ACMT Group of Colleges

POLYTECHNIC – 2ND YEAR/3RD SEM

DIPLOMA (COMPUTER SCIENCE)

SYSTEM ANALYICS & DESIGN

What is the software development life cycle?

SDLC or the Software Development Life Cycle is a process that produces software with the highest quality and lowest cost in the shortest time possible. SDLC provides a well-structured flow of phases that help an organization to quickly produce high-quality software which is well-tested and ready for production use.

A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle. A life cycle model represents all the methods required to make a software product transit through its life cycle stages. It also captures the structure in which these methods are to be undertaken.

The Software Development Life Cycle (SDLC) is a structured process that enables the production of high-quality, low-cost software, in the shortest possible production time. The goal of the SDLC is to produce superior software that meets and exceeds all customer expectations and demands. The SDLC defines and outlines a detailed plan with stages, or phases, that each encompass their own process and deliverables.

A typical Software Development Life Cycle consists of the following stages –

Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 3: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generatethe code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to thetype of software being developed.

Stage 4: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

Stage 5: Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

Software documentation is written text or illustration that accompanies computer software or is embedded in the source code. The documentation either explains how the software operates or how to use it, and may mean different things to people in different roles.

Documentation is an important part of software engineering. Types of documentation include:

- **Requirements** - Statements that identify attributes, capabilities, characteristics, or qualities of a system. This is the foundation for what will be or has been implemented.
- **Architecture/Design** - Overview of software. Includes relations to an environment and construction principles to be used in design of software components.
- **Technical** - Documentation of code, algorithms, interfaces, and APIs.
- **End user** - Manuals for the end-user, system administrators and support staff.
- **Marketing** - How to market the product and analysis of the market demand.

All software documentation can be divided into two main categories:

- **Product documentation**
- **Process documentation**

Product documentation describes the product that is being developed and provides instructions on how to perform various tasks with it. In general, product documentation includes requirements, tech specifications, business logic, and manuals. There are two main types of product documentation:

- System documentation represents documents that describe the system itself and its parts. It includes requirements documents, design decisions, architecture descriptions, program source code, and FAQs.
- User documentation covers manuals that are mainly prepared for end-users of the product and system administrators. User documentation includes tutorials, user guides, troubleshooting manuals, installation, and reference manuals.

Process documentation represents all documents produced during development and maintenance that describe... well, the process. The common examples of process-related documents are standards, project documentation, such as project plans, test schedules, reports, meeting notes, or even business correspondence.

The main difference between process and product documentation is that the first one records the process of development and the second one describes the product that is being developed.

Importance of documentation

1. A single source of truth saves time and energy
2. Documentation is essential to quality and process control
3. Documentation cuts down duplicative work
4. It makes hiring and onboarding so much easier
5. A single source of truth makes everyone smarter

Data and facts gathering techniques:-

For an Analyst Data and fact gathering is important step, on which he/she can develop better understanding of existing system and its problems, based on this analyst can understand the requirements of new system. There are various techniques to gather data and facts of system.

some of them re as follows : 1. Record view and Background reading 2. Interviews 3. Questionnaires 4. Group communication 5. Presentation 6. Site visiting 7. Observation Record view and Background reading : Information

• related to system and organization is already available in some type documents and records(like system user manual , system review/audit, brochures etc.) or is published in the sources like newspapers, magazines, journals etc. Study of already available document is the fastest and independent way of gathering fact and information based on which analyst can prepare questions for further gathering exercise.

Interviews :

This method is used to collect the information from groups or individuals. Analyst selects the people who are related with the system for the interview. In this method the analyst sits face to face with the people and records their responses by which analyst learn about the existing system, its problem and expectation with the system. The interviewer must plan in advance the type of questions he/ she is going to ask and should be ready to answer any type of question. The information collected is quite accurate and reliable as the interviewer can clear and cross check the doubts there itself. This method also helps gap the areas of misunderstandings and help to discuss about the future problems.

Questionnaires :

This method seeks information from the person in written and prescribed format. This is a quickest way for gathering information if respondents are scattered geographically or there is no time for the interviews. Questions can be : structured or unstructured. structured question where the answers are in the form of YES/NO , multiple choice option selection , ratings, fill int hte blanks. unstructured questions where person is asked for his opinion and he/she can answer it freely.

Group Communication :

This method is often used when there no time for personal interview and information is required from face to face sessions. As there are many person present many type of ideas can be heard. Scheduling such sessions is a skillful matter because it has many problems such as : discussion may be dominated by one person others may shy to respond, presence of seniors in the group may not allow others to present their views freely, discussion may lead to verbal fight etc.

Presentation :

Sometime presentation can also be conducted by analyst for presenting his understanding for the system and problems with it. Such presentation may include showing slide , interacting with people and talking to them regarding system, asking questions, answering questions etc. Presentations are useful when users are passive or too busy to actively explain things.

Site Visiting :

it is the process of examining the problems which had previously solved by other sources that can be either human or documents. To solve the requirements of problem, the analyst visits to other organization that had previously experienced for similar problems.

Module Specification

Module is the way to improve the structure design by break down the problem for solving it into independent task.

Advantages of Module –

1. It breakdown the problem into independent modules so the complexity of the problem can be minimized.
2. Each independent module can be easily assigned to the various members of the development team.
3. Module can be easily run and tested independently from another.

Top-Down design approach:-

It is a technique of breakdown a problem into major tasks to be performed. Each task is then further broken down into separate sub task and so on until each sub task is sufficiently simple to be written as a self contained module.

In Top-Down design we initially describes the problem at the highest level that descript what must be done and It does not show how it must be done. Top-Down methods are used throughout the system analysis and design process. The value of using top-down approach, starting at general level and to understand and gain the system and moving down to the levels of greater details.

Advantages of Top-Down Approach

1. By dividing of the problem into number of sub problems we have made it easier to share problem development.
2. It is easy to debug a large program as a number of smaller units rather than one big problem.
3. It is good way to delay decision on problems whose solution is not readily prepare.
4. It allows a programmer to remain on top of a problem and view the developing solutions. The solution always proceeds from the highest level to the lowest level.
5. It becomes an ideal structure for managing the implementation of a computer program using team of programmers.

Bottom-Up design approach:-

When we face a large and complex problem , it is difficult to see how the whole thing can be done so it may easier to solve the part of the problem individual, taking the common and easy aspects first and then more difficult task and finally gather them all together to form complete solution, this is called bottom-up approach.

The bottom-up approach suffers from disadvantage that the part of the program may not fit together very easily and there may be a lack of consistency betweenmodules and reprogramming have to be done.

Waterfall Model

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a **linear-sequential life cycle model**. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

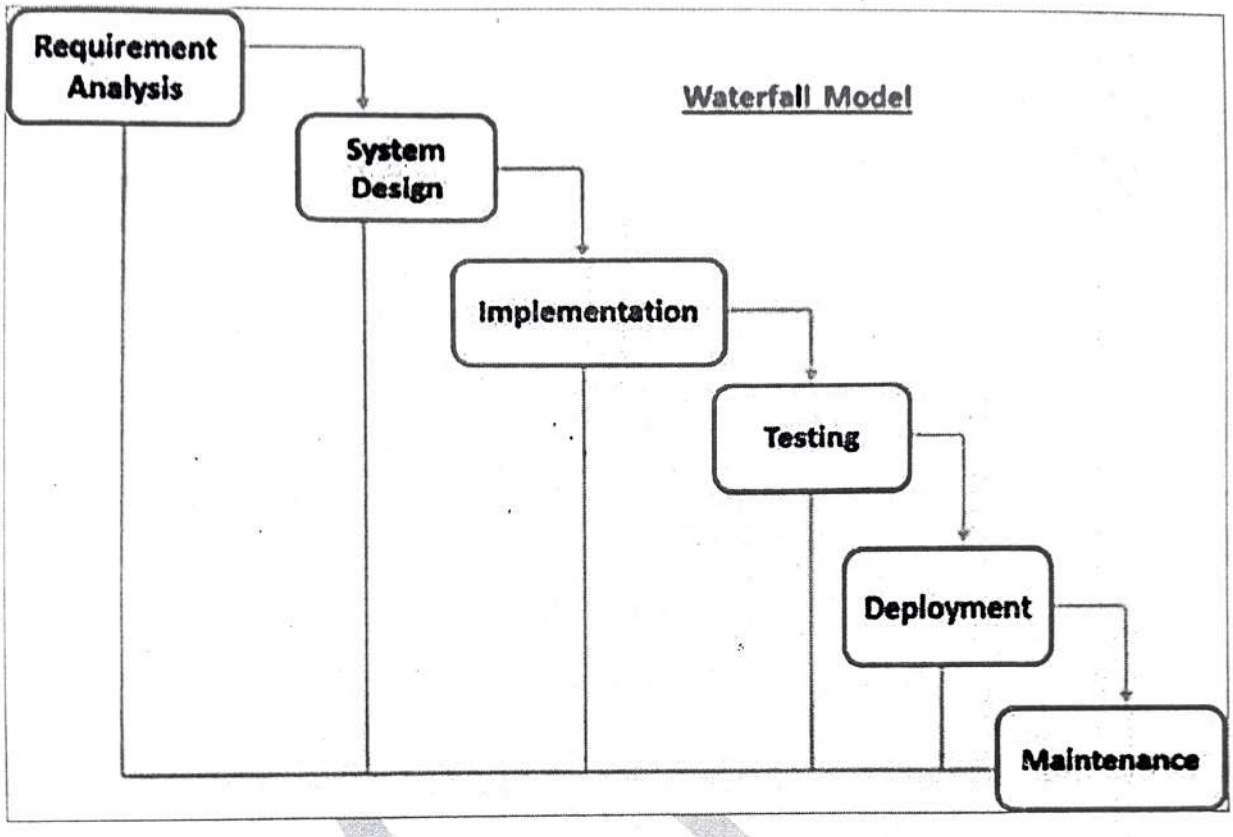
The Waterfall model is the earliest SDLC approach that was used for software development.

The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Waterfall Model - Design

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of

software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the nextphase. Each unit is developed and tested for its functionality, which is referred toas Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

Waterfall Model - Advantages

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

Waterfall Model - Disadvantages

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.

Spiral Model

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

Identification

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

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

Design

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

Construct or Build

The Construct phase refers to production of the actual software product at every spiral. In the baseline spiral, when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.

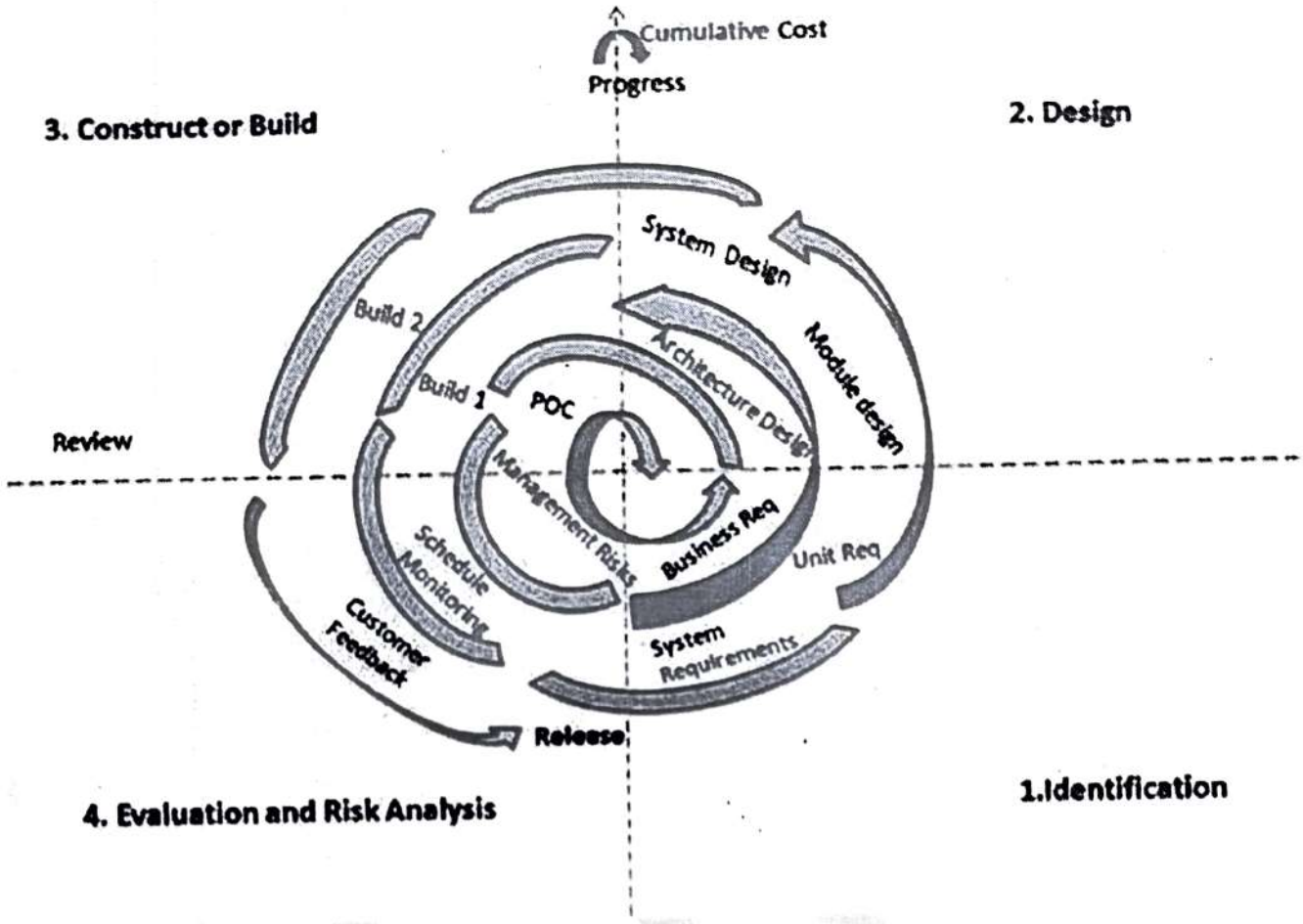
Then in the subsequent spirals with higher clarity on requirements and design details a working model of the software called build is produced with a version number. These builds are sent to the customer for feedback.

Evaluation and Risk Analysis

Risk Analysis includes identifying, estimating and monitoring the technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

The advantages of the Spiral SDLC Model are as follows –

- Changing requirements can be accommodated.
- Allows extensive use of prototypes.
- Requirements can be captured more accurately.
- Users see the system early.
- Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.



The disadvantages of the Spiral SDLC Model are as follows –

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

Prototype Model

The prototype model requires that before carrying out the development of actual software, a working prototype of the system should be built. A prototype is a toy implementation of the system. A prototype usually turns out to be a very crude versionof the actual system, possible exhibiting limited functional capabilities, low reliability,

and inefficient performance as compared to actual software. In many instances, the client only has a general view of what is expected from the software product. In such a scenario where there is an absence of detailed information regarding the input to the system, the processing needs, and the output requirement, the prototyping model may be employed.

Steps of Prototype Model

1. Requirement Gathering and Analyst
2. Quick Decision
3. Build a Prototype
4. Assessment or User Evaluation
5. Prototype Refinement
6. Engineer Product

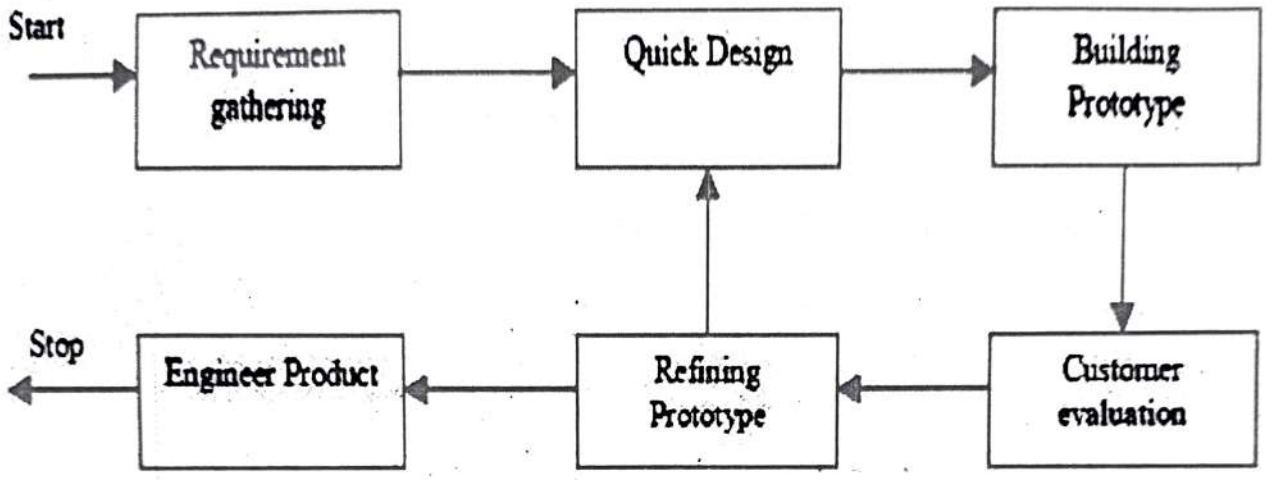
Advantage of Prototype Model

1. Reduce the risk of incorrect user requirement
2. Good where requirement are changing/uncommitted
3. Regular visible process aids management
4. Support early product marketing
5. Reduce Maintenance cost.
6. Errors can be detected much earlier as the system is made side by side.

Disadvantage of Prototype Model

1. An unstable/badly implemented prototype often becomes the final product.
2. Require extensive customer collaboration
    - Costs customer money
    - Needs committed customer
    - Difficult to finish if customer withdraw
    - May be too customer specific, no broad market
3. Difficult to know how long the project will last.

4. Easy to fall back into the code and fix without proper requirement analysis, design, customer evaluation, and feedback.
5. Prototyping tools are expensive.
6. Special tools & techniques are required to build a prototype.
7. It is a time-consuming process.



Prototyping Model

System Analyst

A systems analyst is an IT professional who works on a high level in an organization to ensure that systems, infrastructures and computer systems are functioning as effectively and efficiently as possible. System analysts carry the responsibilities of researching problems, finding solutions, recommending courses of actions and coordinating with stakeholders in order to meet specified requirements. They study the current system, procedures and business processes of a company and create action plans based on the requirements set.

Systems analysts need to be familiar with different operating systems, hardware configurations, programming languages, and software and hardware platforms. They can be involved starting from the analysis phase of the project until the post deployment assessment review.

System Analyst duties and responsibilities

As a role critical to smooth operation and safe, secure computer systems, a System Analyst job description should include many of the duties and responsibilities below:

- Maintaining and upgrading existing systems as required
- Designing new computer systems and frameworks
- Troubleshooting technical issues
- Risk mitigation planning
- Collaborating with Business Analysts, Project Leads and IT team to resolve issues and ensuring solutions are viable and consistent
- Creating system guidelines and manuals for the organization
- Running training sessions and workshops on system processes
- Conducting regular reviews of systems and generating reports on efficiencies and improvement areas
- Structuring and prioritizing business requirements and communicating plans with stakeholders for review and approval

Whether formally qualified or not, a System Analyst should also possess these skills:

- Critical thinking ability
- Strong problem-solving capacity
- High-level written and verbal communication skills
- Project management skills
- Ability to work under pressure and to tight deadlines
- Knowledge of data modeling and data visualization tools

Entity-Relationship Diagrams

ER-modeling is a data modeling method used in software engineering to produce a conceptual data model of an information system. Diagrams created using this ER-modeling method are called Entity-Relationship Diagrams or ER diagrams or ERDs.

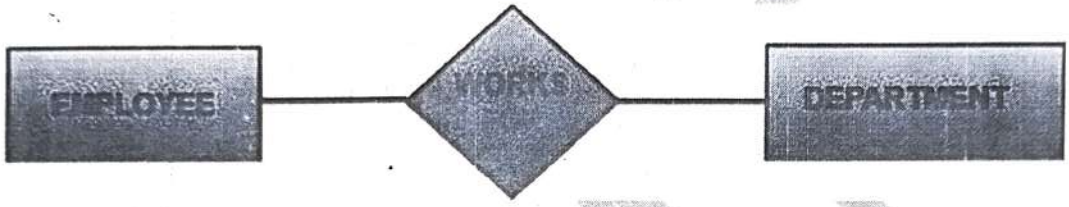
Purpose of ERD

- The database analyst gains a better understanding of the data to be contained in the database through the step of constructing the ERD.
- The ERD serves as a documentation tool.
- Finally, the ERD is used to connect the logical structure of the database to users. In particular, the ERD effectively communicates the logic of the database to users.

Components of an ER Diagrams

1. Entity

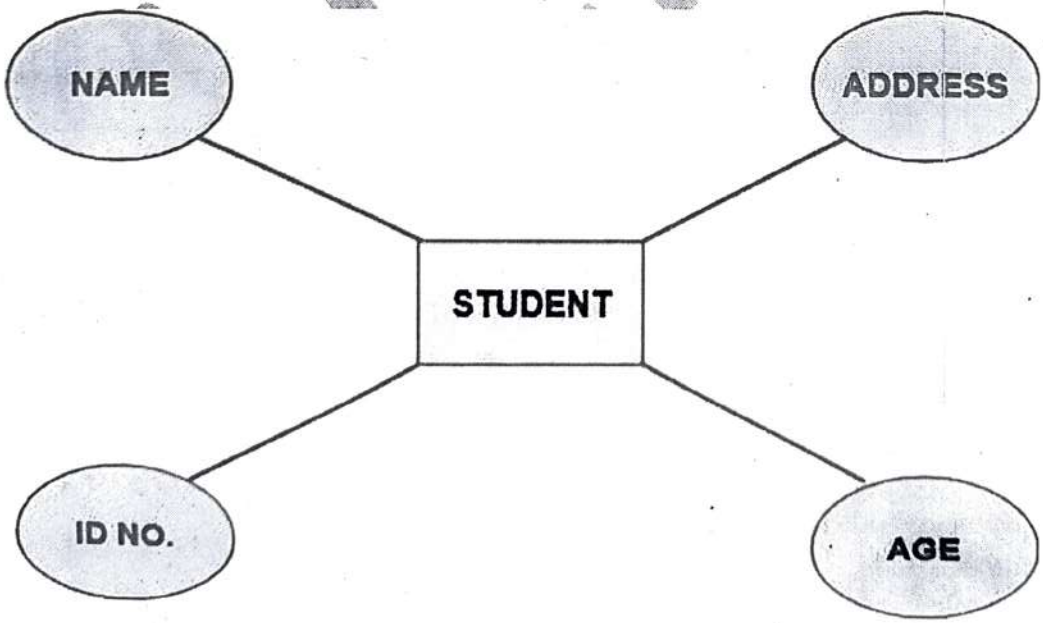
An entity can be a real-world object, either animate or inanimate, that can be merely identifiable. An entity is denoted as a rectangle in an ER diagram. For example, in a school database, students, teachers, classes, and courses offered can be treated as entities. All these entities have some attributes or properties that give them their identity.



2. Attributes

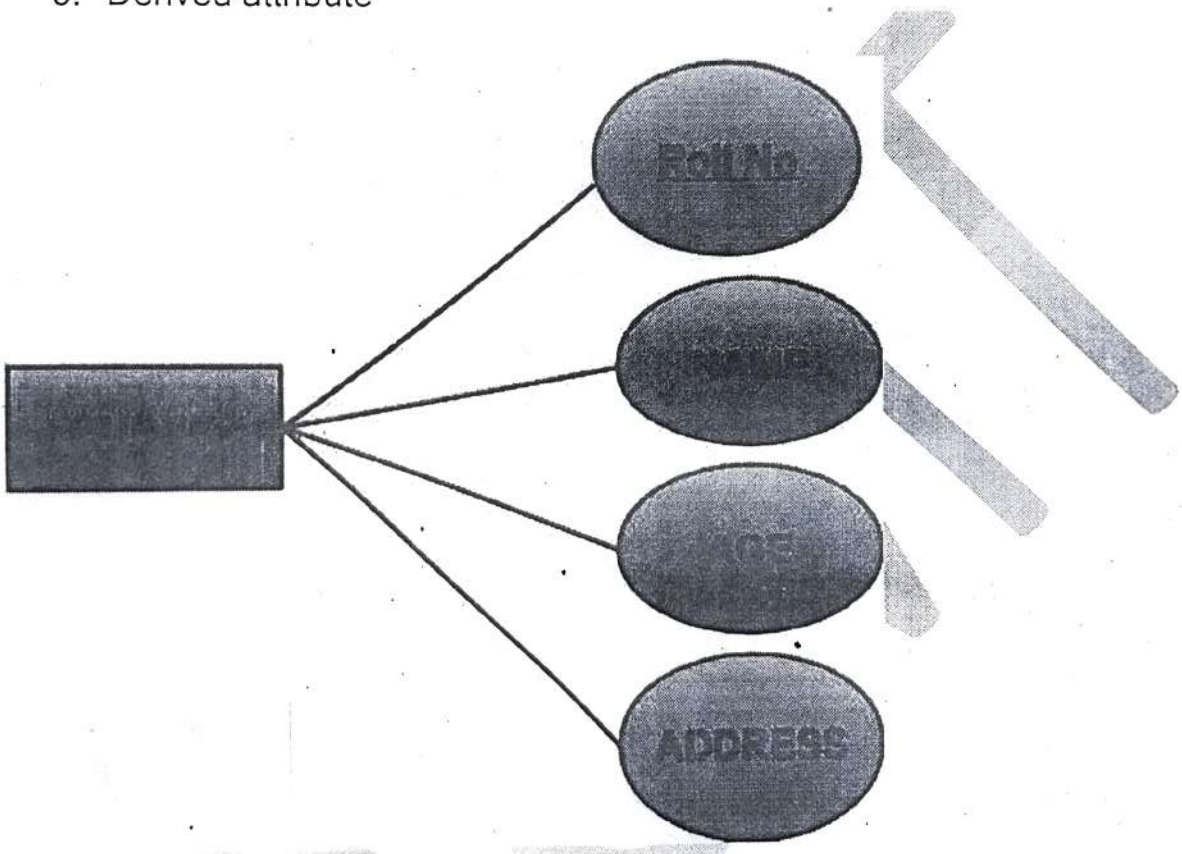
Entities are denoted utilizing their properties, known as attributes. All attributes have values. For example, a student entity may have name, class, and age as attributes.

There exists a domain or range of values that can be assigned to attributes. Forexample, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.



There are four types of Attributes:

1. Key attribute
2. Composite attribute
3. Single-valued attribute
4. Multi-valued attribute
5. Derived attribute



1. Key attribute: Key is an attribute or collection of attributes that uniquely identifies an entity among the entity set. For example, the roll_number of a student makes him identifiable among students.

2. Composite attribute: An attribute that is a combination of other attributes is called a composite attribute. For example, In student entity, the student address is a composite attribute as an address is composed of other characteristics such as pin code, state, country.

5. Derived attribute: Derived attributes are the attribute that does not exist in the physical database, but their values are derived from other attributes present in the database. For example, age can be derived from date_of_birth. In the ER diagram, Derived attributes are depicted by the dashed ellipse.

3. Relationships

The association among entities is known as relationship. Relationships are represented by the diamond-shaped box. For example, an employee works_at a department, a student enrolls in a course. Here, Works_at and Enrolls are called relationships.

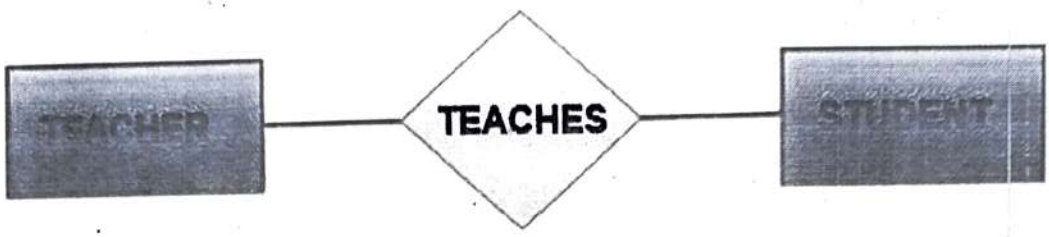


Fig: Relationships in ERD

Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

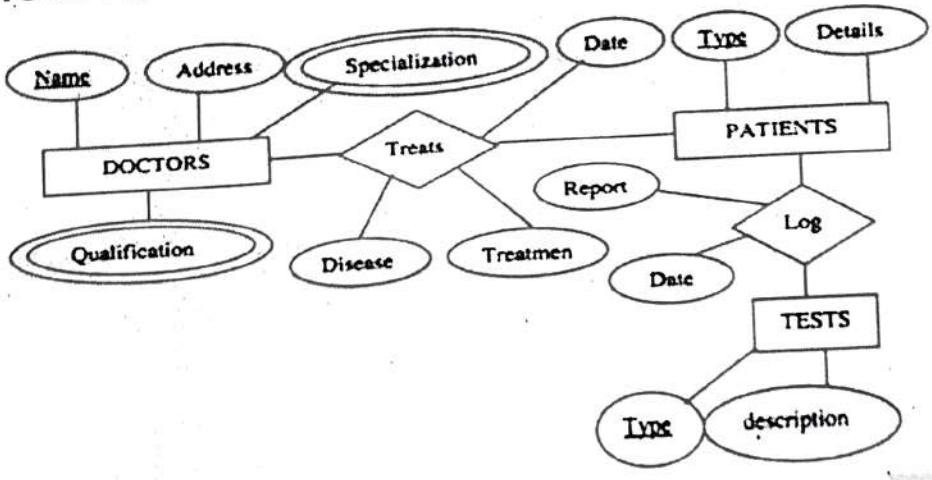
It shows how data enters and leaves the system, what changes the information, and where data is stored.

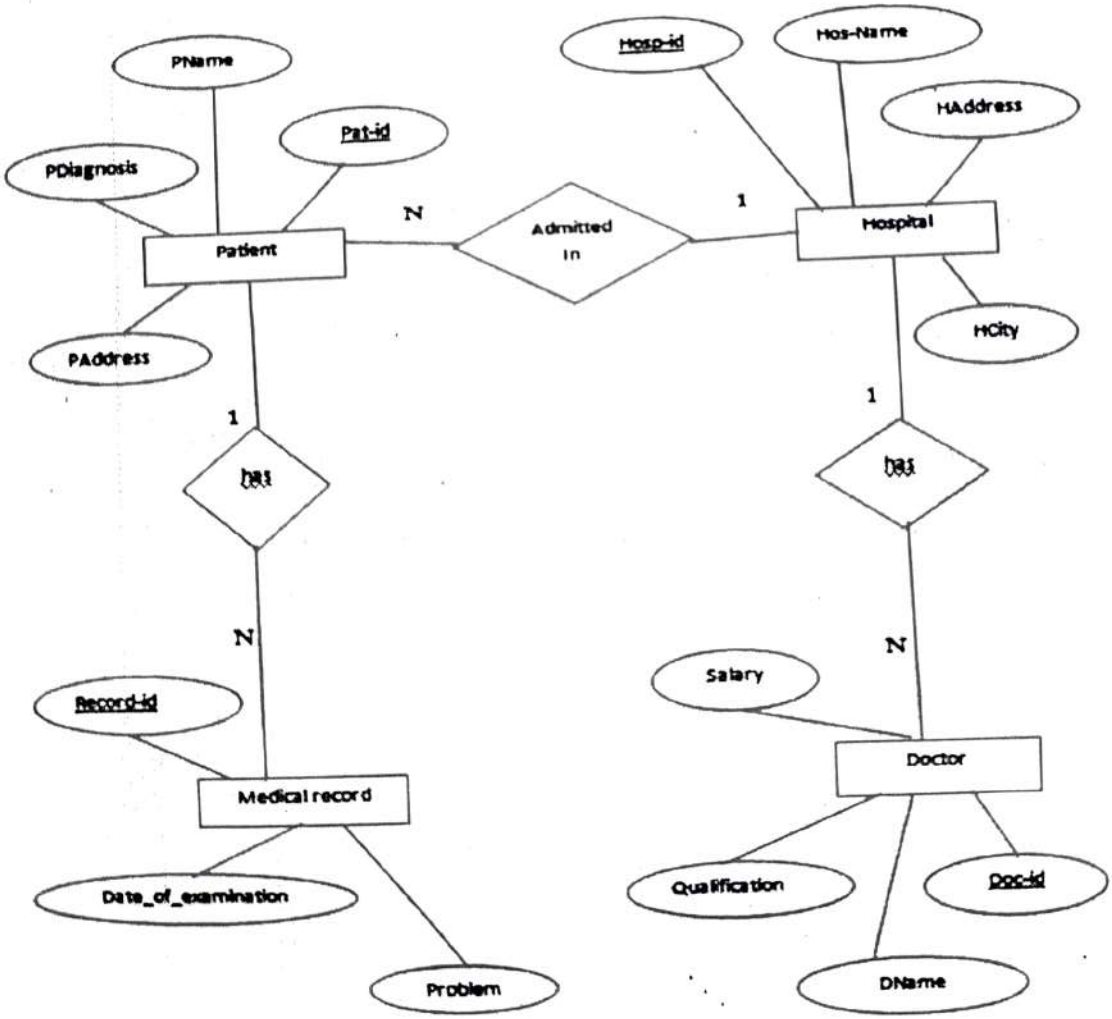
The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. Structure of DFD allows starting from a broad overview and expand it to a hierarchy of detailed diagrams. DFD has often been used due to the following reasons:

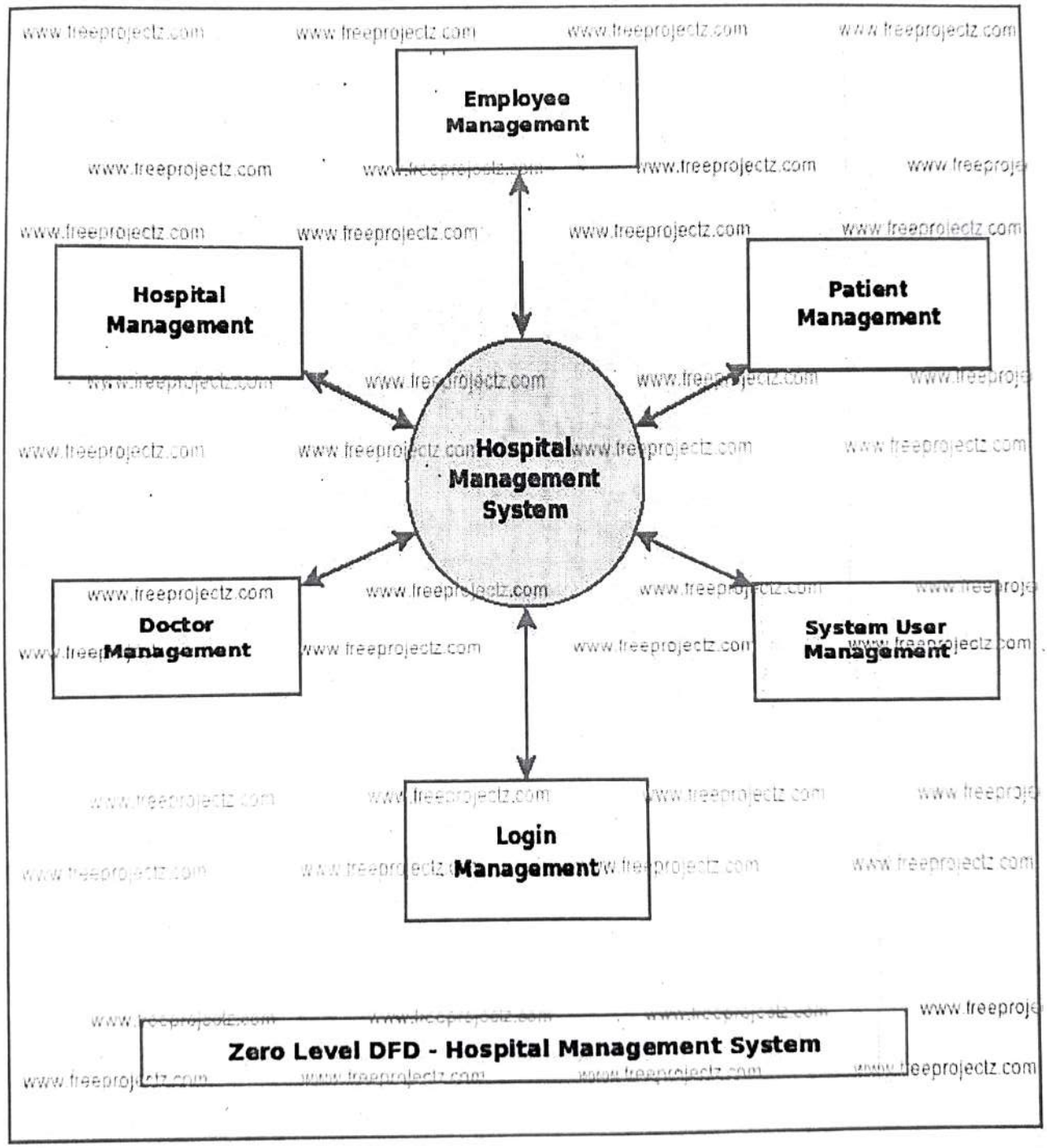
- Logical information flow of the system
- Determination of physical system construction requirements
- Simplicity of notation
- Establishment of manual and automated systems requirements

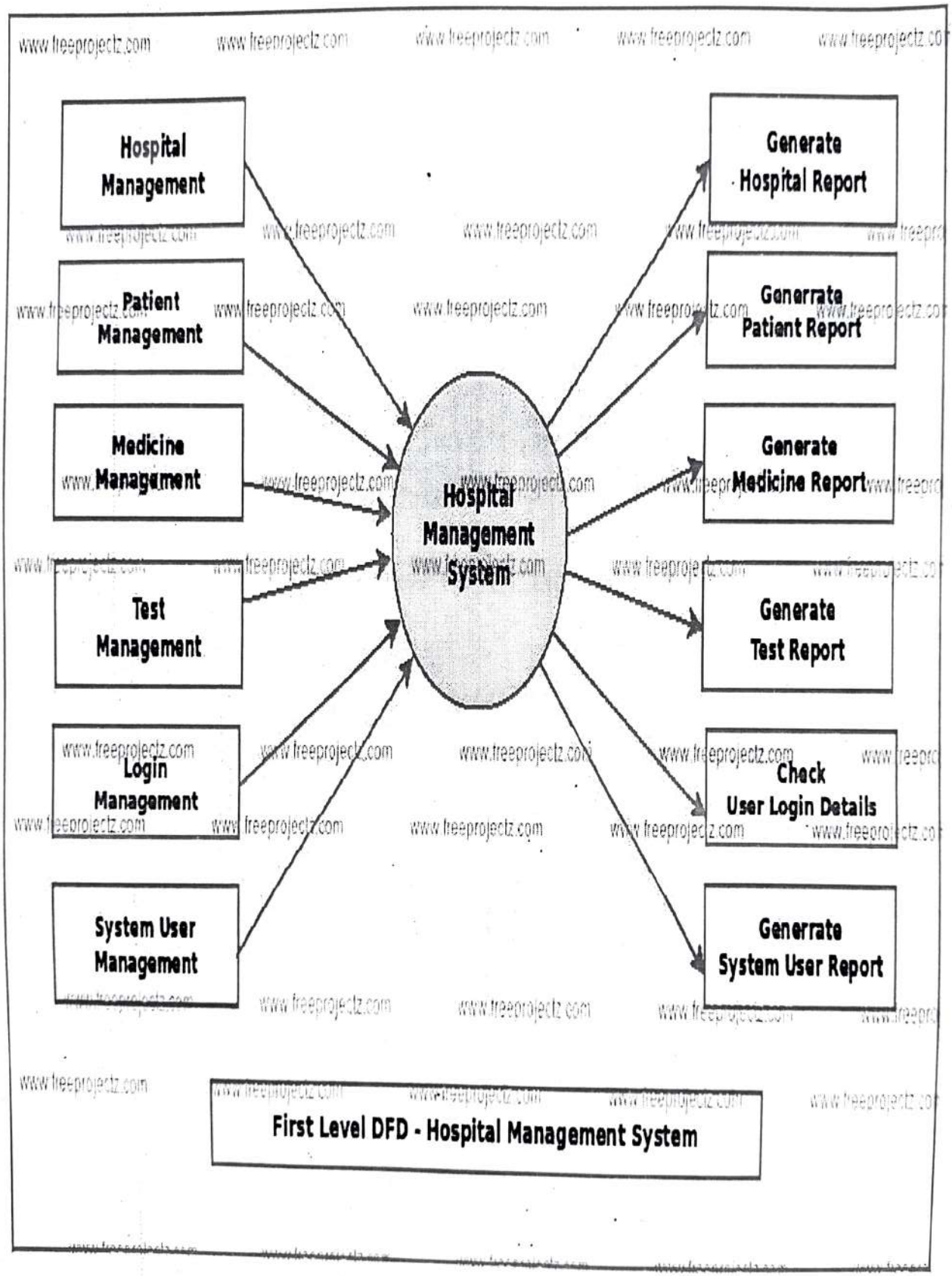
HOSPITAL MANAGEMENT SYSTEM ER-DIAGRAM-



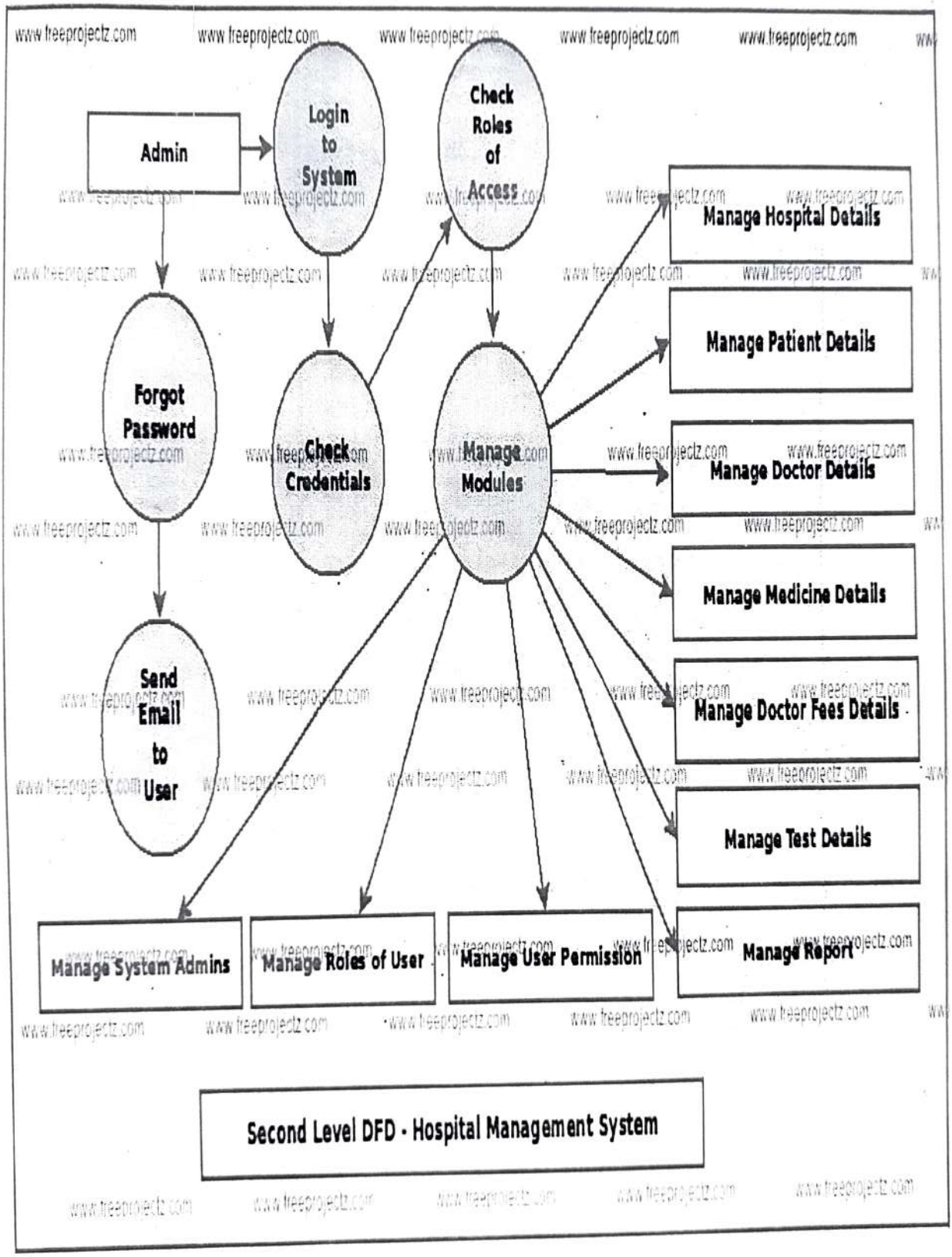


DATA FLOW DIAGRAM-





First Level DFD - Hospital Management System


Admin
Login to System
Check Roles of Access
Forgot Password
Check Credentials
Manage Modules
Send Email to User
Manage Hospital Details
Manage Patient Details
Manage Doctor Details
Manage Medicine Details
Manage Doctor Fees Details
Manage Test Details
Manage Report
Manage System Admins
Manage Roles of User
Manage User Permission
Second Level DFD - Hospital Management System

PROJECT-

A project is defined as a sequence of tasks that must be completed to attain a certain outcome.

A project is a set of tasks which must be completed in order to arrive at a particular goal or outcome. Depending on the size and scope of the project, these tasks may be simple or elaborate, but all projects can be broken down into objectives and what needs to be done to achieve them.

Characteristics of a project

A project is a set of interdependent tasks that have a common goal. Projects have the following characteristics:

1. A clear start and end date – There are projects that last several years but a project cannot go on forever. It needs to have a clear beginning, a definite end, and an overview of what happens in between.
2. A project creates something new – Every project is unique, producing something that did not previously exist. A project is a one-time, once-off activity, never to be repeated exactly the same way again.
3. A project has boundaries - A project operates within certain constraints of time, money, quality, and functionality. We'll see more about this in later sections.
4. A project is not business as usual – Projects are often confused with processes. A Process is a series of routine, predefined steps to perform a particular function, say, expense reimbursement approvals. It's not a one-off activity. It determines how a specific function is performed every single time.

CRITICAL PATH METHOD-

A critical path in project management is certain tasks that need to be performed in a clear order and for a certain period.

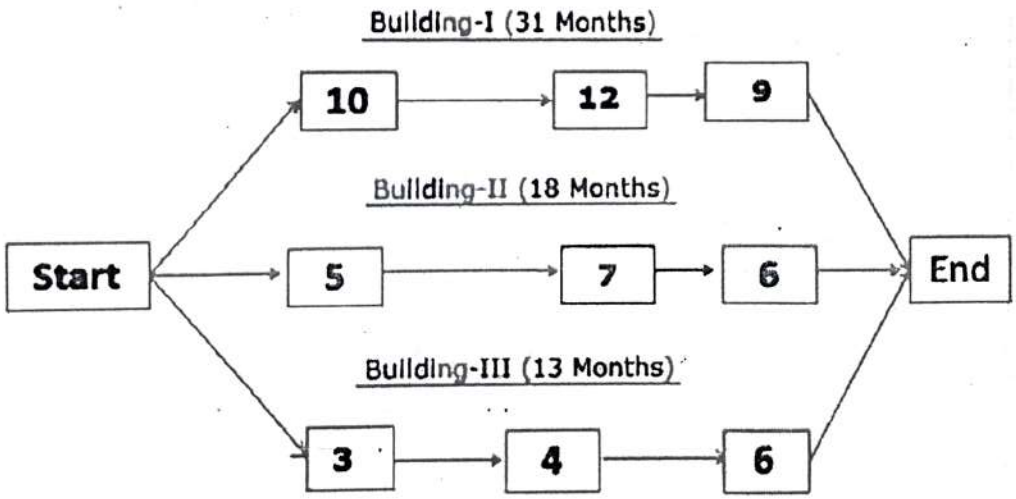
If part of one task can be slowed down or postponed for a term without leaving work on others, then such a task is not critical. While tasks with a critical value cannot be delayed during the implementation of the project and are limited in time.

Critical Path Method (CPM) is an algorithm for planning, managing and analyzing the timing of a project. The step-by-step CPM system helps to identify critical and non-critical tasks from projects' start to completion and prevents temporary risks.

Critical tasks have a zero run-time reserve. If the duration of these tasks changes, the terms of the entire project will be "shifted". That is why critical tasks in project management require special control and timely detection of risks.

FEATURES OF CPM-

- Quick Calculations Save Time and Effort
- Track Progress To Know If You're Behind
- Recalculate As Project Schedule Changes
- Keep Track of Task Dependencies
- Set Milestones and Save Important Dates
- Get Insightful Data When Planning Tasks
- Create Schedule Baseline for Project Variance



PERT CHART-

Program Evaluation Review Technique (PERT) is a project management planning tool used to calculate the amount of time it will take to realistically finish a project. PERT

charts are used to plan tasks within a project – making it easier to schedule and coordinate team members.

PERT charts were created in the 1950s to manage the creation of weapons and defense projects for the US Navy. While PERT was being introduced in the Navy, the private sector simultaneously gave rise to a similar method called critical path. PERT is similar to critical path in that they are both used to visualize the timeline and the work that must be done for a project. However with PERT, you create three different time estimates for the project:

- The shortest possible amount of time each task will take
- The most probable amount of time
- The longest amount of time tasks might take if things don't go as planned

Advantages of PERT Chart

- Activities are mapped to find out the timelines of the project completion
- The most likely, most optimum and pessimistic times can be calculated
- You can evaluate the resources required for each activity
- Also provides the critical path of the activities.
- Helps in setting goals for the project
- Helps to understand the slack period of non critical activities.
- Helps to understand the activities independently and in combination
- Helps in doing the what-if and other critical analysis of the project

Disadvantages of PERT Chart

- It is time intensive and resource intensive to develop the PERT chart
- It is sometimes subjective especially when it comes to the date of completion of the activities and allocation of the resources.