SYSTEM DEVELOPMENT LIFE CYCLE METHODOLOGY

2.1 SYSTEMS DEVELOPMENT PROCESS: Systems development is a process of examining a business situation with the intent of improving it through better procedures and methods. Two major components of system development are:

- System Analysis: It is the process of gathering facts, diagnosing problems, and recommending improvements to the system.
- System Design: It plans a new business system or to replace an existing system.

2.1.1 Achieving System Development Objectives: There are many reasons why organization fails to achieve system development objective and some of them are as follows: [**NRI Lacks Salman**]

- **New technologies**: With advanced technology achieving system development objectives is more difficult because users would not be familiar with that technology.
- **Resistance to change**: People have a natural tendency to resist change. When employee perceives that the project will result in personnel cutbacks, then the development project is doomed to failure.
- **Inadequate testing and user training**: New systems must be tested before installation for its operation and users must be trained effectively for the new system.
- **Lack of senior management support** and involvement in information systems development: Employees and Developers follow those project where senior management are involved because they consider those projects as important and act accordingly by shifting their efforts to those project and therefore other project receive less attention.
- **Lack of standard project management** and systems development methodologies Some organizations do not formalize their project or system development methodologies thereby making it very difficult to complete projects on time or within budget.
- **Lack of user participation** Users must participate in the development effort, must feel ownership for project success, etc. This will reduce resistance to change.
- **Shifting user needs**. User requirements are many and keep on changing. So it becomes difficult for developers to develop system as originally stated.

2.1.2 System Development Team: Several people in the organization are responsible for systems development. In large systems, the worth of a project is typically decided by a top management level steering committee. The steering committee ensures that systems development activities are consistently aimed at satisfying the requirements of managers and users within the organization. System analysts are subsequently assigned to determine user requirements, design the system and assist in development and implementation activities. In end-user developed systems, the end-user is ultimately responsible for the system. Generally, the end-user seeks guidance from information centre personnel while developing the system.

2.2 SYSTEMS DEVELOPMENT METHODOLOGY: A system development methodology is a formalized, standardized, documented set of activities used for system development project. Each methodology is best suited to specific kinds of projects. The methodology is characterized by the following: [**TDTDPP - Total Dil To Dil Policy & Procedure available**]

- The system must be **tested** thoroughly prior to implementation to ensure that it meets users' needs.
- Each project is **divided** into a number of identifiable processes and each process has a starting and an ending point. The division of the project facilitates both project planning and project control.
- User must be provided with adequate **training** of the new system.
- Specific reports and other documentation called **Deliverables** are generated periodically during system development so that execution of system development is proper.
- Users, managers must **participate** in the project to provide approvals for system developed which are often called signoffs.
- A **post-implementation** review must be performed to assess the effectiveness and efficiency of the new system.

Approaches to System Development: Since organizations vary significantly in the way they automate their business procedures and since each new type of system differs from the other, several different system development approaches are used within an organization. All these approaches are not mutually exclusive, which means that it is possible to perform some prototyping while applying the traditional approach. These approaches are as follows:

1. **Waterfall:** Linear framework type
2. **Prototyping:** Iterative framework type
3. **Incremental:** Combination of linear and iterative framework type
4. **Spiral:** Combination linear and iterative framework type
5. **Rapid Application Development (RAD):** Iterative Framework Type
6. **Agile Methodologies**

2.2.1 The Traditional / Waterfall Approach / Sequential Approach: The waterfall approach is a traditional development approach in which each developer in a development team works in different phases. It is used on small projects because it eliminates testing to identify problems early in the process. In the traditional approach of system development, activities are performed in sequence (i.e., step by step). Unless earlier step is completed next step is not undertaken under traditional approach.

Framework type: Linear

Basic Principles: [**DEC – deck**]

- Project is **divided** into sequential phases, with some overlap and splash back acceptable between phases.
- **Emphasis** is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.
- Tight **control** is maintained over the life of the project through the use of extensive written documentation.

Strengths: [**COMIc**]

- **Conserves** resources.
- An **orderly sequence** of development steps and design ensures the quality, reliability, etc of the developed software.
- Progress of system development is **measurable**.
- **Ideal** for less experienced project teams or where a composition team fluctuates.

Weaknesses: [**SFLPCE – Slow Forwarding of Little Problem makes Excessive Changes**]

- It is **slow**, costly, and cumbersome due to tight controls.
- Project progresses **forward** with little scope for backward.
- **Little** room for use of iteration, which used will reduce manageability.
- **Problems** cannot be discovered due to lack of system testing.
- **Excessive** documentation at each stage is time-consuming and also difficult for the users to read it regularly.
- Difficult to respond to changes. **Changes** occurring later in the system are more costly and hence discouraged.

2.2.2 The Prototyping Model: The goal of prototyping approach is to develop a small or pilot version called a prototype of part of a system. A prototype is built quickly at a lesser cost and with the intention of being modifying or replacing it. As users work with the prototype, they make suggestions for improvements which are then incorporated and finally a prototype is developed which is turned into the final system or it is scrapped.

Framework type: Iterative.

Basic Principles: Prototyping can be viewed as a series of four steps:

Step 1 - Identify Information System Requirements: Under prototype approach, the design team needs only fundamental system requirements to build the initial prototype i.e., initial requirement of users.

Step 2 - Develop the Initial Prototype: In this step, the designers create an initial base model and emphasizes on system characteristics such as simplicity, flexibility, and ease of use which enables users with data entry display screens, menus, input prompts, and source documents.

Step 3 - Test and Revise: After finishing the initial prototype, the designers demonstrate the model and then give it to users and ask them to record their likes and dislikes. Using this feedback, the design team modifies the prototype and then resubmits the revised model. Thus process continues until the users are satisfied.
Step 4 - Obtain User Signoff of the Approved Prototype: At the end of Step 3, users formally approve the final version of the prototype, which commits them to the current design and establishes a contractual obligation about what the system will, and will not, do or provide.

Strengths: [**IQ DEKHI**]

- Easily **identify** confusing or difficult functions and missing functionality.
- Provides **quick implementation** of incomplete, but functional application.
- Users' needs and requirements are **defined** properly because it requires extensive user involvement in system development.
- Since users of the system experiment with each version of the prototype, **errors** are detected and eliminated early in the developmental process. As a result, the system should be more reliable and less costly to develop.
- **Knowledge** gained in an early iteration is used in development of later iterations.
- Encourages **innovation** and flexible designs.

Weaknesses: [**BAD-QR i.e., BAD ?R i.e. bad questionnaire**]

- Prototyping may cause **behavioral problems** with system users if system developers are unable to meet all user requirements as well as users will be impatience when they have to go through too many interactions of the prototype.
- **Approval** process and control are not strict.
- To make prototype successful users should be willing to **devote** their significant time for experimenting with the prototype and provide suggestions for improvement which lacks.
- Designers design prototype **quickly** without analyzing users' requirements sufficiently resulting in an inflexible design further limiting future system potential.
- Users **requirements** changes frequently.

2.2.3 The Incremental Model

Framework Type: It is a combination of Linear and Iterative.
Basic Principles: In Incremental model, model is designed, implemented and tested incrementally (a little more is added each time) until the product is finished. Here project is divided into parts and each part is built separately. Each part is delivered to the client for review when it is complete. Project is defined as finished when it satisfies all the users' requirements.

Strengths: [**KFMCG – Knowledge of FMCG sector is must**]

- **Knowledge** gained in an early iteration is used in development of later iterations.
- It is more **flexible** and less costly to change scope and requirements.
- Helps to **mitigate** integration and architectural risks earlier in the project.
- **Concrete** evidence of project status can be given throughout the life cycle.
- **Gradual implementation** provides the ability to monitor the effect of incremental changes and make adjustments before the organization is negatively impacted.

Weaknesses: [**MP3**]

- Some **modules** may be completed earlier than others and so well-defined interfaces are required.
- Each **phase** of an iteration is rigid and do not overlap each other.
- **Problems** relating to system architecture design may arise because all of users' requirements for entire system are not obtained all together.
- System designers may have tendency to **push** difficult problems for future for demonstrating early success to management.

2.2.4 Spiral Model

Framework Type: It is a combination Linear and Iterative.

Basic Principles: Also known as the Spiral Lifecycle, it is a Systems Development Method (SDM) which combines the features of the prototyping model and the waterfall model. The spiral model is intended for large, expensive and complicated projects. It combines advantages of top down and bottom up concepts.

- The new system requirements are defined in as much detail as possible which involves interviewing a number of external or internal users.
- A preliminary design is created for the new system where all possible alternatives that can help in developing a cost effective project are analyzed and decided. This phase identify and resolve all the possible risks. If risks indicate any kind of uncertainty, prototyping may be used to proceed with the available data and find out possible solution.
- A first prototype of the new system is constructed and tested by designer himself and know new prototype is prepared with additional functions based on knowledge obtained from earlier prototype and all this steps are repeated again and again until final product is ready.
- A second prototype is evolved by a fourfold procedure :
  evaluating the first prototype in terms of its strengths, weaknesses, and risks;
  defining the requirements of the second prototype;
  planning and designing the second prototype;
  constructing and testing the second prototype.

Strengths: [**ESI**]

- **Enhance** risk avoidance.
- Useful in **selecting** best methodology for development of software based on project risk.
- Can **incorporate** Waterfall, Prototyping, and Incremental methodologies in the framework as special case and provides guidance as to which combination will give best software iteration based on project risk.

Weaknesses: [**EICC**]

- Requires **experienced** project manager who can determine which methodology to apply in given project.
- No firm deadlines, so there is an **inherent risk** of project not meeting budget or schedule.
- To determine exact **composition** for development of iteration is a challenge.
- Highly **complex** for each project and therefore limiting reusability.

2.2.5 Rapid Application Development (RAD)

Framework Type: Iterative.

Basic Principles: It is a type of software development which uses minimum planning for planning a prototype which results in development of software much faster and makes it easier to change requirements.

- Key objective is to develop software fast with high quality at relatively low cost,
- Attempts to reduce inherent project risk by dividing a project into smaller segments.
- Aims to produce high quality systems quickly, primarily through the use of iterative Prototyping, along with active user involvement, etc.
- Key emphasis is on fulfilling the business need.
- Active user involvement is compulsory i.e., it includes Joint Application Development (JAD).
- Produces production software as opposed to a throwaway prototype.
- Produces documentation for future development and maintenance.

Strengths: [**SPCI-RICO i.e., special reco A/c**]

- **Saves** time, money and human effort.
- System is **produced** more quickly with lesser cost.
- Rapidly **change** system design as per requirement of users.
- Quick initial **reviews** are possible.
- Constant **integration** isolates problems and encourages customer feedback.
- **Concentrates** on development of system elements from user point of view.

- The **operational version** is available much easier as compared to Waterfall, Incremental, or Spiral frameworks.

Weaknesses: [**QEIDMPS - ? all EMI's are PaiD**]

- More speed and low cost leads to a lower system **quality**.
- Project will **end up** with more requirements than needed.
- Some **modules** may be completed earlier than others and so well-defined interfaces are required.
- **Inconsistent** system is developed because system is produced quickly.
- System designers may have tendency to **push** difficult problems for future for demonstrating early success to management.
- **Difficult** to reuse module in future systems.

2.2.6 Agile Methodologies: It is based on iterative and incremental approach where requirements and solutions evolve through collaboration between self organizing and cross functional teams. It promotes adaptive planning, evolutionary development and delivery and encourages rapid and flexible response to change. It is a conceptual framework that promotes interactions throughout the development of life cycle.
Basic Principles: Following are the key principles of this methodology:

- Customer satisfaction by rapid delivery of software,
- Changes in the requirements are welcomed even late in development,
- Working software is delivered frequently (weeks rather than months),
- Sustainable development i.e., able to maintain constant pace,
- Daily co-operation between users and developers,
- Face to face conversation which is the best form of communication,
- Continuous attention to technical improvement and good design,
- Simplicity, and
- Regular adaption to changing circumstances.

Strengths: [ADFDH- DAD ask How did you Failed?]

- The team **does not** have to invest time and effort and finally find that the requirement of the user has changed by the time they deliver the product.
- Agile methodology has the concept of **adaptive** team which responds to the changing requirements,
- The **documentation** is crisp but to the point to save time,
- The end result is the **high quality** software in least possible time which results in satisfied user,
- **Face to face** communication and continuous inputs from user representative leaves no space to decide result by guessing.

Weaknesses: [TELL tO Isca]

- Agile increases potential **threats** to business continuity because they have less documentation as they focus on verbal communication rather than documents.
- In some of the software deliverables especially large ones, it is difficult to assess the **efforts** required at the beginning of the software development.
- There is **lack** of emphasis on necessary designing and documentation.
- Due to **lack** of long term planning, re-work is often required on Agile projects when the various components of the software are combined and forced to interact.
- The project can easily get **off track** if the representative is not clear about what they want.
- Agile lacks attention for outside **integration** because they do not invest time in identifying integration points in advance and therefore need for an integration point can become last minute surprise.

2.3 SYSTEM DEVELOPMENT LIFE CYCLE (SDLC): It is a framework which provides system designers and developers to follow a sequence of activities. It consists of steps in which each phase of the SDLC uses the results of the previous one. The SDLC is document driven. A phase of the SDLC is not complete unless a proper document is prepared. This feature of the SDLC is critical for the successful management of an IS project.
The advantages of this system are as follows: [**HQDP- Helps sdlc in Quality Documentation Phase**]

- **Helps** in better planning and control for project managers.

- Ensure better **quality** because it has to comply with set standards.
- **Documentation** is an important feature which is an important measure for communication and control.
- **Phases** are important milestones as it helps the project manager and user to review the system and for signoff.

From the perspective of the IS Audit, the following are the possible advantages: [**DR –Etc**]

- Based on **document** of each phase, IS auditor can have clear understanding of the various phases of the SDLC
- Based on his examination, he can **report** whether IS managers have complied with standards set by management, if any.
- The auditor can **evaluate** methods and techniques used in various development phases of SDLC.
- If IS Auditor has a **technical knowledge**, he can guide various phases of SDLC.

Risks Associated with SDLC: Some of the shortcomings of the SDLC are as follows:

- Development team may find it difficult to built
- The users may find that the end product will not sustain for a long time.
- Rigidity in approach may prolong the duration of project
- It may not be suitable for small and medium sized projects.

PHASE NO	PHASE NAME	NATURE OF ACTIVITY
1	Preliminary Investigation	Determine and evaluate strategic benefits of the system and ensure that the solution fits the business strategy. Includes cost-benefit analysis of the proposed system.
2	Systems Requirements Analysis	Analyzing the system on the basis of the users requirements.
3	Systems Design	Design the system on the basis of the requirement by developing the system flowcharts, system and data flow diagrams, screens and reports.
4	Systems Development/Programming	Programming the system as designed and conduct the continuous testing and debugging
5	Systems Testing	Testing is conducted before the developed system is implemented.
6	Systems Implementation	After final testing and acceptance by management and user, new system is accepted.
7	Post Implementation Review and Maintenance	Continuous evaluation of the system as it functions in the live environment. Maintenance includes continuous evaluation of the system as it functions in the live environment and its updation.

2.4 THE PRELIMINARY INVESTIGATION

Objective: is to determine and analyze the strategic benefits to be derived after implementing the system. The steps involved in the preliminary investigation phase are as follows:

- Identification of Problem
- Identification of objective
- Delineation of scope
- Feasibility Study

The following issues are typically addressed in the Feasibility Study:

- Determine whether the solution is as per the business requirements.
- Determine whether the existing system can rectify the situation without a major modification.
- Define the time frame for which the solution is required.
- Determine the approximate cost to develop the system.
- Determine whether the vendor product offers a solution to the problem.

Document / Deliverable: A preliminary investigation report/ feasibility study for management.

2.4.1 Identification of Problem

The first step in an system development is to define the problem clearly after discussions with the user group. A problem which has a considerable impact on the organization is likely to receive immediate management attention. Thus to identify a problem system analyst is assigned to make a preliminary investigation. After investigation he submits all proposals for the given problem which might be must beneficial to the organization. Thus purpose of the preliminary investigation is to evaluate the project request. System analyst on preliminary investigation should accomplish the following objectives: [Qn: what does system analyst do?]

- Clarify and understand the project request.
- Determine size of the project.
- Determine alternative approaches.
- Assess costs and benefits of alternative approaches.
- Report findings to the management with recommendation for acceptance or rejection of the proposal.

2.4.2 Identification of Objective: After identifying the problem, system analyst must work out for the objectives of the proposed solution. For instance, inability to provide a convenient reservation system, for a large number of intending passengers was the problem of the Railways. So its objective was "to introduce a system wherein intending passengers could book a ticket from source to destination, faster than in real-time."

2.4.3 Delineation of Scope: The scope of a solution should be clear and comprehensible stating what solution will be provided to the users and what solution it will not provide. Outlining scope is essential right from the beginning.

The following questions should be answered while stating the scope:

- Functionality requirements: What functions will be delivered by solution?
- Data to be processed: What data will be required to achieve these functions?
- Control requirements: What are the control requirements for this application?
- Performance requirements: What level of response time, execution time is required?
- Constraints: What are the conditions the input data has to conform to?
- Interfaces: Does the application has to interface with any other hardware/software?
- Reliability requirements: Reliability means whether the application will remain uncorrupted from misuse.

While eliciting information to delineate the scope, few aspects need to be kept in mind:

- Different users will represent the problem and the solution in different ways. So, the system developer should take the need from the initiator as the base of the project.
- While the initiator of the project may be a member of the senior management but the actual users may be from the operating level. So, understanding his profile helps in designing appropriate system.
- While presenting the proposed solution for a problem, the development organization has to clearly quantify the economic benefits to the user organization. The information required has to be gathered at this stage.
- It is also necessary to understand the impact of the solution on the organization. Solutions which have a wide impact are likely to meet with greater resistance.
- While economic benefit is a critical consideration when deciding on a solution, there are several other factors that have to be given weight-age too. These factors have to be considered from the perspective of the user management and resolved.

Two methods that help to define of the scope of the project can be analyzed as follows:

(i) Reviewing internal documents: Before conducting the investigation the analysts must first learn about the organization involved in the project by examining organization charts and studying written procedures, etc. (ii) Conducting Interviews: Written documents tell the analyst how the systems should operate but does not give enough details for taking a decision. To learn these details analysts must conduct interviews which describes the requirement of the project.

2.4.4 Feasibility Study: After possible solutions are identified project feasibility is carried out i.e., whether the system will be useful for the organization is determined. A feasibility study is carried out by the system analysts to evaluate alternative systems so that the most feasible system can be selected for development. The following issues are typically addressed in the Feasibility Study:

- Determine whether the solution is as per the business requirements.
- Determine whether the existing system can rectify the situation without a major modification.
- Define the time frame for which the solution is required.
- Determine the approximate cost to develop the system.
- Determine whether the vendor product offers a solution to the problem.

The Feasibility Study of a system is evaluated under following dimensions: [**what are the different types of feasibility study?**]

- Technical: Is the technology needed available?
- Financial: Is the solution viable financially?
- Economic: Return on Investment?
- Schedule / Time: Can the system be delivered on time?
- Resources: Are human resources reluctant for the solution?
- Operational: How will the solution work?
- Behavioral: Is the solution going to bring any adverse effect on quality of work life?
- Legal: Is the solution valid in legal terms?

Technical Feasibility: It is concerned with the issues pertaining to hardware and software. An analyst analyze whether the proposed system is feasible with the existing hardware and software technology. The technical issues usually raised during the feasibility stage of investigation include the following:

- Does the necessary technology exist to do what is suggested (and can it be acquired)?
- Can the proposed application be implemented with existing technology?
- Will the proposed system provide adequate responses to inquires?
- Can the system be expanded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access, and data security?

Financial Feasibility: The solution proposed may be prohibitively costly for the user organization i.e., proposed solution should be within our cost

Economic Feasibility/Cost-Benefit Analysis: It includes an evaluation of all costs and benefits of the proposed system if implemented. The analysts must determine the amount of time and money required. The financial and economic questions raised by analysts during the preliminary investigation are for the purpose of estimating the following:

- The cost of conducting a full systems investigation.
- The benefits in the form of reduced costs or fewer costly errors.
- The cost if nothing changes (i.e., the proposed system is not developed)

Estimating costs and benefits: After possible solution options are identified, each solution should be analyzed based on costs and benefits. [**What are the costs – benefits, system analyst must determine after solution are identified for system development?**]

Cost: System costs can be sub divided into Development, Operational and Intangible costs.

- **Development costs** for a system include costs of the system development process such as - salaries of the system analysts and computer programmers; cost of converting and preparing data files; cost of preparing new computer facilities; cost of testing and documenting the system, cost of training employees, other start up costs; etc.
- **Operating costs** of a system include - salaries of computer operators and other data processing personnel; salaries of system analysts and computer programmers; cost of input data preparation; and Cost of maintaining proper physical facilities including power, light, heat, air conditioning, building, etc.

- **Intangible costs** are costs that cannot be easily measured. For example, the development of a new system may disrupt the activities of an organization and cause reduction in customer sales or loss of goodwill. Such costs are difficult to measure.

Benefits: The benefits which result from developing new or improved information systems can be divided into tangible and intangible benefits. Tangible benefits are those that can be accurately measured and are directly related to the introduction of a new system, such as decrease in data processing cost. Intangible benefits such as improved business image are harder to measure and define.

Schedule or Time Feasibility: Schedule feasibility involves estimating how long it will take a new or revised system to become operational and communicating this information to the steering committee. For example, if a project takes 16 months to become fully functional then it may rejected in favor of a simpler alternative which can be implemented in a shorter time frame.

Resources Feasibility: This focuses on human resources i.e., whether skilled personnel will be available to operate the proposed software solutions.

Operational Feasibility: It is concerned with ascertaining the views of workers, employees, etc about the use of computer facility. A system can be highly feasible in all respects except the operational and fails miserably because of human problems. Some of the questions which help in testing the operational feasibility of a project are stated below:

- Is there sufficient support for the system from the management as well as from the users?
- Are current business methods acceptable to users?
- Have the users been involved in planning and development of the project?
- Will the proposed system cause harm?
- Will individual performance be poorer after implementation than before?

Behavioral Feasibility: It refers to the systems which is designed, can produce the desired outputs. However, if the data input for the system is not readily available or collectable, then the system may not be successful.

Legal Feasibility: Legal feasibility is concerned with the conflict between a proposed system and the organization's legal obligations. Any system which violates the local legal requirements should also be rejected.

2.4.5 Reporting Results to Management: After the analyst identifies the problem and its scope, he provides alternative solution and estimates the cost and benefits of each alternative, and report the results of the same to the management. The report should recommend further procedures. From the analyst's report, management should determine what to do next i.e., which alternative to be selected or whether combination of two is required or to conduct analyze all over again, etc.

2.5 SYSTEM REQUIREMENT ANALYSIS

Objectives: This phase includes a thorough understanding of the current system, identification of the areas that need modifications, user requirements are analyzed and have fair idea about various systems development tools. The following activities are performed in this phase: [**CAG –MD i.e., Consult Analysis and Gather data to Model and Document activities**]

- To **consult** the stake owners to determine their expectations and resolve their conflicts.
- To **analyze** requirements and determine priorities.
- To **gather** data or find facts using tools like - interviewing, research/document collection, questionnaires, observation.
- To **model** activities such as developing models to document Data Flow Diagrams, E-R Diagrams.
- To **document** activities such as interview, questionnaires, reports etc. and development of a system (data) dictionary to document the modeling activities.

Document/Deliverable: A systems requirements report.

2.5.1 Fact finding Techniques: Every system is built to meet some set of needs. To assess such needs analysts must interact with the users who will be benefited from the system and also to determine what are

their actual requirements. Various fact-finding techniques are discussed below: [**IOQD – Interviewer Observe reaction During Quesstionnaire**]

- **Documents:** Document means manuals, input / output forms, diagrams of how the current system works. Documents are also for procedure manuals, program codes of the current system, etc. Documents are a very good source of information about the user needs and the current system.
- **Questionnaires:** Through questionnaires large amount of information can be obtained from variety of users quickly. If the questionnaire is skillfully drafted, responses can be rapidly analyzed with the help of a computer.
- **Interviews:** Users and managers should be interviewed to extract information in depth. The data gathered through interviews often provide complete picture of the problems and opportunities.
- **Observation:** Observation plays a central role in requirement analysis in prototype approaches. By observing how users react to prototypes of a new system, system can be successfully developed.

2.5.2 Analysis of the Present System: Detailed investigation of the present system involves collecting, organizing and evaluating facts about the system and the environment in which it operates. Survey of existing methods, procedures, data flow, outputs, files, input and internal controls should be intensive in order to fully understand the present system and its related problems.
The following areas should be studied in depth:

- **Review historical aspects:** A brief history of the organization is a starting point for an analysis of the present system. The system analyst should review what system changes have been occurred in the past and whether it was successful or not.
- **Analyze inputs:** A detailed analysis of present inputs is important since they are basic source of data. Analyze should also be conducted to determine what is source input of present system. Further analyze whether input of one area is the output of other. Analyst must also understand the flow of input.
- **Review data files maintained:** The analyst should investigate the data files maintained by each department, where they are located, who uses them number of times they are used, etc. He should also review all on-line and off-line files maintained in the organization.
- **Review methods, procedures and data communications:** Methods and procedures transform input data into useful output. A method is defined as a way of doing something. A procedure is a series of logical steps to accomplish a job. The analyst must analyze data-communications network in the present system to identify whether there is need to revise the network in new system or not.
- **Analyze outputs:** The outputs should be scrutinized carefully by the system analysts to determine how well it meets the organization's needs. The analysts must understand what information is needed and why. Many times information obtained from reports are a carry-over from earlier days which have little or no relevance and therefore such reports should be eliminated.
- **Review internal controls:** A investigation is not complete until internal control is reviewed because it identifies weaknesses that should be removed in the new system.
- **Model the existing physical system:** All the important items in the existing system should be reviewed and analyzed and this process must be properly documented which allows a thorough understanding of numerous details and problems in the current system.
- **Undertake overall analysis of present system:** It is the final phase of analysis which includes thorough analysis of present work volume; user requirements; etc.

2.5.3 Systems Analysis of Proposed Systems: After analyzing each functional area of the present system along with its strengths and short comings, proposed system should be clearly defined which should be in conformity with the project's objectives are as follows:

- Database maintained should have facility of accessing anytime i.e., online processing capabilities.
- Input data are prepared directly from original source documents for processing in the system.
- Methods and procedures that show the relationship of inputs and outputs to the database must utilize data communications where appropriate.
- Work volumes and timings should be considered carefully taking peak periods into consideration.

Output must be the starting point for compiling these specifications because after outputs are determined, it is possible to infer what inputs, database, methods, procedures and data communications must be employed. The output-to-input process is recommended since outputs are related directly to the objectives of the organization.

2.5.4 System Development Tools: Many tools and techniques have been developed to improve current information systems and to develop new ones. Such tools help end users and systems analysts to -

- Clarify, document and communicate the activities and resources involved in the organization;
- analyze present business operations, management decision, etc of the organization;
- Propose and design new or improved information systems to solve business problems.

The major tools used for system development can be grouped into four categories based on the systems features. These are as follows:

System components and flows: They help the analyst to document the data flow among the major activities of a system. Eg: System flow charts are used to show the flow of data among hardware devices. Data flow diagram uses a few simple symbols to illustrate the flow of data among external entities.

User interface: System analyst has to design the new system which can interface between end users and the computer system. Eg: Layout forms and screens are used to construct the formats and contents of input/output media and methods. Dialogue flow diagrams analyze the flow of dialogue between computers and people.

Data attributes and relationships: When data resources are defined, catalogued are designed by this category of tools.

- A Data Dictionary catalogs the description of the attributes (characteristics) of all data elements and their relationships to each other as well as to external systems.
- File layout forms document the type, size and names of the data elements in a system.
- Grid charts help in identifying the use of each type of data element in input/output or storage media of a system.

Detailed system process: These tools contain detailed procedures and processes to design a computer system. Eg: Decision trees and decision tables use a network or tabular form to document the complex conditional logic involved in choosing among the information processing alternatives in a system.

We will now describe some of these tools in detail.

A). Structured English: It also known as Program Design Language (PDL) or Pseudo Code which uses English language with the syntax of structured programming. The Structured English aims at getting the benefits of both the programming language and natural language i.e., it is a combination of English and technical statement. Structured English consists of the following elements:

- Operation statements written as English phrases are executed from top down.
- Conditional statements such as IF, THEN, and ELSE.
- Repetition statements such as DO, WHILE, and UNTIL.

B) Flowcharts: Flowcharting is a graphic technique used by analysts to represent the inputs, outputs and processes of a business in a pictorial form. They are used in analyzing, designing, documenting or managing a process or program in various fields.

[AS PER NEW MODULE BELOW ALL CONTENTS OF FLOWCHART ARE OMITTED]

Flowcharts are divided into four major categories:

- Document flowchart: It shows document flow through systems.
- Data flowchart: It shows data flows in a system.
- System flowchart: It shows controls at a physical level.
- Program flowchart: It shows controls in a program.

Benefits of Flowchart: [**DEEEP – C i.e., deep sea**]

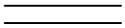
- **D**ocumentation: Flowcharts serve as a good program documentation which is needed for various purposes.
- **E**ffective analysis: With the help of flowchart, problem can be analyzed effectively.
- **E**fficient Program Maintenance: Maintenance of operating program becomes easy with the help of flowchart.
- **E**fficient Coding: Flowcharts act as a guide during systems analysis and development phase.
- **P**roper Debugging: Flowchart helps in debugging process.
- **C**ommunication: Flowcharts are better way of communication to all concerns.

Limitations of Flowcharts: [**CAR**]

- **C**omplex logic: Sometimes program logic is quite complicated which makes flowchart complex and clumsy.
- **A**lterations and Modifications: If any alterations are required then flowchart needs to be redrawn completely.
- **R**eproduction: As flowchart symbols cannot be typed, reproduction of flowchart becomes difficult.

The essentials of what is done can easily be lost in the technical details of how it is done.

C) Data Flow Diagrams: A Data Flow Diagram uses few simple symbols to illustrate the flow of data among external entities i.e., how external entities will interact with the system. Four symbols are combined to show how data are processed, they are as follow:

Symbol	Name	Explanation
	Data Sources and destinations	The user and organizations that send and receive data from the system are represented by square boxes called Data destinations.
	Data flows	The flow of data into or out of a process is represented by curved or straight lines with arrows.
	Transformation Process	Data transformations from inputs to outputs are represented by circles often referred to as bubbles.
	Data stores	The storage of data is represented by two horizontal lines.

D) Decision Tree: A Decision Tree (or tree diagram) helps in decision making with their possible consequences like chances of outcomes, costs, utility, etc. Decision tree is mainly used in decision analysis to identify a strategy which helps to reach goal and to calculate conditional probabilities. It is a tree diagram or tree like graph.

E) Decision Table: A Decision Table is a table which may accompany a flowchart defining the possible contingencies which may be considered within the program and the appropriate course of action for each contingency. The four parts of the decision table are as follows:

1. Condition Stub - It lists the conditions;
2. Action Stub – It lists the actions to be taken;
3. Condition entries - It lists in its columns the possible permutations of answer to the questions in the 1. (i.e., conditions stub); and
4. Action entries - It lists in its columns corresponding to the 3. (i.e., condition entries) the actions contingent upon set of answers to the questions of that column.

F) CASE Tools: The data flow diagram and system flow charts that users review are commonly generated by using the on-screen drawing modules found in CASE (Computer-Aided-Software Engineering) software packages. CASE refers to the automation of anything that humans do to develop systems (i.e., it co-ordinate everything that human do to develop system) and support virtually all phases of development process. An ideal CASE system would have an integrated set of tools and features to perform all aspects in the life cycle.

G) System Components matrix: It documents resources used, the activities performed and the information produced by a system. It can be used as an information system by for both systems analysis and system design. It highlights how the basic activities are accomplished in an information system and how resources are converted into information products.

H) Data Dictionary: A data dictionary is a computer file that contains descriptive information about the data i.e., various data entering into the data warehouse; source of data; various uses of data that are stored in data dictionary; the identity of the computer programs or individuals permitted to access the data; the identity of the computer programs or individuals not permitted to access the data item etc.

When new data is added then computer records it in the data dictionary. Similarly, when new programs are created then existing data are updated with this new program by data dictionary. Finally, when data fields are deleted from the records then their corresponding records are also dropped from the data dictionary.

l) Layout form and Screen Generator, Menu Generator, Report generator, Code Generator: Layout form and Screen Generator: They are for printed report which are used to format or "paint" the desired layouts without entering complex formatting information.
Menu Generator: It outlines the functions which the system should accomplish. They may be linked to other submenus which enables users to understand how the screens and sub-screens will be used for data entry.
Report Generator: Report generator has capacity of performing similar functions as found in screen generators i.e., used to format or "paint" the desired layouts without entering complex formatting information and further indicates totals, paging and breaks in creating samples of the desired report.
Code Generator: They allow the analyst to form modular units of source code from the high level specifications provided by the system analyst and helps in systems development process.

2.5.5 System Specification: At the end of analysis phase, system analyst prepares a document called Systems Requirement Specifications (SRS). A SRS contains the following:

- Introduction: Goals and Objectives of the software development
- Information Description: Like problem description; contents of Information, structure of Hardware, software; human interfaces for external system; etc.
- Functional Description: Diagrammatic representation of functions; processing of each function; interplay among functions; etc.
- Behavioral Description: Response to external events and internal controls
- Validation Criteria: Class of test to be performed to validate functions.
- Appendix: Data flow / Object Diagrams; Tabular Data; Detailed description of charts, graphs; etc.
- SRS Review: The development team makes a presentation and hands over the SRS document to management who reviews the same and if satisfied then signs the document.

2.5.6 Roles Involved in SDLC

1. Steering Committee: Some of the functions of Steering Committee are as follows:

- To provide overall direction.
- To conduct a regular review of progress of the project in the meetings of steering committee.
- Taking corrective actions, if necessary.

2. Project Manager: A project manager is normally responsible for more than one project. He is also responsible for completing the project within the time and budget and to conduct periodically review of the progress of the project.
3. Project Leader: The project leader has to ensure completeness and fulfillment of project along with its objectives and also reviews the project position more frequently.
4. Systems Analyst / Business Analyst: The systems analysts' conducts interviews with users and understand their requirements and then converts the users' requirements in the system requirements.
5. Module Leader / Team Leader: A project is divided into several manageable modules and the development of each module is given to Module Leaders. They are also responsible for the delivering modules within the stipulated time and cost.
6. Programmer / Coder / Developer: Programmers converts design into programs. He also tests the program for debugging activity.
7. Database Administrator (DBA): The DBA handles multiple projects; ensures the integrity and security of information stored in the database. Inclusion of new data has to be done only with the approval of the database administrator.
8. Quality Assurance: This team sets the standards for development and checks compliance with these standards on a periodic basis.
9. Tester: Tester is junior level quality assurance personnel who tests programs as per the plan given by the module and prepare test reports.
10. Domain Specialist: Whenever a project team has to develop an application which is new to them, they take the help of a specialist of that field. For example, if a team undertakes application development in Insurance, about which they have little knowledge, they may seek the assistance of an Insurance expert at different stages. This makes it easier to anticipate user needs. A domain specialist need not have knowledge of software systems.
11. IS Auditor: IS Auditor should be involved at the Design as well as the final Testing Phase to ensure the existence and the operations of the Controls in the new software.

2.6 SYSTEMS DESIGN

After the completion of requirements analysis for a system, design of a system of that alternative which has been selected by management takes place.

Objective: Objective is to designs a system that best suites the users' requirements. It describes the parts of the system and their interaction; how the system shall be implemented, specifies the program and the database specifications and the security plan and control point to prevent uncontrolled access.

Activities: This phase includes activities like describing inputs and outputs, determining the processing steps, stating rules for the new solution, determining database system design; preparing the program specifications, etc.

Document / Deliverable: Blueprint or the design which will be used in system design shall be created

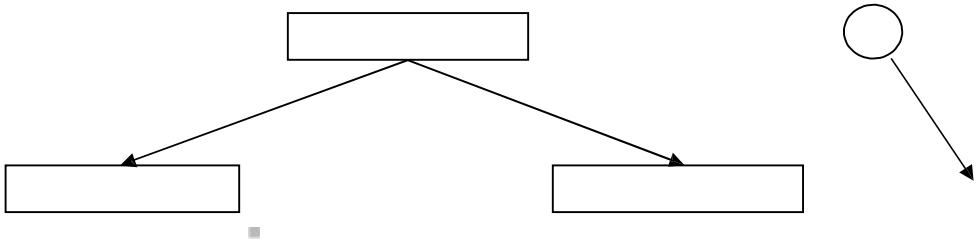
Qn: What are the steps involved in design of a new system?

System design involves first logical design and then physical construction of a system. The logical design describes the main features of the system and how they are interacted to one another. Physical construction produces program software, files and a working system. Design specifications instruct programmers about what the system should do. Once the detailed design is completed, the design is then distributed to the system developers for coding. The design phase involves following steps:

- Architectural Design / Sub System / Decomposition;
- Design of the Data / Information Flow;
- Design of the Database;
- Design of the User-interface;
- Physical Design; and
- Design and acquisition of the hardware/system software platform

2.6.1 Architectural Design: Architectural design deals with the organizing the applications in terms of modules and sub -modules. Here we identify - major modules; function of each module; interface of each module; data received from / sent to / modified in other modules are identified.

The architectural design is made with the help of Functional Decomposition. In this module is represented by a box and connected with arrows. Couple is data element that moves from one module to another which is shown by an arrow with circular tail.



2.6.2 Design of Data / Information flow: It is a major step in design of a new system. In defines the data flow for the proposed system, the inputs required for existing data; problems with the present system; and objective of the new system. All these have been identified in the analysis phase and documented in Software Requirements Specification (SRS).

2.6.3 Design of Database: It determines the scope ranging from local to global structure. The scope is decided on the basis of interdependence among organizational units. Greater the interdependence, greater will be the need for a global structure to prevent sub-optimization by subunits. The design of the database involves four major activities which are as follows:

Conceptual Modeling: These describe the application domain via entities, attributes of these entities and their relationships.

Data Modeling: Conceptual Models need to be translated into data models so that they can be accessed by both high-level and low level programming languages

Storage Structure Design: Decisions must be made on how to partition the data so that it can be stored on some device. Eg: tuples (row) in a relational data model must be assigned to records.

Physical Layout Design: Decisions must be made on how to distribute the storage structure across specific storage media and locations.

2.6.4 Design of User-Interface: Design of user interface means determining how the users will interact with a system like source documents to capture raw data; hard-copy output reports; graphic and color displays; special input/output device; etc.

Designing System Outputs: One of the most important features of a system for users is what output system will generate. The right output must be developed so that users will find the system easy and effective to use.

Input Objectives: Input design consists of developing specifications for data preparation, developing steps to put data into a usable form for processing, and data-entry, i.e., the activity of putting the data into the computer for processing.

Output Objectives: The output from a system must accomplish one or more of the following objectives:

- Convey information about past activities, current status or future projections.
- Signal important events, opportunities or problems.
- Trigger an action.
- Confirmation of an action.

2.6.5 Physical Design: For the physical design, the logical design is transformed into units which are further decomposed into implementation units such as programs and modules. During physical design, the auditor's primary concern is efficiency and effectiveness issues. The auditor should also evidence that designers follow some type of structured approach like – CASE tools.

Design Principles: [**SMART**]

- The design should be as per laid down **standards**.
- The design should be **modular**.
- The design should be based **after analysis**.
- The design should be **related** to business activities.
- There is a **tendency** to develop only one design and consider it as final. However it is recommended two or three design and selects the best out of them.

Modularity: A module is a manageable unit containing data and instructions to be performed in well-defined task. Interaction among modules is based on well-defined interfaces. Modularity is measured by two parameters: Cohesion and Coupling.

Cohesion means the manner in which elements within a module are linked.

Coupling refers to interconnection between modules. In a good modular design, cohesion will be high and coupling low.

2.6.6 Design of the Hardware / System Software Platform: In some cases, the new system requires hardware and system software not currently available in an organization which has to be developed to support the application system. If different hardware and software are not able to communicate with each other then subsequent changes will have to be made and resources should be expanded in trying to make the hardware and software compatible to each other.

2.7. SYSTEM ACQUISITION: After system is designed the next phase of system development is acquisition of hardware, software and services.

2.7.1 Acquisition Standards: Acquisition standards should focus on -

- Ensuring security, reliability, and functionality of existing system.
- Ensuring reviews of vendor, contract, and license and acquiring products compatible with existing systems.
- Including invitations-to-tender and request-for-proposals. Invitations-to-tender involve soliciting bids from vendors while acquiring hardware or software. Request-for-proposals involve soliciting bids when acquiring third-party developed software.

2.7.2 Acquiring Systems Components from Vendors: At the end of the design phase, the organization will get an idea of the type of hardware, software and services required for the system development.

Management should also decide whether the hardware is to be purchased, leased or to be rented.

- **Hardware Acquisition:** While procuring hardware, management normally relies on the time tested selection techniques and the objective selection criteria can be delegated to the technical specialist.

- **Software Acquisition**: After input and output designs are finalized, the systems analyst must determine type of software required. This determination helps the systems developers in deciding about the hardware or software which will be most suited for desired output. They should also determine whether the application software should be created in-house or acquired from a vendor.
- **Contracts, Software Licenses and Copyright Violations**: Contracts between an organization and a vendor should clearly describe the rights and responsibilities of the parties to the contract. The contracts should be in writing and in detail.
  Software license is a license that grants permission to do things with computer software i.e., it authorizes activities which are prohibited for others.
  Copyright laws protect proprietary as well as open-source software. An organization will be exposed to litigation if it uses unlicensed software or violates licensing agreement.
- **Validation of Vendors' proposals**: The contracts and software license consists of evaluation and ranking of various proposals submitted by various vendors which is quite difficult and time consuming because each vendor offers variety of configurations which has to be rigorous evaluated based on the following factors:
  The Performance of each proposed system in relation to its costs;
  Costs - Benefits analysis of each proposed system;
  Maintainability of each proposed system;
  Compatibility of each proposed system with Existing Systems; and
  Vendor Support i.e., after sales service.
- **Methods of validating the proposal**:
  Some of the validation methods are as follows: [**CP = PBT i.e. Cost Price = Profit Before Tax**]
  Checklists: It is the most simple and subjective method for validation and evaluation. The various criteria are put in check list in the form of questions which has to be responded by various vendors.
  Point-Scoring Analysis: It is an objective mean of selecting a final system. There are no absolute rules in the selection process, only guidelines should match with users needs. Here points are allotted to each criteria or question.
  Public Evaluation Reports: Several consultancy agencies compare their hardware and software performance with various other manufacturers and publish their reports. This method is useful when buying staff does not have adequate knowledge of facts.
  Bench marking problem for vendor's proposals: Benchmarking problems for vendors' proposals are sample programs which have been designed to represent proposed requirements Benchmarking is a technique which tests whether design or computer offered by the vendor meets the requirements of the buyer.
  Test problems: They disregard actual job mix (i.e., current configuration) and are devised to test the true capabilities of the hardware, software or system.

2.8. DEVELOPMENT: PROGRAMMING TECHNIQUES AND LANGUAGES

Objective: To convert the specification into a functioning system.
Activities: Application programs are written, tested and documented, conduct system testing.
Document / Deliverable: A fully documented system.

A good coded program should have the following characteristics:
Reliability: It means program should be consistence over a period of time. However poor setting of parameters or hard coding of some data could result in the failure of a program after some time. **Robustness:** It refers to process of taking inputs and outputs in least likely situations.
Accuracy: It refers to what program should do and should also take note of what it should not do.
Efficiency: It means performance should not be affected due to increase in input values.
Usability: It means user-friendly interface and easy-to-understand for any program.
Readability: It refers to the ease of maintenance of program even in the absence of the program developer

2.8.1 Program Coding Standards: Different programmers write the program using different sets of rules but gives the same results and therefore, coding standards should be defined which provides simplicity, efficient utilization of storage and least processing time.

2.8.2 Programming Language: Application programs are coded in the form of statements or instructions which is converted by the compiler to binary machine for the computer to understand and execute. The programming languages commonly used is as follows COBOL, C++, JAVA, JAVAScript, etc.

Choice of Programming Language: The choice of programming language should be decided on the basis of application area; environment in which software has to be executed; data structure complexity; knowledge of software development staff; and capability of in-house staff for maintenance; etc.

2.8.3 Program Debugging: It is testing activity which corrects programming language syntax and diagnoses errors so that program compiles easily. A clean compile successfully converts from the source code into machine language instructions. Debugging can be a tedious task consisting of following four steps:

- Inputting the source program to the compiler,
- Letting the compiler find errors in the program,
- Correcting lines of code that are erroneous, and
- Resubmitting the corrected source program as input to the compiler.

2.8.4 Test the program: A careful testing of each program should be done so that system can be installed successfully. The programmer should plan the testing considering all possible exceptions.

2.8.5 Program Documentation: Managers and users should carefully review documentation to ensure that the software and system behaves as indicated in the document. If not then documentation should be revised.

2.8.6 Program Maintenance: The requirements of business data processing continuously changes and this calls for modification of the various programs which should be properly maintained.

2.9. SYSTEM TESTING: Testing is a process used to identify the correctness, completeness and quality of developed computer software. Testing should uncover all errors with minimum amount of time and with minimum efforts.

Different levels of Testing are as follows:

2.9.1 Unit Testing: Unit testing is a software verification and validation method in which programmer tests whether individual units are fit for use. The goal of unit testing is to isolate each part of the program and show that the individual parts are correct and behaves as intended.

There are five categories of tests that a programmer typically performs on a program unit:

[**FPSSP – Failed Percentage Student Should Pass**]

- **Functional Tests:** It checks whether programs do what they are supposed to do or not. The programmer also checks whether the actual and expected result matches or not.
- **Performance Tests:** It verifies response time, execution time, throughput, traffic rates on data channels, etc. It also compares new system performance with similar system using well defined bench marking.
- **Stress Tests:** It determines stability of a given system. Testing is done beyond normal capacity in order to observe the results. They are performed during peak hours when there is large volume of data and thus test its performance.
- **Structural Tests:** Structural Testing examines the internal processing of a software system.
- **Parallel Tests:** In Parallel Testing same test data is used in both old and new system and output results are compared.

Types of Unit Testing

(I) Static Analysis Testing

Some important Static Analysis Tests are as follows:

- **Desk Check:** This is done by the programmer himself and checks for syntax errors and deviation from coding standards.
- **Structured walk-through:** The application developer leads other programmers through the text of the program and explanation.

- **Code inspection:** The program is reviewed by a formal committee. Review is done with formal checklists.

(II) Dynamic Analysis Testing

- **Black Box Testing:** Black Box Testing takes an external view to derive test cases. These tests can be functional or non-functional. The designer selects both valid and invalid inputs and derives correct output. This method of testing is applicable to all levels of software testing: unit, integration, functional testing, system and acceptance. The higher the level, greater the need of black box testing to simplify testing. However, one cannot be sure that all existent paths are tested. If a module performs a function which it is not supposed to do, then the black box testing cannot identify it.

- **White Box Testing:** White box testing takes an internal view to derive test cases. The tester chooses inputs to exercise paths and derives the appropriate outputs through codes. It is applicable at the unit, integration and system levels of the testing. Normally tests paths within a unit but it can also test paths between units during integration and also between subsystems during a system level testing. This test ensures working of internal operation with specifications.

- **Gray Box Testing:** Gray box testing uses a combination of black box testing and white box testing. Here, tester applies a limited number of test cases to the internal workings of the software under test and in the remaining part applies black box approach.

2.9.2 Integration Testing: Integration testing is an activity of software testing where individual modules are combined and tested as a group. It occurs after unit testing and before system testing. It evaluates whether information received by the module is the same which has been send by earlier module. It is carried out in the following manner:

- **Bottom-up Integration:** Bottom-up integration is the traditional strategy used to integrate the components of a software system into a functioning whole. It consists of unit testing, followed by sub-system testing, and then testing of the entire system. The disadvantage is that testing of major decisions is deferred to a later period.
- **Top-down Integration:** Top-down integration starts with the main routine, and stubs are substituted, for the modules directly subordinate to the main module. Once the main module testing is complete, stubs are substituted with real modules one by one, and these modules are tested with stubs. The difficulty arises in the top-down method, because the high-level modules are tested, not with real outputs from subordinate modules, but from stubs.
- **Regression Testing:** Each time a new module is added as part of integration testing, the software changes and this causes problems with functions that previously worked flawlessly. In the context of the integration testing, the regression tests ensure that changes or corrections have not introduced new errors. The data used for the regression tests should be the same as the data used in the original test.

2.9.3 System Testing: System testing is a process where software and other systems are tested as a whole. The purpose of system testing is to ensure that the new or modified system functions properly i.e., it checks whether all of them work together properly and also simultaneously. The types of testing that might be carried out are as follows:

- **Recovery Testing:** It tests how well application can recover from crashes, hardware failures and other similar problems. Recovery testing is the forced failure of the software to verify whether recovery is properly performed.
- **Security Testing:** It determine whether system protects data and maintains functionality as intended or not. This testing technique also ensures the existence and proper execution of access controls in the new system.
- **Stress Tests:** It determines stability of a given system. Testing is done beyond normal capacity in order to observe the results. They are performed during peak hours when there is large volume of data and thus test its performance.
- **Performance Tests:** It verifies response time, execution time, throughput, traffic rates on data channels, etc. It also compares new system performance with similar system using well defined bench marking.

2.9.4 Final Acceptance Testing: Final Acceptance Testing is conducted when the system is ready for implementation. System testing is conducted by a company on the system supplied by the system developer. This testing ensures that the new system satisfies the quality standards of the business and the system satisfies the users. It has two major parts:

- **Quality Assurance Testing**: It ensures that the new system satisfies the prescribed quality set standards and the development process is as per the organization's requirements.
- **User Acceptance Testing:** It ensures that the functional aspects required by the users have been properly executed in the new system. There are two types of the user acceptance testing :

  Alpha Testing: This is the first stage, often performed by the users within the organization.

  Beta Testing: This is the second stage, generally performed by the external users. This is the last stage of testing, and normally involves sending the product outside the development environment for real world exposure.

2.10. SYSTEMS IMPLEMENTATION:

Objective: To implement the new system i.e. put it into operation.

Activities: The activities involved in System Implementation are as follows:

- Conversion of data to the new system files.
- Training of end users.
- Proper documentation.
- System changeover.
- Evaluation of the system at regular intervals.

Document / Deliverable: A full functional / documented system in its operational environment.

This process ensures that new system is operational and also allows users to take over its operation for use and evaluation is called Systems Implementation.

2.10.1 Activities during Implementation Stage

The activities involved in system implementation stage are as follows:

(I) Equipment Installation: Hardware required to support the new system should be selected prior to the implementation phase. While installation adequate time should be given to allow completion of the following activities:

- **Site Preparation:** An appropriate environment for operating equipment must be found which should be as per vendor's temperature, humidity and dust control specifications.
- **Installation of new hardware / software**: The equipment must be physically installed by the manufacturer with proper connection to the power and wired to communication lines, if required.
- **Equipment check out**: The equipment must be tested thoroughly under normal operating conditions by both the vendor and the implementation team to ensure that equipments are proper and in working condition.

(II) Training Personnel: A system can succeed or fail depending on the way it is operated and used. Therefore, the proper training should be given to the personnel involved with the system for successful implementation of system.

(III) System Implementation Conversion Strategies: Conversion or changeover is the process of changing from the old system to the new system. It requires careful planning for actual changeover. The four types of implementation strategies are as follows:

- **Direct Implementation / Abrupt change-over**: In this strategy, the changeover is done in one operation ie. completely replacing the old system at once o a set date. Direct Implementation usually takes place on a set date often during a break in production or a holiday period so that new system is installed without causing too much disruption.
- **Phased implementation**: In this strategy old system is converted into new system in phased manner i.e. after one phase is successfully converted next phase begins until last phase is successfully completed uptil the new system fully replaces the old one.
- **Pilot implementation**: In this strategy the new system replaces the old one at once but only on a small scale so that any errors can be rectified or further any benefits can be incorporated throughout the whole system with the least disruption. For example - it might be tried out in one branch of the

company or in one location. If successful then the pilot is extended until it eventually replaces the old system completely.

- **Parallel running implementation**: This is considered the most secure method where the both systems run parallel for a short period where both are operated independently. If all goes well the old system is being replaced by new system.

2.10.2 Activities involved in conversion: Conversion includes all those activities which must be completed to successfully convert from the previous system to the new system. These activities are as follows:

- **Procedure conversion**: Operating procedures should be completely documented for the new system that applies to both computer-operations and functional area operations. Information on input, data files, methods, procedures, output, and internal control must be presented in clear, concise and understandable terms for the average reader.
- **File conversion**: This phase should be started long before because large files of information must be converted from one medium to another. In order for the conversion to be as accurate as possible, file conversion programs must be thoroughly tested. The existing computer files should be kept for a sufficient period of time until files are properly backed up.
- **System conversion**: After on-line and off-line files have been converted and the reliability of the new system has been confirmed for a functional area, daily processing can be shifted from the existing information system to the new one. All transactions are then processed on the new system. Simultaneously the old system should be operated for some more time to check the total results of both systems.
- **Scheduling personnel and equipment**: Scheduling data processing operations of a new system for the first time is a difficult task for the system manager. As users become more familiar with the new system the job becomes more routine.

2.11. POST IMPLEMENTATION REVIEW AND SYSTEMS MAINTENANCE

Objective: To assess and review the complete working of a new system after implementation.

Activities: Some of the Systems maintenance activities are as follows:

- Adding new data elements;
- Modifying reports;
- Adding new reports; and
- Changing calculations.

Document / Deliverable: A document should state any scope for improvements, if any like-

- Could further training can improve the degree of benefit being generated?
- Are there further functional improvements or changes that would deliver greater benefit?
- Are specific improvements required in procedures, documentation, support, etc?
- What learning points are there for future projects?

2.11.1 Post Implementation Review: A Post Implementation Review should answer the most important question i.e. “Did we achieve what we set out to do in business terms?” It should further ascertain the success of the project, in particular in achieving its objectives and should also measure improvements which can be incorporated. A Post-Implementation Review should be conducted some time after the new system is deployed, say ranging from 6 weeks to 6 months. The review should be able to answer the following question:

- **Development evaluation**: Due to the uncertainty and mystique associated with system development, very few system are developed within scheduled time and given budget and therefore development process is primarily concerned with whether the system was developed on or before schedule date and within given budget.
- **Operation evaluation**: This evaluation pertains to answer whether the hardware, software and personnel are capable to perform their duties. Operation evaluation is relatively straightforward if evaluation criteria are established in advance.
- **Information evaluation**: An system should also be evaluated in terms of information it provides ie., what information is being provided by new system should be evaluated here. It should be able to provide information that supports the organizational decision system.

2.11.2 System Maintenance: Maintaining the system is an important aspect of SDLC. Maintenance can be categorized in the following ways:

- **Scheduled maintenance**: Scheduled maintenance is anticipated and can be planned for. For example, the implementation of a new inventory coding scheme can be planned in advance.
- **Rescue maintenance**: Rescue maintenance refers to previously undetected errors that were not anticipated but require immediate solution. A system that is properly developed and tested should have few occasions of rescue maintenance.
- **Corrective maintenance**: Corrective maintenance deals with fixing bugs in the code or defects found. A defect can result from design errors, logic errors; coding errors, data processing and system performance errors.
- **Adaptive maintenance**: Adaptive maintenance consists of adapting software to changes in the environment, such as the hardware or the operating system. The term environment in this context refers to the totality of all conditions from outside upon the system, for example, business rule, government policies, etc.
- **Perfective maintenance**: Perfective maintenance mainly deals with accommodating to new or changed user requirements and activities to increase the system's performance or to enhance its user interface.
- **Preventive maintenance**: Preventive maintenance concerns activities aimed at increasing the system's maintainability, such as updating documentation, adding comments, and improving the modular structure of the system.

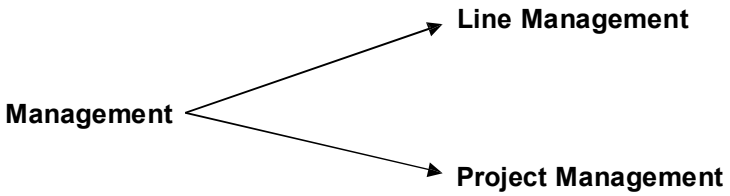
2.12. OPERATION MANUALS: A user's guide is commonly known as an Operation Manual which is a technical communication document which gives assistance to user while using a particular system. It is usually written by a technical writer. Operation manuals are most commonly associated with electronic goods, computer hardware and software. The section of an operation manual includes the following:

- A cover page, a title page;
- A preface containing details of information on how to use the user's guide;
- A contents page;
- A guide on how to use the main functions of the system;
- A troubleshooting section stating possible errors and problems that may occur, along with solution;
- A FAQ (Frequently Asked Questions);
- Where to find further help along with contact details;
- A glossary and for larger documents an index.

2.13. ORGANIZATIONAL STRUCTURE OF IT DEPARTMENT [AS PER OLD MODULE]

We will now give a brief introduction about the management structure of IT department.

2.13.1 Management Structure



Line Management Structure: The information system management subsystems in an organization attempt to ensure that the development, implementation, operation and maintenance of information systems proceed in a planned and controlled manner and operated on a day-to-day basis. Several levels of management subsystems have been identified which are as follows:

- **Top Management**: They must ensure that the data processing installation and are also responsible for long run policies decisions on how computers will be used in the organization.

- **IS Management**: IS management has overall responsibility for planning and control of all computer activities and translates long run policies into short run goals and objectives

- **Systems Development Management**: Systems Development Management is responsible for the design, implementation and maintenance of systems.
- **Programming Management**: They are responsible for programming new systems and maintaining old system.
- **Data Administration**: They are responsible for the control and use of an organization's data.
- **Security Administration**: They are responsible for the physical security of the data processing.
- **Operations Management**: They control the day-to-day operations of data processing systems. It is responsible for data preparation; the data flow through the installation, maintenance of hardware; etc.
- **Quality Assurance Management**: They undertake an in-depth quality review of data in each system. This review involves a detailed check of the authenticity, accuracy and completeness of input, processing and output.

2.13.2 Project Management Structure: In project management, project requests are submitted to and prioritized by the steering committee. The project manager should be given complete operational control of the project and be allocated the appropriate resources for the successful completion of the project.
Duties and Responsibilities: The structure of an IT Department is divided into two main areas of activity:

- Information processing: It is primarily concerned with the operational aspect of the information-processing environment and often includes computer operations, systems programming, telecommunications and librarian functions.
- System development and enhancement: It is concerned with the development, acquisition and maintenance of computer application systems and performs systems analysis and programming functions.

Data entry: The data entry supervisor has to ensure whether the data entered is authorized, accurate and complete. The information is of two forms: first, it may be raw data to be processed and; second, it may be instructions to direct the system to execute particular processes, prepare particular types of output, etc.
File Library: The file librarian is responsible for recording, issuing, receiving and safeguarding all data files that are maintained on computer tapes or disks. This involves three functions. First, files must be used only for the purposes intended. Second, the storage media used for files must be maintained in good working condition. Third, a file backup and retention strategy must be implemented.
Control Group: The control group manages the flow of data and is responsible fore the collection of input and balancing the distribution of output to the user. They must be kept in a separate area and only authorized personnel should be permitted.
Operations: Operations management is responsible for the daily running of hardware and software so that the production system can complete their work and development staff can develop their system.
Security Administration: The security administrator is responsible for physical security of system. So that it can safeguard the system from threats that affect the continuity of operation.
Physical Security: They must take into account the possibility of all physical attacks on the database ranging from disclosure of a password to theft of the physical storage devices. It can be achieved by encrypting data.
Data Security: Database management systems provide control over data manipulation facilities. When database combines with online transaction processing, controls can be ensured through internal database which limit the transaction or program.
Conducting a Security program: A security program is a series of ongoing and regular evaluation conducted to ensure that the physical facilities are properly safeguarded. Firstly the security administrator should consider the possible threats to the organization, evaluate the adequacy of controls, etc. Then it must focus on threats that may occur due to the purchase of new hardware or software, etc.
Production Work Flow Control: Production workflow control in a system is the responsibility of a control section who manages the flow of data between users and the system, and between data preparation and the computer room.
Quality Assurance: Quality Assurance group must test and verify whether the program changes and

documentation are as per standards before the programs are moved into production. The control section facilitates the orderly flow of data and also checks that the inputs are in order.

Systems Analysis: System analysts are responsible for designing the system based on the needs of the user. From the auditor's viewpoint, the auditor is concerned with whether the design meets strategic requirements; efficiency of the system and what resources will be required to run the system and; safeguarding access and data integrity.

Applications Programming: Application programmers are responsible for developing new systems and for monitoring systems in production. They should work in a test environment only. Application programmers should not have access to system program libraries.

Systems programming: System programmers are responsible for maintaining the systems software including the operating systems.

Local Area Network (LAN) Administration: LAN administrator is responsible for technical and administrative control over the local area network which includes transmission links functions properly, ensuring backups of the system are occurring and purchased hardware / software are authorized and properly installed.

Help Desk Administration: The Help Desk Administrator is responsible for monitoring, improving and controlling system performance in mainframe and client/server hardware and software. They are useful when data entry is not based upon a dedicated source document.

AS PER NEW MODULE [above whole 2.13 para has been omitted and it has been substituted by this para but there's no notification for its applicability for MAY'12, so I have inserted both the things to be on safer side.]

2.13. Auditors Role in SDLC: The audit of system under development can have three main objectives:

- To provide an opinion on the efficiency, effectiveness and economy of project management,
- To assess the extent to which the system being developed provides adequate audit trails and controls to ensure the integrity of data processed; and
- To assess the controls being provided for the management of the system's operation.

For the first objective to achieve, an auditor will have to attend project and steering committee meetings; examine project documentation and conduct interviews. This is required to ensure what project control standards are to be complied with and extend of compliance achieved.

For the second objective, auditor can examine system documentation to arrive at opinion on controls. The auditor's opinion will be based on the degree to which the system satisfies the general control objectives. For the third objective, the auditor should provide list of standard controls for operational concerns such as response time, CPU usage, etc

An auditor may adopt a rating system like 1 to 10 to the various phases of SDLC. Eg in rating a feasibility study, auditor can review feasibility study report and can also interview personnel who have conducted this feasibility study. Based on this auditor can arrive at a rating (10 being the best). After deriving such a rating for all the phases, the auditor can form his overall opinion about the SDLC phases. In order to audit technical work products such as physical design, auditor may opt to include a technical expert to seek his opinion. However, auditor will have to give control objectives, directives, etc. Some of them are:

- Documented policy and procedures;
- Established project team with all infrastructure and facilities;
- Development is carried over as per standards, functional specifications;
- Separate test environment for development / test production / test plans;
- Business owners testing and approval before system going alive;
- Adequate audit trails are provided in system; and
- Appropriateness of methodology selected.

Further, post implementation review is performed to determine whether the system meets business requirements. Auditor should also determine whether the expected benefits of the new system are realized and whether users are satisfied with the new system. Further auditor should review which of the SDLC phases have not met desired objectives and whether any corrective action was taken. If differences exist between expected and actual result then auditor should determine the reasons for the same.