Lesson No: 1 **Lesson Name : Overview of System Analysis & Design**

1.0 Objectives

- Defining a system
- The role of computer in information systems
- What are the characteristic and element of information system
- What are the various types of information system and models
- What are the different types of specialised information system

1.1 Introduction

In business, System Analysis and Design refers to the process of examining a business 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 employees 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.

System analysis and design focus on systems, processes and technology.

1.2 Over View of System Analysis and Design

Systems development can generally be thought of as having two major components: Systems analysis and Systems design. *System design* is the process of planning a new business system or one to replace or complement an existing system. But before this planning can be done, we must thoroughly understand the old system and determine how computers can best be used to make its operation more effective. *System analysis*, then, is the process of gathering and interpreting facts, diagnosing problems, and using the information to recommend improvements to the system. This is the job of the systems analyst.

Consider, for example, the stockroom operation of a clothing store. To better control its inventory and gain access to more up – to – date information about stock levels and reordering, the store asks a system analyst, to "computerize" its stockroom operations. Before one can design a system to capture data, update files, and produce reports, one needs to know more about the store operations: what forms are being used to store information manually, such as requisitions, purchase orders, and invoices and what reports are being produced and how they are being used.

To proceed, you then seek out information about lists of reorder notices, outstanding purchase orders, records of stock on hand, and other reports. You also need to find out where this information originates, whether in the purchasing department, stockroom, or accounting department. In other words, you must understand how the existing system works and, more specifically, what the flow of information through the system looks like.

You also must know why the store wants to change its current operations. Does the business have problems tracking orders, merchandise, or money? Does it seem to fall behind in handling inventory records? Does it need a more efficient system before it can expand operations?

Only after you have collected these facts can you being to determine how and where a computer information system can benefit all the users of the system. This accumulation of information, called a *systems study,* must precede all other analysis activities.

Systems analysts do more than solve current problems. They are frequently called upon to help handle the planned expansion of a business. In the case of the clothing store, the systems study is future oriented, since no system currently exists. Analysts assess as carefully as possible what the future needs of the business will be and what changes should be considered to meet these needs. In this instance and in most others, analysts may recommend alternatives for improving the situation. Usually more than one strategy is possible.

Working with managers and employees in the organization, systems analysts recommend which alternative to adopt, based on such concerns as the suitability of the solution to the particular organization and setting, as well as the employee support the solution is likely to have. Sometimes the time required to develop one alternative, compared with others, is the most critical issue. Costs and benefits are also important determinants. In the end, management, which will pay for and use the result, actually decides which alternative to accept.

Once this decision is made, a plan is developed to implement the recommendation. The plan includes all systems design features, such as new data capture needs, file specifications, operating procedures, equipment and personnel needs. The

systems design is like the blueprint for a building: it specifies all the features that are to be in the finished product.

Designs for the stockroom will provide ways to capture data about orders and sales to customers and specify the way the data will be stored, whether on paper forms or on a computer – readable medium, such as magnetic tape or disk. The designs will also designate work to be performed by people and by computers. Designs vary in their division of human and computer tasks.

The stockroom personnel will also need information about the business. Each design describes output to be produced by the system, such as inventory reports, sales analyses, purchasing summaries, and invoices. The systems analysts will actually decide which outputs to use, as well as how to produce them.

Analysis specifies *what* the system should do. Design states *how* to accomplish the objective. Notice that each of the processes mentioned involves people. Managers and employees have good ideas about what works and what does not, about what flows smoothly and what causes problems, about where change is needed and where it is not, and especially about where change will be accepted and where it will not. Despite technology, people are still the keys that make the organizations work. Thus, communicating and dealing with people are very important parts of the systems analyst's job.

1.3 Business System Concepts

The word system is widely used. It has become fashionable to attach the word system to add a contemporary flair when referring to things or processes. People speak of exercise system, investment system, delivery system, information system, education system, computer system etc. System may be referred to any set of components, which function in interrelated manner for a common cause or objective.

1.3.1Definition:

The term system is derived form the Greek word *systema*, which means an organized relationship among functioning units or components. A system exists because it is designed to achieve one or more objectives. We come into daily contact with the

transportation system, the telephone system, the accounting system, the production system, and, for over two decades, the computer system. Similarly, we talk of the business system and of the organization as a system consisting of interrelated departments (subsystems) such as production, sales, personnel, and an information system. None of these subsystems is of much use as a single, independent unit. When they are properly coordinated, however, the firm can function effectively and profitably.

There are more than a hundred definitions of the word system, but most seem to have a common thread that suggests that a system is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective. The word component may refer to physical parts (engines, wings of aircraft, car), managerial steps (planning, organizing and controlling), or a system in a multi level structure. The component may be simple or complex, basic or advanced. They may be single computer with a keyboard, memory, and printer or a series of intelligent terminals linked to a mainframe. In either case, each component is part of the total system and has to do its share of work for the system to achieve the intended goal. This orientation requires an orderly grouping of the components for the design of a successful system.

The study of systems concepts, then, has three basic implications:

1. A system must be designed to achieve a predetermined objective.
2. Interrelationships and interdependence must exist among the components.
3. The objectives of the organization as a whole have a higher priority than the objectives of its subsystems. For example, computerizing personnel applications must conform to the organization's policy on privacy, confidentiality and security, as will as making selected data (e.g. payroll) available to the accounting division on request.

1.4 Characteristics of a System

Our definition of a system suggests some characteristics that are present in all systems: organization (order), interaction, interdependence, integration and a central objective.

1.4.1 Organization

Organization implies structure and order. It is the arrangement of components that helps to achieve objectives. In the design of a business system, for example, the hierarchical relationships starting with the president on top and leading downward to the blue – collar workers represents the organization structure. Such an arrangement portrays a system – subsystem relationship, defines the authority structure, specifies the formal flow of communication and formalizes the chain of command. Like – wise, a computer system is designed around an input device, a central processing unit, an output device and one or more storage units. When linked together they work as a whole system for producing information.

1.4.2 Interaction

Interaction refers to the manner in which each component functions with other components of the system. In an organization, for example, purchasing must interact with production, advertising with sales and payroll with personnel. In a 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. The interrelationship between these components enables the computer to perform.

1.4.3 Interdependence

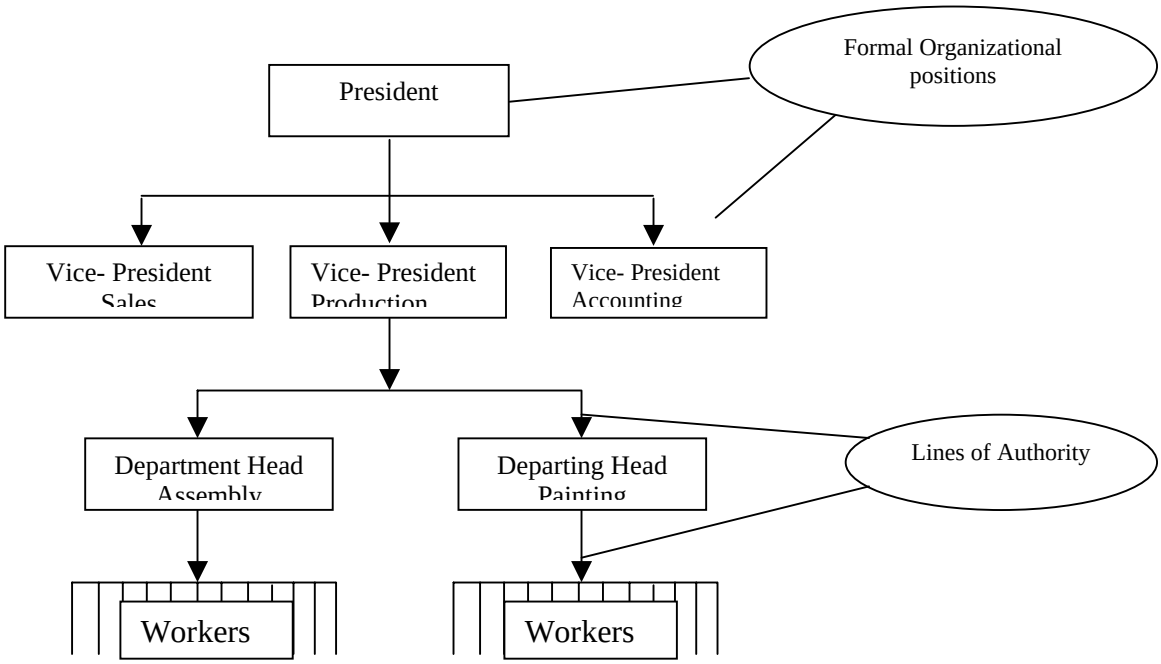
Interdependence means that parts of the organization or computer system depend on one another. They are coordinated and linked together according to a plan. One subsystem depends on the input of another subsystem for proper functioning: that is, the output of one subsystem is the required input for another subsystem. This interdependence is crucial in systems work.

An integrated information system is designed to serve the needs of authorized users (department heads, managers, etc.) for quick access and retrieval via remote terminals. The interdependence between the personnel subsystem and the organization's users is obvious.

In summary, no subsystem can function in isolation because it is dependent on the data (inputs) it receives from other subsystems to perform its required tasks. Interdependence is further illustrated by the activities and support of systems analysts, programmers, and the operations staff in a computer centre. A decision to computerize an

application is initiated by the user, analyzed and designed by the analyst, programmed and tested by the programmer, and run by the computer operator. None of these persons can perform property without the required input from others in the computer center subsystem.

Figure 1-1: Organization Structure – An Example



1.4.4 Integration

Integration refers to the holism of systems. Synthesis follows analysis to achieve the central objective of the organization. Integration 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. Successful integration will typically produce a synergistic effect and greater total impact than if each component works separately.

1.4.5 Central objective

The last characteristic of a system is its central objective. Objectives may be real or stated. Although a stated objective may be the real objective, it is not uncommon for an organization to state one objective and operate to achieve another. 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. Political as well as organizational considerations often cloud the real objective. This means that the analyst must work around such obstacles to identify the real objective of the proposed change.

1.5 Elements of a System

In most cases, systems analysts operate in a dynamic environment where change is a way of life. The environment may be a business firm, a business application, or a computer system. To reconstruct a system, the following key elements must be considered:

1. Outputs and inputs.
2. Processor(s).
3. Control.
4. Feedback.
5. Environment.
6. Boundaries and interface.

1.5.1 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.

1.5.2 Processor(s)

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.

1.5.3 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 behaviour 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.

1.5.4 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.

1.5.5 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.

1.5.6 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.

1.6 Types of systems

The frame of reference within which one views a system is related to the use of the systems approach for analysis. Systems have been classified in different ways. Common classifications are: (1) physical or abstract, (2) open or closed, and (3) "man – made" information systems.

1.6.1 Physical or abstract systems

Physical systems are tangible entities that may be static or dynamic in operation. For example, the physical parts of the computer center are the officers, desks, and chairs that facilitate operation of the computer. They can be seen and counted; they are static. In contrast, a programmed computer is a dynamic system. Data, programs, output, and applications change as the user's demands or the priority of the information requested changes.

Abstract systems are conceptual or non-physical entities. They may be as straightforward as formulas of relationships among sets of variables or models – the abstract conceptualization of physical situations. A model is a representation of a real or a planned system. The use of models makes it easier for the analyst to visualize relationships in the system under study. The objective is to point out the significant elements and the key interrelationships of a complex system.

1.6.2 Open or Closed Systems

Another classification of systems is based on their degree of independence. 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. An information system falls into this category, since it must adapt to the changing demands of the user. In contrast, a closed system is isolated from environmental influences. In reality, a completely closed system is rare. In systems analysis, organizations, applications and computers are invariably open, dynamic systems influenced by their environment.

Figure: 1.3 Gantt Chart – An Example

Gantt Chart

Departments	Number or workers	Capacity per week	May 5		6 12													
Stamping	75	3,000			25		28		22			29						
Sanding	10	400			21				25									
Assembly	60	2,400	19						20									
Painting	8	320	13					1	4									

A focus on the characteristics of an open system is particularly timely in the light of present – day business concerns with computer fraud, invasion of privacy, security controls, and ethics in computing. Whereas the technical aspects of systems analysis deal with internal routines within the user's application area, systems analysis as an open system tends to expand the scope of analysis to relationships between the user area and other users and to environmental factor that must be considered before a new system is finally approved. Furthermore, being open to suggestions implies that the analyst has to be flexible and the system being designed has to be responsive to the changing needs of the user and the environment.

Five important characteristics of open systems can be identified.

1. **Input from outside:** Open systems are self – adjusting and self-regulating. When functioning properly, an open system reaches a steady state or equilibrium. In a retail firm, for example, a steady state exists when goods are purchased and sold without being either out of stock or overstocked. An increase in the cost of goods forces a comparable increase in prices or decrease in operating costs. This response gives the firm its steady state.
2. **Entropy:** All dynamic systems tend to run down over time, resulting in entropy or loss of energy. Open systems resist entropy by seeking new inputs or modifying the processes to return to a steady state. In our example, no reaction to increase in cost of merchandise makes the business unprofitable which could force it into insolvency – a state of disorganization.
3. **Process, output and cycles:** Open systems produce useful output and operate in cycles, following a continuous flow path.
4. **Differentiation:** Open systems have a tendency toward an increasing specialization of functions and a greater differentiation of their components. In business, the roles of people and machines tend toward greater specialization and greater interaction. This characteristic offers a compelling reason for the increasing value of the concept of systems in the systems analyst's thinking.
5. **Equifinality:** The term implies that goals are achieved through differing courses of action and a variety of paths. In most systems, there is more of a consensus on goals than on paths to reach the goals.

Understanding system characteristics helps analysts to identify their role and relate their activities to the attainment of the firm's objectives as they undertake a system project. Analysts are themselves part of the organization. They have opportunities to adapt the organization to changes through computerized application so that the system does not "run down." A key to this process is information feedback from the prime user of the new system as well as from top management.

The theme of the process of designing information systems borrows heavily from a general knowledge of systems theory. The objective is to make a system more efficient by modifying its goals or changing the outputs.

1.6.3 Man – Made Information Systems

Ideally, information reduces uncertainty about a state or event. For example, information that the wind is calm reduces the uncertainty that the boat trip will be pleasant. An information system is the basis for interaction between the user and the analyst. It provides instruction, commands and feedback. It determines the nature of the relationships among decision-makers. In fact, it may be viewed as a decision center for personnel at all levels. From this basis, an information system may be defined as a set of devices, procedures and operating systems designed around user based criteria to produce information and communicate it to the user for planning, control and performance. In systems analysis, it is important to keep in mind that considering an alternative system means improving one or more of these criteria.

Many practitioners fail to recognize that a business has several information systems; each is designed for a purpose and works to accommodate data flow, communications, decision making, control and effectiveness. The major information systems are formal, informal and computer based.

Formal Information system

A formal information system is based on the organization represented by the organization chart. The chart is a map of positions and their authority relationships, indicated by boxes and connected by straight lines. It is concerned with the pattern of authority, communication and workflow. Information is formally disseminated in instructions, memos, or reports from top management to the intended user in the organization. This structure also allows feedback up the chain of command for follow –

up. In Figure 1-1 input form the environment provides impetus for policy decision by top management. Policies are generalizations that specify what an organization ought to do. Policies are translated into directives, rules and regulations and transmitted to lower-level management for implementation. The output represents employee performance.

1.7 Systems Models

In no field are models used more widely and with greater variety than in systems analysis. The analyst beings by creating a model of the reality (facts, relationships, procedures, etc.) with which the system is concerned. Every computer system deals with the real world, a problem area, or a reality outside itself. For examples, a telephone switching system is made up of subscribers, telephone handsets, dialing, conference calls, and the like. The analyst beings by modeling this reality before considering the functions that the system is to perform.

Various business system models are used to show the benefits of abstracting complex system to model form. The major models are schematic, flow, static and dynamic system models.

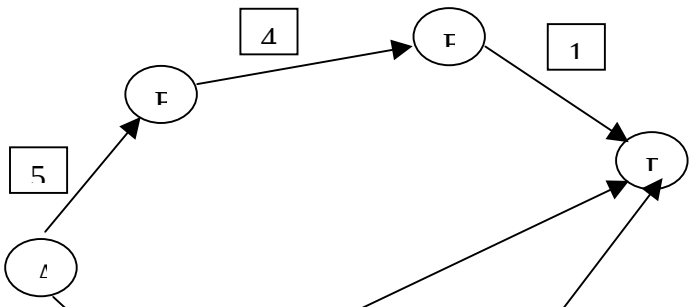
1.7.1 Schematic Models.

A schematic model is a two – dimensional chart depicting system elements and their linkages. Different arrows are used to depict information flow, material flow and information feedback. Various elements of the system are depicted in boxes.

1.7.2 Flow system Models.

A flow system model shows the flow of the material, energy and information that hold the system together. There is an orderly flow of logic in such models. A widely known example is PERT (Program Evaluation and Review Technique). It is used to abstract a real world system in model form, manipulate specific values to determine the critical path, interpret the relationships and relay them back as a control. The probability of completion within a time period is considered in connection with time, resources and performance specifications as shown in the figure1.2.

Figure 1.2 PERT an example



1.7.3 Static system models.

This type of model exhibits one pair of relationships such as activity – time or cost – quantity. The Gantt chart, for example, gives a static picture of an activity- time relationship. Planned activities (stamping, sanding etc.) are plotted in relation to time are shown in figure 1.3. The date column has light lines that indicate the amount of time it takes to complete a given activity. The heavy line represents the cumulative time schedule for each activity. The stamping department, for example, is scheduled to start working on order number 25 Wednesday morning and complete the job by the same evening. One day is also scheduled for order number 28, two days for order number 28, two days for order number 22 and two days (May 10-11) for order number 29. The heavy line opposite the stamping department represents the total of six days. The broken line indicates that the department is two days behind schedule. The arrowhead indicates the date when the chart is to be in effect.

1.7.4 Dynamic System Models.

Business organizations are dynamic systems. A dynamic model approximates the type of organization or application that analysts deal with. It depicts an ongoing, constantly changing system. It consists of (1) inputs that enter the system, (2) the processor through which transformation takes place, (3) the program(s) required for processing and (4) the output(s) that result from processing.

1.8 Categories of Information

There are three categories of information related to managerial levels and the decision managers make. The first level is strategic information, which relates to long –

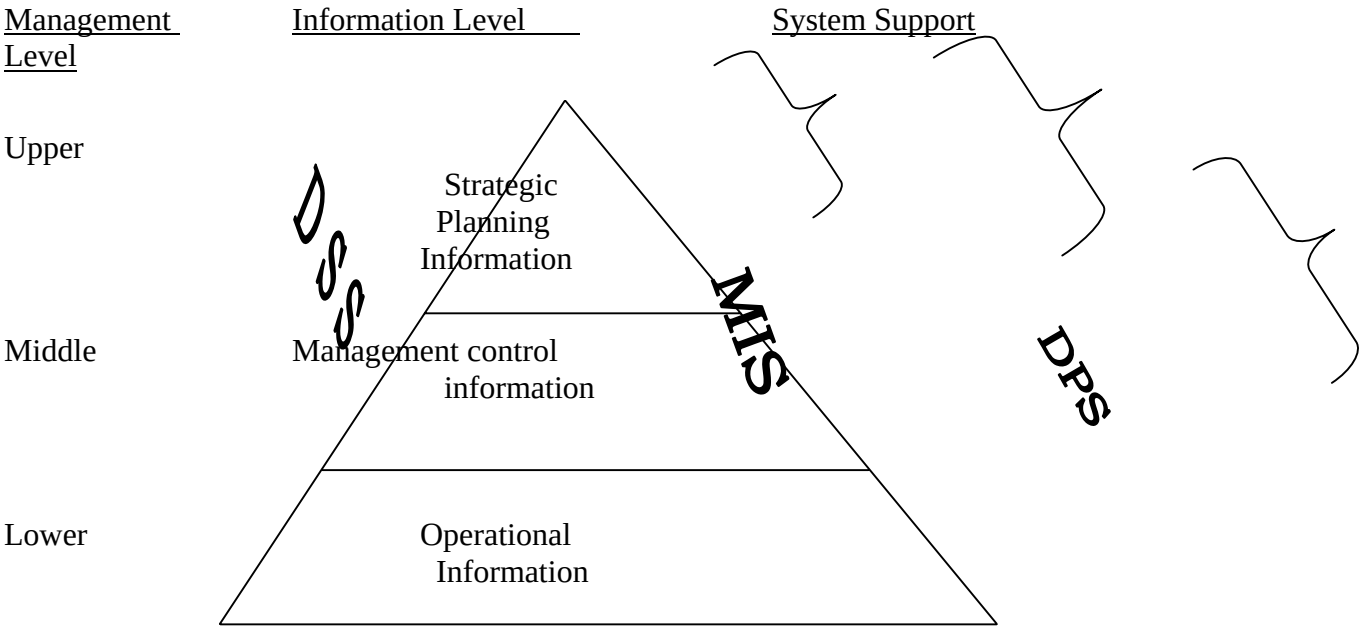
range planning policies that are of direct interest to upper management. Information such as population growth, trends in financial investment and human resources changes would be of interest to top company officials who are responsible for developing policies and determining long-range goals. This type of information is achieved with the aid of Decision Support System (DSS).

The second level of information is managerial information. It is of direct use to middle management and department heads for implementation and control. Examples are sales analysis, cash flow projection and annual financial statements. This information is of use in short – and intermediate -range planning – that is months rather than years. It is maintained with the aid of management information systems (MIS).

The third information level is operational information, which is short-term, daily information used to operate departments and enforce the day-to-day rules and regulations of the business. Examples are daily employee absent sheets, overdue purchase orders and current stocks available. Operational information is established by data processing systems (DPS). Figure 1.4 shows the same.

The nature of the information and managerial levels is also related to the major types of decision making: structured and unstructured decision making. An organizational process that is closed, stable and mechanistic tends to be more structured, computational and relies on routine decision making for planning and control. Such decision making is related to lower-level management and is readily supported with computer systems. In contrast, open, adaptive, dynamic processes increase the uncertainty associated with decision making and are generally evidenced by a lack of structure in the decision – making process. Lack of structure as well as extra-organizational and incomplete information makes it difficult to secure computer support. Table 1-2 summarizes the characteristics of decision making and the information required at different managerial levels.

Figure 1-4: Management and Information Levels in a Typical Organization.



Therefore, in designing an information system, the analyst needs to determine the type of information needed, the level of the information, how it is structured and in what format it is before deciding on the system needed to produce it. This is another reason for having a background in systems theory and organizations.

1.8.1 Informal Information Systems

The formal information system is a power structure designed to achieve company goals. An organization's emphasis on control to ensure performance tends to restrict the communication flow among employees. As a result, an informal information system develops. It is an employee based system designed to meet personnel and vocational needs and to help solve work – related problems. It also funnels information upward through indirect channels. In this respect, it is a useful system because it works within the framework of the business and it's stated policies.

In doing a systems study, the analyst should have a knowledge of the chain of command, the power-authority-influence network, and how decisions are made to get a feel for how much support can be expected for a prospective installation. Furthermore, knowledge about the inner workings of the employee- based system is useful during the exploratory phase of analysis. Employee cooperation and participation are crucial in preventing sabotage and training users. Since computers cannot provide reliable information without user staff support, a proper interface with the informal communication channels could mean the difference between the success and failure of new systems.

1.8.2 Computer – Based Information Systems

A third class of information system relies on the computer for handling business applications. The computer is now a required source of information. Systems analysis relies heavily on computers for problem solving. This suggests that the analyst must be familiar with computer technology and have experience in handling people in an organizational context.

1.8.2.1 Management Information Systems (MIS)

The computer has had a significant impact on the techniques used by management to operate a business. The level of the manager in the organization is also a factor in determining the kind of information needed to solve a problem. Lower – level management needs detailed internal information to make day – to – day, relatively structured control decisions. Higher – level management, for whom long – range objectives are the primary concerns, requires summarized information from a variety of sources to attain goals. In either case, management action is based on information that is accurate, relevant, complete, concise, and timely. MIS has been successful in meeting these information criteria quickly and responsively.

MIS is a person – machine system and a highly integrated grouping of information – processing functions designed to provide management with a comprehensive picture of specific operations. It is actually a combination of information systems. To do the job, it should operate in real time, handling inquires as quickly as they are received. Management information must also be available early enough to affect a decision. Operationally, MIS should provide for file definition, file maintenance and

updating, transaction and inquiry processing and one or more databases linked to an organizational database. Within a MIS, a single transaction can simultaneously update all related data files in the system. In so doing, data redundancy (duplication) and the time it takes to duplicate data are kept to a minimum, thus insuring that data are kept current at all times.

A key element of MIS is the database – a non-redundant collection of interrelated data items that can be processed through application programs and available to many users. All records must be related in some way. Sharing common data means that many programs can use the same files or records. Information is accessed through a data base management system (DBMS). It is a part of the software that handles virtually every activity involving the physical database.

There are several advantages to a data base system:

1. Processing time and the number of programs written are substantially reduced.
2. All applications share centralized files.
3. Storage space duplication is eliminated.
4. Data are stored once in the database and are easily accessible when needed.

The two primary drawbacks of a database are the cost of specialized personnel and the need to protect sensitive data from unauthorized access.

The primary users of MIS are middle and top management, operational managers and support staff. Middle and top management use MIS for preparing forecasts, special requests for analysis, long – range plans and periodic reports. Operational managers use MIS primarily for short- range planning, periodic and exception reports. The support staff finds MIS useful for the special analysis of information and reports to help management in planning and control. Providing data for use in MIS is the function of most levels of personnel in the organization. Once entered into the system, the information is no longer owned by the initiating user but becomes available to all authorized users.

Today's typical MIS poses several problems. Most MIS reports are historical and tend to be dated. Another problem is that many installations have databases that are not in line with user requirements. This means that many MIS environments have not been congruent with the real world of the user. Finally, an inadequate or incomplete update of the database jeopardizes the reliability for all users.

A major problem encountered in MIS design is obtaining the acceptance and support of those who will interface with the system. Personnel who perceive that their jobs are threatened may resist the implementation of MIS. In understanding both technology and human behavior, the analyst faces the challenge of selling change to the right people for a successful installation.

1.8.2.1 Decision Support Systems (DSS)

One reason cited in the literature of management's frustration with MIS is the limited support it provides top management for decision making. DSS advances the capabilities of MIS. It assists management in making decisions. It is actually a continually evolving model that relies heavily on operations research.

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.

Beginning with management decision systems in the early 1970’s the concept of interactive computer – based systems supporting unstructured decision making has been expanded to include everything but transaction processing systems. A typical early definition required an interactive computer – based system to help users use data and models to solve unstructured problems. There are authors today who view DSS as an extension of MIS, DSS as independent of MIS, or MIS as a subset of DSS. The commonly accepted view in the literature views DSS as a second – generation MIS. MIS is generated when we add predefined managerial reports that are spun out of the transaction processing, report generation and online inquiry capabilities – all integrated with a given functional area such as production MIS or personnel MIS. DSS result from adding external data sources, accounting and statistical models and interactive query capabilities. The outcome is a system designed to serve all levels of management and top management in particular, in dealing with “what if” unstructured problem situations. It is a system with the intrinsic capability to support ad hoc data analysis as well as decision – modeling activities.

The intelligence phase of decision making involves the awareness of a problem at a symptomatic level; it requires a closer look at the problem and a through evaluation of the variables and their relationships. The more intelligence management has about the cause of a problem, the better is the likelihood of designing a good decision. A DSS can provide intelligence through information retrieval and statistical packages.

The design phase of decision making focuses on the evaluation of decision alternatives. During this phase, computer – based deterministic or stochastic models may be used for decision design. DSS plays a major role in decision design under uncertainty. The output of the model(s) is the basis of the choice phase of decision-making.

1.10 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. System 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, operation in cycles through feedback, differentiation, and equifinality. Three level of information in organization that require 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. Future emphasises 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.

1.9 Questions:

1. Define system. Give examples.
2. What is man made information system.
3. Explain the features of a system.
4. Elaborate the different types of systems.
5. A system leads to a lot of planning and less of implementation. Do you agree, justify your answer.

Lesson No: 2 **Lesson Name : System Development Life Cycle**

2.0 Objectives

- How to build the computer based information system
- What are the different steps in system development life cycle
- What prompts users to change their request
- What are the various components of feasibility study
- What are the factors to consider in a candidate system

- How to plan and control for the system success

2.1 Introduction

The system analyst gives a system development project meaning & direction. A candidate system is approached after the analyst has a through understanding of user needs & problems. A viable solution is worked out and then communicates the same. Candidate systems often cut across the boundaries of users in the organization. For example, a billing system may involve users in the sales order department, the credit department, the warehouse and the accounting department. To make sure that all users' needs are met, a project from that represents each user works with the analysis to carry out a system development project.

2.2 Stages of system development Life cycle

The system development life cycle method is classically thought of as the set of activities that analysts, designers and users carry out to develop and implement an information system. The various stages in the business are closely related to each other, even the order of the steps in these activities is difficult to determine.

2.2.1 Project Selection

One must know what the problem is before it can be solved. The basis for a candidate system is recognition of a need for improving an information system or a procedure. For example, a supervisor may want to investigate the system flow in purchasing, or a bank president has been getting complaints about the long lines in the drive – in. This need leads to a preliminary survey or an initial investigation to determine whether an alternative system can solve the problem. It entails looking into the duplication of effort, bottlenecks, inefficient existing procedures, or whether parts of the existing system would be candidates for computerization.

If the problem is serious enough, management may want to have an analyst look at it. Such an assignment implies a commitment, especially if the analyst is hired from the outside. In larger environments, where formal procedures are the norm, the analyst's first task is to prepare a statement specifying the scope and objective of the problem. He/She then reviews it with user for accuracy. At this stage, only a rough " ball park" estimate of

the development cost of the project may be reached. However, an accurate cost of the next phase- the feasibility study – can be produced.

Impetus for system Change

The idea for change originates in the environment or from within the firm (see Figure 2-1). Environment-based ideas originate from customers, vendors, government sources, and the like. For example, new unemployment compensation regulations may make it necessary to change the restructures. Customer complaints about the delivery of orders may prompt an investigation of the delivery schedule, the experience of truck drivers, or the volume of orders to be delivered. When investigated, each of these ideas may lead to a problem definition as a first step in the system life cycle process.

Ideas for change may also come from within the organization- top management, the user, and the analyst. As an organization changes its operations or faces advances in computer technology, someone within the organization may feel the need to update existing applications or improve procedures. Here are some examples:

- An organization acquires another organization.
- A local bank branches into the suburbs.
- A department spends 80 percent of its budget in one month.
- Two departments are doing essentially the same work, and each department head insists the other department should be eliminated.
- A request for a new form discloses the use of bootleg (unauthorized) forms.

Serious problems in operations, a high rate of labor turnover, labor intensive activities, and high reject rates of finished goods, also prompt top management to initiate an investigation. Other examples are:

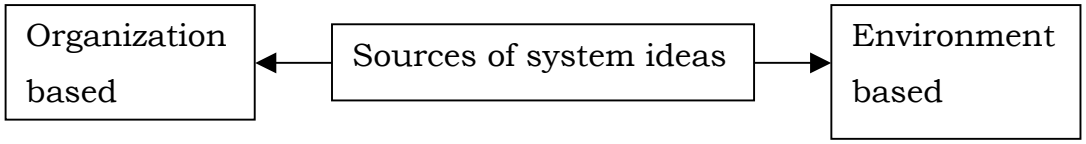
- A report reaches a senior vice president and she suspects the figures.
- The company comptroller reads an IRS audit report and starts thinking.
- An executive read about decision support systems for sales forecasting and it gives him an idea.

Many of these ideas lead to further studies by management request, often funneled downward and carried out by lower management.

User- originated ideas also prompt initial investigations. For example, a bank's head teller has been noticing long customer lines in the lobby. She wants to know whether they are due to the computers slow response to inquires, the new teller's limited training or just a sudden increase in bank business. To what extent and how quickly a user- originated idea is converted to a feasibility study depend on several factors:

- The risks and potential returns.
- Management's bias toward the user.
- Financial costs, and the funds, available for system work.
- Priorities of other projects in the firm.
- The persuasive ability of the user.

All these factors are crucial for a prompt response to a user request for change. A systems analyst is in a unique position to detect and even area of operations make him/ her a convenient resource for ideas. The role and status of the analyst as a professional add credibility to the suggestions made.



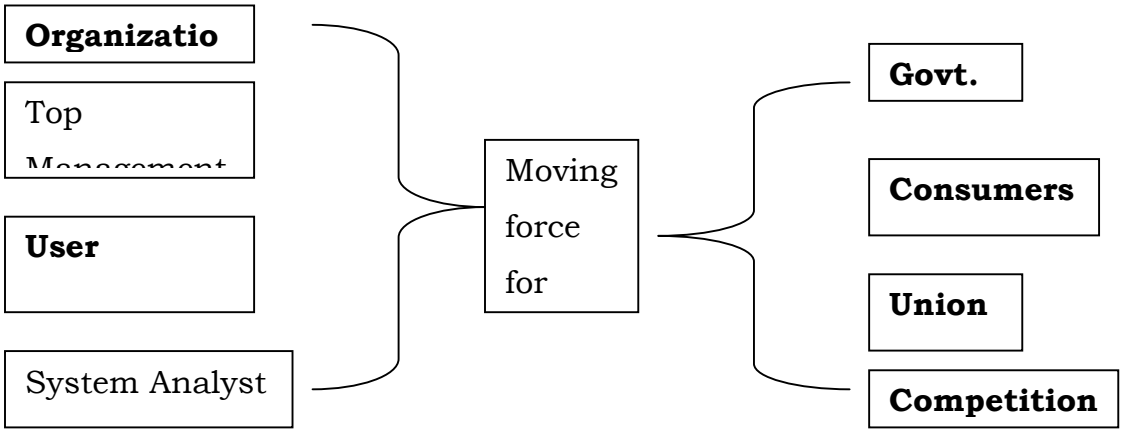


Figure 2.1 Major Sources of Change

2.2.2 Feasibility Study

Depending on the results of the initial investigation, the survey is expanded to a more detailed feasibility study. A feasibility study is a test

of a system proposal according to its workability. Impact on the organization, ability to meet user needs, and effective use of resources. It focuses on three major questions:

1. What are the user's demonstrable needs and how does a candidate system meet them?
2. What resources are available for given candidate systems? Is the problem worth solving?
3. What is the likely impact of the candidate system on the organization? How well does it fit within the organization's master MIS plan?

Each of these questions must be answered carefully. They revolve around investigation and evaluation of the problem, identification and description of candidate systems, specification or performance and the cost of each system and final selection of the best system.

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. It consists of the following:

1. Statement of the problem – a carefully worded statement of the problem that led to analysis.
2. Summary of findings and recommendations- a list of the major findings and recommendations of the study. It is ideal for the user who requires

quick access to the results of the analysis of the system under study. Conclusions are stated followed by a list of the recommendations and a justification for them.

3. Details of findings- an outline of the methods and procedures undertaken by the existing system followed by coverage of the objectives and procedures of the candidate system. Included are also discussions of output reports, file structures, and costs and benefits of the candidate system.

4. Recommendations and conclusions- specific recommendations regarding the candidate system including personnel assignments, costs, project schedules, and target dates.

After management reviews the proposal, it becomes a formal agreement that paves the way for actual design and implementations. This is a crucial decision point in the life cycle. Many project die here, whereas the more promising ones continue through implementations. Changes in the proposal are made in writing, depending on the complexity size, and cost of the project. It is simply common sense to verify changes before committing the project design.

2.2.3 Analysis

Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system. A key question is, what must be done to solve the problem? One aspect of analysis is defining the boundaries of the system and determining whether or not a candidate system should consider other related systems. During analysis, data are collected on the available files, decision points, and transactions handled by the present system. Data flow diagrams interviews, on – site observations, and questionnaires are examples of the analysis tools. The interviews is a commonly used tool in analysis, it requires special skills and sensitivity to the subjects being interviewed. Bias in data collection and interpretation can be a problem. Training, experience, and common sense are required for collection of the information needed to do the analysis.

Once analysis is completed the analyst has a firm understanding of what is to be done. The next step is to decide how the problem might be solved. Thus, in systems, design we move from the logical to the physical aspects of the life cycle.

2.2.4 Design

The most creative and challenging phase of the system life cycle is system design. The term design describes a final system and the process by which it is developed. It refers to the technical specifications (analogous to the engineer's blueprints) that will be applied in implementing the candidate system. It also includes the construction of programs and program testing. The key questions here is: How should the problem be solved? The major steps in design are shown in Figure 2.2.

The first step is to determine how the output is to be produced and in what format. Samples of the output (and input) are also presented. Second, input data and master files (database) have to be designed to meet the requirements of the proposed output. The operational (processing) phases are handled through program construction and testing including a list of the programs needed to meet the system's objectives and complete documentation. Finally, details related to justification of the system and an estimate of the impact of the candidate system on the user and the organization are documented and evaluated by management as a step toward implementation.

The final report prior to the implementation phase includes procedural flowcharts, record layouts, report layouts, and a workable plan for implementing the candidate system. Information on personnel, money, hardware, facilities, and their-estimated cost must also be available. At this point, projected costs must be close to actual costs of implementation.

In some firms, separate groups of programmers do the programming, whereas other firms employ analyst- programmers who do analysis and design as well as code programs. For this discussion, we assume that two separate persons carry out analysis and programming. There are certain functions, though, that the analyst must perform while programs are being written. Operating procedures must also be developed.

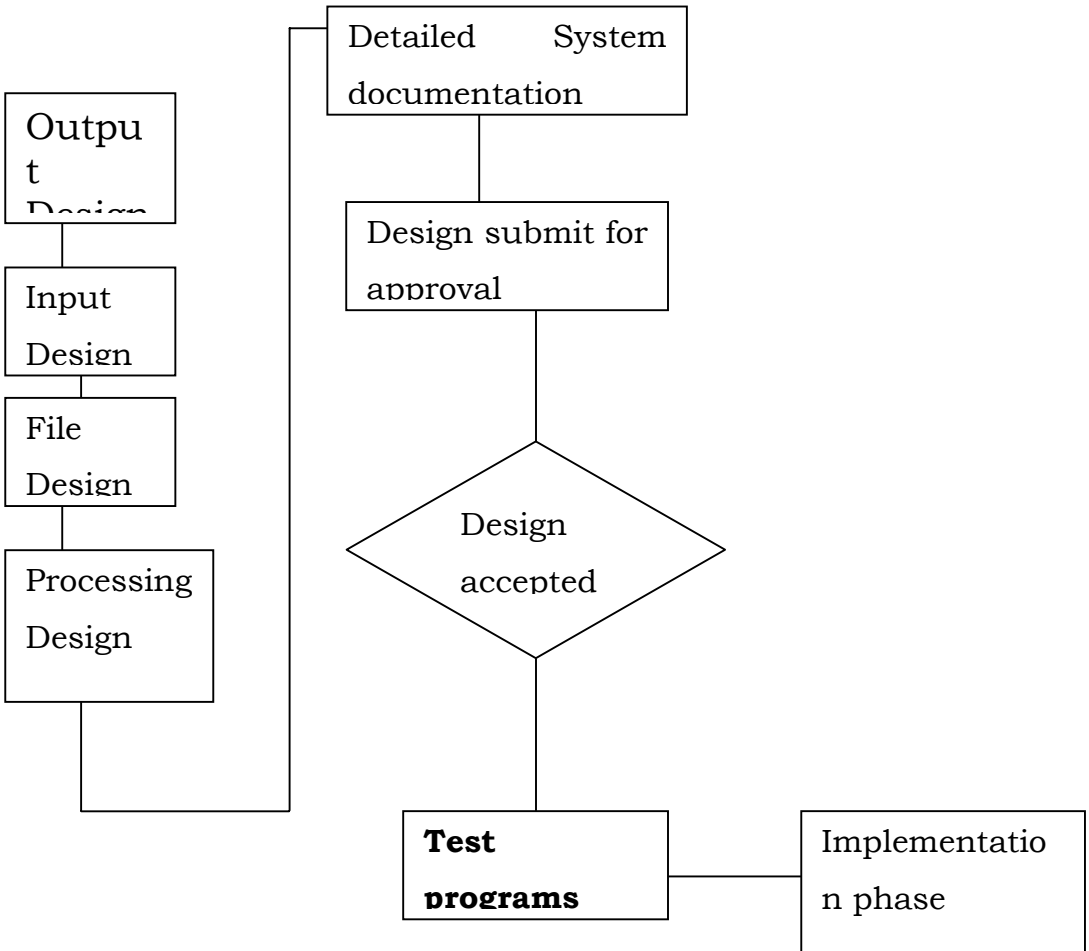
2.2.5 Implementation

The implementation phase is less creative than system design. It is primarily concerned with user training site preparation, and file conversion. When the candidate system is linked to terminals or remote sites, the telecommunication network and tests of the network along with the system are also included under implementation.

During the final testing, user acceptance is tested, followed by user training. Depending on the nature of the system, extensive user training may be required. Conversion usually takes place at about the same time the user is being trained or later.

In the extreme, the programmer is falsely viewed as someone who ought to be isolated from other aspects of system development. Programming is itself design work, however. The initial parameters of the candidate system should be modified as a result of programming efforts. Programming provides a " reality test" for the assumptions made by the analyst. It is therefore a mistake to exclude programmers from the initial system design.

Figure 2.2 Steps in systems design



System testing checks the readiness and accuracy of the system to access, update and retrieve data from new files. Once the programs become available, test data are read into the computer and processed against the file(s) provided for testing. If successful, the program(s) is then run with " live" data. Otherwise, a diagnostic procedure is used to locate and correct errors in the program. In most conversions, parallel run is conducted where the new system runs simultaneously with the "old" system. This method, though costly, provides added assurance against errors in the candidate system and also gives the user staff an opportunity to gain experience through operation. In some cases, however, parallel processing in not practical. For example, it is not plausible to run parallel two online point-of-sale (POS) systems for a retail chain. In any case, after the candidate system proves itself, the old system is phased out.

2.2.5.1 Post – Implementation and Maintenance

After the installation phase is completed and the user staff is adjusted to the changes created by the candidate system, evaluation and maintenance begin. Like any system there is an aging process that requires periodic maintenance of hardware and software. If the new information is inconsistent with the design specifications, then changes have to be made. Hardware also requires periodic maintenance to keep in tune with design specifications. The importance of maintenance is to continue to bring the new system to standards.

User priorities, changes in organizational requirements, or environmental factors also call for system enhancements. To contrast maintenance with enhancement, if a bank decided to increase its service charges on checking accounts from Rs 3.00 to Rs 4.50 for a minimum balance of Rs 300, it is maintenance. However, if the same bank decided to create a personal loan on negative balances when customers overdraw their account, it is enhancement. This change requires evaluation program modifications, and further testing.

Project Termination

A system project may be dropped at any time prior to implementation although it becomes more difficult (and costly) when it goes past the design phase. Generally, projects are dropped if, after a review process, it is learned that:

- Changing objectives or requirements of the user cannot be met by the existing design.
- Benefits realized from the candidate system do not justify commitment to implementation.
- There is a sudden change in the user's budget or an increase in design costs beyond the estimate made during the feasibility study.
- The project greatly exceeds the time and cost schedule.

In each case, a system project may be terminated at the user's request. In contrast project termination is new system failure. There are many reasons a new system does not meet user requirements:

- User requirements were not clearly defined or understood.
- The user was not directly involved in the crucial phases of system development.
- The analyst, programmer, or both were inexperienced.
- The systems analyst (or the project team) had to do the work under stringent time constraints. Consequently not enough thought went into the feasibility study and system design.

- User training was poor.
- Existing hardware proved deficient to handle the new application.
- The new system left users in other departments out of touch with information that the old system had provided.
- The new system was not user-friendly.
- Users changed their requirements.
- The user staff was hostile.

The list can be expanded to include many more causes. The important point is that although advances in computer systems and software make life easier for the analyst, the success of a system project depends on the experience, creative ability, and knowledge of the analyst and the support from the user staff. This suggests that the analyst be skilled in the state of the art (hardware and software) as well as in dealing with people.

2.3 Considerations for candidate system

In today's business, there is more demand for computer services than there are resources available to meet the demand. The demand is made up of the following:

1. Operations of existing system.
2. Maintenance that focuses on "patching" programs – often representing over 50 percent of maintenance.
3. Enhancements that involve major modifications in program structure or equipment.
4. Requests for candidate systems.

All these demands require resource – human, financial, and technological. On the human side, the computer department has to provide the following:

- ❖ Computer operators to run equipment.
- ❖ Data entry personnel.
- ❖ Systems analysts to define and design specifications.
- ❖ Application programmers to convert system specifications to computer programs

- Maintenance programmers to repair errors.
- Supervisors, project leaders, and managers to coordinate the jobs with the users.

Thus, the basic problem is to match the demands for service with the available resources. How much one project is favored over another depends on technical, behavioral, and economic factors.

The technical factor involves the system department's ability to handle a project. Much depends on the availability of qualified analysts, designers, and software specialists to do the work. This is especially true in designing databases and implementing complex systems for large concerns. The alternative to abandoning a project because of limited talent on the inside is free – lancing it to an outside consulting firm. The cost of developing the project has to be weighed against the total benefits expected.

The behavioral factor involves (1) the user's past experience with an existing system (2) the success record of the analyst, and (3) the influence the user can exert on upper management to finance a candidate system. Political considerations that subjectively favor one project over another, the status of the department, and its performance record are additional factors that bear on funding a candidate system.

Perhaps the most important criterion in selecting a project is the economic factor. It focuses on the system's potential return on investment. What is considered an acceptable rate varies with different formulas, the variables chosen, and the like. System consultants suggest an annual rate of return of just over 20 percent.

2.3.1 Political considerations

In conjunction with the preceding considerations is the political factor, which is partly behavioral. Imagine this setting: managers in a production firm are considering two office automation proposals: proposal A – a teleconferencing system designed to reduce travels costs, and proposal B- a sales support system. Proposal B (poorly presented and

justified) was sponsored by an influential executive and had the support of the committee. It passed because the right people were convinced it should.

Politics is the art of using influence and building coalitions when routine procedures do not achieve the right results. When system projects are developed, a collaborative relationship with the end user is helpful. A user who participated in building a system rarely criticizes it. If such a participative relationship comes too late, resistance can crop up and politics comes into play. The trick is to anticipate resistance early and turn it into support.

2.4 Planing and control for system success

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.

Most of this work falls under project management and control. The main idea behind the system development life cycle is to formalize a means structured at three major levels for effective control of the project. At the lowest level, work assignments are broken down into small manageable tasks. A task is usually a well – defined, structured work unit that can be carried out by one individual. The task can be easily budgeted and scheduled and its quality measured. It can be easily completed independent of other tasks and other project team members. If rework is necessary, there is minimal loss or impact on other tasks, except where time is critical.

The second level at which work units are structured involves activities that have larger scope and are designed to produce substantial results. An activity is a group of logically related tasks that serve one phase of the system development life cycle.

A phase, a third level of control, is a set of activities that bring the project to a critical milestone. Milestones are steppingstones that make up the entire project.

In planning a project, the following steps should be taken:

1. Identify the activities in each phase and the tasks within each activity.
2. Calculate the budget for each phase and obtain agreement to proceed.
3. Review, record, and summarize progress on activities periodically.
4. Prepare a project progress report at the end of a reporting month.

In summary, system development should not be regarded merely as some procedure that deals with hardware and software. The original assumptions upon which system specifications were based should be tested and re-evaluated with the user in mind. Managing system projects includes the important responsibility of seeing to it that all features of the candidate system – technological, logical, and behavioural – are considered before implementation and maintenance.

2.6 Summary:

System analysis and design are keyed to the system development life cycle(SDLC). The stages are project selection, feasibility, analysis, Design, implementation, and post implementation stages. t he idea for the project is originates in the environment or from within the organization. Once the problem is verified an initial investigation is conducted to determines whether change is feasible. If the answer is yes, a feasibility study is conducted. Analysis is a detailed study of the various operation performed by a system. System design refer to the technical specifications that will be applied in implementing the candidate system. Implementation is concerned with details of the candidate system. After implementation, maintenance begins includes enhancements, modifications, or any changes from the original specifications. To ensure the success of the system, careful and often extensive planning is required. The overall management process is crucial to the successful completion of system.

2.7 Questions:

1. Why is a system proposal so crucial for system design.

2. What is System Development Life Cycle.

3. What is the difference between analysis and design. Explain.

4. How would an analysis determine the users' needs for a system. Explain.

5. Distinguish between initial investigation and feasibility study. In what way are they related.6. How does system design simplify implementation.

7. What is testing.

8. How is testing different from evaluation.

9. There are several considerations in deciding on a candidate system. What are they. Why are they important.

Lesson No: 3 **Lesson Name : Project Selection**

3.0 Objectives:

- How the project selection will be done initially
- What are the different sources of the project request within and outside the organizarion
- How the user information is gathered and what are the various strategies to gather that information
- How the information is gathered from the existing system
- How the project selection and reviewing will be done by different committee
- How preliminary investigation and interview will be conducted
- How the different types of feasibility will be done

3.1 Introduction:

The first step in the system development life cycle is the identification of a need. 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 objective of project selection 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 build a new one.

The user's request form should specify the following:

1. User – assigned title of work requested.
2. Nature of work requested
3. Problem definition
4. Date request was submitted
5. Date job should be completed
6. Job objectives – purpose of job requested
7. Expected benefits to be derived from proposed change
8. Input / output description – quantity and frequency of inputs and outputs of proposed change.
9. Requester's signature, title, department etc.

10. Signature, title, department etc, 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 final. The success of a system depends largely on how accurately a problem is defined. The user's request must be communicated if the organization's personnel and other resources are to be successfully mobilized to build and maintain a viable information system plan.

3.2 Sources of project requests

There are four primary sources of project requests. The requesters inside the organization are department managers, senior executives, and systems analysts. In addition, government agencies outside the organization may request information systems projects. Depending on the origin of the request and the reason for it, requesters may seek either completely new applications or changes in existing ones.

3.2.1 Department Managers

Frequently, persons who deal with day-to-day business activities, whether employees or managers, are looking for assistance within their departments. For example, a business manager in a large medical clinic supervises the preparation of patient claim forms submitted to insurance companies, which reimburse the clinic for medical care. Even though the business manager knows that preparing insurance claims is necessary to aid the patient and ensure that the clinic is reimbursed, he or she may be dissatisfied with the amount of time the staff devotes to the task, especially when much insurance information (such as patient name, address, age, and the name of the attending physician) is already available in the patient's records. Pointing out the duplication of work, the bookkeepers express their desire to be free of the clerical tasks involved in processing claims.

After discussing the insurance problem with administrators in other clinics, the business manager asks the clinic's management com-preparing insurance forms and maintaining patient records about insurance payments.

This example is typical of cases where managers ask for systems projects. An ongoing activity needs improvement, either to solve a problem (for example, too many errors, excessive costs, or inconsistent work) or to improve the efficiency of job.

The department manager requesting a systems project many not consider the interaction between departments, even though the potential for such interaction can be high. For example, the manager who requests an inventory forecasting system for ordering materials and supplies may be looking primarily at ways to eliminate out – of – stock conditions. The request may not discuss the implications in other areas, such as fewer production problems due to material shortages, lower carrying costs for materials stored, or better process through quantity purchasing. Yet, on an organization – wide basis, these may be even more important reasons to consider the project. The point here is that project requests submitted by department managers seeking specific operating assistance may actually have wider implications that can affect other departments.

3.2.2 Senior Executives

Senior executives, such as presidents, board chairpersons, and vice presidents, usually have information about the organization that is not available to department managers. That information, coupled with the broader responsibilities these executives assume (they manage entire organizations rather than individual departments), influences the systems project requests they make. For example, the vice-president for manufacturing who knows that an additional production planning system one that will enable management to plan manufacturing at both plants at the same time. This project spans several departments (including manufacturing inventory control and purchasing) at two locations and involves many other managers.

The project requests submitted by senior executives are generally broader in scope than those prepared by department managers. Consider how many departments and divisions of an organization are included within the scope of a system request to design and implement a new corporate – wide budget system or a financial planning model.

Such projects tend to cut across more of the organization than does an inventory control system.

Multi-department projects are also more difficult to manage and control, however, departmental projects, in contrast, are more likely to be successful, especially if the actual users take an active role early in the project.

3.2.3 Systems analysts

Sometimes systems analysts see areas where projects should be developed and either write a systems proposal themselves or encourage a manager to allow the writing of a proposal on their behalf. For instance, an analyst who sees that a university's course – registration procedure is slow, error-prone, and generally inefficient may prepare a project proposal for a new registration system. The request prescribes the development of a system that takes advantage of new easy – to – use data entry terminals to speed registration.

Normally, department managers prepare proposals for operating systems, such as those for course registration. However, in this case the analyst has information about new equipment and technology that makes a more efficient registration system possible. The department manager, who is not responsible for researching computer technology, may not take the initiative for developing a systems proposal to facilitate registration procedures.

Do not forget that system analysts and developers can also be users themselves. Project management systems, file-monitoring packages, or programming library projects are typical of the application projects that systems personnel might request.

3.2.4 Outside Groups

Developments outside the organization also lead to project requests. For example, government contractors are required to use special cost accounting systems with government – stipulated features. The internal Revenue Service requires organizations to

keep careful payroll records and to account for employee income tax withheld. The internal Revenue Service also specifies the format for many of the tax documents that must be prepared; the employer has no choice in the matter.

Quite often, new demands from external groups bring about project requests, either for new systems or changes in current ones. Projects originating from this source are just as important as those from within the organization. Ins some cases, such as when there are strict deadlines imposed by the outside agency, these projects take on a higher priority than ones from, say, department managers.

3.3 Determining the user's Information Requirements

Shared, complete and accurate information requirements are essential in building computer – based information systems. Unfortunately, determining the information each user needs is particularly difficult task. In fact, it is recognized as one of the most difficult tasks in system development. The Association for Computing Machinery (ACM) Curriculum Committee on Computing Education for Management recognized this by suggesting two distinct job titles for systems developments: " information analyst" and " systems designer" rather than the more general term " systems analyst". 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 other 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 reasons why it is difficult to determine user requirements:

1. Systems requirements change and user requirements must be modified to account for those these changes.
2. The articulation of requirements is difficult, except for experienced users. Functions and processes are not easily described.
3. Heavy user involvement and motivation are difficult. Reinforcement for their work is usually not realized until the implementation phase – too long to wait.

4. The pattern of interaction between users and analysts in designing information requirements is complex.

Users and analysts traditionally do not share a common orientation toward problem definition. For example, in the analyst's view the problem definition must be translatable into a system design expressed quantitatively in terms of outputs, inputs, processes and data structures. This is the best of situations, and within time constraints. In contrast, the user seems to be satisfied with a qualitative definition that specifies the system in generalities. Flexibility is a key consideration. System specifications must change with their needs, as must the system after implementation.

Based on these contrasting views, users who try to define their information requirements with the analyst's views find themselves in a predicament. According to Scharer, they defend themselves by producing strategies that will satisfy the analyst·

1. In the kitchen sink 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.

2. The smoking 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 to one that is realistic, manageable, and achievable.

3. The same thing 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. Here the analyst has little chance of succeeding because only the usr can fully discover the real needs and problems.

3.4 Strategies for Determining Information Requirements

There are three key strategies or general approaches for eliciting information regarding the user's requirements: (1) asking, (2) getting information form the existing information system, and prototyping.

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 problems. There are three key asking methods:

1. Questions may be open-ended or closed. An open-ended question allows the respondent to formulate a response. It is used when feeling or opinions are important. For example, “How do you evaluate the latest addition to your hardware?” In contrast, a closed question requests one answer from a specific set or responses. It is used when factual responses are known. For example, “How long have you been manager of the computer centre?”

2. Brainstorming is a technique used for generating new ideas and obtaining general information requirements. This method is approach to brainstorming asks each participant to define ideal solutions and then select the best feasible one. It works well for users who have system knowledge but have difficulty accepting new ideas.

3. Group consensus asks participants for their expectations regarding specific variables. In a Delphi inquiry, for example, each participant fills out a questionnaire. The results are

summarized and given to participants along with a follow- up questionnaire. Participants are invite to change their responses. The results are again summarized and fed back to the participants. This debate by questionnaire continues until participants’ responses have converged enough. This method is an advantage over brainstorming in that participants are not subjected to psychological pressure from others with presumed authority or influence.

3.5 Getting Information from the Existing Information System.

Determining information from an existing application has been called the data analysis approach. It simply asks the user what information is currently received and what other information is required. It relies heavily on the user to articulate information needs. The analyst examines all reports, discusses with the user each piece of information

examined, and determines unfulfilled information needs by interviewing the user. The analyst is primarily involved in improving the existing flow of data to the user. In contrast to this method is decision analysis. This 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 the decision areas identified and the specific steps required to complete each major decision. Then the analyst and the user refine the decision process and the information requirements for a final statement of information requirements.

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.

In the decision analysis 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, though, is that information requirements may change when the user is promoted or replaced.

3.6 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 n which to base requirements or a difficulty in visualizing candidate systems. In this case, the user needs to anchor on real-life systems from which adjustments can be made. Therefore, the iterative discovery approach captures an initial set of information requirements and builds a system to meet these requirements. As user gain experience in its use, they request additional requirements or modifications (iterations), in the system in essence, information requirements are discovered by using the system. Prototyping is suitable in environments where it is difficult to formulate a concrete model for defining information requirements and where the information needs of the usr are evolving, such as in DSS.

Which of the three strategies is selected depends on uncertainties in the process of determining information requirements – that is, uncertainly with respect to the stability of

information requirements, the user's ability to articulate information requirements, and the ability of the analyst to elicit requirements and evaluate their accuracy. Thus, the asking strategy is appropriate for low- uncertainty information requirements determinations, whereas the prototyping strategy is appropriate for high uncertainty information requirements determination

3.7 Managing Project Review and Selection

Many more request for systems development are generated than most firms can pursue. Someone must decide which requests to pursue and which to reject (or perhaps solve by other means). The decision to accept or reject a request can be made in a number of different ways and by various members of the organization. The systems analysts are not the final arbiters.

One of the more common methods or reviewing and selecting projects for development is by committee.

3.7.1 Steering committee method

In many organizations, steering (also called operating committees, operating councils, or project selection boards) supervise the review of project proposal. The steering committee typically consists of key managers from various departments of the organization, as well as members of the information systems group. However, systems specialists do not dominate the committee. The information systems steering committee referred to in "The Good Old Days of Information Systems" at the beginning of this chapter included only two information systems specialists among its ten members. A typical seven to ten – person committee would consist of the following membership:

1. Upper – management members:
   Executive Vice president
   Vice President for manufacturing
2. Departmental management:
   Manager of retail marketing
   Credit manager

3. Technical managers:
   Manager of research and development
   Quality control coordinator
4. Information system group:
   Data processing manager
   Senior systems analyst

The committee receives proposals and evaluated them. The major responsibility of the committee is to make a decision, which often requires more information than the proposal provides, therefore, a preliminary investigation, is often requested to gather those details.

The steering – committee method brings high respectability and visibility to the review of project proposals. The committee consists of managers with the responsibility and the authority to decide which projects are in the best interest of the entire firm. Because several levels of management are included on the committee, members can have informed discussions on matters relating to day – to – day operations (treating patients, ordering materials, or hiring staff members) and long – range plans (new facilities, new programs) that many have a bearing on the project request. The managers provide practical information and insight about operations and long – term development. Systems specialists on the committee provide technical and developmental information that is useful in reaching decisions about project management.

The steering committee approach is often favored because systems projects are business investments. Management, not systems analysts or designers, selects projects for development, decisions are made on the basis of the cost of the project, its benefit to the organization, and the feasibility of accomplishing the development within the limits of information systems technology in the organization.

This is the method used by Peter Wallington's employer in "The Good Old Days of Information systems".

3.7.2 Information System Committee Method.

In some organizations, the responsibility for reviewing project requests is assigned to a committee of managers and analysts in the information systems department. Under this method, all request for service and development are submitted directly to a review committee within the information systems department. The information system committee approved or disapproved projects and sets priorities, indicating which projects are most important and should receive immediate attention.

This method can be used when many requests are for routine service or maintenance on existing applications. For these projects, information systems staff members can offer good insight into project requirements. In addition, by working with other projects (and by coordinating their efforts with the organization's business planning committee) systems developers can have access to information about where the firm is moving overall – an important consideration for effective project selection.

Sometimes, such as when major equipment decision must be made or when long – term development commitment are needed to undertake a project, the decision authority is shared with senior executives who determine whether a project should proceed. However, sharing project decision authority may confuse users who want to know how the committee will make the decision about a request. In addition, if top managers and systems- committee members disagree about the merit or priority of a request, the potential for conflict can disrupt the handling of future project proposals. In still other cases, users may attempt to submit a request directly to senior executives after it has been disapproved by the information systems committee. If upper management approves the request, the authority of the information systems committee is undermined.

3.7.3 User-group committee method

In some organizations, the responsibility for project decisions is delegated to the user themselves. Individual department or divisions hire their own analysts and designers, who handle project selection and carry out development. In effect, departments form their

own selection committees – user – group committees – controlling what is developed and when it is implemented.

Although the practice of having user committees both choose and develop systems does take some of the burden from the systems development group, it can have disadvantages for the users. For example, a number of small departments working independently toward the same goal could unknowingly waste resources and miss the opportunity to coordinate planning of a shared and integrated information system that could benefit the entire firm. A company's computer facilities can be unduly strained if the systems development team is not made aware of the future demands on the facilities that are being planned throughout the firm some user groups may find are being planned throughout the firm. Some user groups may find themselves with defective or poorly designed systems that require additional time and effort to undo any damage caused by the misinformation that such systems could generate. Although users groups may find the decision of steering committees and information systems committees disappointing at times, the success rate for who take on the development job is not very encouraging.

Membership often rotates under each of these committee formats, with individuals serving for, say six – or twelve – month periods. Membership changes are staggered to avoid changing the entire membership at one time. The chairperson of each committee should have experience in serving as a committee member and in reviewing systems proposals and making decisions about project requests.

3.7.4 Other methods

Other approaches are also tried from time to time, although usually with much less success than the methods already discussed. Some organization have management planning committees that propose new projects, which are in turn evaluated by the systems department staff members. This method suffers form lack of user involvement, as well as limited insight into technology.

In still other cases, department managers are able to bypass the organizations information systems departments to contract with independent systems companies, which handle all analysis and design work for projects. A disadvantage of this approach is the possibility that a department can sponsor the development of a system while the information system group or upper management is completely unaware that a project is in the making.

3.8 Preliminary investigation

Whether a system will be developed by means of the systems development life cycle method (SDLC) prototyping strategy, or the structured analysis method, or a combination of these methods, a project request should first be reviewed. The choice of development strategy is secondary to whether a request merits the investment of organization's resources in an information system project.

It is advisable for all proposals to be submitted to the selection committee for evaluation to identify those projects that are most beneficial to the organization. The preliminary investigation is then carried out by systems analysts, working under the direction of the selection committee.

3.8.1 Scope of study

The purpose of the preliminary investigation is to evaluate project requests. It is not a design study, nor does it include the collection of details to completely describe the business system. Rather, it is the collecting of information that permits committee members to evaluate the merits of the project request and make an informed judgement about the feasibility of the proposed project.

Analysts working on the preliminary investigation should accomplish the following objectives:

1. Clarify and understand the project request. What is being done? What is required? Why? Is there an underlying reason different form the one the requester identifies?

   Example: The user justifies a request for developing an accounts receivable system on the basis of wanting faster processing. However, the preliminary investigation may reveal that the need for better control of cash handling outweighs the need for sped. Lost checks, not speed of processing, are the real problem, but the requester has not described this specific need clearly.
2. Determine the size of the project.

   Example: Does a request for a course-registration project call for new development or for modification of the existing system? The investigation to answer this question will also gather the details useful in estimating the project. Since many enhancements of existing systems are costly, they are treated in the same way as new projects by the project selection committee.
3. Assess costs and benefits of alternative approaches.

   Example: What are the estimated costs for developing a patient information system, as requested by the hospital's chief of staff? What expenses will be incurred to train medical and nursing personnel and install the system? Will the proposed system reduce operating costs? Is it likely that the cost of errors will decrease?
4. Determine the technical and operational feasibility of alternative approaches.

   Example: Does the necessary technology to link office word processing systems to the main computer exist or can it be acquired? How workable is the request to enable administrative assistants to retrieve sales information form the main system and insert it directly into typewritten reports prepared on a word processor?
5. Report the findings to management, with recommendations outlining the acceptance or rejection of the proposal.

   Example: A proposal for the installation of an order entry system should be modified to allow all salespersons to submit their orders through ordinary telephone connections directly into the computer. The modification will improve

the usefulness of the system and increase the financial benefits to the organization.

3.9 Conducting the Investigation

The data that the analysts collect during preliminary investigations are gathered through two primary methods: reviewing documents and interviewing selected company personnel.

3.9.1 Reviewing Organization Documents

The analysts conducting the investigation first learn about the organization involved in, or affected by, the project. For example, to review an inventory systems proposal means knowing first how the inventory department operates and who the managers and supervisors are Analysts can usually learn these details by examining organization charts and studying written operating procedures. The procedures describe how the inventory process should operate and identify the most important steps involved in receiving, managing, and dispensing stock.

3.9.2 Conducting Interviews

Written documents tell the analysts how the systems should operate, but they may not include enough detail to allow a decision to be made about the merits of a systems proposal, nor do they present user views about current operations. To learn these details, analysts use interviews.

Interviews allow analysts to learn more about the nature of the project request and the reason for submitting it. To accomplish the purpose of the interviews, analysts must be sure to emphasize the request and the problem it addresses. In other words, interviews should

provide details that further explain the project and show whether assistance is merited economically, operationally, and technically. Working out a solution to the situation comes later, during the detailed investigation.

Usually, preliminary investigation interviews involve only management and supervisory personnel.

3.10 Testing Project Feasibility

Preliminary investigations examine project feasibility, the likelihood the system will be useful to the organization. Three tests of feasibility-all equally important-are studied: operational, technical and financial.

3.10.1 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.

3.10.2 Technical Feasibility

The technical issues usually raised during the feasibility stage of the investigation include these:

1. Does the necessary technology exist to do what is suggested (and can it be acquired)?
2. Does the proposed equipment have the technical capacity to hold the data required to use the new system?
3. Will the proposed system provide adequate responses to inquiries, regardless of the number or location of users?
4. Can the system be expanded if developed?
5. Are there technical guarantees of accuracy, reliability, ease of access, and data security?

   For example, if the proposal includes a printer that prints at the rate of 15,000 lines per minute, a brief search shows that this

specification is technically feasible. (Whether it should be included in the configuration is an economic decision.)On the other hand, if a user is requesting voice input to write, read, and change stored data, the proposal may not be technically feasible.

3.10.3 Financial and Economic Feasibility

A system that can be developed technically and that will be used installed must still be a good investment for the organization. Financial benefits must equal or exceed the costs. The financial and economic questions raised by analysts during he preliminary investigation are for the purpose of estimating the following:

1. The cost to conduct a full systems investigation
2. The cost of hardware and software for the class of application being considered.
3. The cost nothing changes (i.e., the proposed system is not developed

To be judged feasible, a project proposal must passed all these tests. Otherwise, it is not a feasible project. For example, a personnel record feasible if the necessary technology does not exit. A medical system that can be developed at reasonable costs but that nurses will avoid using cannot be judged operationally feasible.

3.11 Handling Infeasible Projects

Not all projects submitted for evaluation and review are judged acceptable. Requests that fail to pass feasibility tests are not pursued further, unless they are reworked and resubmitted as new proposals. In some cases, only part of a project is actually unworkable, and the selection committee may decide to combine the workable part of the project with another feasible proposal.

In still other cases, preliminary investigations produce enough new information to suggest that improvements in management and supervision, not the development of information systems, are the actual solutions to reported problems.

3.12 Summary:

The first step in the system development life cycle is the identification of a need. 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 objective of project selection 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 build a new one. There are four primary sources of project requests. The requesters inside the organization are department managers, senior executives, and systems analysts. Shared, complete and accurate information requirements are essential in building computer – based information systems. The Association for Computing Machinery (ACM) Curriculum Committee on Computing Education for Management recognized this by suggesting two distinct job titles for systems developments: " information analyst" and " systems designer" rather than the more general term " systems analyst". There are three key strategies or general approaches for eliciting information regarding the user's requirements: (1) asking, (2) getting information form the existing information system, and 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. Managing Project Review and Selection will be done by different methods such as Steering committee method, Information System Committee Method, User-group committee method and others method. The data that the analysts collect during preliminary investigations are gathered through two primary methods: reviewing documents and interviewing selected company personnel. Preliminary investigations examine project feasibility, the likelihood the system will be useful to the organization. Three tests of feasibility-all equally important-are studied: operational, technical and financial. Not all projects submitted for evaluation and review are judged acceptable.

3.13 Questions:

1. Explain closed questions and open-ended questions with examples.

2.What planning dimensions determine information system development? Elaborate.

3. Why is it difficult to determine user requirements.

4. What is the purpose of preliminary investigation.

5. What is an infeasible project and how are they handled.

Lesson No: 4 **Lesson Name : Feasibility Study**

4.0 Objectives:

- How the project selection will be conducted
- What are the various steps involves in defining system performance
- What are the key considerations are involved in feasibility analysis
- How to conduct the feasibility study

4.1 Introduction: -

The project selection phase is over resulting in completion of various activities:

- ❖ Recognition of need.
- ❖ Determination of user requirements.
- ❖ An initial investigation.
- ❖ Verification of objectives, constraints, required outputs & required inputs.

The next step is to determine exactly what the candidate system is to do by defining its expected performance. Thus, a feasibility study is carried out to select the best system that meets performance requirements. It comprises of identification, description & evaluation of candidate system and finally selection of the best system for the job. The feasibility study recommends to the management either the most effective system or concludes that the system may not be evolved.

4.2 System Performance Definition

A system's required performance be defined by describing its output in a user-acceptable format and at a higher level of detail than what was described in the initial investigation. This involves three steps:

1. Statement of constraints.
2. Identification of specific system objectives.
3. Description of outputs.

This phase builds on the previous phase in that much of the work may already have been done.

4.2.1 Statement of constraints

Constraints are factors that limit the solution of the problem. Some constraints are identified during the initial investigation and are discussed with the user. There are

general constraints that might have a bearing on the required performance of a candidate system. Let's consider safe deposit billing system to illustrate these points. The current billing system and the department handling billing and customer payments face problems. The result of the fact-finding phase of the initial investigation revealed the following general constraints:

1. The president views safe deposit billing as a low priority. He has a false impression that computers are not needed as long as customers can have access to their boxes.

2. The senior vice president is worried that a billing system might require the transfer of safe deposit staff to other departments. Considering Florida's level of unemployment and the cost of retraining, a candidate system has to do more than produce reports.

3. The accounting department has been pushing for installing a computer-based general ledger application for months. The vice president of operations, bogged down with auditing and operations problems, continued to shelve the request.

4. Management has a limited knowledge of computers, although it has several applications on the computer: checking and savings, installment loans, commercial loans and trusts. The president, in his early sixties and interested in "the bottom line" of the financial statement, is traditionally reluctant to spend money on computers.

5. Safe deposit, while doing better than breaking even, is not projected to grow as fast as it did in the early 1980s. The community's recent success in controlling burglaries had an adverse impact on the demand for box rentals in general.

6. If an online system is to be installed, it must interface with the existing checking/savings application to allow for the automatic payment of box rentals.

7. A proposed design must be compatible with the bank's Burroughs computer system.

4.2.2 Identification of specific System Objectives

Once the constraints are spelled out, the analyst proceeds to identify the system's specific performance objectives. They are derived from the general objectives specified

in the project directive at the end of the initial investigation. The steps are to state the system's benefits and then translate them into measurable objectives. In our scenario, the candidate system's anticipated benefits are as follows:

1. Improved collection schedule.
2. Cost reduction.
3. Physical space reduction.
4. Improved customer service.

Each benefit is analyzed and translated into measurable objectives.

1. Collection is improved by billing 30 days in advance of the box renewal data, and one more notice is sent within two weeks. It also improves the account receivables payment "float."
2. Cost reduction is realized by reducing the payroll by two employees. The new online billing system requires less than two hours of labor per day, compared with six hours under the current system.
3. Placing the microcomputer in the place of one of the four existing desks reduces physical space requirements. The remaining desks are removed, allowing an extra cubicle for customer use.
4. Placing master cards and rental information online improve customer service, thus reducing the waiting time of entry from 3 minutes to 30 seconds.

These objectives are effective in comparing the performance of the candidate system with that of the current system. The information – oriented flowchart, input/output analysis sheet and data flow diagram produced in the initial investigation lead to the conclusions that (1) the current system is inefficient and (2) a new online, terminal – oriented system would be the solution. This conclusion was reflected in the general project directive submitted to the user for approval. This information is used as a basis for preparing specific objectives for the candidate system:

1. To establish a billing system with six five-day cycles per month.

2. To mail customers no later than the close of the billing cycle and no later than 25 days prior to the box renewal date.

3. To mail customers a reminder two weeks after the initial statement for box renewal.

4. To speed collections and reduce the “float” by 40 percent.

5. To examine the availability of boxes by size, rental fees and location.

6. To evaluate the ratio of rented to available boxes at all times.

7. To produce periodic reports to management on the performance of the safe deposit department.

4.2.3 Description of output

A final step in system performance definition is describing the output required by the user. An actual sketch of the format and contents of the reports (layout) as well as a specification of the media used, their frequency and the size and number of copies required are prepared at this point. Specifying exactly what the output will look like leads to an estimate of the computer storage requirements that form the basis for the file design to be undertaken in the design phase of the life cycle. The analyst is now ready to evaluate the feasibility of candidate systems to produce these outputs.

4.3 Feasibility Study

Many feasibility studies are disillusioning for both users and analysts. First, the study often presupposes that when the feasibility document is being prepared, the analyst is in a position to evaluate solutions. Second most studies tend to overlook the confusion inherent in system development-the constraints and the assumed attitudes. If the feasibility study is to serve as decision document it must answer three key questions:

1. Is there a new and better way to do the job that will benefit the user?
2. What are the costs and savings of the alternative (s)?
3. What is recommended?

The most successful system projects are not necessarily the biggest or most visible in a business but rather those that truly meets user expectations. More projects fail because of inflated expectations than for any other reason.

4.3.1 Feasibility Considerations

Three key considerations are involved in the feasibility analysis: economic, technical and behavioral. Let’s briefly review each consideration and how it relates to the systems effort.

4.3.2 Economic Feasibility

Economic analysis is the most frequently used method for evaluating the effectiveness of a candidate 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. Otherwise, further justification or alterations in the proposed system will have to be made if it is to have a chance of being approved. This is an ongoing effort that improves in accuracy at each phase of the system life cycle.

4.3.3 Technical Feasibility

Technical feasibility centers around the existing computer system (hardware, software, etc.) and to what extent it can support the proposed addition. For example, if the current computer is operating at 80 percent capacity-an arbitrary ceiling- then running another application could overload the system or require additional hardware. This involves financial considerations to accommodate technical enhancements. If the budget is a serious constraint, then the project is judged not feasible.

4.3.4 Behavioral Feasibility

People are inherently resistant to change, and computers have been known to facilitate change. An estimate should be made of how strong a reaction the user staff is likely to have toward the development of a computerized system. It is common knowledge that computer installations have something to do with turnover, transfers, retraining and changes in employee job status. Therefore, it is understandable that the introduction of a candidate system requires special effort to educate, sell and train the staff on new ways of conducting business.

In safe deposit example, three employees are more than 50 years old and have been with the bank over 14 years, four of which have been in safe deposit. The remaining two employees are in their early thirties. They joined safe deposit about two years before the study. Based on data gathered from extensive interviews, the younger employees want the programmable aspects of safe deposit (essentially billing) put on a computer. Two of the three older employees have voiced resistance to the idea. Their view is that billing is no problem. The main emphasis is customer service – personal contact with customers. The decision in this case was to go ahead and pursue the project.

4.4 Steps in Feasibility Analysis

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.

1. Form a project Team and Appoint a Project Leader

The concept behind a project team is that future system users should be involved in its design and implementation. Their knowledge and experience in the operations area are essential to the success of the system. For small projects, the analyst and an assistant usually suffice; however, more complex studies require a project team. The team consists of analysts and user staff - enough collective expertise to devise a solution to the problem. In many cases, an outside consultant and an information specialist join the team until the job is completed.

Projects are planned to occupy a specific time period, ranging from several weeks to months. The senior systems analyst is generally appointed as project leader. He/she is usually the most experienced analyst in the team. 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.

Regarding the safe deposit case, since the whole user area consists of five employees, the analyst handled most of the work.

2. Prepare System Flowcharts

The next step in the feasibility study is to prepare generalized system flowcharts for the system. Information – oriented charts and data flow diagrams prepared in the initial investigation are also reviewed at this time. The charts bring up the importance of inputs, outputs and data flow among key points in the existing system. All other flowcharts needed for detailed evaluation are completed at this point.

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. In the safe deposit case, it was found that virtually any microcomputer system with more than 128k –byte memory an dual disk drive will do the job. It was also learned that a microcomputer can be designed to interface with the bank's mainframe. In this design, actual processing is handled by the microcomputer, whereas information such as payments and credits are transmitted to the main computer files for proper adjustment through the customer's checking account. The question here is: which microcomputer (IBM, Apple, Digital etc.) should be selected? This is taken up in step 6 of the study.

An important aspect of hardware is processing and main memory. There are a large number of computers with differing processing sizes, main memory capabilities and software support. The project team may contact vendors for information on the processing capabilities of the system available.

4. Describe and Identify Characteristics of Candidate System

From the candidate systems considered, the team begins a preliminary evaluation in an attempt to reduce them to a manageable number. Technical knowledge and expertise in the hardware / software area are critical for determining what each candidate system can and cannot do. In the safe deposit example, a search for the available microcomputers and safe deposit billing packages revealed the information summarized in Table 4-1.

These packages were the result of a preliminary evaluation of more than 15 other packages – all purporting to meet the requirements, of the safe deposit billing system. When the number is reduced to three key packages, the next step is to describe in some detail the characteristics of each package. For example, the first candidate system runs on an IBM PC with a minimum of 128K-bytes of memory. The software is written in Oracle, a relatively new language. In case of enhancements, change has to be made through the software house, since the source code is not available to the user.

Table-4-1 Safe Deposit Billing package and Selected Characteristics

Characteristics	IBM PC	HP 100	Apple III
Memory required (K bytes)	128	64	264
Source language	Oracle	VB	VB
Source Code Available	No	Yes	No
Purchase terms	Purchase (License)	Purchase (License)	Purchase (License)
Purchase price	Rs.99500	Rs.80000	Rs.1,09500
Number installed to date	200	30	50
Date of first installation	2002	2001	2000

The first package was installed in January 2002. More than 200 packages have been installed to date.

The next two candidate systems are similarly described. The information along with additional data available through the vendor highlights the positive and negative features of each system. The constraints unique to each system are also specified. For example, in the IBM PC package, the lack of an available source code means that the user has to secure a maintenance contract that costs 18 percent of the price of the package per year. In contrast the HP 100 package is less expensive and offers a source code to the

user. A maintenance contract (optional) is available at 18 percent of the price of the package.

5. Determine and Evaluate Performance and Cost Effectiveness of Each Candidate System

Each candidate system's performance is evaluated against the system performance requirements set prior to the feasibility study. Whatever the criteria, there has to be as close a match as practicable, although trade-offs are often necessary to select the best system. In the safe deposit case, the criteria chosen in advance were accuracy, growth potential, response time less than five seconds, expandable main and auxiliary storage, and user-friendly software. Often these characteristics do not lend themselves to quantitative measures. They are usually evaluated in qualitative terms (excellent, good, etc.) based on the subjective judgement of the project team.

The cost encompasses both designing and installing the system. It includes user training, updating the physical facilities and documenting. System performance criteria are evaluated against the cost of each system to determine which system is likely to be the most cost effective and also meets the performance requirements. The safe deposit problem is easy. The analyst can plot performance criteria and costs for each system to determine how each fares.

Costs are more easily determined when the benefits of the system are tangible and measurable. An additional factor to consider is the cost of the study design and development. The cost estimate of each phase of the safe deposit project was determined for the candidate system (IBM PC). In many respects, the cost of the study phase is a "sunk cost" (fixed cost). Including it in the project cost estimate is optional.

6. Weight System Performance and Cost Data

In some cases, the performance and cost data for each candidate system show which system is the best choice? This outcome terminates the feasibility study. Many times, however, the situation is not so clear – cut. The performance / cost evaluation matrix at times does not clearly identify the best system, so the next step is to weight the

importance of each criterion by applying a rating figure. Then the candidate system with the highest total score is selected.

The procedure for weighting candidate systems is simple: -

1. Assign a weighting after factor to each evaluation criterion based on the criterion' effect on the success of the system. For example, if the usability criterion is twice as important as the accuracy factor, usability is assigned weight 4 and accuracy is assigned weight 2.
2. Assign a quantitative rating to each criterion's qualitative rating. For example, ratings (poor, fair, good, very good, excellent) may be assigned respective values (1,2,3,4,5).
3. Multiply the weight assigned to each category by the relative rating to determine the score.
4. Sum the score column for each candidate system.

Thus, the weighted candidate evaluation matrix is prepared using these steps, which in it self helps in the next step.

7. Select the Best Candidate System

The system with highest total score is judged the best system. This assumes the weighting factors are fair and the rating of each evaluation criterion is accurate. The criterion of growth potential is generally given the maximum weight, thus the greatest effect on the total score. Additionally, system development and user training are also given high weights.

Most feasibility studies select from more candidate systems than we have mentioned in our example. The criteria chosen and the constraints are also more complex. In any case, management should not make the selection without having the experience to do so. Management cooperation and comments, however, are encouraged.

8. Feasibility Report

The culmination of the feasibility study is a feasibility report directed to management; it evaluates the impact of the proposed changes on the area(s) in question. The report is a formal document for management use, brief enough and sufficiently non-

technical to be understandable, yet detailed enough to provide the basis for system design.

There is no standard format for preparing feasibility reports. Analysts usually decide on a format that suits the particular user and system. Most reports, however, begin with a summary of findings and recommendations, followed by document details. Starting with summary information highlights the essence of the report, giving management the option of reviewing the details later. The report contains the following sections:

1. Cover letter formally presents the report and briefly indicates to management the nature, general findings and recommendations to be considered.
2. Table of content specifies the location of the various parts of the report. Management quickly refers to the sections that concern them.
3. Overview is a narrative explanation of the purpose scope of the project, the reason for undertaking the feasibility study and the department(s) involved or affected by the candidate system. Also included are the names of the persons who conducted the study, when it began, and other information that explains the circumstance surrounding the study.
4. Detailed findings outline the methods used in the present system. The system's effectiveness and efficiency as well as operating costs are emphasized. The section also provides a description of the objectives and general procedures of the candidate system. A discussion of output reports, costs and benefits gives management a feel for the pros and cons 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. A return on investment (ROI) analysis of the project is also included.
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.

Disapproval of the feasibility report is rare if it has been conducted properly. When a feasibility team has maintained good rapport with the user and his/ her staff it makes the recommendations easier to approve. Technically, the report is only a recommendation, but it is an authoritative one. Management has the final say. Its approval is required before system design is initiated.

Oral Presentation

The feasibility report is a good written presentation documenting the activities involving the candidate system. The pivotal step, however, is selling the proposed change. Invariably the project leader or analyst is expected to give an oral presentation to the end user. Although it is not as polished as the written report, the oral presentation has several important objectives. 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 and (2) the ability to answer questions, clarify issues, maintain credibility and pick up on any new ideas or suggestions.

The substance and form of the presentation depend largely on the purposes sought. Table 4.2 suggests a general outline. The presentation may aim at informing, confirming, or persuading.

1. **Informing.** This simply means communicating the decisions already reached on system recommendations and the resulting action plans to those who will participate in the implementation. No detailed findings or conclusions are included.

2. **Confirming.** A presentation with this purpose verifies facts and recommendations already discussed and agreed upon. Unlike the persuading approach, no supportive evidence is presented to sell the proposed change, nor is there elaborate reasoning behind recommendations and conclusions. Although the presentation is not detailed, it should be complete. Confirming is itself part of the

process of securing approval. It should reaffirm the benefits of the candidate system and provide a clear statement of results to be achieved.

TABLE 4.2 Oral Presentation – Suggested Outline

1. Introduction
a. Introduce self
b. Introduce topic.
c. Briefly describe current system.
i. Explain why it is not solving the problem
ii. Highlight user dissatisfaction with it.
iii. Briefly describe scope, objectives and recommendation of the proposed system.
2. Body of presentation.
a. Highlight weaknesses of current system.
b. Describe proposed system. How is it going to solve the problem?
c. Sell proposed system.
i. Specify savings and benefits, costs and expenses.
ii. Use visual aids to justify project and explain system.
d. Summarize implementation plan and schedule.
e. Review human resources requirements to install system.
3. Conclusion.

a. Summarize proposal
b. Restate recommendations and objectives of proposal.
c. Summarize benefits and savings.
d. Ask for top-level management support. Solicit go-ahead for project.
4. Discussion period- Answer questions convincingly.

3. **Persuading.** This is a presentation pitched toward selling ideas- attempts to convince executives to take action on recommendations for implementing a candidate system.

Regardless of the purpose sought, the effectiveness of the oral presentation depends on how successful the project team has been in gaining the confidence of frontline personnel during the initial investigation. How the recommendations are presented also has an impact. Here are some pointers on how to give an oral presentation:

1. Rehearse and test your ideas before the presentation. Show that you are in command. Appear relaxed.
2. Final recommendations are more easily accepted if they are presented as ideas for discussion, even though they seem to be settled and final.
3. The presentation should be brief, factual and interesting Clarity and persuasiveness are critical. Skill is needed to generate enthusiasm and interest throughout the presentation.
4. Use good organization. Distribute relevant material to the user and other parties in advance.
5. Visual aids (graphs, charts) are effective if they are simple, meaningful and imaginative. An effective graph should teach or tell what is to be communicated.

6. Most important, present the report in an appropriate physical environment where the acoustics, seating pattern, visual aid technology and refreshments are available.

The most important element to consider is the length of the presentation. The duration often depends on the complexity of the project, the interest of the user group and the competence of the project team. A study that has company wide applications and took months to complete would require hours or longer to present. The user group that was involved at the outset would likely permit a lengthy presentation, although familiarity with the project often dictates a brief presentation. Unfortunately, many oral presentations tend to be a rehash of the written document with little flare or excitement. Also, when the analyst or the project leader has a good reputation and success record from previous projects, the end user may request only a brief presentation.

4.5 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. Three key considerations are involved in feasibility analysis are economic, technical, and behavioural 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.

4.6 Questions:

1. Elaborate System performance.
2. Explain economic, technical & behavioral feasibility.
3. Explain the importance of oral presentation.
4. Briefly discus the various steps in Feasibility Analysis.
5. What is the importance of feasibility study.

Lesson No: 5 **Lesson Name : Cost/Benefit Analysis**

5.0 Objectives:

- What is involved in data analysis
- What are cost and benefit categories
- How to identify and classify cost and benefits
- What are various evaluation methods for cost/benefit analysis

5.1 Introduction

Each problem has generally more than one solution. Each such approach has costs & benefits that are compared with those of other approaches before a final recommendation is made. The result is a project proposal. The findings of the analysis are summarized and the design is recommended.

5.2 Data Analysis

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 deign requirements are identified, which could be:

1. Better customer service.
2. Faster information retrieval
3. Quicker reports.
4. Less time consuming.
5. Accuracy.
6. Reduce data redundancy.
7. Improved staff efficiency.
8. Lower processing & operating costs.

To achieve these design objectives, several alternatives, must be evaluated, there is seldom just one alternative. The analyst selects those that are feasible economically, technically and operationally. Each approach has its benefits and drawbacks. An analysis of the costs & benefits of each alternative guides the selection process.

5.3 Cost and Benefit Categories

In developing cost estimates for a system, we need to consider several cost elements. Among them are 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 one-time costs and are incorporated into the overall cost estimate of the candidate system.

4. **Operating 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.

A system is also expected to provide benefits. The first task is to identify each benefit and then assign a monetary value to it for cost/ benefit analysis. Benefits may be tangible and intangible, direct or indirect.

The two major benefits are improving performance and minimizing the cost of processing. The performance category emphasizes improvement in the accuracy of or

access to information and easier access to the system by authorized users. Minimizing costs through an efficient system – error control or reduction of staff- is a benefit that should be measured and included in cost/benefit analysis.

5.4 Procedure for Cost/ Benefit Determination

There is a difference between expenditure and investment. We spend to get what we need, but we invest to realize a return on the investment. Building a computer – based system is an investment. Costs are incurred throughout its life cycle. Benefits are realized in the form of reduced operating costs, improved corporate image, staff efficiency, or revenues. To what extent benefits outweigh costs is the function of cost /benefit 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. Identify the costs and benefits pertaining to given project.
2. Categorize the various costs and benefits for analysis.
3. Select a method of evaluation.
4. Interpret the results of the analysis.
5. Take action.

Costs and Benefits Identification

Certain costs and benefits are more easily identifiable than others. For example, direct costs, such as the price of a hard disk, are easily identified form company invoice payments or canceled checks. Direct benefits often relate one-to-one to direct costs, especially savings from reducing costs in the activity in question. Other direct costs and benefits, however, may not be well defined, since they represent estimated costs or benefits that have some uncertainty. An example of such costs is reserve for bad debt. It is a discerned real cost, although its exact amount is not so immediate.

A category of costs or benefits that is not easily discernible is opportunity costs and opportunity benefits. These are the costs or benefits forgone by selecting one alternative over another. They do not show in the organization’s accounts and therefore are not easy to identify.

5.5 Classifications of Costs and Benefits

The next step in cost and benefit determination is to categorize costs and benefits. They may be tangible or intangible, direct or indirect, fixed or variable. Each category is reviewed as follows:

5.5.1 Tangible or Intangible Costs and Benefits

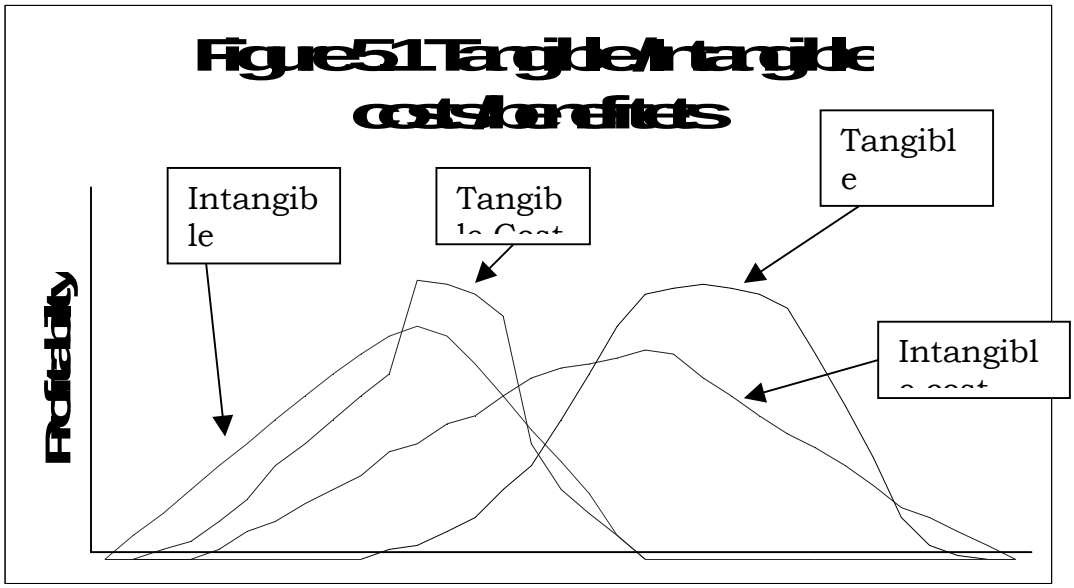
Tangibility refers to the ease with which costs or benefits can be measured. An outlay of cash for a specific item or activity is referred to as a tangible cost. They are usually shown as disbursements on the books. The purchase of hardware or software, personnel training and employee salaries are examples of tangible costs. They are readily identified and measured.

Costs that are known to exist but whose financial value cannot be accurately measured are referred to as intangible costs. For example, employee morale problems caused by a new system or lowered company image is an intangible cost. In some cases, intangible costs may be easy to identify but difficult to measure. For example, the cost of the breakdown of an online system during banking hours will cause the bank to lose deposits and waste human resources. The problem is by how much? In other cases, intangible costs may be difficult even to identify, such as an improvement in customer satisfaction stemming from a real-time order entry system.

Benefits are also classified as tangible or intangible. Like costs, they are often difficult to specify accurately. Tangible benefits, such as completing jobs in fewer hours or producing reports with no errors, are quantifiable. Intangible benefits, such as more satisfied customers or an improved corporate image, are not easily quantified. Both tangible and intangible costs and benefits, however, should be considered in the evaluation process.

Management often tends to deal irrationally with intangibles by ignoring them. According to Oxenfeldt, placing a zero value on intangible benefits is wrong. Axelrod reinforces this point by suggesting that if intangible costs and benefits are ignored, the outcome of the evaluation may be quite different from when they are included. Figure 5.1 is a hypothetical representation of the probability distribution of tangible and intangible costs and benefits. It indicates the degree of uncertainty surrounding the estimation of

costs and benefits. If the project is evaluated on a purely tangible basis, benefits exceed costs by a substantial margin; therefore, such a project is considered cost effective.
On the other hand, if intangible costs and benefits are included, the total tangible and intangible costs exceed the benefits, which make the project an undesirable investment. Furthermore, including all costs increases the spread of the distribution (compared with the tangible – only distribution) with respect to the eventual outcome of the project.



5.5.2 Direct or Indirect Costs and Benefits

From a cost accounting point of view, costs are handled differently depending on whether they are direct or indirect. Direct costs are those with which an exact figure can be directly associated in a project. They are applied directly to the operation. For example, the purchase of a box of diskettes is a direct cost because we can associate the diskettes with the money spent. Direct benefits also can be specifically attributable to a given project. For example, a new system that can handle 25 percent more transactions per day is a direct benefit.

Indirect costs are the results of operations that are not directly associated with a given system or activity. They are often referred to as overhead. A system that reduces overhead realizes a saving. If it increases overhead, it incurs an additional cost. Insurance, maintenance, protection of the computer center, heat, light and air conditioning are all tangible costs, but it is difficult to determine the proportion of each

attributable to a specific activity such as a report. They are overhead and are allocated among users, according to a formula.

Indirect benefits are realized as by-product of another activity or system. For example, our proposed safe deposit billing system that provides profiles showing vacant boxes by size, location and price, will help management decide on how much advertising to do for box rental. Information about vacant boxes becomes an indirect benefit of the billing even though it is difficult to specify its value. Direct and indirect costs and benefits are readily identified for tangible costs and benefits, respectively.

5.5.3 Fixed or Variable- Costs and Benefits

Some costs and benefits are constant, regardless of how well a system is used. Fixed costs are sunk costs. They are constant and do not change. Once encountered, they will not recur. Examples are straight – line depreciation of hardware, and insurance. In contrast, variable costs are incurred on a regular (weekly, monthly) basis. They are usually proportional to work volume and continue as long as the system is in operation. For example, the costs of computer forms vary in proportion to the amount of processing or the length of the reports required.

Fixed benefits are also constant and do not change. An example is a decrease in the number of personnel by 20 percent resulting from the use of a new computer. The benefit of personnel savings may recur every month. Variable benefits, on the other hand, are realized on a regular basis. For example, consider a safe deposit tracking system that saves 20 minutes preparing customer notices compared with the manual system. The amount of time saved varies with the number of notices produced.

5.6 Savings versus Cost Advantages

Savings are realized when there is some kind of cost advantage. A cost advantage reduces or eliminates expenditures. So we can say that a true savings reduces or eliminates various costs being incurred.

There are savings, however, those do not directly reduce existing costs. To illustrate, examine the following case:

A systems analyst designed an online teller system that requires 14 new terminals. No reduction in personnel is immediately planned. Renovation of the bank lobby and the teller cages will be required. The primary benefits are:

1. Savings in teller's time to update account and post transaction.

2. Faster access and retrieval of customer account balances.

3. Available of additional data for tellers when needed.

4. Reduction of transaction processing errors.

5. Higher employee morale.

6. Capability to absorb 34 percent of additional transactions.

This is a case where no money can be realized as a result of the costs incurred for the new installation. There might be potential savings if additional transactions help another department reduce its personnel. Similarly, management might set a value (in terms of savings) on the improved accuracy of teller activity, on quicker customer service, or on the psychological benefits form installing an online teller system. Given the profit motive, savings (or benefits) would ultimately be tied to cost reductions. Management has the final say on how well the benefits can be cost-justified.

5.7 Select Evaluation Method

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:

1. Net benefit analysis.
2. Present value analysis.
3. Net Present value.
4. Payback analysis.
5. Break- even analysis.
6. Cash-flow analysis

1. Net Benefit Analysis:- Net benefit analysis simply involves subtracting total costs from total benefits. It is easy to calculate easy to interpret, and easy to present. The main

drawback is that it does not account for the time value of money and does not discount future cash flow. Period 0 is used to represent the present period. The negative numbers represent cash outlays. The time value of money is extremely important in evaluation processes. What is suggested here is that money has a time value. Today's dollar and tomorrow's dollar are not the same. The time lag accounts for the time value of money.

The time value of money is usually expressed in the form of interest on the funds invested to realize the future value. Assuming compounded interest, the formula is: $F = P (1 + i)^n$

Where

F= Future value of an investment

P= Present value of the investment.

I= Interest rate per compounding period.

N= Number of years.

2. Present Value Analysis:- In developing long-term projects, it is often difficult to compare today's costs with the full value of tomorrow's benefits. As we have seen, the time value of money allows for interest rates, inflation and other factors that alter the value of the investment. Furthermore certain investments offer benefit periods that varies with different projects. Presents value analysis controls for these problems by calculating the costs and benefits of the system in terms of today's value of the investment and then comparing across alternatives.

A critical factor to consider in computing present value is a discount rate equivalent to the forgone amount that the money could earn if it were invested in a different project. It is similar to the opportunity cost of the funds being considered for the project.

Suppose that Rs. 3,000 is to be invested in a microcomputer for our safe deposit tracking system and the average annual benefit is Rs. 1,500 for the four-year life of the system. The investment has to be made today, whereas the benefits are in the future. We compare present values to future values by considering the time value of money to be invested. The amount that we are willing to invest today is determined by the value of the

benefits at the end of a given period (year). The amount is called the present value of the benefit.

To compute the present value, we take the formula for future value ($F = P * (1 + i)^n$ and solve for the present value (P) as follows:

$P = F / (1 + i)^n$

So the present value of Rs. 1,500 invested at 10 percent interest at the end of the fourth year is:

$P= 1,500/(1+0.10)^4 =$ Rs. 1,027.39

That is, if we invest Rs. 1,027.39 today at 10 percent interest, we can expect to have Rs. 1,500 in four years. This calculation can be represented for each year where a benefit is expected.

3. Net Present Value:- The net present value is equal to discounted benefits minus discounted costs. Our Rs. 3,000 microcomputer investment yields a cumulative benefit of Rs. 4,758.51 or a net present gain of Rs.1,758.51. This value is relatively easy to calculate and accounts for the time value of money. The net present value is expressed as a percentage of the investment- in our example:

1,758.51/3,000 = 0.55 percent

4. Payback Analysis:- The payback method is a common measure of the relative time value of a project. It determines the time it takes for the accumulated benefits to equal the initial investment. Obviously the shorter the payback period, the sooner a profit is realized and the more attractive is the investment. The payback method is easy to calculate and allows two or more activities to be ranked. Like the net profit, though, it does not allow for the time value of money.

The payback period may be computed by the following formula:

Overall cost outlay/ Annual cash return = (A*B)+ (C*D)/ (5+2)=

Years + Installation time (G) / Years to recover

5. Break –even Analysis:- Break –even is the point where the cost of the candidate system and that of the current one are equal. Unlike the payback method that compares

costs and benefits of the candidate system, break-even compares the costs of the current and candidate systems. When a candidate system is developed, initial costs usually exceed those of the current system. This is an investment period. When both costs are equal, it is break-even. Beyond that point, the candidate system provides greater benefit (profit) than the old one--a return period.

A break–even chart compares the costs of the current and candidate systems. The attributes are processing cost and processing volume. Straight lines are used to show the model's relationships in terms of the variable, fixed and total costs of the two processing methods and their economic benefits. Intersection indicates the point where the total cost of processing transactions by the current system is equal to the total cost of using the candidate system. Beyond that point is the return period. Before the intersection is the investment period. According to the chart, then, it would be more economical to process manually when volume is below the number of break even point transactions. Higher processing volume favors the candidate system.

6. Cash – Flow Analysis:- Some projects, such as those carried out by computer and word processing services, produce revenues from an investment in computer systems. Cash–flow analysis keeps track of accumulated costs and revenues on a regular basis. The "spread sheet" format also provides break – even and payback information. It is revenue minus expense on a period by period basis.

Drawbacks of the Cash flow analysis are: It ignores time value of money. For a limited period, it does not take into account the profitability of the project. It ignores behavioral implications of the numbers in the financial statement. However the major advantage of the cash flow analysis is that it combines benefits of break even and payback methods.

5.8 Interpret Results of the Analysis and Final Action

When the evaluation of the project is complete, the results have to be interpreted. This entails comparing actual results against a standard or the result of an alternative investment. The interpretation phase as well as the subsequent decision phase is subjective, requiring judgment and intuition. Depending on the level of uncertainty, the analyst may be confronted with a single known value or a range of values. In either case,

simpler measures such as net benefit analysis are easier to calculate and present than other measures, although they do not discount future cash flows. If it can be modified to include the time value of money, the net benefit method would be comparable to the net present value method. More complex measures such as net present value account for the time value of money but are more difficult to evaluate and present.

The decision to adopt an alternative candidate system can be highly subjective, depending on the analyst's or end user's confidence in the estimated costs and benefits and the magnitude of the investment.

In summary, cost/ benefit analysis is a tool for evaluating projects rather than a replacement of the decision-maker. In real-life business situations, whenever a choice among alternatives is considered, cost / benefit analysis is an important tool. Like any tool, however, it has problems:

1. **Valuation problems:-** Intangible costs and benefits are difficult to quantify and tangible costs are generally more pronounced than tangible benefits. In most cases, then, a project must have substantial intangible benefits to be accepted.

2. **Distortion problems:-** There are two ways of distorting the results of cost/benefit analysis. One is the intentional favoritism of an alternative for political reasons. The second is when data are incomplete or missing from the analysis.

3. **Completeness problems:-** Occasionally an alternative is overlooked that compromises the quality of the final choice. Furthermore, the costs related to cost/ benefit analysis may be on the high side or not enough costs may be considered to do a complete analysis. In either case, the reliability of the final choice is in doubt.

5.9 The System Proposal

The final decision following cost/benefit analysis is to select the most cost-effective and beneficial system for the user. At this time, the analyst prepares a feasibility report on the major findings and recommendations. It outlines the options and recommendations. It is presented to management for determining whether a candidate system should be designed. Effective reports follow carefully planned formats that management can

understand and evaluate without having to read the entire document. The content and format of the feasibility report are summarized in Figure 5.2.

Figure 5.2 Feasibility Report- An Outline

A written feasibility report should include the following:

TITLE PAGE Defines the name of the project and who it is for

I. TABLE OF CONTENTS List various parts, features and exhibits, showing page numbers

II. SCOPE Present a brief explanation of the system boundaries

III. STATEMENT OF PROBLEM Describe current system, Describe proposed system, Indicate how proposed system will solve the problem(s)

IV. SUMMARY/ ABSTRACT(optional) Give executive a summary of project, high-lighting benefits

V. COST/BENEFIT STATEMENT List benefits and savings in quantitative terms Present figures of savings versus costs, Summarize cost of new equipment, one – time charges, etc. Quantify net savings and expected returns.

VI. IMPLEMENTATION SCHEDULE Submit implementation plan, Specify human resources requirements, systems and procedures, etc. Include PERT-CPM or Gantt Chart

VII. HARDWARE CONFIGURATION (optional) Lay out computer configuration. Describe terminal network and equipment (CRTs, printers, etc.). List communication equipment (data sets, lines, etc.)

VIII. CREDITS Give credit to those who contributed to the project study.

APPENDIX Include exhibits, correspondence on project, and other miscellaneous documentation.

5.10 Summary

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 deign requirements are identified, and alternative system evaluated. In developing cost estimates for a system, we need to consider several cost elements. Among them are

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

When the evaluation of the project is complete, the results have to be interpreted. In summary, cost/ benefit analysis is a tool for evaluating projects rather than a replacement of the decision-maker. In real-life business situations, whenever a choice among alternatives is considered, cost / benefit analysis is an important tool. The final decision following cost/benefit analysis is to select the most cost-effective and beneficial system for the user.

5.11 Questions

1. Compare the various evaluation methods.
2. How are tangible costs different from direct costs.
3. What are the various cost benefit categories.
4. Why is it necssary to conduct cost/benefit analysis
5. What do you understand by system proposal.

Lesson No: 6 **Lesson Name : System Requirement Specifications &Analysis**

6.0 Objectives :

6.0 Objectives

- What is the importance of system requirement specification
- How the facts are find and what are the methods
- What tools are used in structured analysis’s

- 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

6.1 Introduction

Analysis is the heart of the process. It is the key component of the first two phases of the cycle. In analysis the present system, the analyst collects a great deal of relatively unstructured data through interviews, questionnaires, on–site observations, procedures manuals, and the like. The traditional approach is to organize and convert the data though system flowcharts, which support future developments of the system and simplify communication with the user. But the system flowchart represents a physical rather than a logical system. It makes it difficult to distinguish between what happens and how it happens in the system.

There are other problems with the traditional approach.

1. The system life cycle provides very little quality control to ensure accurate communication from user to analyst. They have no language in common.
2. The analyst is quickly overwhelmed with the business and technical details of the system. Much of the time is spent gathering information. The details are needed and must be available, but the analyst does not have the tools to structure and control the details.
3. Present analytical tools have limitations.
    a. English narrative descriptions of a system are often too vague and make it difficult for the user to grasp how the parts fit together. Furthermore, English is inherently difficult to use where precision is needed.
    b. System and program flowcharts commit to a physical implementation of the system before on has complete understanding of its logical requirements.
4. Problems also relate to system specifications:-

a. System specifications are difficult to maintain or modify. A simple change in the user's requirements necessitates changes in several parts of the document.

b. They describe user requirements inn terms of physical hardware that will implement the system rather than what the user wants the system to do.

c. They are monolithic and redundant; that is, to find out information about a particular part of the system, the user has to search the entire document. Furthermore, the same information is found in numerous locations with no cross-reference.

Because of these drawbacks, the 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 the data flow diagram (DFD). There are other tools as well. The use of several tools in structured analysis, including the following:

1. Data flow diagram (DFD).
2. Data dictionary.
3. Structured English.
4. Decision trees.
5. Decision tables.

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 manner in which a systems investigation is conducted will determine whether the appropriate information is gathered. In turn, having the right information influences the quality of the application that follows. In other words, good system design, whether

developed through the SDLC method, prototyping, or structured methods, begins by documenting the current system and properly diagnosing systems requirements.

6.2 What is Requirements Determination?

Requirements determination involves studying the current business system to find out how it works and where improvements should be made. Systems studies result in an evaluation of how current methods are working and whether adjustments are necessary or possible. These studies consider both manual and computer methods, they are not merely computer studies.

A requirement is a feature that must be included in a new system. It may include a way of capturing or processing data, producing information, controlling a business activity, or supporting management. The determination of requirements thus entails studying the existing system and collecting details about it to find out what these requirements are.

Since systems analysts do not work as managers or employees in user departments (such as marketing, purchasing, manufacturing, or accounting), they do not have the same base of acts and details as the managers and users in those areas. Therefore, an early step in analysts, investigation is to understand the situation. Certain types of requirements are so fundamental as to be common in most all situations. Developing answers to a specific group of questions (to be discussed in this section) will help you understand these basic depending on whether the system is transaction – or decision – oriented and whether the system cuts across several departments. For example, the need to inform the inventory manager of an unusually large order that is forthcoming under scores the importance of linking the sales, purchasing, and warehouse departments.

6.2.1 Activities in requirement Determination

It is helpful to view requirements determination through the three major activities of requirements anticipation, requirements investigation, and requirements specification.

6.2.1.1 Requirements Anticipation

Having had experience in a particular business area or having encountered systems in an environment similar to the one currently under investigation will influence systems analysts study. They may foresee the likelihood of certain problems or features and requirements for a new system. As a result, the features they investigate for the current system, questions they raise, or methods employed may be based on this familiarity.

Requirements anticipation can be a mixed blessing. On the one hand, experience form previous studies can lead to investigation of areas that would otherwise go unnoticed by an inexperienced analyst. Having the background to know what to ask or which aspect to investigate can be a substantial benefit to the organization.

On the other hand, if a bias is introduced or shortcuts are taken in conducting the investigation, requirements anticipation is a problem. We will point out guidelines for structuring an investigation around basic questions to avoid the undesirable consequences of requirements anticipation.

6.2.1.2 Requirements Investigation

This activity is at the heart of systems analysis. Using a variety of tools and skills, analysts study the current system and document its features for further analysis.

Requirements investigation relies on the fact-finding techniques and includes methods for documenting and describing system features.

6.2.1.3 Requirements Specifications

The data produced during the fact-finding investigation are analyzed to determine requirements specifications, the description of features for a new system. This activity has three interrelated parts:

- Analysis of Factual Data

The data collected during the fact – finding study and included in data flow and decision analysis documentation are examined to determine how well the system is performing and whether it will meet the organization's demands.

- Identification of Essential Requirements

  Features that must be included in a new system, ranging from operational details to performance criteria, are specified.

- Selection of Requirements Fulfillment Strategies

The methods that will be used to achieve the stated requirements are selected. These from the basis for system design, which follows requirements specification.

All three activities are important and must be performed correctly.

6.2.2 Basic Requirements

Analysts structure their investigation by seeking answers to these four major questions:

What is the basic business process?

What data are used or produced during that process?

What are the limits imposed by time and the volume of work?

What performance controls are used?

6.2.3 Understand the Process

Begin with the basics. Analysts must raise questions that, when answered, will provide a background of fundamental details about the system and describe it. Asking the following questions will help acquire the necessary understanding.

What is the purpose of this business activity?

What steps are performed?

Where are they performed?

Who performs them?

How long does this take?

How often is it done?

Who uses the resulting information?

Suppose you are reinvestigating an inventory reordering system, something about which you know very little. Where should you begin? Listed below are brief answers to basic questions about the inventory reordering system. These are the kinds of answers you would need to seek for any system you were studying.

What is the purpose of inventory reordering?

To ensure that adequate quantities of stock and materials are on hand and available for use without carrying an excessive and therefore costly quantity.

What steps are performed?

Verifying stock on hand. Determining future requirements and optimum times to place orders. Determining quantities to order.

Where are they performed?

The purchasing department, using information provided by manufacturing, sales, and inventory staff members, as well as by its own records, handles ordering and lead – time. Projection.

Who perform them?

Purchasing manages approve all orders. Stock managers assemble buying instructions and write orders.

How long does this take?

The process may take a few minutes for simple and routine high – prices item or other special circumstance.

How often is it done?

This is a continuous process. Different items are always being ordered.

Who uses the resulting information?

> Information produced as a by – product of this process is used to manage inventory, schedule service and manufacturing monitor purchasing, and pay suppliers, as well as meet unexpected requirements for purchasing and inventory reorder information.

Notice how quickly answers to these questions provide a broad understanding of what inventory reordering is all about and show that the objective of inventory reordering is more than just buying stock. But analysts cannot stop here. There is not yet enough information to fully understand inventory reordering. Instead, the background acquired enables to raise more detailed questions.

6.2.4 Identify data used and information produced

Analysts next need to find out what data are used to perform each activity for example, to reorder inventory, the buyer might require data describing the quantity of an item of hand, expected demand for the item, supplier name, and item cost. To know when to place an order, the buyer would also consider the necessary lead-time (how for in advance the item should be ordered to be on hand when needed).

Most business transactions also produce information that is useful to managers when they evaluate employee, business, and systems performance and that may be useful in another context to both manager and analyst. Inquiring analysts will find out, for example that data about inventory reordering and stocking also provide information about warehouse demands, purchasing practices, sales, and cash flow.

6.2.5 Determine process timing and volume

The frequency of business activities varies greatly. For example, some activities, such as paying taxes, occur only a few times a year, whereas paying employees is a weekly activity. Therefore, analysts should learn how often the activity is repeated. Knowing whether an activity occurs frequently may lead the analyst to raise many

additional and important questions to determine the reason for the frequency and its effect on business activities.

Many times the easiest way to get this information is to identify the reason for the activity: what causes the activity to be performed? Analysts sometimes refer to the direct cause as the trigger function. (it triggers the activity.) Activities can be triggered by customers of an application to open a new bank, charge, or credit account), and by the passage of time (the ending of the day, week, or month). Unless analysts know what triggers an activity, they may misunderstand the reason for the activity and give it more or less importance in the system than it merits.

Some activities, such as completing a purchase requisition, take only a few seconds. Others, such as deciding whether to accept a merger offer, occur infrequently but require a great deal of time when they do take place. Timing alone does not determine the importance of an activity, but it does affect the way analysts evaluate certain steps in carrying out the performance. For example, making a telephone call to obtain stock price information during a merger decision is quite acceptable, since a merge is an infrequent occurrence. But making a telephone call to obtain information every time a purchase requisition is processed is another matter.

The volume of items to be handled may increase the amount of time needed to complete the activity. Savings banks prepare consumer account statements (summaries of deposits, withdrawals, interest accumulations, and balances) only four times a year. Although the frequency of this activity is very low, when the calendar triggers this activity at the end of each quarter, the volume of work is very high, sometimes running into tens of thousands of statements to be prepared. The sheer quantity of item making up an activity can produce special problems for the analyst to study, even though the activity occurs infrequently.

6.2.6 Identity controls

In business situations that are well controlled either by management or process monitoring, determining whether an activity has been performed properly may be no problem. But during the analysis stage, the analysts must examine control methods: are there specific performance standards? Who compares performance against standards? How are mistakes caught? How are error handled? Are the errors excessive? Weak or missing controls area an important discovery in any system investigation. In the vignette the beginning of this chapter, the failure of the two junior systems analyst to give proper attention to weak or mission controls when they studied receiving room activities had serious consequences.

6.2.7 User transaction Requirements

Transaction – level systems captures, process, and store data for a reason. In an order system, for example, sale order form customers are processed so that specified item can be shipped. This simple procedure applies to every order that is received.

Analysis's assigned to work on an order entry system would want to know more about how these transactions are processed. To under – stand these transaction requirements they would undoubtedly ask questions such as the following:

What makes up the transaction being processed?
What initiates the transaction?
Who actually initiates the order? For what purpose?
How often do order occur?
What volume is associated with each?
Are there different conditions that can affect how orders are processed?
What details are needed to process the transaction?
What information is generated? What data is stored?

6.2.8 User decision Requirements

Decision, unlike transaction activities, may not follow a specific procedure. Routines are not as clear – cut, and controls may be very vague. Decisions are made by integrating information in such a way that managers can know what actions to take. Decision systems may focus on the past, the present, or the future. Some may support recurring decisions (such as merchandise pricing, while other are unique and do not recur (such as the merger example used earlier). They may used data that originate inside the firm, such as through transaction processing, or outside, for example form trade associations or commercial sources (such as marketing research firms who sell information to organizations). In some cases, transaction data are processed to provide new information for decision making. For instance, summarized sales transaction data tell managers which products sell and which do not.

Analysts investigating decision support systems should raise the same questions about timing and frequency discussed previously. But other questions should also be posed to determine decision requirements:

1. What information is used to make the Decision?
2. What is the source of the information? Which transaction system produce the data used in the decision process? Which data are required but do not result from processing transactions? Which data originate from sources outside the organization?
3. How would data be processed to produce the necessary information?
4. How should the information be presented?

These questions also point out the relationship between transaction and decision systems. If transaction systems do not capture and store the data needed for decision, important information will be unavailable. Inventory systems capture details about ongoing ordering, receipt, sale, and shipment of items, the data they store are further processed to produce information periodically to analyze sales, determine pricing policy, or decide on marketing plan for product lines.

This means (1) that analysts investigating decision systems must be aware of supporting transaction systems and (2) that effective decision systems require suitable transaction processing procedures to be place first.

6.3 Fact – Finding Techniques

The specific methods analysts use for collecting data about requirements are called fact – finding techniques. These include the interview, questionnaire, record inspections (on – site review) and observation. Analysts usually employ more that one of these techniques to help ensure an accurate and comprehensive investigation.

6.3.1 Interview

Analysts use interviews to collect information from individuals or from groups. The respondents are generally current users of the existing system or potential users of the proposed system. In some instances, the respondents may be managers or employees who provide data for the proposed system or who will be affected by it. Although some analysts prefer the interview to other fact – finding techniques, it is not always the best source of application data. Because of the time required for interviewing, other methods must also be used to gather the information needed to conduct an investigation.

It is important to remember that respondents and analysts converse during an interview – the respondents are not being interrogated. Interviews provide analysts with opportunities for gathering information form respondents who have been chosen for their knowledge of the system under study. This method is frequently the best source of qualitative information (opinions, policies, and subjective descriptions of activities and problems). Other fact finding methods are likely to be more useful for collecting quantitative data (numbers, frequencies, and quantities).

This method of fact – finding can be especially helpful for gathering information from individuals who do not communicate effectively in writing or who may not have the time to complete questionnaires. Interviews allow analysts to discover areas of

misunderstanding, unrealistic expectations, and even indications of resistance to the proposed system.

Interviews can be either structured or unstructured. Unstructured interviews, using a question – and – answer format, are appropriate when analysts want to acquire general information about a system. This format encourages respondents to share their feelings, ideas, and beliefs. Structured interviews use standardized questions in either an open – response or closed – response format. The former allows respondents to answer in their own words; the latter uses a set of prescribed answers. Each approach has advantages and disadvantages.

The success of an interview depends on the skill or the interviewer and on his or her preparation for the interview. Analysts also need to be sensitive to the kinds of difficulties that some respondents create during interviews and know how to deal with potential problems. They need to consider not only the information that is acquired during an interview, but also its significance. It is important to have adequate verification of data through other data collection methods.

6.3.2 Questionnaire

The use of questionnaires allows analysts to collect information about various aspects of a system from a large number of persons. The use of standardized question formats can yield more reliable data than other fact – finding techniques, and the wide distribution ensures greater anonymity for respondents, which can lead to more honest responses. However, this method does not allow analysts to observe the expressions or reactions or respondents. In addition, response may be limited, since completing questionnaires may not have high priority among the respondents.

Analysts often use open – ended questionnaires to learn about feeling, opinions, and general experiences or to explore a process or problem. Closed questionnaires control the frame of reference by presenting respondents with specific responses form which to choose. This format is appropriate for electing factual information.

The high cost of developing and distributing questionnaires demands that analysts carefully consider the objective of the questionnaire and determine what structure will be most useful to the study and most easily understood by the respondents. Questionnaires should also be tested and, if necessary, modified before being printed and distributed.

As with interviewees, recipients, of questionnaires would be selected for the information they can provide. The analysts should ensure that the respondents, background and experiences qualify them to answer the questions.

6.3.3 Record Review

Many kinds of records and reports can provide analysts with valuable information about organizations and operations. In record reviews, analysts examine information that has been recorded about the system and user. Record inspection can be performed at the beginning of the study, as an introduction, or later in the study, as a basis for comparing, actual operations with the records indicate should be happening.

Records include written policy manuals, regulations and standard operating procedures used by most organizations and a guide for managers and employees. They do not show what activities are actually occurring, where the decision – making power lies, or how tasks are performed. However, they can help analysts understand the system by familiarizing them with what operations must be supported and with formal relations within the organization.

6.3.4 Observation

Observation allows analysts to gain information they cannot obtain by any other fact – finding method. Through observation, analysts can obtain firsthand information about how activities are carried out. This method is most useful when analysts need to actually observe how documents are handled, how processes are carried out, observers know what to look for and how to assess the significance of what they observe.

6.4 WHAT IS 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:

1. Use graphics wherever possible to help communicate better with the user.
2. Differentiate between logical and physical systems.
3. 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. The system development life cycle with structured analysis. The primary steps are:

Process 2.1: Study affected user areas, resulting in a physical DFD. The logical equivalent of the present system results in a logical DFD.

Process 2.2: Remove the physical checkpoints and replace them with a logical equivalent, resulting in the logical DFD.

Process 2.3: Model new logical system. So far no consideration is given to modifying methods called for in the feasibility report. This step incorporates the changes the begins to describe the candidate system. It is essentially a paper model system to be installed.

Process 2.4: Establish man/ machine interface. This process modifies the logical DFD for the candidate system and considers the hardware needed to implement the system. The combination results in the physical DFD of the candidate system.

Process 2.5 and 2.6: Quantify costs and benefits and select hardware. The purpose of this step is to cost- justify the system, leading to the selection of hardware for the candidate system. All that is left after this step is writing the structured specification.

The structured specification consists of the DFDs that show the major decomposition of system functions and their interfaces, the data dictionary documenting all interface flows and data stores on the DFDs and documentation of the intervals of DFDs in a rigorous manner through structured English, decision trees, and decision tables.

In summary, structured analysis has the following attributes:

1. It is graphic. The DFD for example, presents a picture of what is being specified and is a conceptually easy-to – understand presentation of the application.
2. The process is partitioned so that we have a clear picture of the progression from general to specific in the system flow.
3. It is logical rather than physical. The elements of system do not depend on vendor or hardware. They specify in a precise, concise, and highly readable manner the working of the system and how it hangs together.
4. It calls for a rigorous study of the user area, a commitment that is often taken lightly in the traditional approach to systems analysis.
5. Certain tasks that are normally carried out late in the system development life cycle are moved to the analysis phase. For example, user procedures are documented during rather than later in implementation.

6.4.1 The Data Flow Diagram (DFD)

The first step is to draw a 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.

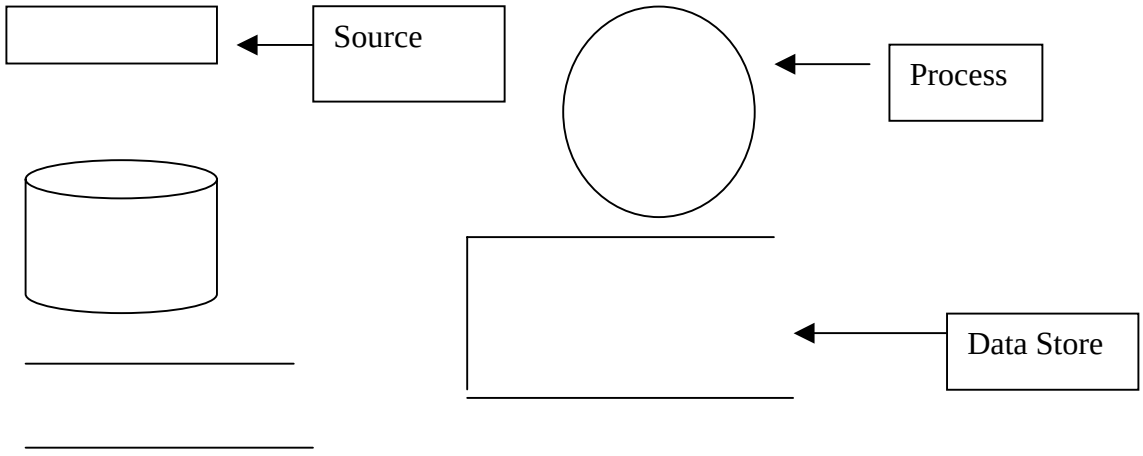
A DFD also known as a “bubble chart,” has the purpose of clarifying system requirements and identifying major transformations that will be come programs in system design. So it is the starting point of the design phase that functionally decompose the requirements specifications down to the lowest level of detail. A DFD consists of a series of bubbles joined by lines. The bubbles represent data transformations and the lines represent data flows in the system. The system takes orders from the customer (bookstore, library, etc.), checks them against an index (file) listing the books available, verifies customer credit through a credit information file, and authorizes, shipment with an invoice.

6.4.1.1 DFD Symbols

In the DFD, there are four symbols

A square defines a source (originator) or destination of system data.

1. An arrow identifies data flow- data in motion. It is a pipeline through which information flows.
2. A circle or a “bubble” (some people use an oval bubble) represents a process that transforms incoming data flow(s) into outgoing data flow(s).
3. An open rectangle is a data store- data at rest, or a temporary repository of data.



Note that a DFD describes what data flow (logical) rather than how they are processed so it does not depend on hardware, software, data structure, or the organization. The key question that we are trying to answer is: What major transformations must occur for input to be correctly transformed into output?

Elaborate on the logical functions of the system. first, incoming orders are checked for correct book titles, author's names and other information and their batched with other book orders from the same bookstore to determine how many copies can be shipped through the warehouse. Also, the credit status of each bookstore is checked before shipment is authorized. Each shipment has a shipping notice detailing the kind and number of books shipped. This is compared to the original order received (by mail or phone) to ascertain its accuracy. The details of the order are normally available in a special file or a data store, called "bookstore orders."

Following order verification and credit check, a clerk batches the order by assembling all the book titles ordered by the bookstore. The batched order is sent to the warehouse with authorization to pack and ship the books to the customer.

Further expansion of the DFD focuses on the steps taken in billing the bookstore. As you can tell by now, each process summarizes a lot of information and can be exploded into several lower-level detailed DFDs. This is often necessary to make sure that a complete documentation of the data flow is available for further reference.

6.4.1.2 Constructing a DFD

Several rules of thumb are used in drawing DFDs.

1. Process should be named and numbered for easy reference. Each name should be representative of the process.
2. 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.

3. When a process is exploded into lower- level details, they are numbered.
4. 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.

How detailed should a DFD be? As mentioned earlier, the DFD is designed to aid communication. If it contains dozens of processes and data stores, it gets too unwieldy. The rule of thumb is to explode the DFD to a functional level, so that the next sublevel does not exceeded 10 processes. Beyond that, it is best to take each function separately and expand it to show the explosion of the single process. If a user wants to know what happens within a given process, then the detailed explosion of that process may be shown.

A DFD typically shows the minimum contents of data stores. Each data store should contain all the data elements that flow in and out. Questionnaires can be used to provide information for a first cut. All discrepancies, missing interfaces, redundancies, and the like are then accounted for- often through interviews.

The DFD methodology is quite effective, especially when the required design is unclear and the user and the analyst need a notational language for communication. The DFD is easy to understand after a brief orientation.

The main problem however is the large number of iterations that often are required to arrive at the most accurate and complete solution.

6.4.2 Data Dictionary

In our data flow diagrams, we give names to data flows, processes and data stores. Although the names are descriptive of the data, they do not give details. So following the DFD our interest is to build some structured pace to keep details of the contents of data flows, processes and data store. A data dictionary is a structured repository of data. It is a set of rigorous definitions of all DFD data elements and data structure.

A data dictionary has many advantages. The most obvious is documentation: it is a valuable reference in any organization. Another advantage is improving analyst/ user

communication by establishing consistent definitions of various elements, terms and procedures. During implementation, it serves as a common base against which programmers who are working on the system compare their data descriptions. Also control information maintained for each data element is cross- referenced in the data dictionary. For example, programs that use a given data element are cross- referenced in a data dictionary, which makes it easy to identify them and make any necessary changes. Finally a data dictionary is an important step in building a database. Most data base management systems have a data dictionary as a standard feature.

Data have been described in different ways. For example, in tape and disk processing, IBM called a file data set. In data base technology, the term file took on a different meaning IBM's information Management

FIGURE Project Data Element Form – A Sample

PROJECT DATA ELEMENT SHEET

PROJECT NAME _____ DATE_____

DATA ELEMENT DESCRIPTION	DATA ELEMENT ABBREVATION	ELEMENT PICTURE	ELEMENT LOCATION	ELEMENT SOURCE

System's (IMS) manual defines data as fields divided into segments, which, in turn, are combined into databases. The Conference on Data System Languages (CODASYL) defines data as data items combined into aggregates, which, in turn are

combined into records. A group of related records is referred to as a set. If we choose words that represent the general thinking of common vocabulary. There are three classes of items to be defined:

1. Data element: The smallest unit of data that provides for no further decomposition. For example, " data" consists of day, months and year. They hand together for all practical purposes.
2. Data structure: a group of data elements handled as a unit. For example, " phone" is a data structure consisting of four data elements: Area code- exchange – number –extension- for example, 804-924-3423-236. "BOOK DETAILS" is a data structure consisting of the data elements author name, title, ISBN (International Standard Book Number), LOCN (Library of Congress Number), publisher's name and quantity.
3. Data flows and data stores. As defined earlier, data flows are data structures in motion, whereas data stores are data structures at rest. A data store is a location where data structures are temporarily located.

6.4.2.1 Describing Data Elements

The description of a data element should include the name, description and an alias (synonym). For example:

AUTHOR NAME –first	WHISKEY – name
- middle	- distiller
- last	- vintage
- alias	

The description should be a summary of the data element. It may include an example. We may also want to include whether or not the data elements(s) has:

1. A different name. For example a PURCHASE ORDER may exist as PUR.ORDER, PUCHASE ORD., or P.O. We want to record all these in the data dictionary and include them under the PUCHASE ORDER definition and separately with entires of their own. One example is "P.O. alias of (or see also)

PUCHASE ORDER." Then we look up PUCHASE ORDER to find the details. It is an index.

2. Usage characteristics, such as a range of values or the frequency of use or both. A value is a code that represents a meaning. Here we have two types of data elements:

    a. Those that take a value within a range: for example, a payroll check amount between $ 1and $10,000 is called continuous value.

    b. Those that have a specific value: for example. Departments in a firm may be coded 100 (accounting), 110 (personnel), etc. In a data dictionary, it is described as follows:

    100 means "Accounting Department"

    101 means " Accounts Receivable Section"

    102 means " Accounts Payable Section"

    108 means " General Ledger Section"

3. Control information such as the source, date of origin, users, or access authorizations.

4. Physical location in terms of a record, file or data base.

6.4.2.2 Describing Data Structures

We describe any data structure by specifying the name of each data structure and the elements it represents, provided they are defined else- where in the data dictionary. Some elements are mandatory, whereas others are optional. To illustrate, let us take "BOOK- DETAILS". The data elements of this data structure are as follows:

6.4.2.3 Describing Data Flows and Data Stores

The contents of a data flow may be described by the name (s) of the data structures(s) that passes along it. In our earlier example, BOOK-DETAILS express the content of the data flow that lead to process 4. Additionally, we may specify the source of the date flow, the destination, and the volume (if any). Using the BOOK- ORDER example, data flows may be described as follows:

Data Flow	Comments
BOOK-DETAILS	From Newcomb Hall Bookstore (source)
AUTHOR –NAME	
TITLE OF BOOK	
EDITION	Recent edition required
QUANTITY	Minimum 40 copies

A data store is described by the data structures found in it and the data flows that feed it or are extracted from it. For example, the date store BOOK STORE- ORDER is described by the following contents:

	Comments
ORDER	Data flow/data structure feeding date store
ORDER-NUMBER	
CUSTOMER –DETAILS	Content of data store
BOOK- DETAIL	Data flow/data structure extracted from data store

6.4.2.4 Describing Processes.

This step is the logical description. We want to specify the inputs and outputs for the process and summarize the logic of the system. In constructing a data dictionary, the analyst should consider the following points:

1. Each unique data flow in the DFD must have one data dictionary entry. There is also a data dictionary entry for each data store and process.
2. Definitions must be readily accessible by name.
3. There should be no redundancy or unnecessary definitions in the data definition. It must also be simple to make updates.
4. The procedure for writing definitions should be straightforward but specific. There should be only one way of defining words.

In summary a data dictionary is an integral component of the structured specification. Without a data dictionary, the DFD lacks rigor, and without the DFD, the data dictionary is of no use. Therefore, the correlation between the two is important.

6.4.3 DECISION TREES

As you know well, people often have different ways of saying the same thing. For example, the discount conditions discussed in the last example can also be stated in the following ways:

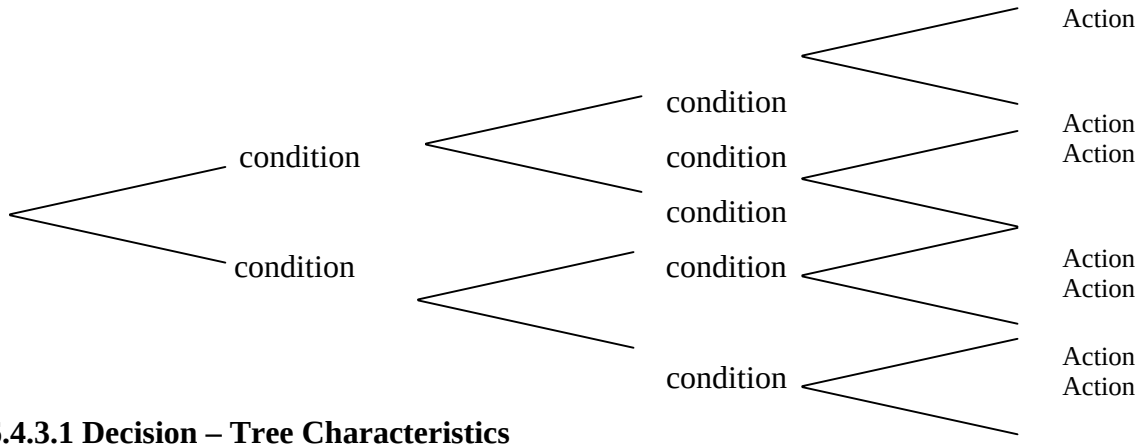
1. Greater than $10,000, grater than or equal $ 5,000 but less than or equal to $ 10,000, and below $5,000
2. Not less than $10,000, not more than $ 10,000 but at least $ 5,000, and not $5,000 or more

Having different ways of saying the same thing can create difficulties in communication during systems studies (analyst and manager may misunderstand each other's comments or forget to discuss all the details). Thus, analysts seek to prevent misunderstandings. They also need to organize information collected about decision making.

Decision trees are one of the methods for describing decisions, while avoiding difficulties in communication.

CONDITION	ACTION
Order is signed	Begin order verification process.
Order is unsigned	Begin merchandise acceptance processing.

CONDITION	ACTION
Size of order : Over $ 10,000	Take 3 % discount form invoice total.
$5,000 to $10,000	Take 2 % discount form invoice
Less than $5,000	Pay full invoice amount.

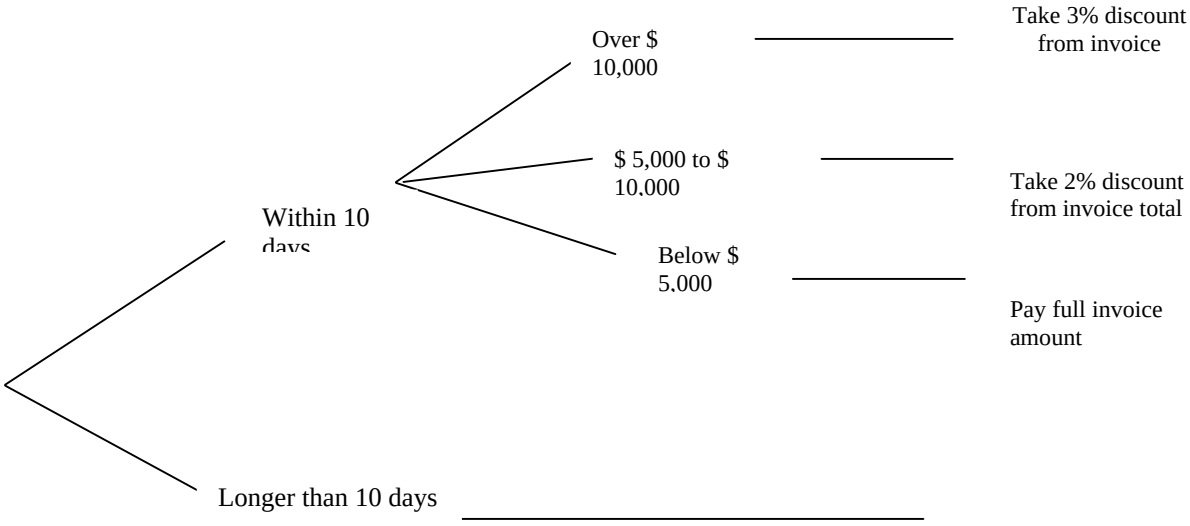


6.4.3.1 Decision – Tree Characteristics

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.



6.4.3.2 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 nonquantitative variables.

It is possible, for instance, to show what discount action to take, depending on the number of dollar spent by customers. When an organization opens accounts with dealers and suppliers, it formalizes an agreements for taking discounts from the full invoice price. Two conditions are specified in this agreement: first, the invoice must always be paid within ten days of its receipt, and, second, the size of the discount will depend on the value of the invoice. It is agreed that under some conditions the organization can take the action of deducting a 3 percent discount; under other conditions, a 2 percent discount; and under all other conditions, no discount is allowed.

Decision trees also force analysts to consider the sequence of decisions. Consider the sequence of decision in the example. you can quickly determine that one condition is amount of the invoice-does not matter unless another condition is met and the invoice is paid within the time established by the supplier – ten days. The other conditions are relevant only if that condition is true. Therefore, the decision tree identifies the time condition first and shows two values (within ten days and longer than ten days). The discount condition is described next, but only for the branch of the tree for WITHIN TEN DAYS. The LONGER THAN TEN DAYS branch has no other relevant conditions and so show the resulting action (unconditionally). This tree shows that the action PAY FULL INVOICE AMOUNT applies under two different conditions. It also shows implicitly that there is no reason to pay invoice of less than $5,000 within ten days, since there is no discount available for these amounts.

The decision tree shows the nonquantitative conditions for processing accounts payable: signed invoice, authorized purchase, and correct pricing. Depending on the set of conditions that exist, one of two actions can be taken: payment can be authorized or the submitted invoice can be rejected. Notice how clearly each alternative is shown in the decision tree. Sometimes in more complex business situations, the specific action most appropriate under several conditions is not readily apparent without formal analysis of this nature.

Analysts find that in accounts payable processing, it is necessary to determine whether a purchase order is available and valid and whether the invoice is processed properly before it is actually paid. In turn, they must learn of the conditions for proper invoice processing. Full development and examination of the decision tree also shows clearly that there are only two ways an invoice can be authorized for payment but many conditions under which payment can be rejected.

The sequence of decision is easy to see in this example. The condition of having a valid purchase order does not matter unless the invoice has been signed. The signature is

important, since the condition of having a signed invoice must be met before processing can continue. Then analysts can consider the authorization condition.

6.4.4 Identifying Data Requirements

We have already pointed out the use of decision trees to formally highlight the sequential nature of many business decision, and we have shown that decision trees are effective when describing business problems of more than one dimension or condition. However, they also identify critical data requirements surrounding the decision process; that is, they indicate the sets of data the manager requires to formulate decision or select action. The explicit data in the payment example are payment data, amount of invoice, and discount allowance percentage. There are other important data elements such as invoice details (number, supplier name and address), new invoice amount payable, and adjustments to "discount taken" that are indict (not directly expressed in the decision tree). The analyst must identify and list all the data used in the decision process, even must identify and list all the data used in the decision process, even though the decision tree does not show all the individual data items.

It decision trees are assembled after the completion of data flow analysis (which tracks the flow of data through busbies processes), critical data may already be defined in the data dictionary (which describes system data and where they are used). If decision trees are identify each data element needed to make a decision trees are identify each data element needed to make a decision. The data dictionary format, is useful for listing and describing data elements as they are identified and understood.

The date requirements discussed for decision trees also apply to the other decision – analysis methods that will be discussed. Analysis's need to describe and define all data used in decision making, so that the system can be designed to produce data properly.

6.4.4.1 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.

6.4.5 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.

FIGURE Structured English- Using Data Dictionary Values.

COMPUTE-DISCOUNT

Add up the number of copies per book title

IF order is from bookstore

And – IF ORDER –SIZE is SMALL

THEN: Discount is 25%

ELSE (ORDER –SIZE is MINIMUM)

So: no discount is allowed

ELSE (order is from libraries or individual customers)

So-IF ORDER –SIZE is LARGE

Discount is 15%

ELSE IF OREDR SIZE is EMDIUM

Discount is 10%

ELSE IF ORDER-SIZE is SMALL

Discount is 5%

ELSE (ORDER –SIZE is MINIMUM)

So: no discount is allowed

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 .

FIGURE Decision Table- Discount Policy

Condition Stub Condition Entry

---

1 2 3 4 5 6

Customerisbookstore? Order_size6 co	Y Y	Y N	N N Y Y	N N Y N Y	N N Y N N Y	N N Y N N N

Allow 25% discount Allow 15% discou	X	X	X	X	X	X

Action Stub Action Entry

FIGURE Elements and Definitions in a Decision Table

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 conditon

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 Form the answers to the conditions in the Condition entry quadrant

The answers are represented by a Y to signify yes, an N to signify no, or a blank to show that the condition involved has not been tested. In the action entry quadrant an X(or a check mark will do) indicates the response to the answer(s) entered in the condition entry quadrant. Furthermore, each column represents a decision or a rule. For example, rule 1 states:

IF customer is a bookstore and order size is 6 copies or more.

THEN allow 25% discount

So, according to the decision table, we have six decisions and therefore six rules. A look at the table provides each decision (answer) immediately the following rules should be followed in constructing decision tables:

1. A decision should be given a name, shown in the top left of the table.
2. The logic of the decision table is independent of the sequence in which the condition rules are written, but the action takes place in the order in which the events occur.
3. Standardized language must be used consistently.
4. Duplication of terms or meanings should be eliminated, where possible.

6.5 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:

1. 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.[7] The user often finds it confusing initially.

2. 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.
3. Structured English is best used when the problem requires sequences of actions with decisions.
4. Decision trees are sued to verify logic and in problems that involve a few complex decisions resulting in limited number of actions.
5. 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.

6.6 Summary

Analysis is the heart of the process. It is the key component of the first two phases of the cycle. In analysis the present system, the analyst collects a great deal of relatively unstructured data through interviews, questionnaires, on–site observations, procedures manuals, and the like. Requirements determination involves studying the current business

system to find out how it works and where improvements should be made. Systems studies result in an evaluation of how current methods are working and whether adjustments are necessary or possible. These studies consider both manual and computer methods, they are not merely computer studies.

The specific methods analysts use for collecting data about requirements are called fact – finding techniques. These include the interview, questionnaire, record inspections (on – site review) and observation. Analysts usually employ more that one of these techniques to help ensure an accurate and comprehensive investigation. 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 first step is to draw a 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 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. It is also a method of showing the relationship of each condition and its permissible actions. 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. The pros and cons of the tools are;

- 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.

6. 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.
7. Structured English is best used when the problem requires sequences of actions with decisions.
8. 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.

6.7 Questions

1. What type of information is best obtained through interview.
2. What is systems requirement.
3. What advantages do decision trees present. from analysts.
4. Discuss the pros and cons of the various tools of doing analysis.
5. What is structured analysis.

Lesson No: 7 **Lesson Name : Detailed Design**

7.0 Objectives:

- What is the process of system deign
- What are the alternative methods of file organizations

- Objectives of data base
- Types of data structure
- The difference between schema and subschema
- How to normalise files

7.1 Introduction:

The design translates the system requirements into ways of operationalizing them. The design is a solution, a " how to " approach, compared to analysis, a "what is" orientation. The design phase focuses on the detailed implementation of the system recommended in the feasibility study. Emphasis is on translating performance specifications into design specifications. The design phase is a transition from a user-oriented document to a programmer-oriented document.

7.2 Modularization:

One way to plan a new system is to focus on each functional subsystem as a separate entity or application area. Using such an approach, each application area is treated as if it were totally independent. There is minimal sharing of information and systems processes between areas. For example, it two major systems efforts were being simultaneously undertaken. One in the order department & the other in the accounts section. The orders affect the amount of receivables, amount of receivables affect customer's credit, validate the order and much more. From an applications point of view, the order processing subsystem should be designed to meet accounts receivable functional requirements and vice versa. However, there would be no need to review each application area for common internal processes. Both the systems would be performing certain same steps in each of their systems individually.

The modular systems approach divides each application area into a number of smaller units called modules. These modules may apply to a particular application, or they may be common to two or more application areas. Modules may be used only once, or they may be used several times during the processing of an application. Breaking up of a problem into smaller manageable parts is certainly beneficial.

The advantages of modularization are:-

1. It can speed up the systems process in general & the computer programming function in particular.
2. It eliminates unnecessary duplications.
3. It can result in higher quality because of the concentrated effort devoted to the development of common modules.
4. It provides better control over the total system project, since work can be segmented and assigned in smaller, more controllable units.
5. It more efficiently maintains the system as a correction at one place rectifies the entire problem.
6. It allows flexibility as additional features may be added later.
7. Small parts of the system can be tested separately.

Certainly these factors present a strong argument in favor of the modularization. However there are certain limitations to it as follows:-

1. Numerous unique application requirements which must be incorporated in common modules. If a single module is to accommodate all situations, it will become very large & complex.
2. Many systems, changes for particular application areas. Many times a high rate of change means a high rate of potential error. When these changes and errors affect common modules, the negative consequences can be widespread.

Modular systems design is best viewed as one aspect of a broader planning issue, but it is not a required step in the design process. The analyst, based upon the in-depth understanding of problem, specifies the level of modularization.

Files: The data is stored in files according to user requirements. Some records are processed daily whereas other are updated at random. Depending upon the way the data will be used, the file is organized.

7.2.1 Basic file 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.

7.3 File Design

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, indexed – sequential, inverted list and direct access. Each method is explained.

7.3.1 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.

7.3.2 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.

In an airline reservation file, the index area might contain pointers to the Chicago and Delhi flights. The Chicago flight points to the Chicago flight information stored in the prime area. The Delhi flight points to the Delhi flight information in the prime area. Lack of space to store the Brisbane flight in sequential order make it necessary to load it in the overflow area. The overflow pointer places it logically in sequential order in the prime area. The same arrangement applies to the other flights.

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.

7.3.3 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 necessarily stored in particular sequence. They are placed in the data storage area, but indexes are updated for the record keys and location.

Data for our flight reservation system has a separate Index area and a data location area. The index area may contain flight number and a pointer to the record present in the data location area. The data location area may have record numbers along with all the details of the flight such as the flight number, flight description, and flight departure time. These are all defined as keys, and a separate index is maintained for each. In the data location area, flight information is in no particular sequence. Assume that a passenger needs information about the Delhi flight. The agent requests the record with the flight description “Delhi flight”. The Data Base Management System (DBMS) then reads the single-level index sequentially until it finds the key value for the Delhi flight. This value may have two records associated with it. The DBMS essentially tells the agent the departing time of the flight. Looking at inverted-list organization differently, suppose the passenger requests information’s on a Delhi flight that departs at 8:15. The DBMS first searches the flight description index for the value of the “Delhi flight.” It finds both the records. Next it searches the flight departure index for these values. It finds that one of them departs at 10:10, but the other departs at 8:15. The later record in the data location area is displayed for follow-up.

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.

7.3.4 Direct- Access organization

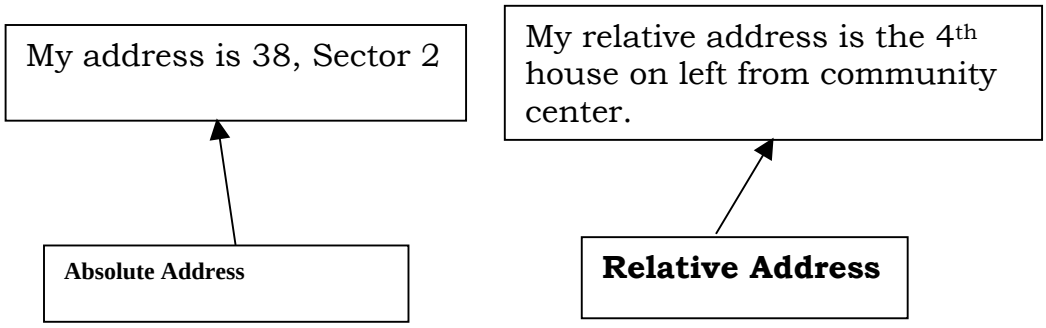
In direct – access file organization, records are placed randomly throughout the file. Records need not be in sequence because they are updated directly and rewritten

back in the same location. New records are added at the end of the file or inserted in specific locations based on software commands.

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 (see Figure 7.1). 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.

FIGURE 7.1 Absolute and Relative Addressing – An Example



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.

FIGURE 7.2 File Organization Methods – A Summary

Method	Advantages	Disadvantages
Sequential	Simple to design. Easy to program. Variable length & blocked records available Best use of software space	Records cannot be added to middle of file.
Indexed sequential	Records can be inserted or Updated in middle of file. Processing may be carried out Sequentially or randomly	Unique keys required Processing occasionally slow. Periodic reorganization of file required.
Inverted list	Used in applications request-ing specific data on multiple keys.	

Random	Records can be inserted or middle of file required for Better control over record Allocation.	Calculating address updated in processing. Variable- length records nearly impossible to process.

7.4 Data Base Design

A decade ago, database was unique to large corporations with mainframes. Today it is recognized as standard of MIS and is available for virtually every size of computer. Before the data base concept became operational, users had programs the handled their own data independent of other users. It was a conventional file environment with no data integration or sharing of common data across applications. In a database environment, common data are available and used by several users. Instead of each program (or user) managing its own data, data across applications are shared by authorized users with the data base software managing the data as an entity. A program now requests data through the data base management system (DBMS), which determines data sharing.

7.4.1 Objectives of Data Base

The general theme behind a database is to handle information as an integrated whole. There is none of the artificiality that is normally embedded in separate file or applications. 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:

1. **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.

2. **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.

3. **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.

4. **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.

5. **Accuracy and integrity: -** The accuracy of a database ensures that data quality and content remain constant. Integrity controls detect data inaccuracies where they occur.

6. **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.

7. **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.

8. **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.

In a data base environment, the DBMS is the software that provides the interface between the data file on disk and the program that requests processing. The DBMS stores and manages data. The procedure is as follows:

1. The user requests a sales report through the application program. The application program uses a data manipulation language (DML) to tell the DBMS what is required.

2. The DBMS refers to the data model, which describes the view in a language called the data definition language (DDL). The DBMS uses DDL to determine how data must be structured to produce the user's view.

3. The DBMS requests the input/output control system (IOCS) to retrieve the information from physical storage as specified by the application program. The output is the sales report.

To summarize,

1. DML manipulates data; it specifies what is required.
2. DDL describes how data are structured.
3. DBMS manages data according to DML requests and DDL descriptions.

DBMS performs several important functions:

1. Storing, retrieving, and updating data.

2. Creating program and data independence. Either one can be altered independently of the other.

3. Enforcing procedures for data integrity. Data are immune from deliberate alteration because the programmer has no direct method of altering physical databases.

4. Reducing data redundancy. Data are stored and maintained only once.

5. Providing security facilities for defining users and enforcing authorization. Access is limited to authorized users by passwords or similar schemes.

6. Reducing physical storage requirements by separating the logical and physical aspects of the database.

7.4.2 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.

7.4.3 Schemas and Subschemas

The schema is the view the helps the DBMS decide what data in storage it should act upon as requested by the application program. An example of a schema is the arrival and departure display at an airport. Scheduled flights and flight numbers (schema) remain the same, but the actual departure and arrival times may vary. The user's view might be a particular flight arriving or departing at a scheduled time. How the flight actually takes off or lands is of little concern to the user. The latter view is of subschema. It is a programmer's (pilot's) view. Many subschemas can be derived from one schema, just as different pilots visualize different views of a landing approach, although all (it is hoped) arrive at the scheduled time indicating on the CRT screen display (schema).

Different application programmers visualize different subschemas. The software provides the relationships among the schema, subschema and physical structure.

7.5 Data Structure

Data are structured according to the data model. In any sales example, sales items are linked to the salesperson that sold them. The salesperson is called an entity and the item sold is also an entity. An entity is a conceptual representation of an object. Relationships between entities make up a data structure. A data model represents a data structure that is described to the DBMS in DDL.

7.6 Types of Relationships

Three types of relationships exist among entities: one-to-one, one-to-many, and many-to-many relationships.

A one-to-one (1:1) relationship is an association between two entities. For example, in our culture, a husband is allowed on wife (at a time) and vice versa, and an employee has one social security number.

A one–to-many (1:M) relationships describes an entity that may have two or more entities related to it. For example, a father may have many children, and an employee may have many skills.

A many-to-many (M:M) relationship describes entities that may have many relationship in both directions. For example, children may have many toys, and students may have many courses.

7.7 Types of Data Structure

Data structuring determines whether the system can create 1:1, 1:M, or M:M relationships among entities. Although all DBMSs have a common approach to data management, they differ in the way they structure data. There are three types of data structure: hierarchical, network and relational.

7.7.1 Hierarchical Structuring

Hierarchical (also called tree) structuring specifies that an entity can have no more than one owning entity, that is we can establish a 1:1 or 1:M relationship. The owning entity is called the parent; the owned entity, the child. A parent with no owners is called the root. There is only one root in a hierarchical model.

For example, a parent can have many children (1:M), whereas a child can have only one parent. Elements at the ends of the branches with no children are called leaves. Trees are normally drawn upside down, with the root at the top and the leaves at the bottom.

The hierarchical model is easy to design and understand. Some applications, however, do not conform to such a scheme, such as for a firm dealing in sale of spare parts being manufactured by more than one company. Thus, we would have a non-hierarchical structure, which complicates programming or the DBMS description. The problem is resolved by using a network structure.

7.7.2 Network Structuring

A network structure allows 1:1, 1:M, or M:M relationships among entities. For example, an auto parts shop may have dealings with more than one manufacturer (parent). Spare parts may come from two companies, so they are owned by both entities-a structure that can best be supported by a network. Now consider the manufacturer and the auto parts shops it deals with. If the manufacturer sold spare parts to only one shop (say, a new car dealer), then there is a 1:1 relationship. If it supplied to many other dealers, then there is a 1:M relationship. The 1:1 and 1:M relationships can be supplied by a hierarchy. When auto parts dealers are supplied by many manufacturers, however, there is an M:M relationship, which is a network structure.

A network structure reflects the real world, although a program structure can become complex. The solution is to separate the network into several hierarchies with duplicates. This simplifies the relationship to no more complex than 1:M. A hierarchy, then becomes a subview of the network structure.

7.7.3 Relational Structuring

In relational structuring, all data and relationships are represented in a flat, two-dimensional table called a relation. A relation is equivalent to a file, where each line represents a record. For example, a relation that describes the entity EMPLOYEE by social security number, name and years with the firm. All the entries in each field are of the same kind. Furthermore, each field has a unique name. Finally, no two rows in the table are identical. A row is referred to as a tuple.

A relational DBMS has several features:

1. It allows the user to update (add, modify, or delete) the table’s contents. Any position can be changed.

2. It provides inquiry capabilities against a label. Using our example, an inquiry might be: “How many years has Boynton been with the firm?” The response is “6.”

3. Two or more tables can be merged to form one relation. Unlike hierarchical or network structuring where all relationships are predefined, a relational DBMS develops new relations on user commands.

4. A relational structure is simpler to construct than a hierarchical or a network structure. It may be inefficient, though, since a relational DBMS responds to queries by an exhaustive review of the relations involved.

7.8 Entities and Attributes

An entity is something of interest to the user about which to collect or store data. It is also called a data aggregate because it represents a number of data elements. In our sales status system, the “sales” entity contains data elements such as the salesperson’s number, name and date of employment, and the sales period covered by the report. The “item” entity has data elements such as item number, item description, and the sale price of each item.

Data entities are explained by the use of several terms: attribute, value key and instance of an entity. For example, a salesperson (entity) is described by attributes such a number, name, sex, age and height. So attributes describe an entity. They are physically stored in fields or data elements.

Each attribute takes on a unique value. For example, “11306801” is a unique value of the attribute “ salesperson number.” An attribute, then, takes on a value for a specific occurrence (or instance) of an entity.

A key is a unique identifier of the entity. In our example, the key 11306801 is a unique identifier of Jim Arnold. Sex, age and height are not identifiers, since they are not unique. They are non-key identifiers.

7.9 Normalization

Data structuring is refined through a process called normalization. Data are grouped in the simplest way possible so that later changes can be made with a minimum of impact on the data structure. When too many attributes are grouped together to form entities, some attributes are found to be entities themselves. Further normalization of

these entities into attributes linked by common data elements to form relationships improves the effectiveness of the DBMS.

7.10 Summary

The design translates the system requirements into ways of operationalizing them. The design is a solution, a " how to " approach, compared to analysis, a "what is" orientation. The design phase focuses on the detailed implementation of the system recommended in the feasibility study.

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 organizatiion uses an index for each key type. Records are not necessarily in a particular sequences.
- Direct access organization has records placed randomly through out the file. Records are updated directly and independently of the other records.

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. Data base design minimizes the artificially embedded in using separate files. The primary objectives are fast response time to inquiries, more information at low cost, control of redundancy, clarity and ease of use, data and program independence, accuracy and integrity of the system, fast recovery, privacy and security of information, and availability of powerful end user languages.

The heart of the data base is DBMS. It manages and controls the data base file and handle request from the application program. A data structure defines relationships among ententes. There are three types of relationships one-to-one, one-to-many and

many-to-many. Although all DBMS have a common approach to data management, they are differ in the way they structure data. The three types of data structure are hierarchical , network and relational.

Three are four views of data. The first three views are logical: users view application program (called Subschema), and overall logical view (called Schema). The data structure can be refined through a normalization process that group the data in the simplest way possible so that later changes can be made with ease. Normalization is designed to simplify relationships and establish logical links between files without losing information.

7.11 Questions

1 How is modularization a better approach than the traditional approach.

2 Give examples of various types of relationships.

3 Explain all the file organizations.

4 What is normalization.

5 Briefly explain the need of detailed design.

Lesson No: 8 **Lesson Name : System Control and Quality Assurance**

8.0 Objectives:

- The goals of quality assurance in system development process
- What are the system design objectives
- What are the various software design and documentation tools
- Why system are tested
- The activity network of system testing
- What steps are taken to test systems
- What is a audit trails

8.1 Introduction

No program or system design is perfect. Communication between the user and the designer is not always complete or clear and time is usually short. The result is errors. The number and nature of errors in a new design depend on several factors.

1. Communication between the user and the designer.

2. The programmer's ability to generate a code that reflects exactly the system specifications.

These factors put an increasing burden on systems analysts to ensure the success of the system developed. The quality of a system depends on its design, development, testing and implementation.

8.2 Design Objectives

The two operational design objectives continually sought by developers are systems reliability and maintainability.

8.2.1 Reliable Systems

A system is said to have *reliability* if it does not produce dangerous or costly failures when it is used in a reasonable manner, that is, in a manner that a typical user expects is normal. This definition recognizes that systems may not always be used in the ways that designers expect. There are changes in the ways users use a system and also in business operations. However, there are steps analysts can take to ensure that the system is reliable when it is installed and that the reliability can be maintained after implementation.

8.2.1.1 Approaches to Reliability

There are two levels of reliability. The first is that the system is meeting the right requirements. For instance, a system might be expected to have specific security features or controls built into it by the users. But if the design fails to specify them and permits the loss of funds or merchandise for a lengthy time before someone detects the problem, the system is not reliable. Reliability at the design level is possible only if the analyst performed a thorough and effective determination of systems requirements. A careful and thorough systems study is needed to satisfy this aspect of reliability.

The second level of systems reliability involves the actual workings of the system delivered to the user. At this level, systems reliability is interwoven with software engineering and development.

An error occurs whenever the system does not produce the expected output. While it is true that no program is ever fully debugged or fully tested, nor proven correct – a fact that startles many users and aspiring programmers – errors are not limited to the correct use of programming syntax alone.

The computing industry, largely through the work of Glenford Myers, has come to distinguish between error and failures. A failure is the occurrence of a software error, weighted by its seriousness. For example, if an inventory program is developed to truncate rather than round half – rupees when calculating the value of materials on handed, it is an error if specifications call for rounding. But it may be of no consequence to the user, who in fact does not consider this a failure. However, if the program regularly skips certain items or indicates they are out of stock when in fact the records show they are in stock, there is a serious failure.

8.2.2 Error Avoidance

There are three approaches to reliability namely, error avoidance, error detection and error tolerance. Under error avoidance, developers and programmers make every attempt to prevent errors from occurring at all. The emphasis on early and careful identification of user requirements in another way this objective is pursued.

Analysts must assume that it is impossible to fully achieve this objective. Errors will occur despite the best efforts of very competent people.

8.2.3Error Detection and Correction

This method uses design features that detect errors and make necessary changes to correct either the error while the program is in use or the effect on the user, so that a failure does not occur. Correcting user errors, such as misspelling keywords or entering invalid commands, is one remedy. Error detection in programs is handled in a similar manner. For example, a program that calculates the productivity of a waiter or waitress in a restaurant by dividing the total revenue from meals served into the hours worked should not fail when employees do not serve anything. When a blinding snowstorm prevents customers from coming to a restaurant, employees will accumulate working time but will not have sales. The program should detect the divide – by – zero error and correct for it in order to keep the system running properly. Unfortunately, many programs fail when a situation like this one occurs. Even though it may not happen for several years after the system is installed, the error is there from the day of development. The failure occurs later.

8.2.4 Error Tolerance

Error tolerance strategies keep the system running even in the presence of errors. The United States National Aeronautics and Space Administration (NASA), for example, designs its systems to be error – tolerant through the use of redundant hardware. In one space program, redundant on – board computers and computer voting are used to process data in parallel, so results can be compared. Two computers process the data on location, course correction and compare the results with those produced by two other computers processing the same data. A fifth computer is available to break a tie should one occur. If needed, a sixth computer stored away in an accessible storage compartment can quickly replace one of the other computers that has been damaged or failed.

Another manner of error tolerance is the use of degraded processing. With this strategy, the user receives less service than the system was designed to provide, but that is considered a better alternative in some cases than having no service at all. For example, many electric power generation and distribution facilities in North America are computer – controlled. Suppose that on a record – breaking hot day the system becomes overloaded and the computer control center is unable to correctly process allocation data and keep up with the power demands. Rather than risk damaging the power distribution network, the computer automatically shuts down part of the network. By providing degraded service, the computer tolerates a software error without failing.

8.2.5 Causes of Errors

The software aspects of systems design are different from concerns about hardware reliability. In hardware, for example, any design errors are reproduced in every copy of the item manufactured. However, application systems are often unique and design errors are not widely distributed. Of course, if you are working on a system that will be sold commercially, there is considerable concern over development and marketing of software packages that is rampant with design errors.

Manufacturing errors are introduced during the actual production process. They are not a property of the design and, in fact, may not be in every item produced. Manufacturing errors may exist only in items made during a specific time period, either because of unknown problems with material quality or mistakes made by people newly assigned to a step in the process. In software systems, the equivalent of manufacturing errors is the small chance that, when disk or tape copies of programs are made for

distribution, errors will be introduced. This problem seldom occurs, however, and should not be a major concern to the analyst.

Hardware failures occur as equipment is used and begins to wear out. There is no equivalent in software; that is, we do not find software unusable because it is worn out. The medium on which it is carried (such as magnetic tape or disk) may become worn or damaged, but the software will not.

Therefore, the primary software problem is designing and developing software that will not fail. It is impossible to prove that there are no errors in a particular system.

The causes of errors that interest the analyst are (1) not obtaining the right requirements, (2) not getting the requirements right, and (3) not translating the requirements in a clear and understandable manner so that programmers implement them properly.

The transition from systems design to software development is an additional opportunity for introducing translation errors. These are the result of the programmer's not properly understanding or interpreting the design specifications produced by analysts. Conversely, they also occur when analysts force programmers to translate specifications that are incomplete. In the latter case, the programmer is forced to make design decision while coding the software.

When such misunderstanding exist and implementation occurs before they are detected, the result is a need for maintenance.

8.3 Maintenance of Systems

When systems are installed, they generally are used for long periods. The average life of a system is 4 to 6 years, with the oldest application often in use for over 10 years. However, this period of use brings with it the need to continually maintain the system. Because of the use a system receives after it is fully implemented, analysts must take precautions to ensure that the need for maintenance is controlled through design and testing and the ability to perform it is provided through proper design practices.

8.3.1 Maintenance Issues

Many private, university and government studies have been conducted to learn about maintenance requirements for information systems. The studies have generally concluded the following:

1. From 60 to 90 percent of the overall cost of software during the life of a system is spent on maintenance.
2. Often maintenance is not done very efficiently. In documented cases, the cost of maintenance, when measured on the basis of the cost of writing each instruction in code form, is more than 50 times the cost of developing a system in the first place.
3. Software demand is growing at a faster rate than supply. Many programmers are spending more time on systems maintenance than on new development. Studies have documented that in some sites, two – thirds of the programmes are spending their time on the maintenance of software. There is a backlog of new development work. Moreover, there is a hidden backlog, requests for development work that users do not bother even to submit because they know it will be years before development can being.

Several studies of maintenance have examined the type of tasks performed under maintenance. The broad classes maintenance found in information systems environments are corrective, adaptive and perfective. Once systems are installed, the need for debugging and correcting errors or failures on an emergency basis is comparatively low: less than 20 percent of the tasks are for correction.

Information systems and the organizations they serve are in a constant state flux. Therefore, the maintenance of systems also involves adaptations of earlier versions of the software. Approximately 20 percent of all maintenance is performed to accommodate changes in reports, files and data. This also includes adaptations required when new hardware or software is installed in a particular processing center.

The greatest amount of maintenance work is for user enhancement, improved documentation, or recoding systems components for greater efficiency. Sixty percent of all maintenance is for this purpose. Yet, many of the tasks in this category can be avoided if systems engineering is carried out properly.

8.3.2 Maintainable Designs

The keys to reducing the need for maintenance, while making it possible to do essential tasks more efficiently, are these:

1. More accurately defining the user's requirements during systems development.
2. Assembling better system documentation.
3. Using more effective methods for designing processing logic and communicating it to project team members.
4. Making better use of existing tools and techniques.
5. Managing the systems engineering process effectively.

As indicated by the preceding comments, design is both a process and a product. The design practices followed for software dramatically affect the maintainability of a system: good design practices produce a product that can be maintained.

8.4 Software Design

These principles should guide software design:

- *Modularity and Partitioning.*

Each system should consist of a hierarchy of modules. Lower level modules are generally smaller in scope and size compared to higher – level modules and serve to partition processes into separate functions.

- *Coupling*

  Modules should have little dependence on other modules in a system.

- *Cohesion*

  Modules should carry out a single processing function.

- *Span of Control*

Modules should interact with and manage the functions of a limited number of lower-level modules.

- *Size*

The number of instructions contained in a module should be limited to that module size is generally small.

- *Shared Use*

Functions should not be duplicated in separate modules, but established in a single module that can be invoked by any other module when needed.

8.4.1 Top-Down Structure of Modules

Top – down methods are used throughout the analysis and design process. The value of using a top-down approach, starts at the general levels to gain an understanding of the system and gradually moves down to levels of greater detail. In the process of moving from top downward, each component is “exploded” into greater detail. One data flow diagram became several at the next lower level.

During the discussion of input and menu design, a top-down approach was emphasized. The main menu contains several choices. Making one choice produces another menu in, which more detailed options are presented to the user. This capability provides users with an easy – to – understand method for using the system and selecting option. They do not have to make all decision together but instead can make one at a time.

The top – down method is also widely used in systems engineering and software design. Each function, the system will perform is first identified, and then developed in greater detail. Program designers term this as stepwise refinement: the procedures and processes are developed a step at a time, from general to specific.

For example, an accounting system consists of many separate modules that are invoked one at a time as users indicate the particular function they wish to perform. Each upper – level module in turn leads to using one or several lower-level modules until the desired function is performed.

8.4.2 Coupling

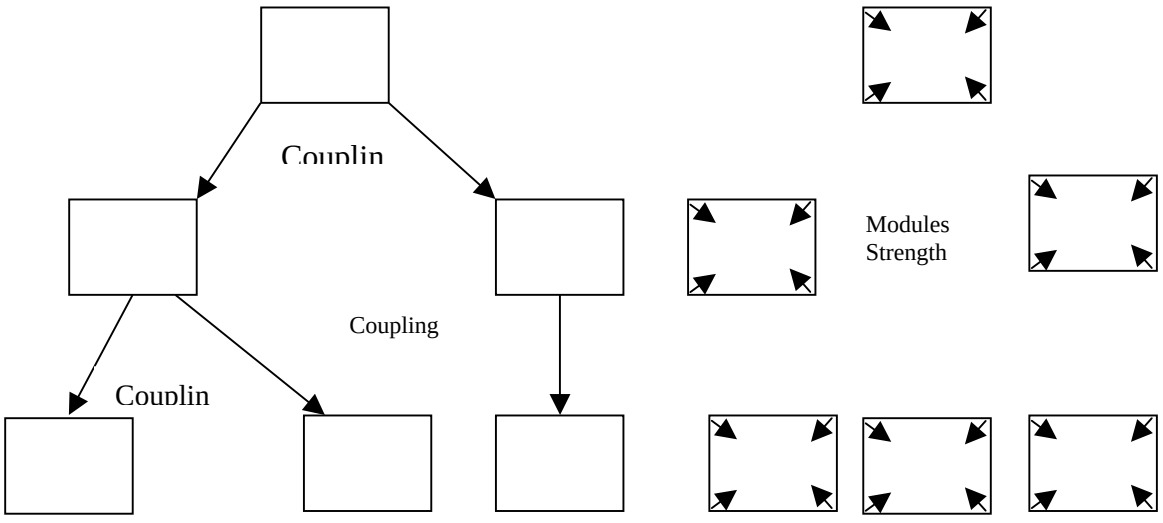
Coupling refers to the strength of the relationship between modules in a system. In general, good designers seek to develop the structure of a system so that one module has little dependence on any other module.

Loose coupling minimizes the interdependence between modules. We can achieve this in the following ways.

- ❖ Control the number of parameters passed between modules
- ❖ Avoid passing unnecessary data to called modules
- ❖ Pass data (whether upward or downward) only when needed
- ❖ Maintain superior/subordinate relationship between calling and called modules.
- ❖ Pass data, not control information

Consider the manner in which data are passed in an accounting system. In editing a vendor record (for the accounts payable portion), two alternative designs for editing a vendor record (for the accounts payable portion) are available. In the first, typified by tight coupling, which is undesirable, the calling module passes the vendor name, vendor identification number, address, tax status and date. The called module returns the customer record, along with an end – of – file flag.

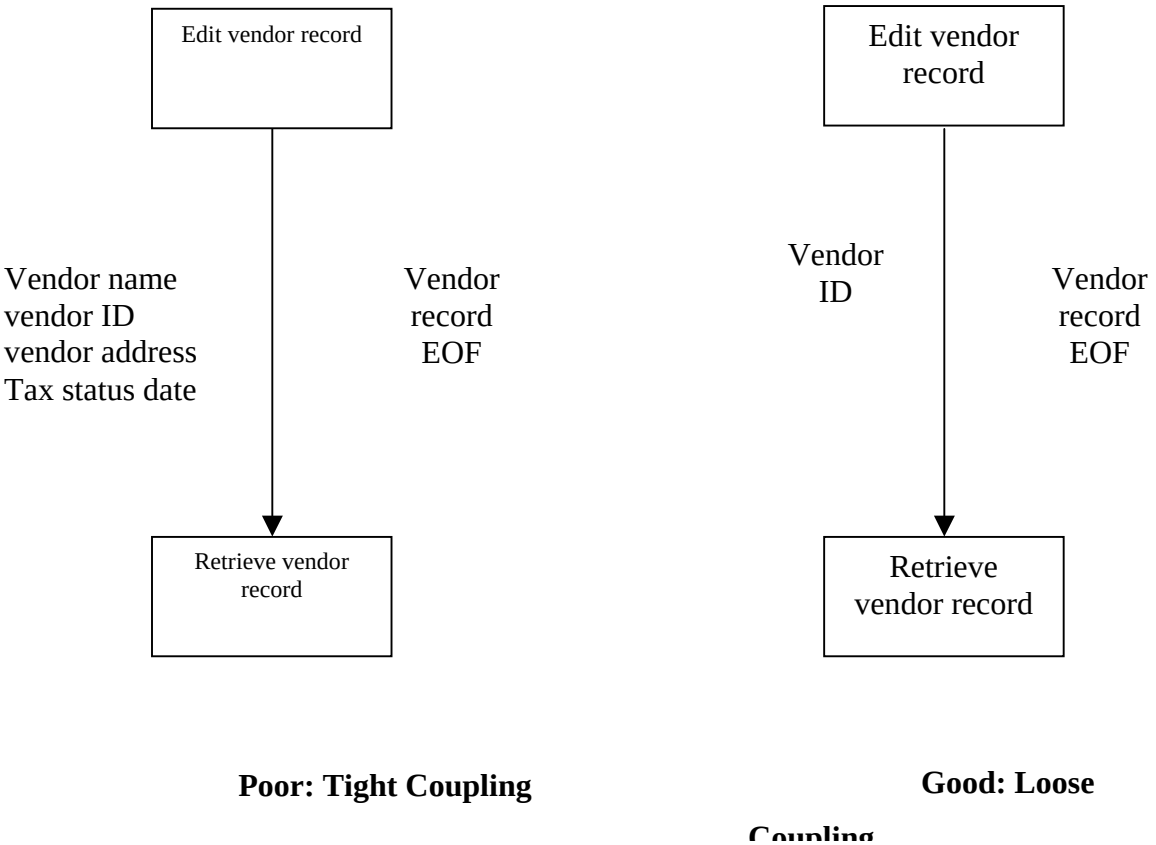
Figure: 8.1 Coupling & Cohesion in software design



Coupling: Strength of relations Between

Cohesion: Strength of relation Within modules

Figure 8.2 Coupling and strength of relations between modules



Poor: Tight Coupling **Good: Loose Coupling**

Compare this with the preferred loosely coupled version in which only the vendor ID is passed to retrieve the same record of information. Not only does this design move less data (only non-superfluous data), but also there is far less dependence between modules. Only the vendor identification is needed to distinguish one vendor's record from another. Since it is likely to be the record key, it is also unlikely to change. Other items in the record may change. Hence, the loosely coupled alternative is better suited to achieving the stated design and maintenance objectives.

Several poor design features should be avoided. Passing too little data can make it impossible to perform the task. For example, if the calling module does not pass the vendor ID, how does the subordinate module know which record to locate?

Designs that create floating data should also be avoided. This occurs when one module produces data that are not needed by the calling module but by another elsewhere in the system. The details are passed through the system (hence the term “floating”), until they finally reach the function that requires them. Redesigning, to establish loose coupling, along with the creation of more cohesive modules, will avoid this difficulty.

8.4.3 Cohesion

Using the top- down approach to planning the software for a system is no guarantee that errors will be avoided or that the system will be maintainable. In properly modularized, cohesive systems, the contents of the modules are so designed that they perform a specific function and are more easily understood by people than systems designed by other methods. There are four general types of modules contents:

1. Module contents determined by function performed,
2. Module contents determined by data used,
3. Module contents determined by logic of processing and
4. Module contents not closely related.

The least desirable type of grouping of module contents consists of steps that do not perform any complete function or that do not logically go together. This extreme case can arise if programmers work according to strict rules and divide modules into section of 50 statements each (or some other imposed limit). Input, output, calculation and file – handling activities are thus performed in a single module. If this case occurs, it is usually created as result of the programmer’s working without explicit design specifications or a clear understanding of how to handle a task.

Module contents may also be grouped together because they logically go together. A module that handles all input or all output operations or one that handles all order – processing activities, regardless of type of customer or data – handling needs for each customer, uses logical grouping.

The elements may also be related by the time at which they are performed; that is, they logically seem to go together and are performed at the same time. A module that

initializes all variables and opens files is logically bound. This level of modularity is preferable to the first type because all the elements are executable at one time.

Modules that are logically bound are difficult to modify because the cost will be shared for each type of activity. Even the simplest change can affect all types of transactions. A better solution is to separate each type of transaction into its own module.

Module contents may also be determined by the data used. A module in which the contents refer to the same data is preferable to one that is developed only on the basis of processing logic. For example, a module can be designed so that all operations on a set of data are performed at one time: a file is prepared to be printed on paper, spooled to disk, and also duplicated for backup purposes. A module that reads the next transaction and updates the master file by adding, deleting, or changing records, including the errors checking required for each type of function, shares a common set of data. This type of binding is better than the other types discussed, but it is not as good functional grouping.

Functional grouping permits more thorough testing of the module. If changes are needed at a later time, analysts and programmers can quickly determine how the module is constructed and how it processes data and interacts with other modules in the system.

The emphasis on reliability and maintainability is constant throughout systems development.

8.4.4 Span of Control

Span of control refers to the number of subordinate modules controlled by a calling module. In general, we should seek to have no more than five to seven subordinate modules.

On the one hand, excessive span of control, meaning a high number of subordinate modules, creates considerable overhead in determining which module to invoke under certain conditions and in establishing calling sequences to pass data and receive results. On the other hand, it typically results from not adhering to the coupling and cohesion objectives discussed previously.

8.4.5 Module Size

How large should a program module be? While it is impossible to fix a specific number of instructions, there are useful guidelines to manage module size.

Some organizations have established rules to manage module size. A common one is that no module should contain more than 50 instructions. Another is that the listing of source code for a module should fit on a single printer page. In some situations, these rules are appropriate, but in others they result in arbitrary decision (for example, “If the module cannot be coded in 50 instructions, create a second module that is called by the first”) that miss the point of managing module size.

In general, we should seek designs in which the modules focus on a single purpose, are highly cohesive, and are loosely coupled. Yet the modules should not be too small (when that occurs, modules should be combined). The size of the module depends on the language used as well. Approximately 50 statements in COBOL may be an acceptable upper limit (programmers should not have to redesign if, say 51 or 55 statements are needed, providing other design objectives are met). On the other hand, 50 instructions in a powerful fourth – generation language (where one instruction replaces from 10 to 50 equivalent COBOL statements) will probably be unacceptable.

8.4.6 Shared Modules

Shared use results from the desire to have a certain function, calculation, or type of processing performed in one place in a system. We want to design the system so that a certain calculation (such as determination of sales tax in an order processing system) is performed once. Then the module can be shared throughout the system by invoking it from various calling modules.

Why share modules? There are several reasons. First, sharing modules minimizes the amount of software that must be designed and written. Second, it minimizes the number of changes that must be made during system maintenance For instance, if tax calculation procedures change, only one module must be modified under the shared module principle. And third, having a single shared module reduces the chance of error: there is a greater likelihood that redundant modules will follow different calculation procedures, if not initially, after the introduction of maintenance changes (one module may not be changed because it is overlooked).

Many systems establish library modules – predefined procedures – which are included in the system’s program library. A single command or call quickly invokes the routine.

Shared library modules are one of the best ways in which good designers can design solutions.

8.5 Software Design and Documentation Tools

Well – designed, modular software is more likely to meet the maintenance, reliability, and testing requirements. Three specific tools are discussed: Structured flow-charts, HIPO diagrams, and Warnier / Orr diagrams.

8.5.1 Structured Flowcharts

Structured flowcharts, also called Nassi-Schneiderman Charts, are graphic tools that force the designer to structure software that is both modular and top- down. They provide a structure that can be retained by programmers who develop the application software. Organization responsibilities vary. In some organizations, analysts are responsible for developing module logic, while in others that responsibility is delegated to the programmer. In either case, the programmer should be well versed in the use of structured flowcharts.

8.5.1.1 Basic Elements

There are three basic elements used in developing structured flowcharts: process, decision, and iteration. (There are many similarities between these elements and the components used in structured English.)

Process: A rectangular box, the process symbol, represents Simple processes or steps in a program. This symbol represents initialization of values, input and output activities, and calls to execute other procedures.

A name of brief description written in the box states the purpose of the process. The succession of steps is shown using several process boxes.

8.5.1.2 Decision

The decision symbol represents alternative conditions that can occur and that the program must have a manner of handling. They show the equivalent of the IF-THEN-ELSE structures common in many programming languages. As examples will show, the decision symbol may show actions for more than two alternatives at the same time.

8.5.1.3 Iteration

The iteration symbol represents looping and repetition of operations while a certain condition exists or until a condition exists. The form of the iteration symbol

clearly shows the scope of the iteration, including all processes and decisions that are contained within the loop. The left – hand portion of the symbol shows the path of repetition to follow until the conditions controlling the iteration are satisfied.

8.5.1.4 Using Structured Flowcharts

Structured flowcharts use no arrows or continuations on separate pages. Each structured flowchart is shown on a single sheet of paper (or single display screen, if developed online).

When designing a structured flowchart the systems analyst specifies the logic in a top – down fashion. The first consideration in a process or decision is the top element. The second in sequence is the next one shown, and so forth. Similarly, there is a single exit from the process.

The analyst beings with a major process and introduces other symbols to subdivide the process. Each process is named, but, if the name is not underlined, it is a reference to some other diagram or description. This simple convention makes it possible to link together easily different procedures that are performed to complete an entire activity.

The structure chart reads from top to bottom and left to right. Each activity is nested within the iteration and alternative processes of which it is part. In addition, each condition is clearly shown.

Individual parts of processes are often further described in lower-level diagrams. Individual modules are referenced to handle the processing for each type of transaction.

An important use of structured flowcharts for the designer concerned about verifying systems specifications against planned software logic is to identify conditions and procedures followed when the conditions exist. The fact that the structure chart is easy to read will enable the analyst to determine whether the debit adjustment transaction, for example, has been added by the programmer or is a part of the original systems specifications.

8.5.2 HIPO

HIPO is another commonly used method for developing systems software. IBM developed this method of Hierarchical Input Process Output (HIPO), for large, complex operating systems.

8.5.2.1 Purpose

The assumption on which HIPO is based is that is easy to lose track of the intended function of a system or component in a large system. This is one reason why it is difficult to compare existing systems against their original specifications (and therefore why failures can occur even in systems that are technically well formulated). From the user's view, single functions can often extend across several modules. The concern of the analyst then is understanding, describing, and documenting the modules and their interaction in a way that provides sufficient detail but that does not lose sight of the larger picture.

HIPO diagrams are graphic, rather than prose or narrative, descriptions of the system. They assist the analyst in answering three guiding questions:

1. What does the system or module do? (Asked when designing the system).
2. How does it do it? (Asked when reviewing the code for testing or maintenance).
3. What are the inputs and outputs? (Asked when reviewing the code for testing or maintenance.)

A HIPO description for a system consists of the visual table of contents and the functional diagrams.

8.5.3 Visual Table of contents

The visual table of contents (VTOC) shows the relation between each of the documents making up a HIPO package. It consists of a hierarchy chart that identifies the modules in a system by number and in relation to each other and gives a brief description of each module. The numbers in the contents section correspond to those in the organization section.

The modules are in increasing detail. Depending on the complexity of the system, three to five levels of modules are typical.

8.5.3.1 Functional Diagrams

There is one diagram for each box in the VTOC. Each diagram shows input and output (right to left or top to bottom), major processes, movement of data, and control points. Traditional flowchart symbols represent media, such a magnetic tape, magnetic

disk, and printed output. A solid arrow shows control paths, and open arrow identifies data flow.

Some functional diagrams contain other intermediate diagrams. But they also show external data, as well as internally developed data (such as tables in the invoice example) and the step in the procedure where the data are used. A data dictionary description can be attached to further explain the data elements used in a process.

HIPO diagrams are effective for documenting a system. They also aid designers and force them to think about how specifications will be met and where activities and components must be linked together. However, they rely on a set of specialized symbols that require explanation, an extra concern when compared to the simplicity of, for example, data flow diagrams. HIPO diagrams are not as easy to use for communication purposes as many people would like. And, of course, they do not guarantee error-free systems. Hence, their, greatest strength is the documentation of a system.

8.5.4 Warinier/Orr Diagrams

Warnier/Orr diagrams (also known as logic construction of programs/logical construction of system) were initially developed in France by Jean – Dominique Warnier and in the United States by Kenneth Orr. This method aids the design of program structures by identifying the output and processing results and then working backwards to determine the steps and combinations of input needed to produce them. The simple graphic methods used in Warnier/Orr diagrams make the levels in the system evident and the movement of the data between them vivid.

8.6 Managing 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.

8.6.1 Levels of Assurance

Analysts use four levels of quality assurance: testing, verification, validation, and certification.

8.6.1.1 Testing

Systems 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.

8.6.1.2 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 very 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.

8.6.1.3 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.

8.7 Testing Plans

The philosophy behind testing is to find errors. Test cases are devised with this purpose in mind. A test case is a set of data that the system will process as normal input. However, the data are created with the express intent of determining whether the system will process them correctly. For example, test cases for inventory handling should include situations in which the quantities to be withdrawn form inventory exceed, equal, and are less than the actual quantities on hand. Each test case is designed with the intent of finding errors in the way the system will process it.

There are two general plans for testing software: The strategies of code testing ad specification testing.

8.7.1 Code Testing

The code-testing strategy examines the logic of the program. To follow this testing method, the analyst develops test cases that result in executing every instruction in the program or module; that is, every path through the program is tested. A path is a specific combination of conditions that is handled by the program. For example, in the accounting systems example, one path through the system is to change the account

balances. The correct request is submitted, then the proper passwords, data, and command entries.

On the surface, code testing seems to be an ideal method for testing software. However, the rationale that all software errors can be uncovered by checking every path in a program is faulty. First of all, in even moderately large programs of the size used in typical business situations, it is virtually impossible to do exhaustive testing of this nature. Financial considerations and time limitations alone will usually preclude executing every path through a program, since there may be several thousand.

However, even if code testing can be performed in its entirety, it does not guarantee against software failures. This testing strategy does not indicate whether the code meets its specifications nor does it determine whether all aspects are even implemented. Code testing also does not check the range of data that the program will accept, even though, when software failures occur in actual use, it is frequently because users submitted data outside of expected ranges.

8.7.2 Specification Testing

To perform specification testing, the analyst examines the specifications stating what the program should do and how it should perform under various conditions. Then test cases are developed for each condition or combination of conditions and submitted for processing. By examining the results, the analyst can determine whether the program performs according to its specified requirements.

This strategy treats the program as if it were a black box: the analyst does not look into the program to study the code and is not concerned about whether every instruction or path through the program is tested. In that sense, specification testing is not complete testing. However, the assumption is that, if the program meets the specifications, it will not fail.

Neither code nor specification testing strategy is ideal. However, specification testing is a more efficient strategy, since it focuses on the way software is expected to be used. It also shows once again how important the specifications developed by the analysts are throughout the entire systems development process.

8.7.3 Managing Testing Practices

Regardless of which strategy the analyst follows, there are preferred practices to ensure that the testing is useful. The levels of tests and types of test data, combined with testing libraries, are important aspects of the actual test process.

8.7.4 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.

8.7.5 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 form 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.

8.7.6 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 reindexing 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.

8.7.7 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 judgement 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 full 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 form 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 form 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.

8.8 Designing Test Data

There are two very different sources of test data, live and artificial. Both have distinct advantages and disadvantages for the tester.

8.8.1 Using Live Test Data

Live test data are those that are actually extracted from organization files. After a system partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. For example, in a general ledger accounting system, they may ask someone from the accounting staff to enter the chart of account numbers and a set of account balances, along with transactions affecting those accounts. Then the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirements, assuming that the live data entered are in fact typical, such data generally will not test all the combinations or formats that can enter the system. The bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause systems failure.

8.8.2 Using Artificial Test Data

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data – generating utility program in the information systems department, make possible the testing of all logic and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of tester formulates a testing plan, using the systems specifications.

8.9 Testing Libraries

To assure that all systems are properly tested, many organizations establish test libraries. A testing library is a set of data developed to thoroughly test a system of programs. It is stored in machine – readable form, usually on magnetic disk, and is used by all persons who are involved with a particular system.

For example, a large inventory system consists, of hundreds of computer programs. All share common data and file formats. Each will also process similar transactions and will sometimes update records and other times retrieve data to respond to inquiries or prepare reports and documents. Because these programs are interdependent and process – related transactions, it makes sense to use a common set of data to test each program.

Test libraries are not just for initial testing. As the system evolves and programs are modified and maintained, they must be re-tested. The testing library should be maintained throughout the life of the system so that, as each change is made, reliable data are again available to test the system.

8.10 System controls

A well-designed system should have controls to ensure proper operation and routine auditing. A candidate systems failure often results from lack of emphasis on data control. Therefore, standards of accuracy, consistency and maintainability must be specified to eliminate errors and control for fraud.

A system design introduces new control elements and changes the control procedures. New controls in the form of relational comparisons are designed to detect and check errors that rise form the use of the system. In a manual system, internal control depends on human judgement, personal care and division of labor. In a computer based system the number of persons involved is considerably reduced. In designing a new system the designer should specify the location of error control points and evaluate them on the basis of error frequency, cost and timing of error detection. By identifying points

where potential errors may occur, designers can create control procedures for handling errors immediately.

8.10.1 Processing controls

Several methods have been devised to control processing activities:

1. Data record may be combined into small groups to control totals. If in batch processing, error is encountered, the batch may be held and reviewed to correct the error.
2. Completeness check ensures that all fields in a record are present and are read in the proper sequence. In a multiple record check, the program verifies the self-checking number of the records that make up the transaction. If an error is detected, the entire group of records is rejected.
3. Consistency check refers to the relevance of one type of data to another. Data being accepted through various means need to be checked for its uniformity. All critical paths need to be checked for its proper path selection.
4. Reasonableness check evaluates a transaction against a standard or maximum / minimum value to determine its validity. For example an employee may not have age less than 21 and not more than 60 years.
5. Sequence check verifies that data records are in sequence prior to processing. Duplicate records need to be checked.

8.11 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.

8.11.1 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.

8.12 Summary:

No program or system design is perfect. Communication between the user and the designer is not always complete or clear and time is usually short. The result is errors. The number and nature of errors in a new design depend on several factors. The two operational design objectives continually sought by developers are systems reliability and maintainability. There are three approaches to reliability namely, error avoidance, error detection and error tolerance. Under error avoidance, developers and programmers make every attempt to prevent errors from occurring at all. The emphasis on early and careful identification of user requirements in another way this objective is pursued. Correcting

user errors, such as misspelling keywords or entering invalid commands, is one remedy. Error detection in programs is handled in a similar manner. Error tolerance strategies keep the system running even in the presence of errors. When systems are installed, they generally are used for long periods. The average life of a system is 4 to 6 years, with the oldest application often in use for over 10 years. Several studies of maintenance have examined the type of tasks performed under maintenance. The broad classes maintenance found in information systems environments are corrective, adaptive and perfective. Once systems are installed, the need for debugging and correcting errors or failures on an emergency basis is comparatively low: less than 20 percent of the tasks are for correction. Software design should be guided by modularity and partitioning , coupling, cohesion, span of control , size and shared use. Well – designed, modular software is more likely to meet the maintenance, reliability, and testing requirements. Three specific tools are discussed: Structured flow-charts, HIPO diagrams, and Warnier / Orr diagrams. 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. 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, and human factor testing. A well-designed system should have controls to ensure proper operation and routine auditing. A candidate systems failure often results from lack of emphasis on data control. Therefore, standards of accuracy, consistency and maintainability must be specified to eliminate errors and control for fraud. 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.

8.13 Questions

1. Differentiate between error tolerance & Error avoidance.
2. What are the causes of errors.
3. Explain Coupling & Cohesion.
4. Conduct a comparative study between the various documentation tools.
5. What are the levels of assurance.

Lesson No: 9 **Lesson Name : System Administration and Training**

9.0 Objectives

- What is the objectives of system administration
- What is the purpose of giving training to the user of system
- The different types of training
- How conversion will takes place from existing system to new system
- What is the conversion plan
- What are the phases of conversion
- What is the operating plan

9.1 Introduction

Putting a new system into operation is usually complicated by the fact that there is an older system already in operation. The analyst has to deal with changing from something familiar to something new and different, while also attending to the mechanics of implementation. Since the concern for simultaneous conversion and implementation is usual. New system brings in new equipment. It may represent a change from manual to automated operation or a change in the level of available machine capacity. During implementation, planning plays a decisive factor in determining the ultimate success or failure of system. Due attention should be paid to:

1. Assigning system personnel.
2. Structuring user relationship.
3. Preparing for new equipment.
4. Training user personnel.

1. Assignment of Systems Personnel:

Assign people to the implementation who demonstrate the ability in dealing with the unique problem situations associated with the process.

2. Structuring user Relationships:

Plan for periodic meeting between user and system personnel for the duration of the implementation, to discuss problems being faced. Also there should be provisions to meet when the need arises. Certainly waiting for the meeting in critical problems is not a reasonable approach. Also if people meet only during crisis, they cannot expect a very positive encounter.

3. Preparing for New Equipment:

New equipment means more complexity. For new equipment additional areas of concern are to be taken care of: -

1. Structuring a relationship with the equipment vendor.
2. Preparing a physical site for installation and use of new equipment.
3. Installation of new equipment and removing old equipment.
4. Training personnel to use the new equipment.

4. Training of user Personnel:

Planning for the formal training of user personnel in the operation of the new system is important. A new method may drastically affect people's lives by changing their work methods, work style and relationships with other employees. One of the most effective ways of dealing with the potential impact of these changes is to provide a well-designed program of training. The training program should:

a) Inform the user about the system in general

b) Inform the user about specific operation of the system

c) Give the user some practice in operating the system

d) Provide opportunity for user feed back.

e) Provide ample opportunity to adjust to the new system.

f) Provide answers to the queries raised by the employees.

g) Generate a feeling among employees that the new system is “their” system.

9.2 Training

Even well designed and technically elegant systems can succeed or fail because of the way they are operated and used. Therefore, the quality of training received by the personnel involved with the system in various capacities helps or hinders, and may even prevent, the successful implementation of an information system. Those whose will be associated with or affected by the system must know in detail what their roles will be, how they can use the system, and what the system will or will not do. Both systems operators and users need training.

9.2.1 Training systems operators

Many systems depend on the computer – center personnel, who are responsible for keeping the equipment running as well as for providing the necessary support service. Their training must ensure that they are able to handle all possible operations, both routine and extraordinary. Operator training must also involve the data entry personnel.

If the system calls for the installation of new equipment, such as a new computer system, special terminals, or different data entry equipment, the operators training should include such fundamentals as how to turn the equipment on and use it, how to power it down, and a knowledge of what constitutes normal operation and use. The operators should also be instructed in what common malfunctions may occur, how to recognize them, and what steps to take when they arise. As part of their training, operators should be given both a troubleshooting lists that identifies possible problems and remedies for them, as well as the names and telephone numbers of individuals to contact when unexpected or unusual problems arise.

Training also involves familiarization with run procedures, which involves working through the sequence of activities needed to use a new system on an ongoing basis. These procedures allow the computer operators to become familiar with the actions they need to take (such as mounting magnetic disks or tapes, copying files, changing printer forms, or turning on communication systems), and when these actions must occur. In addition, they find out how long applications will

run under normal conditions. This information is important both to enable users to plan work activities and to identify systems that run longer or shorter than expected – a sign that typically indicates problems with the run.

9.2.2 User Training

User training may involve equipment use, particularly in the case where, say, a microcomputer is in use and the individual involved is both operator and user. In these cases, user must be instructed first in how to operate the equipment. Questions that seem trivial to the analyst, such as how to turn on a terminal, how to insert a diskette into a microcomputer, or when it is safe to turn off equipment without danger of data loss, are significant problems to new users who are not familiar with computers.

User training must also instruct individuals in troubleshooting the system, determining whether a problem that arise is caused by the equipment or software or by something they have done in using the system. Including a troubleshooting guide in systems documentation will provide a useful reference long after the training period is over. There is nothing more frustrating than working with a system, encountering a problem, and not being able to determine whether it is the user's fault or a problem with the system itself. The place to prevent this frustration is during training.

Most user training deals with the operation of the system itself. Training in data coding emphasizes the methods to be followed in capturing data form transactions or preparing data needed for decision support activities. For example, in an accounting system, it may be important to translate customer names into customer account numbers that are input as part of the accounting transaction. Users must be trained so that they know how to determine the customer account number, that it is four digits in length, and that there are no alphabetic characters in it.

Data – handling activities receiving the most attention in user training are adding data (how to store new transactions), editing data (how to change previously stored data), formulating inquiries (finding specific records or getting responses to questions) and deleting records of data. The bulk of systems use involves this set of activities, so it follows that most training time will be devoted to this area.

From time to time, users will have to prepare disks, load paper into printers, or change ribbons on printers. No training program is complete without some time devoted to systems

maintenance activities. If a microcomputer or data entry system will use disks, users should be instructed in formatting and testing disks. They should also actually perform ribbon changes, equipment cleaning and other routine maintenance. It is not enough to simply include this information in a manual, even though that is essential for later reference.

As the above discussion demonstrates, there are two aspects to user training: familiarization with the processing system itself (that is, the equipment used for data entry or processing) and training in using the application (that is, the software that accepts the data, processes it, and produces the results). Weaknesses in either aspect of training are likely to lead to awkward situation that produce user frustration, errors, or both. Good documentation, although essential, does not replace training. There is no substitute for hands – on – operation of the system while learning its use.

9.2.3 Training methods

The training of operators and users can be achieved in several different ways. Training activities may take place at vendor locations; at rented facilities, for example, in hotels or on university campuses; or in house at the employee's organizations. The methods and content of the training often vary, depending on the source and location of the training.

9.2.3.1 Vendor and In-Service Training

Often the best source of training on equipment is the vendor supplying the equipment. Most vendors offer extensive educational programs as part of their services, in some cases, there is a charge, but in many instances training is free. For example, IBM offers complimentary two and three – day courses to purchasers of many of their minicomputers and mainframe computers. The courses, offered by experienced trainers and sales personnel, cover all aspects of using the equipment, from how to turn it on and off, to the storage and removal of data, to handling malfunctions. This training is hands-on, so the participants actually use the system in the presence of the trainers. If questions arise, they can quickly be answered. Since the system is intended for training, there is generally no rush to get training out of the way so that the productive use of the system can start. Training conducted at the organization's location might be rushed, a danger that installation personnel must guard against.

If special software such as a teleprocessing package or database management system is being installed, sending personnel to off – site short courses providing in – depth training is preferable to in – service training. These courses, which are generally provided for a fee, are

presented to personnel from many organizations that are acquiring or using the same system. The benefit of sharing questions, problems, and experiences with persons from other companies is substantial. The personal contacts made during the sessions frequently last for years, with the continual sharing of information benefiting both parties. Short courses often involve additional time and costs for travel to other cities.

9.2.3.2 In – house Training

The advantage of offering training for the system on site is that the instruction can be tailored to the organization where it is being offered and focused on special procedures used in that setting, the organization's plans for growth, and any problems that have arisen. Often, the vendors or training companies negotiate fees and charges that are more economical and that enable the organization to involve more personnel in the training program than is possible when travel is required.

There are also disadvantages. The mere fact that employees are in their own surroundings is a distraction, since telephone calls and emergencies can disrupt training sessions. Moreover, when outside firms' come on – site, they many present courses that emphasize general concepts but that lack sufficient hands – on training. The training coordinator must recognize this possibility and deal with it an advance to ensure that the course content will meet operating needs.

In – house training can also be offered through special purchased instructional materials. A variety of professional training programs on special topics can be rented or purchased from computer training firms such as Edutronics (McGraw – Hill, Inc.); Deltak, Inc.; Professional Development, Inc.; and Learning Corporation of America. Other vendors offer printed and audiovisual programmed instruction materials that are either self - instructional or that supplement other training activities.

However, there is no substitute for hands –on experience. Training manuals are acceptable for familiarization, but the experiences of actually using the equipment, making and correcting mistakes, and encountering unexpected situations are the best and most lasting way of learning.

Training manuals generally take one of two approaches. Some have the user work through different activities step by step. For example, the checklist is provided to list the steps necessary to implement a system. Each individual step is listed in the proper order.

The other common approach is to create a case – study example that include all frequently encountered situations that the system is able to handle and that the users should be able to handle. Then the users must use the system to handle the actual situations; that is enter data as required, process the data, and prepare reports. If the system is inquiry – oriented, the case study should require the user to pose and receive responses to inquiries. If the results produced do not match those provided in the training guide, the users will know that mistakes were made.

During training, systems personnel should be alert to comments made by users or to problems that users may encounter. Although human factors testing, performed earlier, is intended to detect difficulties, some problems may not occur until inexperienced users are directly interacting with the system. Despite testing, awkward keying requirements to enter data, unexpected transactions, or unusual ways of preparing transactions may still arise during training. The trainer must be certain to involve systems personnel when problems in the design are found, while assisting users who are reluctant to change from their old ways to the new methods required to use the system. Of course, the trainer must first be certain that the new methods are necessary and do represent an improvement over current methods.

9.3 Conversion

Conversion is the process of changing form the old system to the new one.

9.3.1 Conversion Methods

There are four methods of handling a systems conversion (Table 9.1). Each method should be considered in light of the opportunities that it offers and problems that it may cause. However, some situations dictate the use of one method over others, even though other methods may be more beneficial. In general, systems conversion should be accomplished as quickly as possible. Long conversion periods increase the possible frustration and difficulty of the task for all persons involved, including both analysts and users.

9.3.1.1 Parallel systems

The most secure method of converting from an old to new system is to run both systems in parallel. Under this approach, users continue to operate the old system in the accustomed manner but they also begin using the new system. This method is the safest conversion approach, since it guarantees that, should problems such as errors in processing or inability to handle certain types of transactions arise in using the new system, the organization can still fall back to the old system without loss of time, revenue, or service.

The disadvantages of the parallel systems approach are significant. First of all, the system costs double, since there are two sets of systems costs. In some instances it is necessary to hire temporary personnel to assist in operating both systems in parallel. Second, the fact that users know they can fall back to the old ways may be a disadvantage if there is potential resistance to the change or if users prefer the old system. In other words, the new system may not get a fair trail.

All in all, the parallel method of systems conversion offers the most secure implementation plan if things go wrong, but the costs and risks to a fair trail cannot be overlooked.

Table 9.1 Methods of Systems Conversion

Method	Description	Advantages	Disadvantages
Parallel system	The old system is operated along with the new system	Offers greatest security. The old system can take over if errors are found in the new system or if usage problems occur.	Doubles operating costs. The new system may not get fair trail.
Direct conversation	The old system replaced by the new one. The organization relies fully on the new system.	Forces users to make the new system work. There are immediate benefits from new methods and controls.	There is no other system to fall back on if difficulties arise with new system. Requires the most careful planning.
Pilot system	Working version of system implemented in one part of the organization. Based on feedback, changes are made and the system is installed in rest of the organization by one of the other methods.	Provides experience and live test before implementation.	May give the impression that the old system is unreliable and not error-free.
Phase – in	Gradually implement system across all users.	Allows some users to take advantage of the system early. Allows training and installation without unnecessary use of resources.	A long phase – in causes user problems whether the project goes well (over enthusiasm) or not (resistance and lack of fair trial).

9.3.1.2 Direct Cutover

The direct cutover method converts from the old to the new system abruptly, sometimes over a weekend or even overnight. The old system is used until a planned conversion day, when it is replaced by the new system. There are no parallel activities. If the analyst must make the change and wants to ensure that the new system fully replaces the old one so that users do not rely on the

previous methods, direct cutover will accomplish this goal. Psychologically, it forces all users to make the new system work; they do not have any other method to fall back on.

The advantage of not having a fallback system can turn into a disadvantage if serious problems arise with the new system. In some instances, organizations even stop operations when problems arise so that difficulties can be corrected.

One organization allocated its entire accounting staff to entering data to start a new automated system. The task took approximately three weeks, during which time none of the regular accounting operations that were to be converted to the new system were performed. Consequently, a three – week backlog of work developed. However, such backlog was expected and management had planned to authorize overtime work and the hiring of temporary assistance to catch up after conversion. Approximately two days before the direct cutover was to take place, a senior manager realized that the accounting department was not planning to preserve the data for accounts receivable aging. The manager stopped the conversion. As a result, the accounting staff had to catch up on three weeks work and reschedule the conversion to a date one-month later, when many of the previous steps had to be restarted. The system was finally implemented three months later, after much extra work, overtime, and staff frustration because of the way the cutover was handled.

Stopping conversion was a particularly drastic measure. It would have been doubly bad had the steps been taken because of technical problems needing correction. If users know that a system was once halted because of difficulties, they may not be fully confident that the system will be reliable, even if analysts tell them that the problems have been corrected. The time it takes to redo work that was stopped because of the conversion can be both lengthy and costly, and time lost can never be recaptured.

Direct cutover require careful planning. Training sessions must be scheduled and maintained. The installation of all equipment must be on time, with ample days allowed in the schedule to correct any difficulties that occur. Any site preparation must be complete before the conversion can be done.

Direct conversions are quite common, particularly with purchased or turnkey systems. For example, a hotel operation decided to install an automated reservation system. The entire system was implemented during a one – week period, when the computer system was set up, the

software loaded, and the system tested. During that week, a separate training crew worked with all the accounting and front desk personnel to familiarize them with the operation and use of the system. These activities occurred Monday through Saturdays. On Sunday, all personnel were brought in to enter reservations, guest charges, and accounting information into the new system so that it coincided with the current system. On Sunday, evening, after the close of business for the day, the new system was started and used permanently. The old paper reservation file was removed, and the cash registers and bookkeeping machines were replaced with the terminals. The new system became live at midnight on Sunday. There was no old system to fall back on.

9.3.1.3 Pilot Approach

When new systems also involve new techniques or drastic changes in organization performance, the pilot approach is often preferred. In this method, a working version of the system is implemented in one part of the organization, such as a single work area or department. The users in this area typically know that they are piloting a new system and that changes can be made to improve the system.

When the system is deemed complete, it is installed throughout the organization, either all at once (direct cutover method) or gradually (phase – in method).

This approach has the advantage of providing a sound proving ground before full implementation. However, if the implementation is not properly handled, users may develop the impression that the system continues to have problems and that it cannot be relied on. For example, they may feel that the difficulties they experienced for two or three weeks may in fact not be gone just because the analysts claim they are.

9.3.1.4 Phase – In Method

The phase- in method is used when it is not possible to install a new system throughout an organization all at once. The conversion of files, training of personnel, or arrival of equipment may force the staging of the implementation over a period of time. Ranging form weeks to months. Some users will begin to take advantage of the new system before others.

For example, a medical system aimed at linking 10 or 15 different clinics to hospital may phase in over a year. The work required to convert patient and insurance records on paper to files stored on magnetic disks requires 2 to 3 weeks for each clinic. A week of user training is also

required for each clinic. Therefore, the analysts may phase this system in one clinic at a time, allowing 3 to 4 weeks for each conversion. It is conceivable in this system that the full conversion will be phased over one year.

Long phase – in periods create difficulties for analysts, whether the conversions go well or not. If the system is working well, early users will communicate their enthusiasm to other personnel who are waiting for implementation. In fact, enthusiasm may reach such a high level that when a group of users does finally receive the system, there is a letdown. In the clinic example, for instance, the medical staff may exaggerate the time savings that accrue from not having to search for medical records or manually prepare insurance claims, activities that will be handled by the new system. Later, when conversion occurs, the staff finds out that the system does not do the processing instantly. The disappointment is understandable.

On the other hand, if there are problems in the early phases of implementation, word of difficulties will spread also. Then the users may expect difficulties at the time of conversion and react negatively to the smallest mistakes. When systems are phased in, they must work well on the first conversion and all that follow.

9.4 Conversion Plan

The conversion plan includes a description of all the activities that must occur to implement the new system and put it into operation. It identifies the persons responsible for each activity and includes a timetable indicating when each activity will occur.

During the pre-implementation stages, when the conversion is being planned, analysts should assemble a lit of all tasks, including the following:

1. List all files for conversion.
2. Identify all data required to build new files during conversion.
3. List all new documents and procedures that go into use during conversion.
4. Identify all controls to be used during conversion. Establish procedures for cross-checking the old and new systems. Determine how team members will know if something has not been completed properly.
5. Assign responsibility for each activity.
6. Verify conversion schedules.

The conversion plan should anticipate possible problems and ways to deal with them. Among the most frequently occurring problems are missing documents, mixed data formats between current and new files, errors in data translation, missing data or lost files, and situations that were overlooked during systems development. The conversion manager must guard against the omission of steps in the conversion. A checklist will prevent missed steps. Personnel absences must also be expected and adequate fallback plans specified.

Conversion timing is challenging, since there are so many aspects of the conversion, ranging from the installation of equipment to the ordering of forms and supplies.

9.5 Operating Plan:

The operating plan is checking of all arrangements. It includes reviewing conversion plans, verifying the delivery of equipment, software, forms, preparing the site and preparing the data and files.

1. **Site Preparation:** Analysts often work with vendor personnel to outline site – preparation guidelines. Due importance should be paid to electrical using, air conditioning needs, humidity controls, space requirements, etc.
2. **Data and File Preparation:** For a new system to begin master files and system files need to be entered into the system before the normal functioning of the system. Master files are generally created manually. The number of records in older system master file should tally with the number of records in new master file.

In case of financial software the balance brought forward should be checked for validation before implementation of the new system.

9.6 Summary

For administration putting a new system into operation is usually complicated by the fact that there is an older system already in operation. The analyst has to deal with changing from something familiar to something new and different, while also attending to the mechanics of implementation. Since the concern for simultaneous conversion and implementation is usual. Planning for the formal training of user personnel in the operation of the new system is important. A new method may drastically affect people's lives by changing their work methods, work style and relationships with other employees. One of the most effective ways of dealing with the potential impact of these changes is to provide a well-designed program of training. The training program should:

h) Inform the user about the system in general

i) Inform the user about specific operation of the system

j) Give the user some practice in operating the system

k) Provide opportunity for user feed back.

l) Provide ample opportunity to adjust to the new system.

m) Provide answers to the queries raised by the employees.

Generate a feeling among employees that the new system is “their” system. the quality of training received by the personnel involved with the system in various capacities helps or hinders, and may even prevent, the successful implementation of an information system. Conversion is the process of changing form the old system to the new one. The conversion plan includes a description of all the activities that must occur to implement the new system and put it into operation. It identifies the persons responsible for each activity and includes a timetable indicating when each activity will occur. The operating plan is checking of all arrangements. It includes reviewing conversion plans, verifying the delivery of equipment, software, forms, preparing the site and preparing the data and files.

Questions

1. Compare the different conversion methods with each other.
2. Explain the operating plan.
3. To what extent training is important.
4. “Conversion is simple”. Do you agree. Justify.
5. What is training system operators.

Lesson No: 10 **Lesson Name : Hardware and Software Selection**

10.0 Objectives

- What kind of hardware and peripherals are required
- How the hardware is selected

- How the computer evaluation and measurement will be conducted
- What is benchmarking
- What are the various financial factors
- How the maintenance and support will be provided to the system
- How the vendor selection is done
- what are the different criteria for software selection
- How the system performance evaluation will be done

10.1 Introduction

A major element in building systems is selecting compatible Hardware & software. The kind of hardware & peripherals required is to be determined. The suitable software has to be selected. The experienced analysts will explore various options regarding it. Hardware/software selection begins with requirements analysis, followed by a request for proposal and vendor evaluation. The final system selection initiates contract negotiations, price, maintenance agreements, vendor selection, acceptance criteria and similar issues.

10.2 Hardware Selection

Gone are the days when a user calls IBM to order a 360 system, which in itself included hardware, software & support. Today, selecting a system is a serious and time concurring activity. Unfortunately, many systems are still selected based on vendor reputation only or other subjective factors. Instead the factors, which are to be considered, should be determining equipment size, capacity needs, financial considerations and acquisition method.

10.3 Determining size and capacity requirements

With computers ranging in size from small microcomputers to large mainframe systems, the number of options to choose from when selecting a system is obviously very large. Even within the lines of a single manufacturer, there are many different models and configurations from which to select. How then does the analyst determine which system to use when a new computer is to be acquired?

The starting point in an equipment decision process is the size and capacity requirements. One particular computer system may be appropriate for one workload and inappropriate for another. Systems capacity is frequently the determining factor. Relevant features to consider include the following:

1. Internal memory size
2. Cycle speed of system for processing
3. Characteristics of display and communication components
4. Types and numbers of auxiliary storage units that can be attached
5. Systems support and utility software provided or available

Frequently, software needs dictate the minimum configuration required. For instance, if a particular program to be run on a microcomputer requires, say, 4 megabytes of storage, the list of feasible candidates will exclude all systems, regardless of their attractiveness, that do not have or that cannot be easily configured to have a memory of at least 4 megabytes.

All systems have limits, depending on what they are designed for. The limits may or may not be a factor in a particular selection decision. For example, some systems communicate data only in a synchronous fashion. If the system has other attractive features and will not be used for data communications or teleprocessing, the synchronous feature may be of little concern. However, if the primary application for the computer requires synchronous transmission of ASCII data, the bisynchronous limitation is important. Likewise, the fact that a particular minicomputer is limited to five ports for connecting terminals and printers may be too restrictive in a teleprocessing system designed to link 23 sites together through terminals and communications lines.

Software needs often dictate hardware requirements such as internal memory sizes, communication ports, disk capacity, and the ability to use magnetic tape. Vendors are reliable sources of configuration requirements. They can provide information on the minimum configuration requirements needed to use their software properly. Trade newspapers and magazines provide regular distribution of information

about hardware and software requirements. In addition, subscription services offer information on operating specifications. These services, which cost several hundred dollars yearly, provide monthly updates (generally using a loose-leaf binder format) and telephone assistance for computer operation, as well as user comments.

Auxiliary storage capacity is generally determined by file storage and processing needs. To estimate the disk storage needed for a system, the analyst must consider the space needed for each master file, the space for programs and software, including systems software, and the method by which backup copies will be made. When using flexible diskettes on a small business system, the analyst must determine whether master and transaction files will be maintained on the same diskette and on which diskette, programs will be stored. Backup considerations, as well as file size, guide the decision about how many disk drives are needed. The configuration should keep scope for backup copies of all disks.

10.4 Computer evaluation and measurement

Comparisons are often made among different computer systems on the basis of actual performance data. Using benchmark data, generated by using synthetic programs, is more effective than simply comparing technical specifications.

10.4.1 Benchmarking

A benchmark is the application of synthetic programs to emulate the actual processing work handled by a computer system. Benchmark programs permit the submission of a mix of jobs that are representative of the users projected workload. They also demonstrate data storage equipment techniques and provide the opportunity to test specific functions performed by the system. Through this technique, the limitations of the equipment become apparent early in the acquisition process. Sometimes user organizations will insist that the results be attached to the sales contract, formally stating that a specific number of transactions can be processed in a given period of time, the response to an inquiry will be given within a stated amount of time, and so forth.

Benchmarks can be run in virtually any type of system environment, including batch and online job streams, and with the users linked to the system directly or through telecommunications method.

Common benchmarks are the speed of the central processor, with typical instruction executed in a set of programs, as well as multiple streams of jobs in a multiprogramming environment. The same benchmark run on several different computers will make apparent any speed and performance differences attributable to the central processor.

Benchmarks also can be centered around an expected language mix for the programs that will be run, a mix of different types of programs, and applications having widely varying input and output volumes and requirements. Their response time for sending and receiving data from terminals is an additional benchmark for the comparison of systems.

Sometimes, rather than running actual benchmark jobs on computer systems, systems simulators are used to determine performance differences. In commercial systems simulators, the workload of a system is defined in terms of, say, how many input and output operations there are, how many instructions are utilized in a computation, and the order in which work is processed. The specifications are fed into a simulator that stores data about the characteristics of particular equipment (such as instruction speed, channel capacity, and read – write times). The simulator in turn processes the data against the operating characteristics and prepares a report of the expected results as if the actual computer were used. Then the system characteristics can be changed to mimic another model of computer and a new set of performance data produced for comparison. The time and expense of running actual benchmark programs on a computer are of concern to analyst and vendor alike. Thus, the use of commercial simulators is an attractive alternative.

10.4.2 Design of Synthetic Programs

A synthetic job is a program written to exercise a computer's resources in a way that allows the analyst to imitate the expected job steam and determine the results. Then the artificial job stream can be can be adjusted and rerun to determine the impact. The process can be repeated, as many times as necessary to see which tasks a comparison set of computer handles well and which they do not handle as well.

The synthetic jobs can be adjusted to produce the same type of activity as actual programs, including perhaps random access of files, sequential searching of files with varying size records, input and output activities, and file accessing in varying random patterns. The types of hardware and software features that are often simulated are listed in Table 10.1.

HARDWARE	SOFTWARE
CPU processing speed. Memory access speed. Interrupt handling abilities. Peripheral channel speed. Printer speeds. Seek time for magnetic disk. Rotational delay for magnetic disk. Communication speeds.	Scheduling algorithm. Compilation algorithm. Code efficiency. Virtual storage management algorithm File handling efficiency. Interrupt handling. Indexing methods. Multiple buffer handling. Communication processing procedure

Table 10.1 Representative benchmarks for hardware & software.

10.4.3 Comparison of Benchmarks

Although some comparison on the basis of equipment performance is better than no comparison at all, there are drawbacks to the use of benchmarks, first of all, the comparisons are made on purely quantitative grounds. They do not relate the learning time needed to become accustomed to the system or the quality of the systems software (such as the quality of the diagnostics prod`uced during compilation or the efficiency of the object code produced).

In addition, benchmarks do not provide reasonable assurances that programs currently being used on an existing system can be converted to the new system or that the new machine will run them efficiently even if they are converted. Vendors may also make sales claims that a specific system can handle additional tasks that another system cannot. Since benchmarks cannot directly verify these claims, the purchaser may insist that statements of certain sales claims be attached in writing to the sale contract.

10.4.4 Plug – Compatible Equipment

For reasons of cost, analysts frequently consider using equipment for a particular make of computer that is not manufactured by the computer vendor. Such components are called plug-compatible equipment. Some companies specialize in manufacturing systems components, such as printers, disk drives, or memory units that can be connected to a vendor's system in place of the same equipment manufactured by the vendor. The central processing unit does not care or know that the equipment is not the same make.

The benefit of plug – compatible equipment is the lower cost of an item compared with one produced by a major computer vendor. Because firms specializing in specific components can develop manufacturing expertise or are likely to have a smaller investment in research and development – they are duplicating components developed by another firm – they are able to offer the same product at a lower cost.

Although there is a large market for plug-compatible equipment because of price differences, the analyst must ensure that the equipment will meet necessary quality levels, that it will perform as well as (or possibly better than) the original equipment, and the computer vendor will not disallow warranties and service agreements on the rest of the system. There is a danger that some service people employed by the vendor will blame malfunctions on the "foreign" agreements on maintenance responsibilities and methods for resolving possible disputes about malfunction.

10.4.5 Financial Factors

The acquisition of and payment for a computer system are usually handled through one of three common methods: rental, lease, or purchase. Determining which option is appropriate depends on the characteristics and plans of the organization at the time the acquisition is made. No one option is always better than the other. (Table 10.2 summarizes the features of each method of acquisition.)

10.4.5.1 Rental

Computer rental is for the short – term use of a system, generally form 1 to 12 months. Each month a payment is made for the use of the equipment. Both the user and supplier have the option of canceling the rental with advance notice, usually 30 or 60 days ahead of the termination date.

Because the commitment is short-term, the renter has a great deal of flexibility. The decision to purchase a system can be delayed until financing is adequate, until a new generation of equipment is available, or until such time as the organization wishes, for whatever reason. Flexibility can be particularly important when an organization is experiencing planned rapid growth and will outgrow a specific system in a brief period, when important reorganizations of divisions and departments that will affect computing resources are in progress, or when the enterprise is in a period of dynamic change.

Compared with other acquisition methods, rental is the most expensive. Monthly payments are higher, and the organization does not receive any tax or ownership benefits, other than deduction of the monthly rental as a business expense. The equipment received is often used, although the rental agreement should be written in such a way that the renter is assured of having a system that runs properly and the will be maintained adequately. The short – notice cancellation provision may not provide enough security for the renter to plan on the continued availability of the system. For this reason, rental is typically a short-term solution that is appropriate perhaps while awaiting the official announcement and delivery of a new system. Many firms refuse to tie up capital or

equipment for short-term rentals. The analyst must ensure that rental systems are even available before making such a decision, since not all suppliers offer short – term rentals.

10.4.5.2 Lease

As lease is a commitment to use a system for a specific time, generally from three to seven years. Payments are predetermined and do not change throughout the course of the lease. Depending on the terms of the lease, payments are monthly, quarterly, semi-annual, or annual and include the cost of equipment service and maintenance. At the end of the lease period the lessor generally does not own the equipment. (If that is not the case, and the equipment becomes the property of the lessor, the Internal Revenue Service considers the agreement a conditional sale and the entire transaction must then be treated as a purchase.)

Compared with rental, leasing is less expensive. Because there is a longer commitment, the supplier will generally provide better service and the user can count of having the system available for use. Leasing protects against technical obsolescence, always a concern when purchasing computer equipment. If the lease term is short, the lessor can upgrade to a more powerful system even though the lease has not expired, providing the system is acquired form the same manufacturer.

No capital investment is required to lease a computer system. Leasing offers specific tax advantages. In addition to deducting the cost of the lease as a business expense, tax credits are sometimes business pays. In some case, the title for the equipment can even be passed to the lessor. Legal assistance is needed to investigate the current terms and conditions allowed by the Internal Revenue Service at the time such a transaction is considered.

10.4.5.3 Purchase

The ownership of computers through outright purchase is he most common method of computer acquisition and is increasing in popularity as lease costs rise. Over

time, the purchase option frequently costs the least, especially in light of the tax advantages that can some – times be gained.

Under purchase, the organization takes title to the equipment. Of course, the money for the purchase must be taken from operating funds or borrowed. And, in a sense the organization is locked in to the system it purchases, since changing to a different computer system is more difficult; either the system must be sold or arrangements must be negotiated to trade it in on a different computer.

The organization must acquire its own maintenance services (for parts and labor), usually from the manufacturer, and pay the monthly charges, which fluctuate from year to year. In addition, if the equipment was financed, payment on the loan must be made periodically. The cash outflow still may be lower than with renting or leasing, depending on the terms arranged by the purchaser. In return for the outgoing cash, purchase offers specific tax advantages:

1. The monthly maintenance charges are deductible as a business expense.
2. Interest on any loan to finance the purchase is deductible as a business expense.
3. The cost of the equipment can be depreciated over time; this also lowers the taxable income and therefore the income taxes paid.
4. Local, state, and federal taxes paid on the purchase may be deductible from income taxes.

The purchase option indicates the use of depreciation to reduce taxes. In a sense then, depreciation deductions on income tax reduce the cost of the computer to the organization. Normally, this benefit is not possible under lease agreements and it is never feasible for short – term rentals. Of course, the tax benefits described apply only to firms that operate for profit. Non profit firms that do not pay income taxes thus do not receive tax benefits form computer purchase.

Table 10.2 Comparison of Computer Systems Financing Options

Method of acquisition	Advantages	Disadvantages
Rental	Short – term commitment. High level of flexibility. Does not require cash up front.	Most expensive option. Little control of equipment change. Not all vendors will rent.
Lease	Predetermined payments for fixed period. Does not require cash up front. Usually better service from vendor than under rental. Little risk of obsolescence. Less expensive than rental	More expensive than purchase. May have limitations on hours of equipment use.
Purchase	Least cost in long run. Distinct tax advantages if a profit – making firm. A business investment. Full control over equipment use.	Risk of obsolescence. Permanent commitment. Full responsibility for all problems. Greater early cash requirements than other options.

1.

10.5 Maintenance and Support

An additional factor in hardware decision concerns the maintenance and support of the system after it is installed. Primary concerns are the source of maintenance, terms, and response times.

a. Maintenance Source

Once the system is delivered and installed, there is a brief warranty period during which time the sales unit is responsible for maintenance. This is typically a 90-day period, although the specific terms fare subject to contract negotiation. After that time, the purchaser has the option acquiring maintenance from various sources.

The most common source of maintenance for new equipment is the firm from which it was purchased. If a mainframe or minicomputer system is purchased through the manufacturer's sales force, there is also generally a maintenance support group that provides service for a standard price. Large companies set national maintenance costs that are adjusted on an annual or semi-annual basis. If the system is a microcomputer or personal computer, the dealer generally provides maintenance as a chargeable service. The buyer may pay a lower purchase price for personal computer purchased form mail-order houses, but may lose the convenience of local service. Lower service costs are one reason some mail-order firms are able to offer lower purchase prices.

Service is also available from companies specializing in providing maintenance service. Third party maintenance companies as these firms are called, frequently provide service in smaller communities, where manufacturers do not find it cost-effective to maintain offices. In addition, sellers of turnkey systems, who deliver and install working hardware and software combinations but do not manufacture the equipment themselves, suggest the use of specific third – party maintenance firms with whom they work directly and inform of changes in hardware, software, and suggested maintenance procedures. Sales organization, the purchaser may have no choice but to use a third-party maintenance firm. Many manufacturers do not service equipment they did not sell.

Terms

In formulating a maintenance agreement, the terms of the agreement are as important as the cost. The contract may be written to cover both labor and parts (all parts, regardless of the number needed or their cost), labor and an allowance for parts, or labor only, with parts charges added on as needed. The type of contract desired depends on the expenditures the organization is willing to make in comparison with how frequently it estimates service will be required. The labor and parts form is the most common type of contract for large systems.

The analyst should also consider how maintenance costs would change. Large manufactures have established policies of adjusting their maintenance charges on an annual or semiannual basis and frequently will not change these policies for any customer. Other suppliers and service companies offer open – ended contracts that allow the adjustment of charges at any time with 30 days notice. Frequently, analysts negotiating service with these companies will seek a cap on maintenance; that is, they will seek agreement, in writing, that the maintenance costs will not increase by any more than a stated maximum amount during a specific period, such as a calendar year. This type of protection ensures that the supplier cannot take advantage of the user who is totally dependent on the service agency. Most service companies are very reputable, but good business practice dictates that adequate protection always is sought in contracting for services.

10.5.1 Service and Response

Maintenance support is useful only if it is available when needed. Two concerns in maintenance are the response time when service is requested and the hours of support.

When a telephone call is placed for emergency maintenance, will a technician or engineer be dispatched immediately? That may be unlikely. However, the user has right to expect a reasonable response time after making an emergency call. Organizations often specify in the contract that the response to a telephone call must be made within 2 hours. Others specify same-day response, and still others accept response no later than the next morning. The degree of dependency the user organization has on the computer system will dictate how these terms are negotiated. An online system that is in use 24 hours a day will need a much quicker response that one that is used intermittently for batch processing.

When desktop computers are in use, an alternative to on-site support is available in the form of carry-in service: The user delivers the computer to the dealer or maintenance agency for repair. Often, service while-you-wait or same-day service is possible. For problems requiring longer repair times, a rental system may be available.

Repair service is often provided only during normal working hours. If an organization wishes evening service or around – the-clock coverage, it is usually available for an extra charge, say, from 10 percent to 50 percent additional cost.

However, equally important is the need for performing preventive maintenance, the routine service of cleaning and adjusting the equipment to prevent breakdowns. Whenever contracting for maintenance, a schedule of preventive maintenance must be agreed on in advance. Information about manufacturers suggested preventive maintenance cycles and procedures should be filed in the systems department and included in service agreements.

In all instances, the stocking of sufficient spare parts is important, since good service is impossible if spare parts are not available. User organizations should obtain sufficient assurances about adequate parts inventories in advance of need.

10.5.2 Options to In-House Systems

Less common options for computer support include the use of service bureaus or facilities management companies. A service bureau is a company that owns computer facilities and makes them available to users for a charge. The user submits data for processing, which is performed at the service bureau on the bureau's computer systems. In some cases, organizations interact directly with the computer though terminals located in users offices. There is usually a monthly cost plus a charge that varies according to the amount of time the user is in communication with the system. Additional fees may be charged for storing data, mounting magnetic disks and tapes, or printing pages.

Some service bureaus provide data processing service. The bureau prepares the data for input, handles all processing, and may even provide pickup and delivery service. Custom programming is available for charge.

The use of service bureaus is very common in accounting and payroll applications. Often, firms that want automatic data processing services in these areas but that do not want to purchase equipment or hire systems personnel will contract with a service bureau. However, as computer costs continue to drop and high – quality commercial software is available the reliance of some firms on service bureaus may change.

Facilities management companies provide a service to companies that wish to develop information systems capabilities but that prefer not to maintain a staff of operators, analysts, and programmers. Under this option, the user organization may purchase a computer system and then contract with a facilities management firm to operate the computer and provide service on the organization's premises. The facilities management company provides the information systems expertise and personnel for a

fee. It also develops software or acquires commercial software to meet the organization needs.

Through facilities management, an organization can obtain professional information processing and service without investing time and resources in managing a systems staff, while still receiving the benefits of owning a computer system.

10.6 Vendor Selection

This step determines the "winner" among the list of vendors available. The vendor with the best combination of reputation, reliability, service record, training delivery time, lease / finance terms & conversion schedule is selected. Initially a decision is made as to which vendor to contact. The sources available to check on vendors include:

1. Users
2. Software houses
3. Trade Associations
4. Universities
5. Publications
6. Vendor software list
7. Vendor referral directories
8. Published Directories
9. Consultants
10. Industry Contacts
11. Vendor's annual financial statement.

After this data is gathered about a vendor, it is matched with the selection criteria. Few of the selected vendors are invited to give presentation of their system. The system chosen goes though contract negotiations before implementation.

10.7 Software Selection

Software selection is a critical aspect of system development. The search starts with the software, followed by the hardware. There are two ways of acquiring software: custom – made or "off – the –shelf" packages. Today's trend is toward purchasing packages, which represent roughly 10 percent of what it costs to develop the same in house. In addition to reduced cost, there are other advantages:

1. A good package can get the system running in a matter of days rather than the weeks or months required for “home-grown” packages.
2. MIS personnel are released for other projects.
3. Packages are generally reliable and perform according to stated documentation.
4. Minimum risks are usually associated with large – scale systems and programming efforts.
5. Delays in completing software projects in house often occur because programmers quit in midstream.
6. It is difficult to predict the cost of “home-grown” software.
7. The user has a change of seeing how well the package performs before purchasing it.

There are drawbacks, however, to software packages:

1. The package may not meet user requirements adequately.
2. Extensive modification of a package usually results in loss of the vendor’s support.
3. The methodology for package evaluation and selection is often poorly defined. The result is a haphazard review based on a faulty process or questionable selection criteria.
4. For first – time software package users, the overall expectation from a package is often unclear and ill defined.

It can be seen, then, that the quality of a software package cannot be determined by price alone. A systematic review is crucial.

10.8 Criteria for Software Selection

Prior to selecting the software the project team must set up criteria for selection. Selection criteria fall into the categories described here.

Reliability.

It is the probability that the software will execute for a specified time period without a failure, weighted by the cost to the user of each failure encountered. It relates to the ease of recovery and ability to give consistent results. Reliability is particularly important to the professional user. For example, a pharmacist relies on past files on patients when filling prescriptions. Information accuracy is crucial.

Hardware may become inoperative because of design errors, manufacturing errors, or deterioration caused by heat, humidity, friction, and the like. In contrast, software does not fail or wear out. Any reliability problems are attributable to errors introduced during the production process. Furthermore, whereas hardware failure is based largely on random failures, software reliability is based on predestined errors.

Although reliable software is a desirable goal, limited progress has been made toward improving it in the last decade. The fact of unreliable software had led to the practice of securing maintenance agreements after the package is in operation. In a sense, unreliability is rewarded.

Software reliability brings up the concept of modularity, or the ease with which a package can be modified. This depends on whether the package was originally designed as a package or was retrofitted after its original development for single installation use. A package with a high degree of modularity has the capacity to operate in many machine configurations and perhaps across manufacturers' product lines.

With modularity come expandability, which emphasizes the sensitivity of a software package to handle an increased volume of transaction or to integrate with other programs. The following questions should be considered:

1. Is there room for expanding the master file?
2. How easily can additional fields and files be added?
3. Are there errors a user can make that will ring down the system?
4. How much of the system becomes unusable when a part of it fails?
5. What are the recovery capabilities?

b. Functionality

It is a definition of the facilities, performance, and other factors that the user requires in the finished product. All such information comes from the user. The following are key questions to consider:

1. Do the input transactions, files, and reports contain the necessary data elements?
2. Are all the necessary computations and processing performed according to specifications?

1. Capacity

ii.Capacity refers to the capability of the software package to handle the user's requirements for size of files, number of data elements, volume of transactions and reports and number of occurrences of data elements. All limitations should be checked.

Flexibility

It is a measure of the effort required to modify an operational program. One feature of flexibility is adaptability, which is a measure of the ease of extending the product.

Usability

This criterion refers to the effort required to operate, prepare the input, and interpret the output of a program. Additional points to be considered are portability and understandability. Portability refers to the ability of the software to be used on different hardware and operating systems. Understandability means that the purpose of the product is clear to the evaluator and that the package is clearly and simply written, is free of jargon, and contains sufficient references to readily available documents so that the reader can comprehend advance contents.

Security.

It is a measure of the likelihood that a system's user can accidentally or intentionally access or destroy unauthorized data. A key question is how well can one control access of software or data file? Control provides system integrity.

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 loads. Each package should be evaluated for acceptance on the user's system.

The language in which a package is written and the operating system are additional performance considerations. If we plan to modify or extend a package, it is easier if it is written in a language that is commonly known to programmers. Likewise, if the package run only under a disk operating system and the installation is under a full operating system, then either the package will have to be upgraded to the larger operating system or the system downgraded to handle the package as is. In either case, the change could be costly and counterproductive.

Serviceability.

This criterion focuses on documentation and vendor support. Complete documentation is critical for software enhancement. It includes a narrative description of the system, system logic and logic instructions. Vendor support assures the user adequate technical support for software installation, enhancements, and maintenance, the user should determine how much on – site technical assistance is provided by the vendor, especially during the first few weeks after the installation.

The user expects on – site training and support as part of most commercial packages. It is vital to inquire about the amount of training provided. The user may require training at several levels--clerical, operations, programming, and management.

Ownership

Who owns the software once it is "sold" to the user? Most of the standard license agreement forms essentially lease the software to the user for an indefinite time. The user does not "own" it, which means that the source code is inaccessible for modification, except by the vendor. Many users enter into an escrow arrangement whereby the vendor deposits code to the user if the vendor goes out of business or is unable to perform the services specified in the license.

In acquiring software, several questions should be asked:

1. What rights to the software is the user buying?
2. Can the user sell or modify the software?
3. If the vendor is modifying the package especially for the user, can the vendor sell it to other within the same industry the user is in?
4. What restrictions are there to copying the software or documentation?

Minimal costs.

Cost is a major consideration in deciding between in – house and vendor software. Cost – conscious users consider the following points:

1. Development and conversion costs.
2. Delivery schedule.
3. Cost and frequency of software modifications.
4. Usable life span of the package.

10.9 Performance Evaluation

Evaluating a system includes the hardware and software as a unit. Hardware selection requires an analysis of several performance categories.

1. System availability. When will the system be available?
2. Compatibility. How compatible is the system with existing programs?
3. Cost. What is the lease or purchase price of the system? What about maintenance and operation costs?
4. Performance. What are the capacity and throughput of the system?
5. Uptime. What is the 'uptime' record of the system? What maintenance schedule is required?
6. Support. How competent and available is the vendor's staff to support the system?
7. Usability. How easy is it to program, modify, and operate the system?

For the software evaluation, the following factors are considered:

1. The programming language and its suitability to the application(s).
2. Ease of installation and training.
3. Extent of enhancements to be made prior to installation.

In addition to hardware/software evaluation, the quality of the vendor's services should be examined. Vendor support service include the following:

1. Backup. Emergency computer backup available from vendor.

2. Conversion. Programming and installation service provided during conversion.
3. Maintenance. Adequacy and cost of hardware maintenance.
4. System development. Availability of competent analysts and programmers for system development.

10.10 Summary

A major element in building systems is selecting compatible Hardware & software. The kind of hardware & 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. Using benchmark data, generated by using synthetic programs, is more effective than simply comparing technical specifications. Software is classified as system software for controlling computer operations and application software for solving user oriented problems. There are several thing to do before selection :

- Define system capabilities that makes sense for the business
- Specify the magnitude of the 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

vendor proposals are evaluated and finalised ad hoc by scoring the characteristics of each system. There are three method of acquisition : Rental , lease , and purchase.

10.11 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.