

SYSTEM ANALYSIS AND DESIGN - QUICK GUIDE

https://www.tutorialspoint.com/system_analysis_and_design/system_analysis_and_design_quick_guide.htm

Copyright © tutorialspoint.com

SYSTEM ANALYSIS AND DESIGN - OVERVIEW

Systems development is systematic process which includes phases such as planning, analysis, design, deployment, and maintenance. Here, in this tutorial, we will primarily focus on –

- Systems analysis
- Systems design

Systems Analysis

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

Analysis specifies **what the system should do**.

Systems Design

It is a process of planning a new business system or replacing an existing system by defining its components or modules to satisfy the specific requirements. Before planning, you need to understand the old system thoroughly and determine how computers can best be used in order to operate efficiently.

System Design focuses on **how to accomplish the objective of the system**.

System Analysis and Design *SAD* mainly focuses on –

- Systems
- Processes
- Technology

What is a System?

The word System is derived from Greek word Systema, which means an organized relationship between any set of components to achieve some common cause or objective.

A system is “an orderly grouping of interdependent components linked together according to a plan to achieve a specific goal.”

Constraints of a System

A system must have three basic constraints –

- A system must have some **structure and behavior** which is designed to achieve a predefined objective.

- **Interconnectivity** and **interdependence** must exist among the system components.
- The **objectives of the organization** have a **higher priority** than the objectives of its subsystems.

For example, traffic management system, payroll system, automatic library system, human resources information system.

Properties of a System

A system has the following properties –

Organization

Organization implies structure and order. It is the arrangement of components that helps to achieve predetermined objectives.

Interaction

It is defined by the manner in which the components operate with each other.

For example, in an organization, purchasing department must interact with production department and payroll with personnel department.

Interdependence

Interdependence means how the components of a system depend on one another. For proper functioning, the components are coordinated and linked together according to a specified plan. The output of one subsystem is the required by other subsystem as input.

Integration

Integration is concerned with how a system components are connected together. It means that the parts of the system work together within the system even if each part performs a unique function.

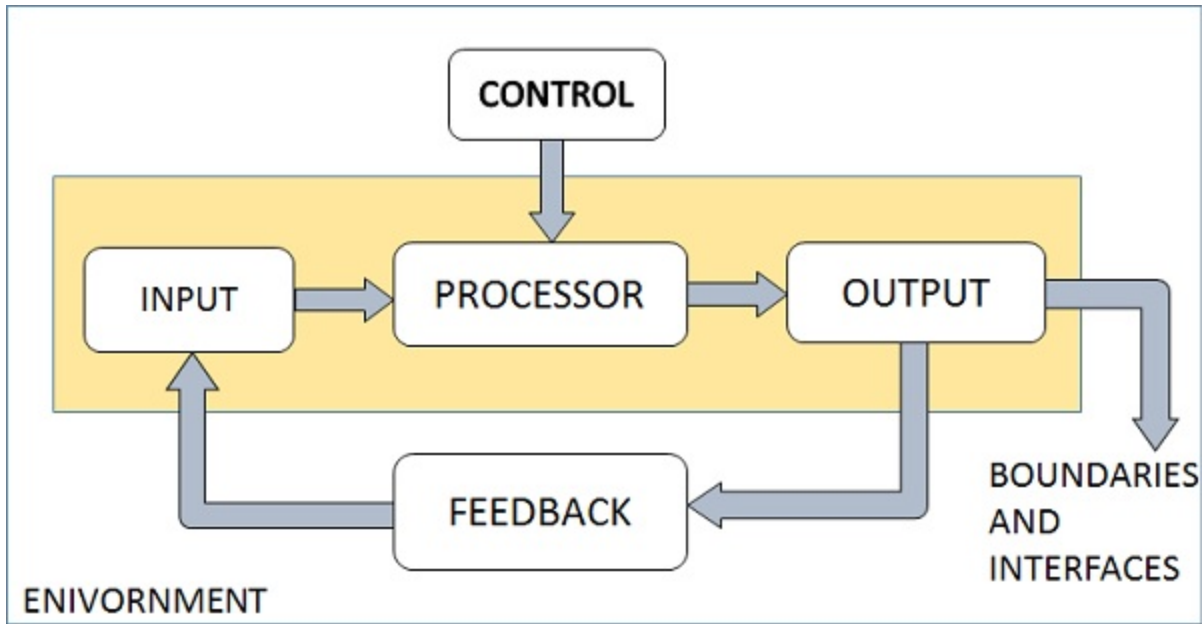
Central Objective

The objective of system must be central. It may be real or stated. It is not uncommon for an organization to state an objective and operate to achieve another.

The users must know the main objective of a computer application early in the analysis for a successful design and conversion.

Elements of a System

The following diagram shows the elements of a system –



Outputs and Inputs

- The main aim of a system is to produce an output which is useful for its user.
- Inputs are the information that enters into the system for processing.
- Output is the outcome of processing.

Processors

- 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 either totally or partially, depending on the output specification.
- As the output specifications change, so does the processing. In some cases, input is also modified to enable the processor for handling the transformation.

Control

- The control element guides the system.
- It is the decision-making subsystem that controls the pattern of activities governing input, processing, and output.
- The behavior of a computer System is controlled by the Operating System and software. In order to keep system in balance, what and how much input is needed is determined by Output Specifications.

Feedback

- Feedback provides the control in a dynamic system.
- Positive feedback is routine in nature that encourages the performance of the system.

- Negative feedback is informational in nature that provides the controller with information for action.

Environment

- The environment is the “supersystem” within which an organization operates.
- It is the source of external elements that strike on the system.
- It determines how a system must function. For example, vendors and competitors of organization’s environment, may provide constraints that affect the actual performance of the business.

Boundaries and Interface

- A system should be defined by its boundaries. Boundaries are the limits that identify its components, processes, and interrelationship when it interfaces with another system.
- Each system has boundaries that determine its sphere of influence and control.
- The knowledge of the boundaries of a given system is crucial in determining the nature of its interface with other systems for successful design.

Types of Systems

The systems can be divided into the following types –

Physical or Abstract Systems

- Physical systems are tangible entities. We can touch and feel them.
- Physical System may be static or dynamic in nature. For example, desks and chairs are the physical parts of computer center which are static. A programmed computer is a dynamic system in which programs, data, and applications can change according to the user's needs.
- Abstract systems are non-physical entities or conceptual that may be formulas, representation or model of a real system.

Open or Closed Systems

- An open system must interact with its environment. It receives inputs from and delivers outputs to the outside of the system. For example, an information system which must adapt to the changing environmental conditions.
- A closed system does not interact with its environment. It is isolated from environmental influences. A completely closed system is rare in reality.

Adaptive and Non Adaptive System

- Adaptive System responds to the change in the environment in a way to improve their performance and to survive. For example, human beings, animals.
- Non Adaptive System is the system which does not respond to the environment. For example, machines.

Permanent or Temporary System

- Permanent System persists for long time. For example, business policies.
- Temporary System is made for specified time and after that they are demolished. For example, A DJ system is set up for a program and it is dissembled after the program.

Natural and Manufactured System

- Natural systems are created by the nature. For example, Solar system, seasonal system.
- Manufactured System is the man-made system. For example, Rockets, dams, trains.

Deterministic or Probabilistic System

- Deterministic system operates in a predictable manner and the interaction between system components is known with certainty. For example, two molecules of hydrogen and one molecule of oxygen makes water.
- Probabilistic System shows uncertain behavior. The exact output is not known. For example, Weather forecasting, mail delivery.

Social, Human-Machine, Machine System

- Social System is made up of people. For example, social clubs, societies.
- In Human-Machine System, both human and machines are involved to perform a particular task. For example, Computer programming.
- Machine System is where human interference is neglected. All the tasks are performed by the machine. For example, an autonomous robot.

Man–Made Information Systems

- It is an interconnected set of information resources to manage data for particular organization, under Direct Management Control *DMC*.
- This system includes hardware, software, communication, data, and application for producing information according to the need of an organization.

Man-made information systems are divided into three types –

- **Formal Information System** – It is based on the flow of information in the form of memos, instructions, etc., from top level to lower levels of management.
- **Informal Information System** – This is employee based system which solves the day to day work related problems.
- **Computer Based System** – This system is directly dependent on the computer for managing business applications. For example, automatic library system, railway reservation system, banking system, etc.

Systems Models

Schematic Models

- A schematic model is a 2-D chart that shows system elements and their linkages.

- Different arrows are used to show information flow, material flow, and information feedback.

Flow System Models

- A flow system model shows the orderly flow of the material, energy, and information that hold the system together.
- Program Evaluation and Review Technique *PERT*, for example, is used to abstract a real world system in model form.

Static System Models

- They represent 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.

Dynamic System Models

- Business organizations are dynamic systems. A dynamic model approximates the type of organization or application that analysts deal with.
- It shows an ongoing, constantly changing status of the system. It consists of –
 - Inputs that enter the system
 - The processor through which transformation takes place
 - The programs required for processing
 - The outputs that result from processing.

Categories of Information

There are three categories of information related to managerial levels and the decision managers make.

Volume of Information	Type of Information	Information Level	Management Level	System Support
Low Consensed	Unstructured	Strategic Information	Upper	DSS
Medium Moderately Processed	Moderately Structured	Management Control Information	Middle	MIS
Large Detail Reports	Highly Structured	Operational Information	Lower	DPS

Strategic Information

- This information is required by topmost management for long range planning policies for next few years. For example, trends in revenues, financial investment, and human resources, and population growth.
- This type of information is achieved with the aid of Decision Support System *DSS*.

Managerial Information

- This type of Information is required by middle management for short and intermediate range planning which is in terms of months. For example, sales analysis, cash flow projection, and annual financial statements.
- It is achieved with the aid of Management Information Systems *MIS*.

Operational information

- This type of information is required by low management for daily and short term planning to enforce day-to-day operational activities. For example, keeping employee attendance records, overdue purchase orders, and current stocks available.
- It is achieved with the aid of Data Processing Systems *DPS*.

SYSTEM DEVELOPMENT LIFE CYCLE

An effective System Development Life Cycle *SDLC* should result in a high quality system that meets customer expectations, reaches completion within time and cost evaluations, and works effectively and efficiently in the current and planned Information Technology infrastructure.

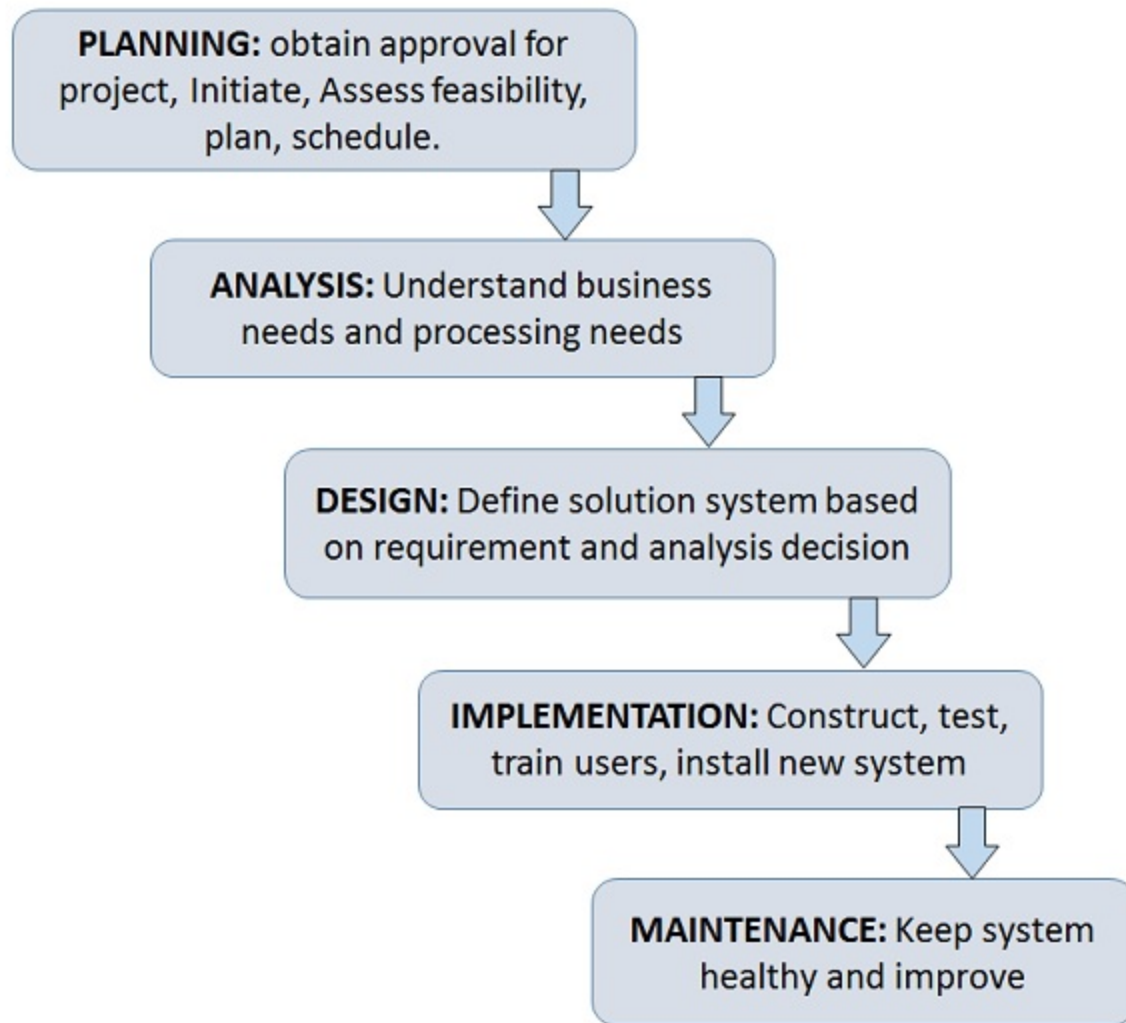
System Development Life Cycle *SDLC* is a conceptual model which includes policies and procedures for developing or altering systems throughout their life cycles.

SDLC is used by analysts to develop an information system. SDLC includes the following activities –

- requirements
- design
- implementation
- testing
- deployment
- operations
- maintenance

Phases of SDLC

Systems Development Life Cycle is a systematic approach which explicitly breaks down the work into phases that are required to implement either new or modified Information System.



Feasibility Study or Planning

- Define the problem and scope of existing system.
- Overview the new system and determine its objectives.
- Confirm project feasibility and produce the project Schedule.
- During this phase, threats, constraints, integration and security of system are also considered.
- A feasibility report for the entire project is created at the end of this phase.

Analysis and Specification

- Gather, analyze, and validate the information.
- Define the requirements and prototypes for new system.
- Evaluate the alternatives and prioritize the requirements.

- Examine the information needs of end-user and enhances the system goal.
- A Software Requirement Specification *SRS* document, which specifies the software, hardware, functional, and network requirements of the system is prepared at the end of this phase.

System Design

- Includes the design of application, network, databases, user interfaces, and system interfaces.
- Transform the SRS document into logical structure, which contains detailed and complete set of specifications that can be implemented in a programming language.
- Create a contingency, training, maintenance, and operation plan.
- Review the proposed design. Ensure that the final design must meet the requirements stated in SRS document.
- Finally, prepare a design document which will be used during next phases.

Implementation

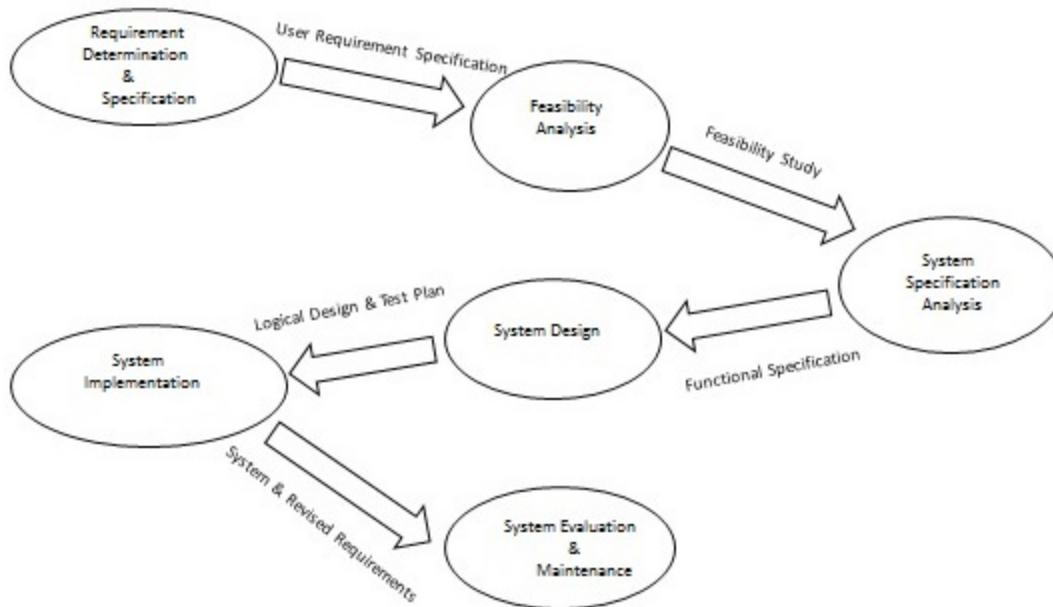
- Implement the design into source code through coding.
- Combine all the modules together into training environment that detects errors and defects.
- A test report which contains errors is prepared through test plan that includes test related tasks such as test case generation, testing criteria, and resource allocation for testing.
- Integrate the information system into its environment and install the new system.

Maintenance/Support

- Include all the activities such as phone support or physical on-site support for users that is required once the system is installing.
- Implement the changes that software might undergo over a period of time, or implement any new requirements after the software is deployed at the customer location.
- It also includes handling the residual errors and resolve any issues that may exist in the system even after the testing phase.
- Maintenance and support may be needed for a longer time for large systems and for a short time for smaller systems.

Life Cycle of System Analysis and Design

The following diagram shows the complete life cycle of the system during analysis and design phase.



Role of System Analyst

The system analyst is a person who is thoroughly aware of the system and guides the system development project by giving proper directions. He is an expert having technical and interpersonal skills to carry out development tasks required at each phase.

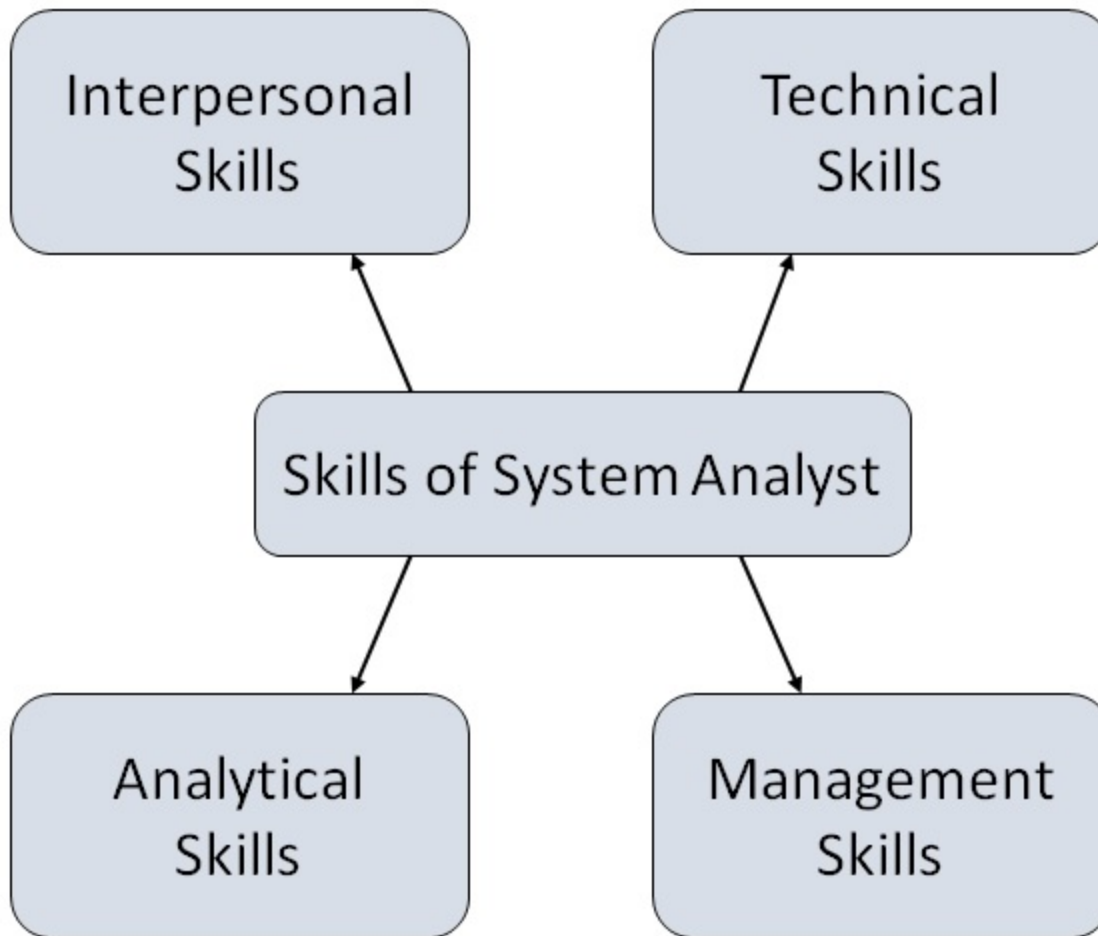
He pursues to match the objectives of information system with the organization goal.

Main Roles

- Defining and understanding the requirement of user through various Fact finding techniques.
- Prioritizing the requirements by obtaining user consensus.
- Gathering the facts or information and acquires the opinions of users.
- Maintains analysis and evaluation to arrive at appropriate system which is more user friendly.
- Suggests many flexible alternative solutions, pick the best solution, and quantify cost and benefits.
- Draw certain specifications which are easily understood by users and programmer in precise and detailed form.
- Implemented the logical design of system which must be modular.
- Plan the periodicity for evaluation after it has been used for some time, and modify the system as needed.

Attributes of a Systems Analyst

The following figure shows the attributes a systems analyst should possess –



Interpersonal Skills

- Interface with users and programmer.
- Facilitate groups and lead smaller teams.
- Managing expectations.
- Good understanding, communication, selling and teaching abilities.
- Motivator having the confidence to solve queries.

Analytical Skills

- System study and organizational knowledge
- Problem identification, problem analysis, and problem solving
- Sound commonsense
- Ability to access trade-off
- Curiosity to learn about new organization

Management Skills

- Understand users jargon and practices.
- Resource & project management.
- Change & risk management.
- Understand the management functions thoroughly.

Technical Skills

- Knowledge of computers and software.
- Keep abreast of modern development.
- Know of system design tools.
- Breadth knowledge about new technologies.

SYSTEM ANALYSIS & DESIGN - SYSTEM PLANNING

What is Requirements Determination?

A requirement is a vital feature of a new system which may include processing or capturing of data, controlling the activities of business, producing information and supporting the management.

Requirements determination involves studying the existing system and gathering details to find out what are the requirements, how it works, and where improvements should be made.

Major Activities in requirement Determination

Requirements Anticipation

- It predicts the characteristics of system based on previous experience which include certain problems or features and requirements for a new system.
- It can lead to analysis of areas that would otherwise go unnoticed by inexperienced analyst. But if shortcuts are taken and bias is introduced in conducting the investigation, then requirement Anticipation can be half-baked.

Requirements Investigation

- It is studying the current system and documenting its features for further analysis.
- It is at the heart of system analysis where analyst documenting and describing system features using fact-finding techniques, prototyping, and computer assisted tools.

Requirements Specifications

- It includes the analysis of data which determine the requirement specification, description of features for new system, and specifying what information requirements will be provided.

- It includes analysis of factual data, identification of essential requirements, and selection of Requirement-fulfillment strategies.

Information Gathering Techniques

The main aim of fact finding techniques is to determine the information requirements of an organization used by analysts to prepare a precise SRS understood by user.

Ideal SRS Document should –

- be complete, Unambiguous, and Jargon-free.
- specify operational, tactical, and strategic information requirements.
- solve possible disputes between users and analyst.
- use graphical aids which simplify understanding and design.

There are various information gathering techniques –

Interviewing

Systems analyst collects information from individuals or groups by interviewing. The analyst can be formal, legalistic, play politics, or be informal; as the success of an interview depends on the skill of analyst as interviewer.

It can be done in two ways –

- **Unstructured Interview** – The system analyst conducts question-answer session to acquire basic information of the system.
- **Structured Interview** – It has standard questions which user need to respond in either close *objective* or open *descriptive* format.

Advantages of Interviewing

- This method is frequently the best source of gathering qualitative information.
- It is useful for them, who do not communicate effectively in writing or who may not have the time to complete questionnaire.
- Information can easily be validated and cross checked immediately.
- It can handle the complex subjects.
- It is easy to discover key problem by seeking opinions.
- It bridges the gaps in the areas of misunderstandings and minimizes future problems.

Questionnaires

This method is used by analyst to gather information about various issues of system from large number of persons.

There are two types of questionnaires –

- **Open-ended Questionnaires** – It consists of questions that can be easily and correctly interpreted. They can explore a problem and lead to a specific direction of answer.
- **Closed-ended Questionnaires** – It consists of questions that are used when the systems analyst effectively lists all possible responses, which are mutually exclusive.

Advantages of questionnaires

- It is very effective in surveying interests, attitudes, feelings, and beliefs of users which are not co-located.
- It is useful in situation to know what proportion of a given group approves or disapproves of a particular feature of the proposed system.
- It is useful to determine the overall opinion before giving any specific direction to the system project.
- It is more reliable and provides high confidentiality of honest responses.
- It is appropriate for electing factual information and for statistical data collection which can be emailed and sent by post.

Review of Records, Procedures, and Forms

Review of existing records, procedures, and forms helps to seek insight into a system which describes the current system capabilities, its operations, or activities.

Advantages

- It helps user to gain some knowledge about the organization or operations by themselves before they impose upon others.
- It helps in documenting current operations within short span of time as the procedure manuals and forms describe the format and functions of present system.
- It can provide a clear understanding about the transactions that are handled in the organization, identifying input for processing, and evaluating performance.
- It can help an analyst to understand the system in terms of the operations that must be supported.
- It describes the problem, its affected parts, and the proposed solution.

Observation

This is a method of gathering information by noticing and observing the people, events, and objects. The analyst visits the organization to observe the working of current system and understands the requirements of the system.

Advantages

- It is a direct method for gleaning information.
- It is useful in situation where authenticity of data collected is in question or when complexity of certain aspects of system prevents clear explanation by end-users.
- It produces more accurate and reliable data.
- It produces all the aspect of documentation that are incomplete and outdated.

Joint Application Development *JAD*

It is a new technique developed by IBM which brings owners, users, analysts, designers, and builders to define and design the system using organized and intensive workshops. JAD trained analyst act as facilitator for workshop who has some specialized skills.

Advantages of JAD

- It saves time and cost by replacing months of traditional interviews and follow-up meetings.
- It is useful in organizational culture which supports joint problem solving.
- Fosters formal relationships among multiple levels of employees.
- It can lead to development of design creatively.
- It Allows rapid development and improves ownership of information system.

Secondary Research or Background Reading

This method is widely used for information gathering by accessing the gleaned information. It includes any previously gathered information used by the marketer from any internal or external source.

Advantages

- It is more openly accessed with the availability of internet.
- It provides valuable information with low cost and time.
- It act as forerunner to primary research and aligns the focus of primary research.
- It is used by the researcher to conclude if the research is worth it as it is available with procedures used and issues in collecting them.

Feasibility Study

Feasibility Study can be considered as preliminary investigation that helps the management to take decision about whether study of system should be feasible for development or not.

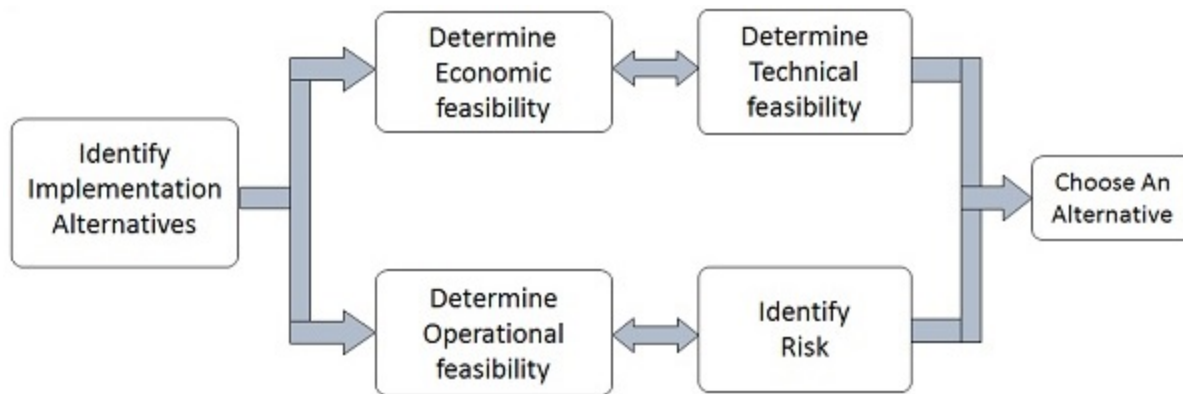
- It identifies the possibility of improving an existing system, developing a new system, and produce refined estimates for further development of system.
- It is used to obtain the outline of the problem and decide whether feasible or appropriate solution exists or not.
- The main objective of a feasibility study is to acquire problem scope instead of solving the problem.
- The output of a feasibility study is a formal system proposal act as decision document which includes the complete nature and scope of the proposed system.

Steps Involved in Feasibility Analysis

The following steps are to be followed while performing feasibility analysis –

- Form a project team and appoint a project leader.

- Develop system flowcharts.
- Identify the deficiencies of current system and set goals.
- Enumerate the alternative solution or potential candidate system to meet goals.
- Determine the feasibility of each alternative such as technical feasibility, operational feasibility, etc.
- Weight the performance and cost effectiveness of each candidate system.
- Rank the other alternatives and select the best candidate system.
- Prepare a system proposal of final project directive to management for approval.



Types of Feasibilities

Economic Feasibility

- It is evaluating the effectiveness of candidate system by using cost/benefit analysis method.
- It demonstrates the net benefit from the candidate system in terms of benefits and costs to the organization.
- The main aim of Economic Feasibility Analysis *EFS* is to estimate the economic requirements of candidate system before investments funds are committed to proposal.
- It prefers the alternative which will maximize the net worth of organization by earliest and highest return of funds along with lowest level of risk involved in developing the candidate system.

Technical Feasibility

- It investigates the technical feasibility of each implementation alternative.
- It analyzes and determines whether the solution can be supported by existing technology or not.
- The analyst determines whether current technical resources be upgraded or added it that fulfill the new requirements.
- It ensures that the candidate system provides appropriate responses to what extent it can support the technical enhancement.

Operational Feasibility

- It determines whether the system is operating effectively once it is developed and implemented.
- It ensures that the management should support the proposed system and its working feasible in the current organizational environment.
- It analyzes whether the users will be affected and they accept the modified or new business methods that affect the possible system benefits.
- It also ensures that the computer resources and network architecture of candidate system are workable.

Behavioral Feasibility

- It evaluates and estimates the user attitude or behavior towards the development of new system.
- It helps in determining if the system requires special effort to educate, retrain, transfer, and changes in employee's job status on new ways of conducting business.

Schedule Feasibility

- It ensures that the project should be completed within given time constraint or schedule.
- It also verifies and validates whether the deadlines of project are reasonable or not.

STRUCTURED ANALYSIS

Analysts use various tools to understand and describe the information system. One of the ways is using structured analysis.

What is Structured Analysis?

Structured Analysis is a development method that allows the analyst to understand the system and its activities in a logical way.

It is a systematic approach, which uses graphical tools that analyze and refine the objectives of an existing system and develop a new system specification which can be easily understandable by user.

It has following attributes –

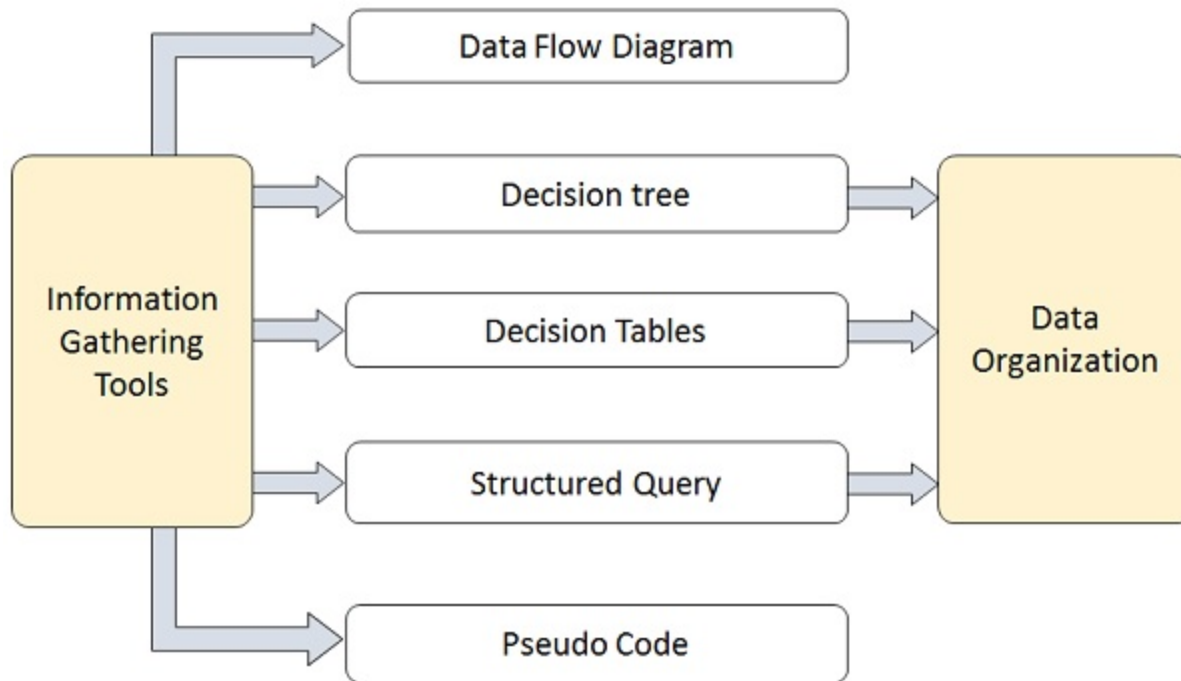
- It is graphic which specifies the presentation of application.
- It divides the processes so that it gives a clear picture of system flow.
- It is logical rather than physical i.e., the elements of system do not depend on vendor or hardware.
- It is an approach that works from high-level overviews to lower-level details.

Structured Analysis Tools

During Structured Analysis, various tools and techniques are used for system development. They are –

- Data Flow Diagrams

- Data Dictionary
- Decision Trees
- Decision Tables
- Structured English
- Pseudocode



Data Flow Diagrams *DFD* or Bubble Chart





It is a technique developed by Larry Constantine to express the requirements of system in a graphical form.

- It shows the flow of data between various functions of system and specifies how the current system is implemented.
- It is an initial stage of design phase that functionally divides the requirement specifications down to the lowest level of detail.
- Its graphical nature makes it a good communication tool between user and analyst or analyst and system designer.
- It gives an overview of what data a system processes, what transformations are performed, what data are stored, what results are produced and where they flow.

Basic Elements of DFD

DFD is easy to understand and quite effective when the required design is not clear and the user wants a notational language for communication. However, it requires a large number of iterations for obtaining the most accurate and complete solution.

The following table shows the symbols used in designing a DFD and their significance –

Symbol Name	Symbol	Meaning
Square		Source or Destination of Data
Arrow		Data flow
Circle		Process transforming data flow
Open Rectangle		Data Store

Types of DFD

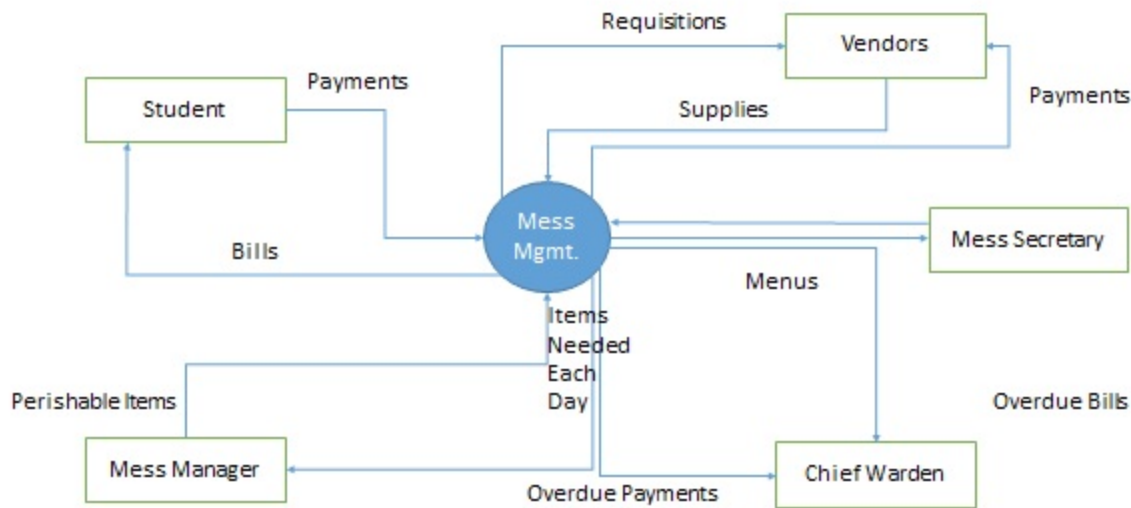
DFDs are of two types: Physical DFD and Logical DFD. The following table lists the points that differentiate a physical DFD from a logical DFD.

Physical DFD	Logical DFD
It is implementation dependent. It shows which functions are performed.	It is implementation independent. It focuses only on the flow of data between processes.
It provides low level details of hardware, software, files, and people.	It explains events of systems and data required by each event.
It depicts how the current system operates and how a system will be implemented.	It shows how business operates; not how the system can be implemented.

Context Diagram

A context diagram helps in understanding the entire system by one DFD which gives the overview of a system. It starts with mentioning major processes with little details and then goes onto giving more details of the processes with the top-down approach.

The context diagram of mess management is shown below.



Data Dictionary

A data dictionary is a structured repository of data elements in the system. It stores the descriptions of all DFD data elements that is, details and definitions of data flows, data stores, data stored in data stores, and the processes.

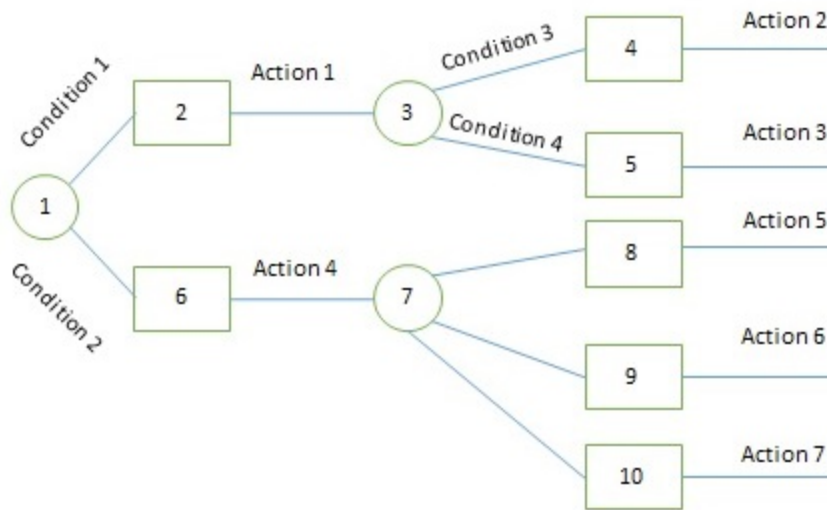
A data dictionary improves the communication between the analyst and the user. It plays an important role in building a database. Most DBMSs have a data dictionary as a standard feature. For example, refer the following table –

Sr.No.	Data Name	Description	No. of Characters
1	ISBN	ISBN Number	10
2	TITLE	title	60
3	SUB	Book Subjects	80
4	ANAME	Author Name	15

Decision Trees

Decision trees are a method for defining complex relationships by describing decisions and avoiding the problems in communication. A decision tree is a diagram that shows alternative actions and conditions within horizontal tree framework. Thus, it depicts which conditions to consider first, second, and so on.

Decision trees depict the relationship of each condition and their permissible actions. A square node indicates an action and a circle indicates a condition. It forces analysts to consider the sequence of decisions and identifies the actual decision that must be made.



The major limitation of a decision tree is that it lacks information in its format to describe what other combinations of conditions you can take for testing. It is a single representation of the relationships between conditions and actions.

For example, refer the following decision tree –



Decision Tables

Decision tables are a method of describing the complex logical relationship in a precise manner which is easily understandable.

- It is useful in situations where the resulting actions depend on the occurrence of one or several combinations of independent conditions.
- It is a matrix containing row or columns for defining a problem and the actions.

Components of a Decision Table

- **Condition Stub** – It is in the upper left quadrant which lists all the condition to be checked.

- **Action Stub** – It is in the lower left quadrant which outlines all the action to be carried out to meet such condition.
- **Condition Entry** – It is in upper right quadrant which provides answers to questions asked in condition stub quadrant.
- **Action Entry** – It is in lower right quadrant which indicates the appropriate action resulting from the answers to the conditions in the condition entry quadrant.

The entries in decision table are given by Decision Rules which define the relationships between combinations of conditions and courses of action. In rules section,

- Y shows the existence of a condition.
- N represents the condition, which is not satisfied.
- A blank - against action states it is to be ignored.
- X or a checkmark will do against action states it is to be carried out.

For example, refer the following table –

CONDITIONS	Rule 1	Rule 2	Rule 3	Rule 4
Advance payment made	Y	N	N	N
Purchase amount = Rs 10,000/-	-	Y	Y	N
Regular Customer	-	Y	N	-
ACTIONS				
Give 5% discount	X	X	-	-
Give no discount	-	-	X	X

Structured English

Structure English is derived from structured programming language which gives more understandable and precise description of process. It is based on procedural logic that uses construction and imperative sentences designed to perform operation for action.

- It is best used when sequences and loops in a program must be considered and the problem needs sequences of actions with decisions.

- It does not have strict syntax rule. It expresses all logic in terms of sequential decision structures and iterations.

For example, see the following sequence of actions –

```

if customer pays advance
then
    Give 5% Discount
else
    if purchase amount >=10,000
    then
        if the customer is a regular customer
        then Give 5% Discount
        else No Discount
        end if
    else No Discount
    end if
end if

```

Pseudocode

A pseudocode does not conform to any programming language and expresses logic in plain English.

- It may specify the physical programming logic without actual coding during and after the physical design.
- It is used in conjunction with structured programming.
- It replaces the flowcharts of a program.

Guidelines for Selecting Appropriate Tools

Use the following guidelines for selecting the most appropriate tool that would suit your requirements –

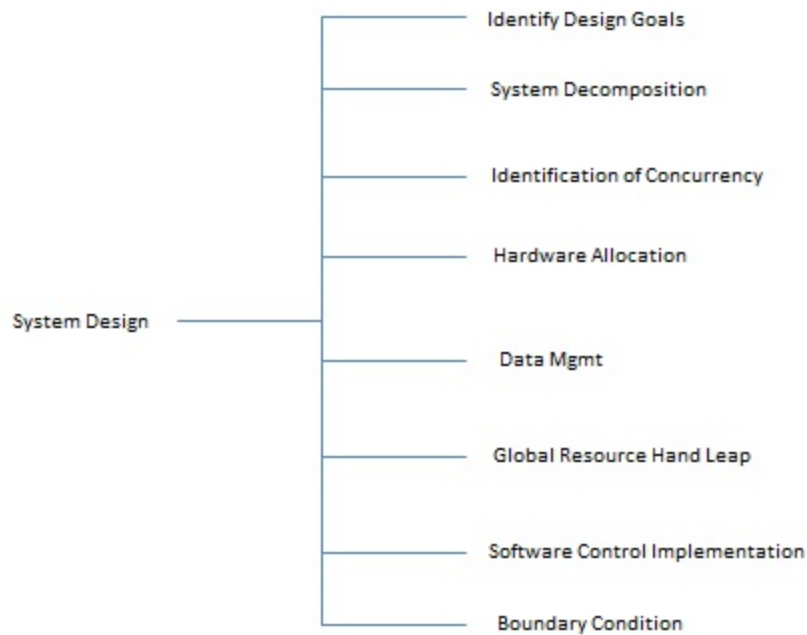
- Use DFD at high or low level analysis for providing good system documentations.
- Use data dictionary to simplify the structure for meeting the data requirement of the system.
- Use structured English if there are many loops and actions are complex.
- Use decision tables when there are a large number of conditions to check and logic is complex.
- Use decision trees when sequencing of conditions is important and if there are few conditions to be tested.

SYSTEM ANALYSIS & DESIGN - SYSTEM DESIGN

System design is the phase that bridges the gap between problem domain and the existing system in a manageable way. This phase focuses on the solution domain, i.e. *“how to implement?”*

It is the phase where the SRS document is converted into a format that can be implemented and decides how the system will operate.

In this phase, the complex activity of system development is divided into several smaller sub-activities, which coordinate with each other to achieve the main objective of system development.



Inputs to System Design

System design takes the following inputs –

- Statement of work
- Requirement determination plan
- Current situation analysis
- Proposed system requirements including a conceptual data model, modified DFDs, and Metadata *dataaboutdata*.

Outputs for System Design

System design gives the following outputs –

- Infrastructure and organizational changes for the proposed system.
- A data schema, often a relational schema.
- Metadata to define the tables/files and columns/data-items.
- A function hierarchy diagram or web page map that graphically describes the program structure.
- Actual or pseudocode for each module in the program.
- A prototype for the proposed system.

Types of System Design

Logical Design

Logical design pertains to an abstract representation of the data flow, inputs, and outputs of the system. It describes the inputs *sources*, outputs *destinations*, databases *datastores*, procedures *dataflows* all in a format that meets the user requirements.

While preparing the logical design of a system, the system analyst specifies the user needs at level of detail that virtually determines the information flow into and out of the system and the required data sources. Data flow diagram, E-R diagram modeling are used.

Physical Design

Physical design relates to the actual input and output processes of the system. It focuses on how data is entered into a system, verified, processed, and displayed as output.

It produces the working system by defining the design specification that specifies exactly what the candidate system does. It is concerned with user interface design, process design, and data design.

It consists of the following steps –

- Specifying the input/output media, designing the database, and specifying backup procedures.
- Planning system implementation.
- Devising a test and implementation plan, and specifying any new hardware and software.
- Updating costs, benefits, conversion dates, and system constraints.

Architectural Design

It is also known as high level design that focuses on the design of system architecture. It describes the structure and behavior of the system. It defines the structure and relationship between various modules of system development process.

Detailed Design

It follows Architectural design and focuses on development of each module.

Conceptual Data Modeling

It is representation of organizational data which includes all the major entities and relationship. System analysts develop a conceptual data model for the current system that supports the scope and requirement for the proposed system.

The main aim of conceptual data modeling is to capture as much meaning of data as possible. Most organization today use conceptual data modeling using E-R model which uses special notation to represent as much meaning about data as possible.







Entity Relationship Model




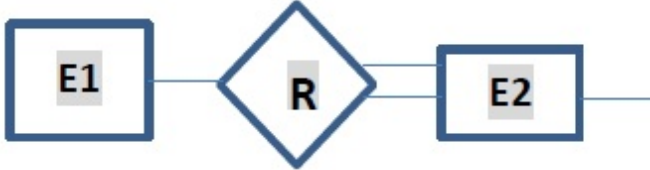
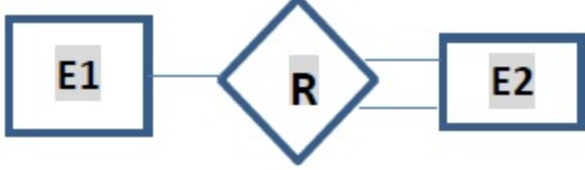
It is a technique used in database design that helps describe the relationship between various entities of an organization.

Terms used in E-R model

- **ENTITY** – It specifies distinct real world items in an application. For example: vendor, item, student, course, teachers, etc.
 - **RELATIONSHIP** – They are the meaningful dependencies between entities. For example, vendor supplies items, teacher teaches courses, then supplies and course are relationship.
 - **ATTRIBUTES** – It specifies the properties of relationships. For example, vendor code, student name.
- Symbols used in E-R model and their respective meanings –

The following table shows the symbols used in E-R model and their significance –

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identity Relationship
	Attributes
	Key Attributes

	Multivalued
	Composite Attribute
	Derived Attributes
	Total Participation of E2 in R
	Cardinality Ratio 1:N for E1:E2 in R

Three types of relationships can exist between two sets of data: one-to-one, one-to-many, and many-to-many.

File Organization

It describes how records are stored within a file.

There are four file organization methods –

- **Serial** – Records are stored in chronological order *in order as they are input or occur*. **Examples** – Recording of telephone charges, ATM transactions, Telephone queues.
- **Sequential** – Records are stored in order based on a key field which contains a value that uniquely identifies a record. **Examples** – Phone directories.
- **Direct relative** – Each record is stored based on a physical address or location on the device. Address is calculated from the value stored in the record's key field. Randomizing routine or hashing algorithm does the conversion.

- **Indexed** – Records can be processed both sequentially and non-sequentially using indexes.

Comparison

	Serial	Sequential	Direct	Index
Type of Access	Batch	Batch	Online	Batch or Online
Data Organization	Serial	Sequentially by key value	No particular order	Sequentially and by index
Flexibility in handling inquiries	No	No	Yes	Yes
Availability of up to date Data	No	No	Yes	Yes
Speed Retrieval	Slow	Slow	Very Fast	Fast
Activity	High	High	Low	High
Volatility	Low	Low	High	High
Example	ATM Transition Queue	Payroll process script billing operation	Online reservation and banking transaction	Customer ordering and billing

File Access

One can access a file using either Sequential Access or Random Access. File Access methods allow computer programs read or write records in a file.

Sequential Access

Every record on the file is processed starting with the first record until End of File *EOF* is reached. It is efficient when a large number of the records on the file need to be accessed at any given time. Data stored on a tape *sequentialaccess* can be accessed only sequentially.

Direct Random Access

Records are located by knowing their physical locations or addresses on the device rather than their positions relative to other records. Data stored on a CD device *direct – access* can be accessed either sequentially or randomly.

Types of Files used in an Organization System

Following are the types of files used in an organization system –

- **Master file** – It contains the current information for a system. For example, customer file, student file, telephone directory.

- **Table file** – It is a type of master file that changes infrequently and stored in a tabular format. For example, storing Zipcode.
- **Transaction file** – It contains the day-to-day information generated from business activities. It is used to update or process the master file. For example, Addresses of the employees.
- **Temporary file** – It is created and used whenever needed by a system.
- **Mirror file** – They are the exact duplicates of other files. Help minimize the risk of downtime in cases when the original becomes unusable. They must be modified each time the original file is changed.
- **Log files** – They contain copies of master and transaction records in order to chronicle any changes that are made to the master file. It facilitates auditing and provides mechanism for recovery in case of system failure.
- **Archive files** – Backup files that contain historical versions of other files.

Documentation Control

Documentation is a process of recording the information for any reference or operational purpose. It helps users, managers, and IT staff, who require it. It is important that prepared document must be updated on regular basis to trace the progress of the system easily.

After the implementation of system if the system is working improperly, then documentation helps the administrator to understand the flow of data in the system to correct the flaws and get the system working.

Programmers or systems analysts usually create program and system documentation. Systems analysts usually are responsible for preparing documentation to help users learn the system. In large companies, a technical support team that includes technical writers might assist in the preparation of user documentation and training materials.

Advantages

- It can reduce system downtime, cut costs, and speed up maintenance tasks.
- It provides the clear description of formal flow of present system and helps to understand the type of input data and how the output can be produced.
- It provides effective and efficient way of communication between technical and nontechnical users about system.
- It facilitates the training of new user so that he can easily understand the flow of system.
- It helps the user to solve the problems such as troubleshooting and helps the manager to take better final decisions of the organization system.
- It provides better control to the internal or external working of the system.

Types of Documentations

When it comes to System Design, there are following four main documentations –

- Program documentation
- System documentation

- Operations documentation
- User documentation

Program Documentation

- It describes inputs, outputs, and processing logic for all the program modules.
- The program documentation process starts in the system analysis phase and continues during implementation.
- This documentation guides programmers, who construct modules that are well supported by internal and external comments and descriptions that can be understood and maintained easily.

Operations Documentation

Operations documentation contains all the information needed for processing and distributing online and printed output. Operations documentation should be clear, concise, and available online if possible.

It includes the following information –

- Program, systems analyst, programmer, and system identification.
- Scheduling information for printed output, such as report, execution frequency, and deadlines.
- Input files, their source, output files, and their destinations.
- E-mail and report distribution lists.
- Special forms required, including online forms.
- Error and informational messages to operators and restart procedures.
- Special instructions, such as security requirements.

User Documentation

It includes instructions and information to the users who will interact with the system. For example, user manuals, help guides, and tutorials. User documentation is valuable in training users and for reference purpose. It must be clear, understandable, and readily accessible to users at all levels.

The users, system owners, analysts, and programmers, all put combined efforts to develop a user's guide.

A user documentation should include –

- A system overview that clearly describes all major system features, capabilities, and limitations.
- Description of source document content, preparation, processing, and, samples.
- Overview of menu and data entry screen options, contents, and processing instructions.
- Examples of reports that are produced regularly or available at the user's request, including samples.
- Security and audit trail information.
- Explanation of responsibility for specific input, output, or processing requirements.

- Procedures for requesting changes and reporting problems.
- Examples of exceptions and error situations.
- Frequently asked questions *FAQs*.
- Explanation of how to get help and procedures for updating the user manual.

System Documentation

System documentation serves as the technical specifications for the IS and how the objectives of the IS are accomplished. Users, managers and IS owners need never reference system documentation. System documentation provides the basis for understanding the technical aspects of the IS when modifications are made.

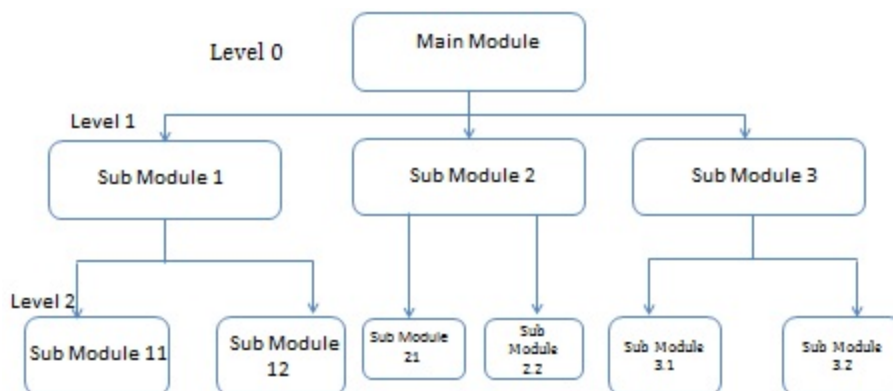
- It describes each program within the IS and the entire IS itself.
- It describes the system's functions, the way they are implemented, each program's purpose within the entire IS with respect to the order of execution, information passed to and from programs, and overall system flow.
- It includes data dictionary entries, data flow diagrams, object models, screen layouts, source documents, and the systems request that initiated the project.
- Most of the system documentation is prepared during the system analysis and system design phases.
- During systems implementation, an analyst must review system documentation to verify that it is complete, accurate, and up-to-date, and including any changes made during the implementation process.

DESIGN STRATEGIES

Top-Down Strategy

The top-down strategy uses the modular approach to develop the design of a system. It is called so because it starts from the top or the highest-level module and moves towards the lowest level modules.

In this technique, the highest-level module or main module for developing the software is identified. The main module is divided into several smaller and simpler submodules or segments based on the task performed by each module. Then, each submodule is further subdivided into several submodules of next lower level. This process of dividing each module into several submodules continues until the lowest level modules, which cannot be further subdivided, are not identified.

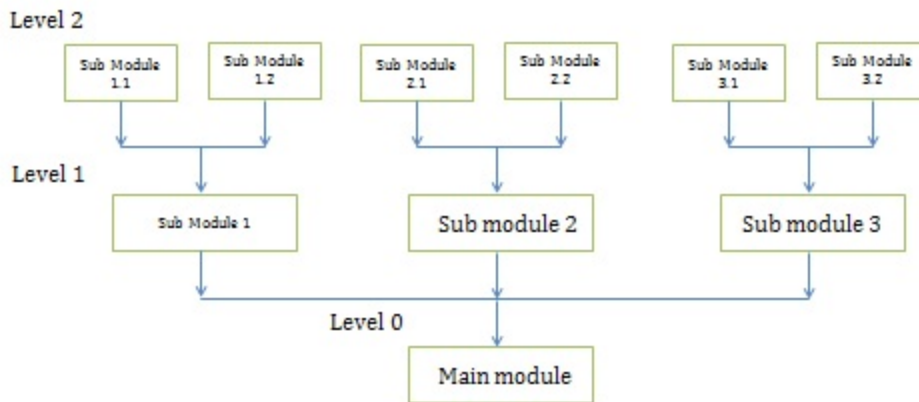


Bottom-Up Strategy

Bottom-Up Strategy follows the modular approach to develop the design of the system. It is called so because it starts from the bottom or the most basic level modules and moves towards the highest level modules.

In this technique,

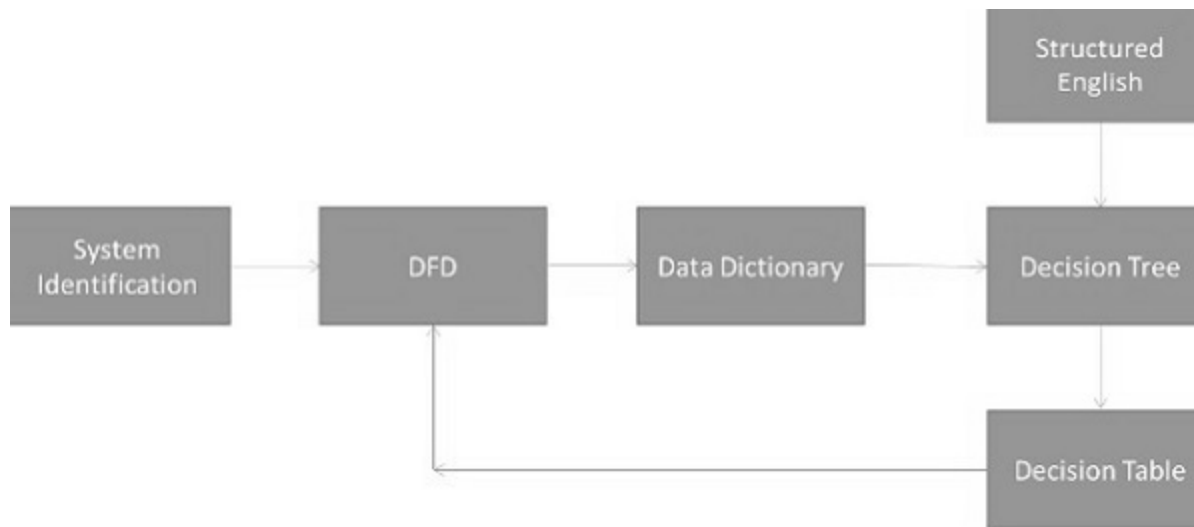
- The modules at the most basic or the lowest level are identified.
- These modules are then grouped together based on the function performed by each module to form the next higher-level modules.
- Then, these modules are further combined to form the next higher-level modules.
- This process of grouping several simpler modules to form higher level modules continues until the main module of system development process is achieved.



Structured Design

Structured design is a data-flow based methodology that helps in identifying the input and output of the developing system. The main objective of structured design is to minimize the complexity and increase the modularity of a program. Structured design also helps in describing the functional aspects of the system.

In structured designing, the system specifications act as a basis for graphically representing the flow of data and sequence of processes involved in a software development with the help of DFDs. After developing the DFDs for the software system, the next step is to develop the structure chart.



Modularization

Structured design partitions the program into small and independent modules. These are organized in top down manner with the details shown in bottom.

Thus, structured design uses an approach called Modularization or decomposition to minimize the complexity and to manage the problem by subdividing it into smaller segments.

Advantages

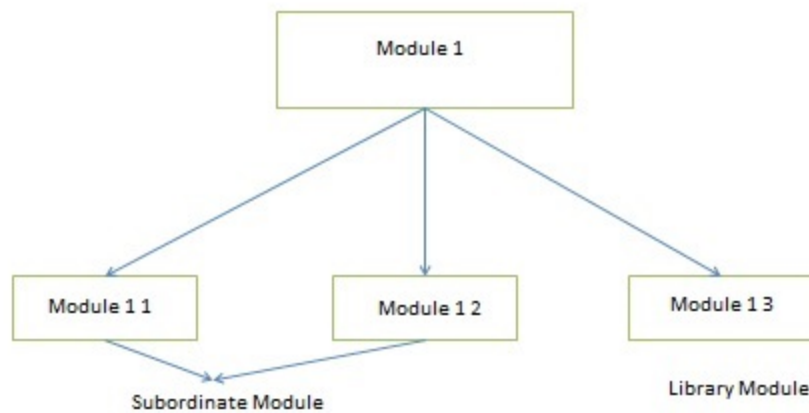
- Critical interfaces are tested first.
- It provide abstraction.
- It allows multiple programmers to work simultaneously.
- It allows code reuse.
- It provides control and improves morale.
- It makes identifying structure easier.

Structured Charts

Structured charts are a recommended tool for designing a modular, top down systems which define the various modules of system development and the relationship between each module. It shows the system module and their relationship between them.

It consists of diagram consisting of rectangular boxes that represent the modules, connecting arrows, or lines.

- **Control Module** – It is a higher-level module that directs lower-level modules, called **subordinate modules**.
- **Library Module** – It is a reusable module and can be invoked from more than one point in the chart.



We have two different approaches to design a structured chart –

- **Transform-Centered Structured Charts** – They are used when all the transactions follow same path.
- **Transaction-Centered Structured Charts** – They are used when all the transactions do not follow the same path.

Objectives of Using Structure Flowcharts

- To encourage a top-down design.
- To support the concept of modules and identify the appropriate modules.
- To show the size and complexity of the system.
- To identify the number of readily identifiable functions and modules within each function.
- To depict whether each identifiable function is a manageable entity or should be broken down into smaller components.

Factors Affecting System Complexity

To develop good quality of system software, it is necessary to develop a good design. Therefore, the main focus on while developing the design of the system is the quality of the software design. A good quality software design is the one, which minimizes the complexity and cost expenditure in software development.

The two important concepts related to the system development that help in determining the complexity of a system are **coupling** and **cohesion**.

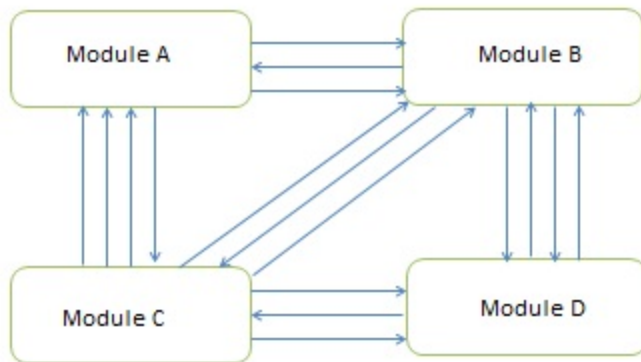
Coupling

Coupling is the measure of the independence of components. It defines the degree of dependency of each module of system development on the other. In practice, this means the stronger the coupling between the modules in a system, the more difficult it is to implement and maintain the system.

Each module should have simple, clean interface with other modules, and that the minimum number of data elements should be shared between modules.

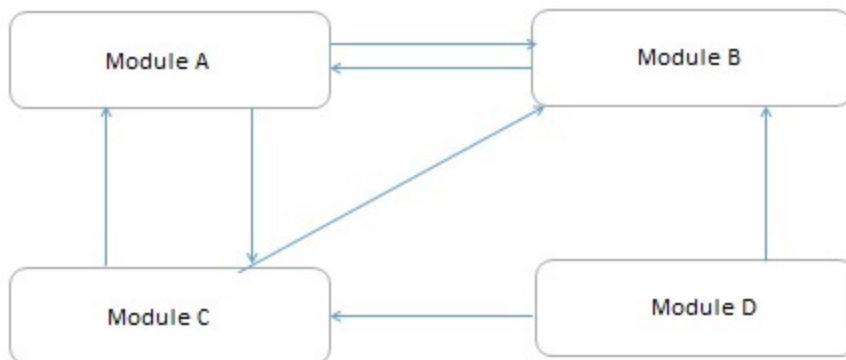
High Coupling

These type of systems have interconnections with program units dependent on each other. Changes to one subsystem leads to high impact on the other subsystem.



Low Coupling

These type of systems are made up of components which are independent or almost independent. A change in one subsystem does not affect any other subsystem.



Coupling Measures

- **Content Coupling** – When one component actually modifies another, then the modified component is completely dependent on modifying one.

- **Common Coupling** – When amount of coupling is reduced somewhat by organizing system design so that data are accessible from a common data store.
- **Control Coupling** – When one component passes parameters to control the activity of another component.
- **Stamp Coupling** – When data structures is used to pass information from one component to another.
- **Data Coupling** – When only data is passed then components are connected by this coupling.

Cohesion

Cohesion is the measure of closeness of the relationship between its components. It defines the amount of dependency of the components of a module on one another. In practice, this means the systems designer must ensure that –

- They do not split essential processes into fragmented modules.
- They do not gather together unrelated processes represented as processes on the DFD into meaningless modules.

The best modules are those that are functionally cohesive. The worst modules are those that are coincidentally cohesive.

The worst degree of cohesion

Coincidental cohesion is found in a component whose parts are unrelated to another.

- **Logical Cohesion** – It is where several logically related functions or data elements are placed in same component.
- **Temporal Cohesion** – It is when a component that is used to initialize a system or set variables performs several functions in sequence, but the functions are related by timing involved.
- **Procedurally Cohesion** – It is when functions are grouped together in a component just to ensure this order.
- **Sequential Cohesion** – It is when the output from one part of a component is the input to the next part of it.

INPUT / OUTPUT & FORMS DESIGN

Input Design

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well designed input forms and screens have following properties –

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- It ensures proper completion with accuracy.

- It should be easy to fill and straightforward.
- It should focus on user's attention, consistency, and simplicity.
- All these objectives are obtained using the knowledge of basic design principles regarding –
 - What are the inputs needed for the system?
 - How end users respond to different elements of forms and screens.

Objectives for Input Design

The objectives of input design are –

- To design data entry and input procedures
- To reduce input volume
- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.
- To use validation checks and develop effective input controls.

Data Input Methods

It is important to design appropriate data input methods to prevent errors while entering data. These methods depend on whether the data is entered by customers in forms manually and later entered by data entry operators, or data is directly entered by users on the PCs.

A system should prevent user from making mistakes by –

- Clear form design by leaving enough space for writing legibly.
- Clear instructions to fill form.
- Clear form design.
- Reducing key strokes.
- Immediate error feedback.

Some of the popular data input methods are –

- Batch input method *Offline data input method*
- Online data input method
- Computer readable forms
- Interactive data input

Input Integrity Controls

Input integrity controls include a number of methods to eliminate common input errors by end-users. They also include checks on the value of individual fields; both for format and the completeness of all inputs.

Audit trails for data entry and other system operations are created using transaction logs which gives a record of all changes introduced in the database to provide security and means of recovery in case of any failure.

Output Design

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design

The objectives of input design are –

- To develop output design that serves the intended purpose and eliminates the production of unwanted output.
- To develop the output design that meets the end users requirements.
- To deliver the appropriate quantity of output.
- To form the output in appropriate format and direct it to the right person.
- To make the output available on time for making good decisions.

Let us now go through various types of outputs –

External Outputs

Manufacturers create and design external outputs for printers. External outputs enable the system to leave the trigger actions on the part of their recipients or confirm actions to their recipients.

Some of the external outputs are designed as turnaround outputs, which are implemented as a form and re-enter the system as an input.

Internal outputs

Internal outputs are present inside the system, and used by end-users and managers. They support the management in decision making and reporting.

There are three types of reports produced by management information –

- **Detailed Reports** – They contain present information which has almost no filtering or restriction generated to assist management planning and control.
- **Summary Reports** – They contain trends and potential problems which are categorized and summarized that are generated for managers who do not want details.
- **Exception Reports** – They contain exceptions, filtered data to some condition or standard before presenting it to the manager, as information.

Output Integrity Controls

Output integrity controls include routing codes to identify the receiving system, and verification messages to confirm successful receipt of messages that are handled by network protocol.

Printed or screen-format reports should include a date/time for report printing and the data. Multipage reports contain report title or description, and pagination. Pre-printed forms usually include a version number and effective date.

Forms Design

Both forms and reports are the product of input and output design and are business document consisting of specified data. The main difference is that forms provide fields for data input but reports are purely used for reading. For example, order forms, employment and credit application, etc.

- During form designing, the designers should know –
 - who will use them
 - where would they be delivered
 - the purpose of the form or report
- During form design, automated design tools enhance the developer's ability to prototype forms and reports and present them to end users for evaluation.

Objectives of Good Form Design

A good form design is necessary to ensure the following –

- To keep the screen simple by giving proper sequence, information, and clear captions.
- To meet the intended purpose by using appropriate forms.
- To ensure the completion of form with accuracy.
- To keep the forms attractive by using icons, inverse video, or blinking cursors etc.
- To facilitate navigation.

Types of Forms

Flat Forms

- It is a single copy form prepared manually or by a machine and printed on a paper. For additional copies of the original, carbon papers are inserted between copies.
- It is a simplest and inexpensive form to design, print, and reproduce, which uses less volume.

Unit Set/Snap out Forms

- These are papers with one-time carbons interleaved into unit sets for either handwritten or machine use.
- Carbons may be either blue or black, standard grade medium intensity. Generally, blue carbons are best for handwritten forms while black carbons are best for machine use.

Continuous strip/Fanfold Forms

- These are multiple unit forms joined in a continuous strip with perforations between each pair of forms.

- It is a less expensive method for large volume use.

No Carbon Required *NCR* Paper

- They use carbonless papers which have two chemical coatings *capsules*, one on the face and the other on the back of a sheet of paper.
- When pressure is applied, the two capsules interact and create an image.

TESTING AND QUALITY ASSURANCE

The software system needs to be checked for its intended behavior and direction of progress at each development stage to avoid duplication of efforts, time and cost overruns, and to assure completion of the system within stipulated time. The software system needs to be checked for its intended behavior and direction of progress at each development stage to avoid duplication of efforts, time and cost overruns, and to assure completion of the system within stipulated time.

System testing and quality assurance come to aid for checking the system. It includes –

- Product level quality *Testing*
- Process level quality.

Let us go through them briefly –

Testing

Testing is the process or activity that checks the functionality and correctness of software according to specified user requirements in order to improve the quality and reliability of system. It is an expensive, time consuming, and critical approach in system development which requires proper planning of overall testing process.

A successful test is one that finds the errors. It executes the program with explicit intention of finding error, i.e., making the program fail. It is a process of evaluating system with an intention of creating a strong system and mainly focuses on the weak areas of the system or software.

Characteristics of System Testing

System testing begins at the module level and proceeds towards the integration of the entire software system. Different testing techniques are used at different times while testing the system. It is conducted by the developer for small projects and by independent testing groups for large projects.

Stages of System Testing

The following stages are involved in testing –

Test Strategy

It is a statement that provides information about the various levels, methods, tools, and techniques used for testing the system. It should satisfy all the needs of an organization.

Test Plan

It provides a plan for testing the system and verifies that the system under testing fulfils all the design and functional specifications. The test plan provides the following information –

- Objectives of each test phase
- Approaches and tools used for testing
- Responsibilities and time required for each testing activity
- Availability of tools, facilities, and test libraries
- Procedures and standards required for planning and conducting the tests
- Factors responsible for successful completion of testing process

Test Case Design

- Test cases are used to uncover as many errors as possible in the system.
- A number of test cases are identified for each module of the system to be tested.
- Each test case will specify how the implementation of a particular requirement or design decision is to be tested and the criteria for the success of the test.
- The test cases along with the test plan are documented as a part of a system specification document or in a separate document called **test specification** or **test description**.

Test Procedures

It consists of the steps that should be followed to execute each of the test cases. These procedures are specified in a separate document called test procedure specification. This document also specifies any special requirements and formats for reporting the result of testing.

Test Result Documentation

Test result file contains brief information about the total number of test cases executed, the number of errors, and nature of errors. These results are then assessed against criteria in the test specification to determine the overall outcome of the test.

Types of Testing

Testing can be of various types and different types of tests are conducted depending on the kind of bugs one seeks to discover –

Unit Testing

Also known as Program Testing, it is a type of testing where the analyst tests or focuses on each program or module independently. It is carried out with the intention of executing each statement of the module at least once.

- In unit testing, accuracy of program cannot be assured and it is difficult to conduct testing of various input combination in detail.
- It identifies maximum errors in a program as compared to other testing techniques.

Integration Testing

In Integration Testing, the analyst tests multiple module working together. It is used to find discrepancies between the system and its original objective, current specifications, and systems documentation.

- Here the analysts are try to find areas where modules have been designed with different specifications for data length, type, and data element name.
- It verifies that file sizes are adequate and that indices have been built properly.

Functional Testing

Function testing determines whether the system is functioning correctly according to its specifications and relevant standards documentation. Functional testing typically starts with the implementation of the system, which is very critical for the success of the system.

Functional testing is divided into two categories –

- **Positive Functional Testing** – It involves testing the system with valid inputs to verify that the outputs produced are correct.
- **Negative Functional Testing** – It involves testing the software with invalid inputs and undesired operating conditions.

Rules for System Testing

To carry out system testing successfully, you need to follow the given rules –

- Testing should be based on the requirements of user.
- Before writing testing scripts, understand the business logic should be understood thoroughly.
- Test plan should be done as soon as possible.
- Testing should be done by the third party.
- It should be performed on static software.
- Testing should be done for valid and invalid input conditions.
- Testing should be reviewed and examined to reduce the costs.
- Both static and dynamic testing should be conducted on the software.
- Documentation of test cases and test results should be done.

Quality Assurance

It is the review of system or software products and its documentation for assurance that system meets the requirements and specifications.

- Purpose of QA is to provide confidence to the customers by constant delivery of product according to specification.
- Software quality Assurance *SQA* is a techniques that includes procedures and tools applied by the software professionals to ensure that software meet the specified standard for its intended use and performance.
- The main aim of SQA is to provide proper and accurate visibility of software project and its developed product to the administration.

- It reviews and audits the software product and its activities throughout the life cycle of system development.

Objectives of Quality Assurance

The objectives of conducting quality assurance are as follows –

- To monitor the software development process and the final software developed.
- To ensure whether the software project is implementing the standards and procedures set by the management.
- To notify groups and individuals about the SQA activities and results of these activities.
- To ensure that the issues, which are not solved within the software are addressed by the upper management.
- To identify deficiencies in the product, process, or the standards, and fix them.

Levels of Quality Assurance

There are several levels of QA and testing that need to be performed in order to certify a software product.

Level 1 – Code Walk-through

At this level, offline software is examined or checked for any violations of the official coding rules. In general, the emphasis is placed on examination of the documentation and level of in-code comments.

Level 2 – Compilation and Linking

At this level, it is checked that the software can compile and link all official platforms and operating systems.

Level 3 – Routine Running

At this level, it is checked that the software can run properly under a variety of conditions such as certain number of events and small and large event sizes etc.

Level 4 – Performance test

At this final level, it is checked that the performance of the software satisfies the previously specified performance level.

SYSTEM IMPLEMENTATION AND MAINTENANCE

Implementation is a process of ensuring that the information system is operational. It involves –

- Constructing a new system from scratch
- Constructing a new system from the existing one.

Implementation allows the users to take over its operation for use and evaluation. It involves training the users to handle the system and plan for a smooth conversion.

Training

The personnel in 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. The success or failure of well designed and technically elegant systems can depend on the way they are operated and used.

Training Systems Operators

Systems operators must be trained properly such that they can handle all possible operations, both routine and extraordinary. The operators should be trained in what common malfunctions may occur, how to recognize them, and what steps to take when they come.

Training involves creating troubleshooting lists to identify 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.

User Training

- End-user training is an important part of the computer-based information system development, which must be provided to employees to enable them to do their own problem solving.
- User training involves how to operate the equipment, troubleshooting the system problem, determining whether a problem that arose is caused by the equipment or software.
- Most user training deals with the operation of the system itself. The training courses must be designed to help the user with fast mobilization for the organization.

Training Guidelines

- Establishing measurable objectives
- Using appropriate training methods
- Selecting suitable training sites
- Employing understandable training materials

Training Methods

Instructor-led training

It involves both trainers and trainees, who have to meet at the same time, but not necessarily at the same place. The training session could be one-on-one or collaborative. It is of two types –

Virtual Classroom

In this training, trainers must meet the trainees at the same time, but are not required to be at the same place. The primary tools used here are: video conferencing, text based Internet relay chat tools, or virtual reality packages, etc.

Normal Classroom

The trainers must meet the trainees at the same time and at the same place. Their primary tools used here are blackboard, overhead projectors, LCD projector, etc.

Self-Paced Training

It involves both trainers and trainees, who do not need to meet at the same place or at the same time. The trainees learn the skills themselves by accessing the courses at their own convenience. It is of two types –

Multimedia Training

In this training, courses are presented in multimedia format and stored on CD-ROM. It minimizes the cost in developing an in-house training course without assistance from external programmers.

Web-based Training

In this training, courses are often presented in hyper media format and developed to support internet and intranet. It provides just-in-time training for end users and allow organization to tailor training requirements.

Conversion

It is a process of migrating from the old system to the new one. It provides understandable and structured approach to improve the communication between management and project team.

Conversion Plan

It contains description of all the activities that must occur during implementation of the new system and put it into operation. It anticipates possible problems and solutions to deal with them.

It includes the following activities –

- Name all files for conversions.
- Identifying the data requirements to develop new files during conversion.
- Listing all the new documents and procedures that are required.
- Identifying the controls to be used in each activity.
- Identifying the responsibility of person for each activity.
- Verifying conversion schedules.

Conversion Methods

The four methods of conversion are –

- Parallel Conversion
- Direct Cutover Conversion
- Pilot Approach
- Phase-In Method

Method	Description	Advantages	Disadvantages
Parallel		Provides fallback when	Causes cost

Conversion	Old and new systems are used simultaneously.	new system fails. Offers greatest security and ultimately testing of new system.	overruns. New system may not get fair trail.
Direct Cutover Conversion	New system is implemented and old system is replaced completely.	Forces users to make new system work Immediate benefit from new methods and control.	No fall back if problems arise with new system Requires most careful planning
Pilot Approach	Supports phased approach that gradually implement system across all users	Allows training and installation without unnecessary use of resources. Avoid large contingencies from risk management.	A long term phase in causes a problem of whether conversion goes well or not.
Phase-In Method	Working version of system implemented in one part of organization based on feedback, it is installed throughout the organization all alone or stage by stage.	Provides experience and line test before implementation When preferred new system involves new technology or drastic changes in performance.	Gives impression that old system is erroneous and it is not reliable.

File Conversion

It is a process of converting one file format into another. For example, file in WordPerfect format can be converted into Microsoft Word.

For successful conversion, a conversion plan is required, which includes –

- Knowledge of the target system and understanding of the present system

- Teamwork
- Automated methods, testing and parallel operations
- Continuous support for correcting problems
- Updating systems/user documentation, etc

Many popular applications support opening and saving to other file formats of the same type. For example, Microsoft Word can open and save files in many other word processing formats.

Post-Implementation Evaluation Review *PIER*

PIER is a tool or standard approach for evaluating the outcome of the project and determine whether the project is producing the expected benefits to the processes, products or services. It enables the user to verify that the project or system has achieved its desired outcome within specified time period and planned cost.

PIER ensures that the project has met its goals by evaluating the development and management processes of the project.

Objectives of PIER

The objectives of having a PIER are as follows –

- To determine the success of a project against the projected costs, benefits, and timelines.
- To identify the opportunities to add additional value to the project.
- To determine strengths and weaknesses of the project for future reference and appropriate action.
- To make recommendations on the future of the project by refining cost estimating techniques.

The following staff members should be included in the review process –

- Project team and Management
- User staff
- Strategic Management Staff
- External users

System Maintenance / Enhancement

Maintenance means restoring something to its original conditions. Enhancement means adding, modifying the code to support the changes in the user specification. System maintenance conforms the system to its original requirements and enhancement adds to system capability by incorporating new requirements.

Thus, maintenance changes the existing system, enhancement adds features to the existing system, and development replaces the existing system. It is an important part of system development that includes the activities which corrects errors in system design and implementation, updates the documents, and tests the data.

Maintenance Types

System maintenance can be classified into three types –

- **Corrective Maintenance** – Enables user to carry out the repairing and correcting leftover problems.
- **Adaptive Maintenance** – Enables user to replace the functions of the programs.
- **Perfective Maintenance** – Enables user to modify or enhance the programs according to the users' requirements and changing needs.

SYSTEM SECURITY AND AUDIT

System Audit

It is an investigation to review the performance of an operational system. The objectives of conducting a system audit are as follows –

- To compare actual and planned performance.
- To verify that the stated objectives of system are still valid in current environment.
- To evaluate the achievement of stated objectives.
- To ensure the reliability of computer based financial and other information.
- To ensure all records included while processing.
- To ensure protection from frauds.

Audit of Computer System Usage

Data processing auditors audits the usage of computer system in order to control it. The auditor need control data which is obtained by computer system itself.

The System Auditor

The role of auditor begins at the initial stage of system development so that resulting system is secure. It describes an idea of utilization of system that can be recorded which helps in load planning and deciding on hardware and software specifications. It gives an indication of wise use of the computer system and possible misuse of the system.

Audit Trial

An audit trial or audit log is a security record which is comprised of who has accessed a computer system and what operations are performed during a given period of time. Audit trials are used to do detailed tracing of how data on the system has changed.

It provides documentary evidence of various control techniques that a transaction is subject to during its processing. Audit trials do not exist independently. They are carried out as a part of accounting for recovering lost transactions.

Audit Methods

Auditing can be done in two different ways –

Auditing around the Computer

- Take sample inputs and manually apply processing rules.
- Compare outputs with computer outputs.

Auditing through the Computer

- Establish audit trail which allows examining selected intermediate results.
- Control totals provide intermediate checks.

Audit Considerations

Audit considerations examine the results of the analysis by using both the narratives and models to identify the problems caused due to misplaced functions, split processes or functions, broken data flows, missing data, redundant or incomplete processing, and nonaddressed automation opportunities.

The activities under this phase are as follows –

- Identification of the current environment problems
- Identification of problem causes
- Identification of alternative solutions
- Evaluation and feasibility analysis of each solution
- Selection and recommendation of most practical and appropriate solution
- Project cost estimation and cost benefit analysis

Security

System security refers to protecting the system from theft, unauthorized access and modifications, and accidental or unintentional damage. In computerized systems, security involves protecting all the parts of computer system which includes data, software, and hardware. Systems security includes system privacy and system integrity.

- **System privacy** deals with protecting individuals systems from being accessed and used without the permission/knowledge of the concerned individuals.
- **System integrity** is concerned with the quality and reliability of raw as well as processed data in the system.

Control Measures

There are variety of control measures which can be broadly classified as follows –

Backup

- Regular backup of databases daily/weekly depending on the time criticality and size.
- Incremental back up at shorter intervals.
- Backup copies kept in safe remote location particularly necessary for disaster recovery.
- Duplicate systems run and all transactions mirrored if it is a very critical system and cannot tolerate any disruption before storing in disk.

Physical Access Control to Facilities

- Physical locks and Biometric authentication. For example, finger print
- ID cards or entry passes being checked by security staff.
- Identification of all persons who read or modify data and logging it in a file.

Using Logical or Software Control

- Password system.
- Encrypting sensitive data/programs.
- Training employees on data care/handling and security.
- Antivirus software and Firewall protection while connected to internet.

Risk Analysis

A risk is the possibility of losing something of value. Risk analysis starts with planning for secure system by identifying the vulnerability of system and impact of this. The plan is then made to manage the risk and cope with disaster. It is done to access the probability of possible disaster and their cost.

Risk analysis is a teamwork of experts with different backgrounds like chemicals, human error, and process equipment.

The following steps are to be followed while conducting risk analysis –

- Identification of all the components of computer system.
- Identification of all the threats and hazards that each of the components faces.
- Quantify risks i.e. assessment of loss in the case threats become reality.

Risk Analysis – Main Steps

As the risks or threats are changing and the potential loss are also changing, management of risk should be performed on periodic basis by senior managers.



Risk management is a continuous process and it involves the following steps –

- Identification of security measures.
- Calculation of the cost of implementation of security measures.
- Comparison of the cost of security measures with the loss and probability of threats.
- Selection and implementation of security measures.
- Review of the implementation of security measures.

OBJECT ORIENTED APPROACH

In the object-oriented approach, the focus is on capturing the structure and behavior of information systems into small modules that combines both data and process. The main aim of Object Oriented Design *OOD* is to improve the quality and productivity of system analysis and design by making it more usable.

In analysis phase, OO models are used to fill the gap between problem and solution. It performs well in situation where systems are undergoing continuous design, adaption, and maintenance. It identifies the objects in problem domain, classifying them in terms of data and behavior.

The OO model is beneficial in the following ways –

- It facilitates changes in the system at low cost.
- It promotes the reuse of components.
- It simplifies the problem of integrating components to configure large system.
- It simplifies the design of distributed systems.

Elements of Object-Oriented System

Let us go through the characteristics of OO System –

- **Objects** – An object is something that exists within problem domain and can be identified by data *attribute* or behavior. All tangible entities *student, patient* and some intangible entities *bankaccount* are modeled as object.
- **Attributes** – They describe information about the object.
- **Behavior** – It specifies what the object can do. It defines the operation performed on objects.
- **Class** – A class encapsulates the data and its behavior. Objects with similar meaning and purpose grouped together as class.
- **Methods** – Methods determine the behavior of a class. They are nothing more than an action that an object can perform.
- **Message** – A message is a function or procedure call from one object to another. They are information sent to objects to trigger methods. Essentially, a message is a function or procedure call from one object to another.

Features of Object-Oriented System

An object-oriented system comes with several great features which are discussed below.

Encapsulation

Encapsulation is a process of information hiding. It is simply the combination of process and data into a single entity. Data of an object is hidden from the rest of the system and available only through the services of the class. It allows improvement or modification of methods used by objects without affecting other parts of a system.

Abstraction

It is a process of taking or selecting necessary method and attributes to specify the object. It focuses on essential characteristics of an object relative to perspective of user.

Relationships

All the classes in the system are related with each other. The objects do not exist in isolation, they exist in relationship with other objects.

There are three types of object relationships –

- **Aggregation** – It indicates relationship between a whole and its parts.
- **Association** – In this, two classes are related or connected in some way such as one class works with another to perform a task or one class acts upon other class.
- **Generalization** – The child class is based on parent class. It indicates that two classes are similar but have some differences.

Inheritance

Inheritance is a great feature that allows to create sub-classes from an existing class by inheriting the attributes and/or operations of existing classes.

Polymorphism and Dynamic Binding

Polymorphism is the ability to take on many different forms. It applies to both objects and operations. A polymorphic object is one whose true type hides within a super or parent class.

In polymorphic operation, the operation may be carried out differently by different classes of objects. It allows us to manipulate objects of different classes by knowing only their common properties.

Structured Approach Vs. Object-Oriented Approach

The following table explains how the object-oriented approach differs from the traditional structured approach –

Structured Approach	Object Oriented Approach
It works with Top-down approach.	It works with Bottom-up approach.
Program is divided into number of submodules or functions.	Program is organized by having number of classes and objects.
Function call is used.	Message passing is used.
Software reuse is not possible.	Reusability is possible.
Structured design programming usually left until end phases.	Object oriented design programming done concurrently with other phases.
Structured Design is more suitable for offshoring.	It is suitable for in-house development.
It shows clear transition from design to implementation.	Not so clear transition from design to implementation.
It is suitable for real time system, embedded system and projects where objects are not the most useful level of abstraction.	It is suitable for most business applications, game development projects, which are expected to customize or extended.
DFD & E-R diagram model the data.	Class diagram, sequence diagram, state chart diagram, and use cases all contribute.
In this, projects can be managed easily due to clearly identifiable phases.	In this approach, projects can be difficult to manage due to uncertain transitions between phase.

Unified Modeling Language *UML*

UML is a visual language that lets you to model processes, software, and systems to express the design of system architecture. It is a standard language for designing and documenting a system in an object oriented manner that allow technical architects to communicate with developer.

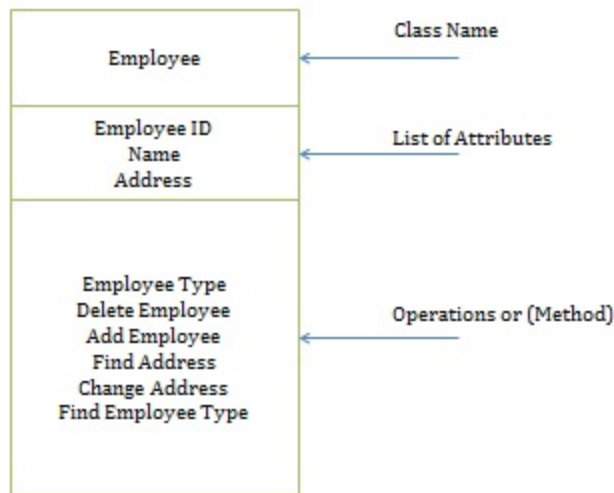
It is defined as set of specifications created and distributed by Object Management Group. UML is extensible and scalable.

The objective of UML is to provide a common vocabulary of object-oriented terms and diagramming techniques that is rich enough to model any systems development project from analysis through implementation.

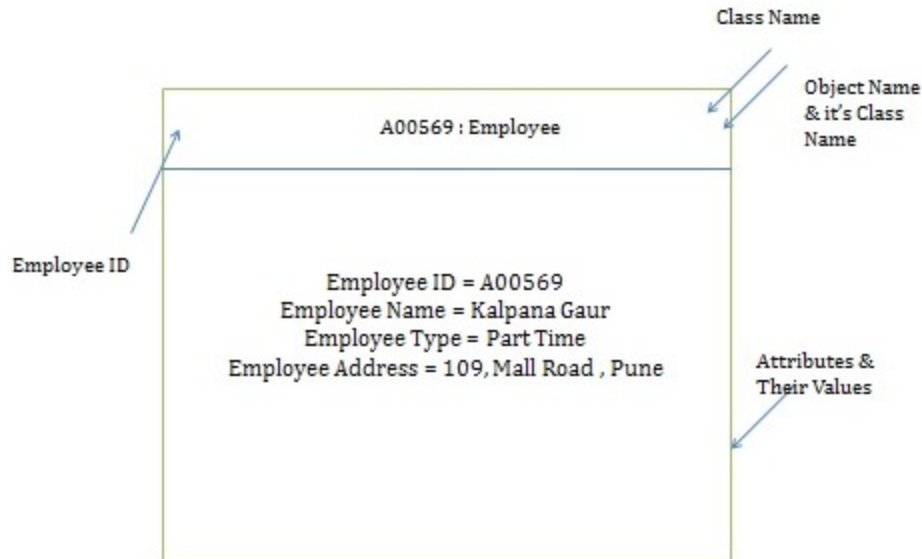
UML is made up of –

- **Diagrams** – It is a pictorial representations of process, system, or some part of it.
- **Notations** – It consists of elements that work together in a diagram such as connectors, symbols, notes, etc.

Example of UML Notation for class



Instance diagram-UML notation



Operations Performed on Objects

The following operations are performed on the objects –

- **Constructor/Destructor** – Creating new instances of a class and deleting existing instances of a class. For example, adding a new employee.
- **Query** – Accessing state without changing value, has no side effects. For example, finding address of a particular employee.
- **Update** – Changes value of one or more attributes & affect state of object For example, changing the address of an employee.

Uses of UML

UML is quite useful for the following purposes –

- Modeling the business process
- Describing the system architecture
- Showing the application structure
- Capturing the system behavior
- Modeling the data structure
- Building the detailed specifications of the system
- Sketching the ideas
- Generating the program code

Static Models

Static models show the structural characteristics of a system, describe its system structure, and emphasize on the parts that make up the system.

- They are used to define class names, attributes, methods, signature, and packages.
- UML diagrams that represent static model include class diagram, object diagram, and use case diagram.

Dynamic Models

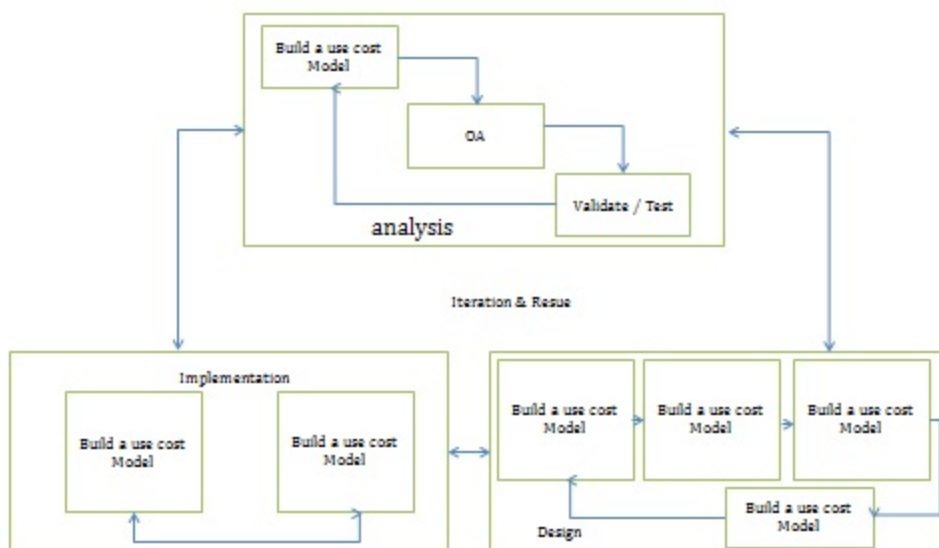
Dynamic models show the behavioral characteristics of a system, i.e., how the system behaves in response to external events.

- Dynamic models identify the object needed and how they work together through methods and messages.
- They are used to design the logic and behavior of system.
- UML diagrams represent dynamic model include sequence diagram, communication diagram, state diagram, activity diagram.

Object Oriented System Development Life Cycle

It consists of three macro processes –

- Object Oriented Analysis *OOA*
- Object oriented design *OOD*
- Object oriented Implementation *OOI*



Object Oriented Systems Development Activities

Object-oriented systems development includes the following stages –

- Object-oriented analysis
- Object-oriented design

- Prototyping
- Implementation
- Incremental testing

Object-Oriented Analysis

This phase concerns with determining the system requirements and to understand the system requirements build a **use-case model**. A use-case is a scenario to describe the interaction between user and computer system. This model represents the user needs or user view of system.

It also includes identifying the classes and their relationships to the other classes in the problem domain, that make up an application.

Object-Oriented Design

The objective of this phase is to design and refine the classes, attributes, methods, and structures that are identified during the analysis phase, user interface, and data access. This phase also identifies and defines the additional classes or objects that support implementation of the requirement.

Prototyping

Prototyping enables to fully understand how easy or difficult it will be to implement some of the features of the system.

It can also give users a chance to comment on the usability and usefulness of the design. It can further define a use-case and make use-case modeling much easier.

Implementation

It uses either Component-Based Development *CBD* or Rapid Application Development *RAD*.

Component-based development *CBD*

CODD is an industrialized approach to the software development process using various range of technologies like CASE tools. Application development moves from custom development to assembly of pre-built, pre-tested, reusable software components that operate with each other. A CBD developer can assemble components to construct a complete software system.

Rapid Application Development *RAD*

RAD is a set of tools and techniques that can be used to build an application faster than typically possible with traditional methods. It does not replace SDLC but complements it, since it focuses more on process description and can be combined perfectly with the object oriented approach.

Its task is to build the application quickly and incrementally implement the user requirements design through tools such as visual basic, power builder, etc.

Incremental Testing

Software development and all of its activities including testing are an iterative process. Therefore, it can be a costly affair if we wait to test a product only after its complete development. Here incremental testing comes into

picture wherein the product is tested during various stages of its development.