits. The user's responsibilities are as follows:

- Identify critical applications why they are critical and how computer unavailability would affect the department.
- Approving data protection procedures.
- Funding the cost of backup.

4.5.1 The Plan

When a disaster/recovery procedure is planned, several questions have to be answered:

1. The time taken to rebuild the computer center or aspects of it
2. The type of accommodation should we look for in a backup installation. How quickly is it available?
3. The equipment is needed to keep the corporation functioning
4. How would reports be transmitted to the user? That is, is there going to be telecommunications network or simply a courier service?
5. That utilities(electric power, air conditioning, etc) are required when a disaster occurs
6. Would there be sufficient experienced staff available for proper recovery?

When these questions are answered and management gives its support for a disaster/recovery plan, the next step is to initiate a plan that involves four phases:

1. Appoint a disaster/recovery team and a team coordinator to develop the plan or procedure.
2. Prepare planning task.

3. Compile a disaster/recovery manual.
4. Dummy run to test the procedure.

4.5.2 The team

A disaster/recovery team should include a cross section of system designers, users, and computer operators. Under the leadership of a coordinator, the team's main functions are to organize the project, monitor progress on the plan, and oversee its completion. The team meets periodically to ensure that the plan is kept up to date, considers new vulnerabilities or exposures to loss, and implements new technology or procedures as needed. More specifically, the objectives of a disaster/recovery team include the following:

1. Secure backup sites for occupation and use.
2. Contract for hardware to meet minimum processing needs.
3. Supply working copies of all operating systems and application programs to meet minimum processing requirements.
4. Supply communication facilities to make reports promptly available to the user.
5. Supply consumables and administrative support.

4.5.3 Planning Task

Disaster/recovery planning tasks are prepared in a cycle similar to that of system development. Briefly, the cycle entails the following:

1. Definition phase sets the objectives of the disaster/recovery project.
2. Requirements phase evaluates applications against disaster. Recovery Objectives, determines what is to be included in the plan, and specifies priorities. The team takes inventor of the hardware, software, telecommunications, backup and clerical procedures, utilities, and personnel assignments. Design phase evaluates design alternatives, potential vendors, and prices and chooses the final design.
3. Testing and implementation phase runs backup systems, compares results, and correct errors. During implementation, procedures are written, sites are prepared, and maintenance plans are developed.

4.5.4 The Manual

Once the team has completed the assignment, a disaster/recovery manual is prepared and copies are made available to team members and management. All copies of the manual should be updated as needed.

4.6. ETHICS IN SYSTEM DEVELOPMENT

Ethics is the study of value concepts such as 'good,' 'bad,' 'right,' 'wrong,' 'ought', applied to actions in relation to group norms and rules. Therefore, it deals with many issues fundamental to practical decision-making. Computer software systems lie at the heart of modern decision making, including data/information storage and manipulation, data availability, and 'alternatives' formulation and selection. In fact, the very use of computer systems can often frame the types of questions that can be asked as well as their possible answers.

This is particularly evident when we incorporate software systems into our knowledge management methods, as they then play an essential role in institutional memory. The ubiquity of software systems in

all aspects of public and private institutions means that the environment that they create needs to be critically examined as they are developed and deployed.

Two major ethical questions must be addressed with regard to software systems. Firstly, can these systems represent the different codes of ethics of the groups affected by software-mediated decisions? Secondly, what ethical considerations should guide the design and development of the software itself?

Consider the following examples of how a system professional might act unethically:

1. The analyst knows that a user's requirement can be adequately met with a simple, inexpensive system yet installs a latest system that takes twice the time to implement.
2. A system analyst accepts a microcomputer from a vendor in return for recommending a system sold or made by the vendor.
3. A new client completes with a past client for whom the analyst has already installed a system. The analyst knows what system will give the new client the competitive edge and offers to install it in return for a gift or company stock.

Check Your Progress 1

1. What do you mean by system security?

   _____

   _____

   _____

2. List the potential threats within a firm.

   _____

   _____

   _____

3. Explain various types of control measures.

   _____

   _____

   _____

4.7 SUMMARY

- Security is critical in system development. The motivation behind security are to keep the organization running, protect data as an asset, and seek management support for more installation.
- There are three levels of controls in data security:
  - Physical Security
  - Data Base Integrity
  - Control Measures
- Control measures are classified as follows:
  - Identification

    - Access Control
    - Audit Control
    - System Integrity
- Disaster/recovery planning is a means of addressing the concern for system availability by identifying potential exposure, prioritizing applications and designing safeguards and minimizes loss if a disaster occurs.

4.8 KEYWORDS

- Data Integrity
- Ethics
- System Security
- Decryption
- Encryption
- Data Security
- Password
- Disaster

4.9 MODEL ANSWERS

Check Your Progress 1

1. System security refers to the technical innovations and procedures applied to the hardware and operating systems to protect against deliberations or accidental damage from a defined threat.
2. The potential threats within a firm are:
    - Errors and omissions
    - Disgruntled and dishonest employees.
    - Fire.
    - Natural disasters.
    - External attack.
3. Control measures are classified as follows:
    - Identification
    - Access Control
    - Audit Control
    - System Integrity

4.10 TERMINAL QUESTIONS

1. What do you mean by system security?
2. What are the major threats to system security? Which one is the most serious? Why?
3. Disaster/recovery planning tasks are prepared in a cycle similar to that of system development. Explain.
4. List the control measures in system security?
5. What is encryption? Discuss.
6. What do you mean by ethics in system development?
7. What is system integrity? Explain.
8. Explain the term risk analysis.

APPENDIX – A

BIBLIOGRAPHY

- Systems Analysis and Design, Elias M. Awad, Galgotia
- Analysis and Design of Information Systems, James A Senn, McGraw Hill International
- Analysis and Design of Information Systems, V Rajaraman, PHI 2002
- Modern Systems Analysis and Design, Jeffrey A Hoffer, Joey F George, Joseph S. Valacich, Pearson Education
- Systems Analysis and Design, Kendall and Kendall, PHI
- Systems Analysis and Design, Igor Hawryszkiewycz, PHI
- Davenport, Thomas (1993), Process Innovation: Reengineering work through information technology, Harvard Business School Press, Boston
- Software Engineering by Roggers S. Pressmen
- http://en.wikipedia.org/wiki/Systems_Development_Life_Cycle

APPENDIX – B

GLOSSARY

Action Entry: The lower right quadrant of a decision table indicates the responses of the question entered in the condition entry.

Action Stub: Lower left quadrant of a decision table outlines in narrative form the conditions that may exist.

Analysis: Breaking a program into successively manageable parts for individual study.

Audit Trail: A feature of data processing system that allow for the study of data as processed from step to step, an auditor can then trace all transactions that affect an account.

Corrective Maintenance: Changes made to a system to repair flaws in its design, coding, or implementation.

Context Diagram: An overview of an organizational system that shows the system boundary, external entities that interact with the system and the major information flows between the entities and the system

Conversion: The organizational process of changing over from the current information system to a new one.

Cost/benefit analysis: the analysis to compare costs and benefits to see whether investing in the development of a new system will be beneficial.

Data: Data are raw facts that collected for the entity in question

Database Management Systems: Software that is used to create, maintain, and provide controlled access to user databases

Data Flow Diagrams (DFD): Graphical depiction of data processes, data flows and data stores in a business system

Data Store: Data at rest, which may take the form of many different physical representations

Decision table: a tabular representation of processing logic containing decision variables, decision variable values, and actions or formulas.

Decision tree: a graphical description of process logic that uses lines organized like branches of a tree.

Decision Support System (DSS): An interactive information system that supports the decision making process through the presentation of information designed specifically for decision maker's problem approach and application needs

Economic Feasibility: A process of identifying the financial benefits and costs associated with a development project.

Gantt chart: a bar chart that represents the tasks and activities of the project schedule.

Gap analysis: The process of discovering discrepancies between two or more sets of data flow diagrams or discrepancies within a single DFD

Implementation: The phase of system development where the information system is coded, tested, installed, and supported in the organization

Information: Information is processed data. Information attributes a meaning to the data.

Information system: a collection of interrelated components that collect, process, store, and provide as output the information needed to complete business tasks.

Management Information System (MIS): A computer based system composed of people, hardware, software and procedures that share a common database to help users interpret data and apply to business

Maintainability: The ease with which software can be understood, corrected, adapted, and enhanced.

Maintenance: Maintenance includes all the software engineering activities that occur following delivery of a software product to the customer.

Operational Feasibility: The process of assessing the degree to which a proposed system solves business problems or takes advantage of business opportunities.

Perfective Maintenance: Changes made to a system to add new features or to improve performance.

Preventive Maintenance: Changes made to a system to avoid possible future problems Prototyping: An iterative process of systems development in which requirements are converted to a working system which is continually revised through close work between an analyst and users.

Schema: The design of a database is called as schema

Structured analysis: is a technique that helps the developer define what the system needs to do (the processing requirements), what data the system needs to store and use (data requirements), what inputs and outputs are needed, and how the functions work together overall to accomplish tasks.

Structure chart: a graphical model showing the hierarchy of program modules produced by the structured design technique.

Structured English: a method of writing process specifications that combine structured programming techniques with narrative English.

Subsystem: a system that is part of a larger system.

Super-system: a larger system that contains other systems

System a collection of interrelated components that function together to achieve a common goal.

Systems Analysis: the process of understanding and specifying in detail what the information system should do.

System boundary: the separation between a system and its environment that inputs and outputs must cross.

Systems Design: the process of specifying in detail how the many components of the information system should be physically implemented.

Systems Analyst: The organizational role most responsible for the analysis and design of information systems

Systems Analysis and Design: The complex organizational process whereby computer-based information systems are developed and maintained.

Systems design: Phase of the SDLC in which the system chosen for development in systems analysis is first described independent of any computer platform (logical design) and is then transformed into technology-specific details (physical design) from which all programming and system construction can be accomplished

System Documentation: Detailed information about a system - design specifications, it's internal workings, and its functionality.

Transactions: Individual, simple events in the life of an organization that contain data about organizational activity.

Transaction Processing System (TPS) A computerized information system developed to process large amounts of data for routine business transactions such as payroll and inventory.

Waterfall method: a method of executing an SDLC where one phase leads (falls) to the next phase.