Date :
Page No. :

Q1. Explain what do you mean by a System.

A system is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective. The study of system concepts has three basic implications :

1. A system must be designed to achieve a pre-determined objective.
2. Interrelationships and interdependence must exist among the components.
3. The objectives of the organization as a whole have a higher priority than the objectives of its subsystems.

Characteristics of a system :

1. Organization : It implies structure and order. It is the arrangement of components that helps to achieve objectives.
   Example : organization of various components like input devices, output devices, CPU and storage devices.
2. Interaction : It refers to the manner in which each component functions with other components of the system. For example, the main memory holds the data that has to be operated by ALU
3. Interdependence : It means that parts of the organization or computer system depend on one another. They are coordinated and linked together according to a plan. One subsystem depends on the output of another subsystem for proper functioning.
4. Integration : It refers to the holism of systems. It is concer

Date :
Page No. :

-ned with how a system is tied together. It is more than sharing a physical part or location. It means that parts of the system work together with the system even though each part performs a unique function.

5. Objective: A system should have a central objective. Objectives may be real or stated. Although a stated objective may be the real objective, it is not uncommon for an organization to state one objective and operate to achieve another. The important point is that users must know the central objective of a computer application early in the analysis for a successful design and conversion.

Q2. What are the various types of system.

1. Open and Closed Systems:

An open system continually interacts with its environment. It receives inputs from and delivers outputs to the outside of system. For example, an information system which must adapt to the changing environmental conditions.

A closed system doesnot interact with its environment. It is isolated from environmental influences. A completely closed system is rare in reality.

2. Physical and Abstract Systems:

Physical systems are tangible entities. We can touch and

Date :
Page No. :

feel them. Physical system may be static or dynamic in nature. For example, desks and chairs are physical parts of a computer center which are static. A programmed computer is a dynamic system in which programs, data and applications can change according to user's need.

Abstract systems are non-physical entities or conceptual that may be formulas, representation, or model of a real system.

3. Deterministic and Probabilistic System :

Deterministic system is the one in which the occurences of all events is perfectly predictable. If we get the description of the system state at a particular time, the next state can be easily predicted. An example of such a system is numerically controlled machine tool.

Probabilistic system is the one in which the occurence of events cannot be perfectly predicted. For example, weather forecasting, mail delivery etc. in which the exact output is not known.

4. Management Information System (MIS) :

Management Information System or 'MIS' is a planned system of collecting, storing and disseminating data in the form of information needed to carry out the functions of management.

Management Information System can be analysed as:

Date :
Page No. :

i) Management : Management covers the planning, control and administration of the operations of concern. The top management handles planning, the middle management concentrates on controlling, and the lower management is concerned with actual administration.

ii) Information : Information in MIS means the processed data that helps the management in planning, controlli-ng and operations. Data is processed i.e, recorded, summarized, compared and finally presented to the management in the form of MIS report.

iii) System : Data is processed into information with the help of a system. A system is made up of inputs, processing, output and feedback or control.

Thus MIS means a system for processing data in order to give proper information to management for performing its functions.

5. Decision Support System (DSS) :

Decision Support Systems are interactive software-based systems intended to help managers in decision-making by accessing large volumes of information generated from various related information systems involved in organizational business processes, such as office automation system, transaction, processing system, etc
DSS uses the summary information, exceptions, pattern

Date :
Page No. :

and trends using the analytical models. A DSS helps in decision-making but does not necessarily give a decision itself. The decision makers compile useful information from raw data, documents, personal knowledge, and/or business models to identify and solve problems and make decisions.

Q3. Explain basic principles of a successful system.

1. System should fulfill user requirements.
2. System should be completed in time.
3. System should be maintained effectively for sustained use. For this to happen, system itself should be maintainable.
4. User should get benefit from the system.
5. System should be well-documented i.e, documentation should be a must feature at every phase of system development.

Q4. Explain Command Maturity Model Integration in detail.

The CMMI (Capability Maturity Model Integration) was developed by Software Engineering Institute (SEI) at Carneige Mellon University as a comprehensive process meta-model that is predicated on a set of system and software engineering capabilities that should be present as organizations reach different levels of capability & maturity. To achieve these capabilities, the SEI contends

Date :
Page No. :

that an organization should develop a process model that conforms to the Capability Maturity Model Integration (CMMI) guidelines.

The CMMI represents a process meta-model in 2 different ways: Staged and Continuous.

Staged Representation:

- uses a predefined set of process areas to define improvement path.
- provides a sequence of improvements, where each part in the sequence serves as a foundation for the next.
- an improved path is defined by maturity level.
- maturity level describes the maturity of processes in organization.
- Staged CMMI representation allows comparison between different organizations for multiple maturity levels.

Continuous Representation:

- allows selection of specific process areas.
- uses capability levels that measures improvement of an individual process area.
- Continuous CMMI representation allows comparison between different organizations on a process-area-by process area basis.
- allows organizations to select processes which require more improvement.
- In this representation, order of improvement of various

processes can be selected which allows the organizations to meet their objectives and eliminate risks.

CMMI Model - Maturity Levels :

In CMMI with staged representation, there are 5 maturity levels described as follows :

Maturity level 1 : Initial

- processes are poorly managed or controlled.
- unpredictable outcomes of processes involved.
- ad hoc and chaotic approach used.
- No KPAs (Key Process Areas) defined.
- lowest quality and highest risk.

Maturity level 2 : Managed

- requirements are managed.
- processes are planned and controlled
- projects are managed and implemented according to their documented plans.
- This risk is involved is lower than Initial level, but still exists.
- Quality is better than initial level.

Maturity level 3 : Defined

- processes are well characterized and described using standards, proper procedures, and methods, tools etc.
- Medium quality and medium risk involved.

Focus is process standardization.

Maturity level 4 : Quantitatively Managed

Date :
Page No. :

- quantitative objectives for process performance and quality are set.
- quantitative objectives are based on customer requirements, organization needs, etc.
- process performance measures are analyzed quantitatively
- higher quality of processes is achieved
- lower risk.

Maturity level 5 : Optimizing

- continuous improvement in processes and their performance
- improvement has to be both incremental and innovative.
- highest quality of processes.
- lowest risk in processes and their performance

CMMI Model - Capability levels :

A capability level includes relevant specific and generic practices for a specific process area that can improve the organization's processes associated with that process area. For CMMI models with continuous representation, there are 6 capability levels as described :

1. Capability level 0 : Incomplete

- incomplete process - partially or not performed.
- one or more specific goals of process area are not met

Date :
Page No. :

- no generic goals are specified for this level.
- this capability level is same as maturity level 1.

2. Capability level 1 : Performed
- process performance may not be stable.
- objectives of quality, cost and schedule may not be met.
- a capability level 1 process is expected to perform all specific and generic practices for this level.
- only a start-step for process improvement.

3. Capability level 2 : Managed.
- process is planned, monitored and controlled.
- managing the process by ensuring that objectives are achieved.
- objectives are both model and other including cost, quality, schedule.
- actively managing processing with the help of metrics.

4. Capability level 3 : Defined
- a defined process is managed and meets the organization's set of guidelines and standards.
- focus is process standardization.

5. Capability level 4 : Quantitatively Managed
- a process is controlled using statistical and quantitative techniques.
- process performance and quality is understood in statistical terms and metrics.
- quantitative objects for process quality and performance

Date :
Page No. :

are established.

6. Capability level 5 : Optimizing

- focuses on continually improving process performance.
- performance is improved in both ways - incremental and innovation.
- emphasizes on studying the performance results across the organization to ensure that common causes or issues are identical and fixed.

Q5. Explain the concept of process model for the development of a software?

A software process model is an abstraction of the software development process. The models specify the stages and order of a process. It is defined as a specified representation of the order of activities of the process and the sequence in which they are performed.

A software process model defines :

i) the task to be performed.
ii) the input and output of each task
iii) the pre and post conditions for each task
iv) the flow and sequence of each task.

The goal of a software process model is to provide guidance for controlling and coordinating the tasks to achieve the end product and objectives as effectively as possible.

Date :
Page No. :

The software development models are the various processes or methadologies that are being selected for the development of the project depending on the projects' aims and goals. There are many development life cycle models that have been developed in order to achieve different required objectives. The models specify the various stages of the process and the order in which they are carried out.

The selection of model has very high impact on the testing that is carried out. It defines the what, where and when of our planned testing, influence regression testing and largely determines which test techniques to use.

There are many kinds of process models for meeting different requirements. These are refered as SDLC models (Software Development Life Cycle models). They are :

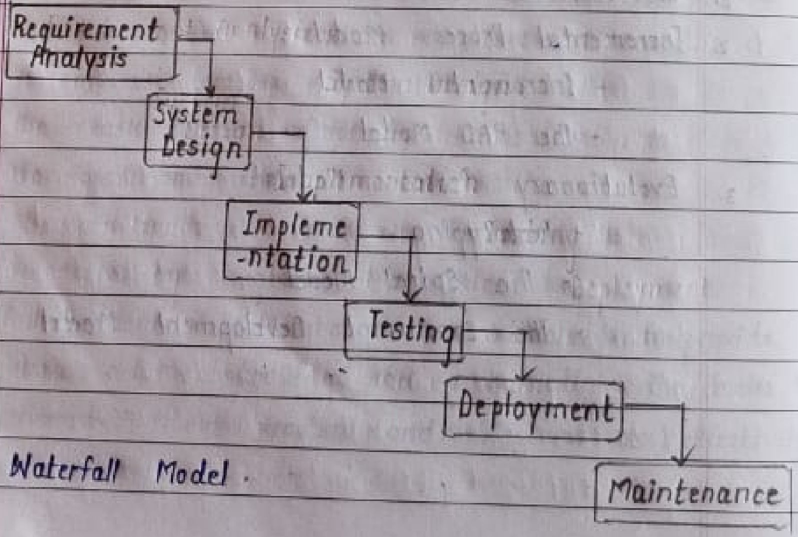
1. The Waterfall Model
2. Incremental Process Models :
   - Incremental Model
   - The RAD Model.
3. Evolutionary Process Models :
   - Prototyping
   - The Spiral Model
   - The Concurrent Development Model.

Date :
Page No. :

The Waterfall Model :

The Waterfall Model was introduced by Winston Royce in 1970. The Waterfall Model was the first model to be introduced and is a classical model used in System Development Life Cycle to create a system with a linear and sequential approach, hence also referred to as Linear-Sequential life Cycle model. It is termed as Waterfall because the model develops systematically from one phase to another in a downward fashion. This model is divided into different phases and the output of one phase is used as the input of the next phase. Every phase has to be completed before the next phase starts and there is no overlapping of the phases.

• The sequential phases described in the Waterfall model are:



Waterfall Model.

Date :
Page No. :

Requirement Gathering and Analysis: Firstly all the requirements regarding the software are gathered from the customer and then gathered requirements are analyzed. These analyzed requirements are documented in a Software Requirement Specification (SRS) document. SRS document serves as a contract between development team and customers. Any future dispute between the customers and developers can be settled by examining the SRS document.

System Design: The requirement specification from the 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 system design, the system is first developed into small programs called units/ modules, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as 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.

Date :
Page No. :

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.

Advantages:

1. This model is simple to implement, also the number of resources that are required for it is minimal.
2. Phases in this model are processed one at a time.
3. Each stage in the model is clearly defined.
4. The release date for the complete product, as well as its final cost can be determined before development.
5. Processes, actions and results are very well documented.

Disadvantages:

1. High amounts of risk and uncertainity, so this model is not suitable for more significant and complex projects.
2. This model cannot accept the changes in requirements during development.
3. It is difficult to measure progress within stages.
4. Since testing is done at a later stage, it doesnot allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare.

Date :
Page No. :

When to use SDLC Waterfall Model:-

Some circumstances where the use of Waterfall Model is most suited are:

1. When the requirements are constant and not changed regularly.
2. Where the tools and technology used is consistent and is not dynamic.
3. A project is short.
4. When the resources are well prepared and are available to use.

INCREMENTAL PROCESS MODEL:

Incremental process model is also known as Successive version model. Incremental Model is a process of software development where requirements are divided into multiple standalone modules of the software development cycle.
In this model each module goes through the requirements, design, implementation and testing phases. Every subsequent release of the module adds function to the previous release. until all designed functionality has been Implemented.
The system is put into production when the first increment is delivered. The first increment is often a core product where the basic requirements are addressed, and supplementary features are added in the

Date :
Page No. :

next increments. Once the core product is analyzed by the client, there is plan development for the next increment.

$$\begin{tikzcd}
\boxed{\text{Analysis}} \arrow[r] & \boxed{\text{Design}} \arrow[r] & \boxed{\text{Code}} \arrow[r] & \boxed{\text{Testing}} & \text{Increment 1} \arrow[d] \\
\boxed{\text{Analysis}} \arrow[r] & \boxed{\text{Design}} \arrow[r] & \boxed{\text{Code}} \arrow[r] & \boxed{\text{Testing}} & \text{Increment 2} \arrow[d] \\
\boxed{\text{Analysis}} \arrow[r] & \boxed{\text{Design}} \arrow[r] & \boxed{\text{Code}} \arrow[r] & \boxed{\text{Testing}} & \text{Increment 3}
\end{tikzcd}$$

Incremental Model

The various phases of Incremental model are as follows:

Requirement Analysis: In the first phase of the incremental model, the product analysis expertise identifies the requirements. And the system functional requirements are understood by the requirement analysis team. To develop the software under incremental model, this phase performs a crucial role.

Design and Development: In this phase, the design of the system functionality and the development method are finished with success. When the software develops new practicality, the incremental model uses style and development phase.

Coding: In this phase, coding is done according to the purpose of requirements. The coding standards must be followed without any unnecessary hard codes & defaults. This phase also enables the implementation of the designs

Date :
Page No. :

which is done practically. By completing this phase, the quality of the working product can be upgraded and enhanced.

Testing : This is the final phase of the Incremental Process Model. In this phase, the performance of each of the existing functions, as well as other additional functionality are checked. Also, various methods are used to test the various behaviours of each task.

Advantages :

1. The cost of the initial delivery is reduced.
2. Since each incremental phase is tested, there will be numerous testing for the software which will, in turn, lead to fewer defects and better results.
3. It is easy to identify errors when using this model.
4. Risk management is easy because the risky parts are identified and properly handled during iteration.
5. This model is flexible. A new feature is added to the product after a new release.

Disadvantages :

1. The total cost of the complete system is high.
2. The model requires an efficient design to ensure the inclusion of the required functionality as well as providing for changes later in the project.
3. More management attention is required,
4. More resources and highly skilled resources are

Date :
Page No. :

required for risk analysis.

5. Not suitable for smaller projects.

When to use Incremental Process Model?

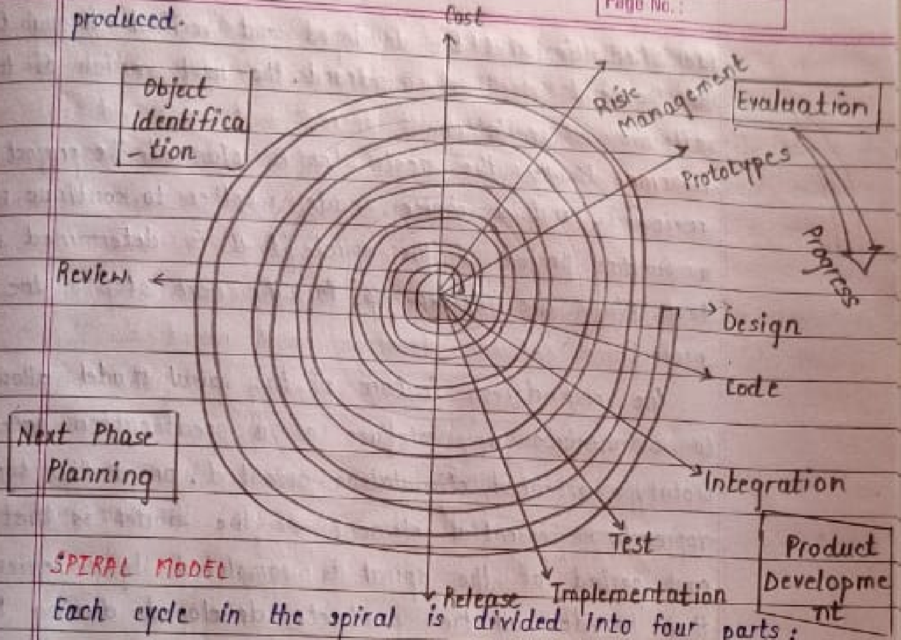
1. This model can be used when the requirements of the complete system are clearly defined and understood.
2. When demand for an early release of a product arises
3. When Software Engineering team are not well-skilled or trained.
4. When there are some high risk features and goals involved.
5. When a project has a lengthy development schedule.

SPIRAL MODEL :

The Spiral Model, initially proposed by Boehm, is an evolutionary software process model that couples the iterative feature of prototyping with the controlled and systematic aspects of linear Sequential Model. It implements the potential for rapid development of new versions of the software. Using this model, the software is developed in a series of incremtal releases. During the early iterations, the additional release may be a paper model or prototype. During later iterations more and more complete versions of the engineered system are

Date :
Page No. :

produced.



SPIRAL MODEL

Each cycle in the spiral is divided into four parts :

Objective Setting: Each cycle in the spiral starts with the identification of purpose for that cycle, the various alternatives that are possible for achieving the targets, and the constraints that exist.

Risk Assessment and Reduction: During the second quadrant, all possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.

Development and Validation: During the third quadrant, the

Date :
Page No. :

identified features are developed and verified through testing. At the end of third quadrant, the next version of the software is available.

Planning: Finally the next step is planned. The project is reviewed, and a choice made whether to continue with a further period of the spiral. If it is determined to keep, plans are drawn up for the next step of the project.

The risk-driven feature of the spiral model allows it to accomodate any mixture of a specification-oriented, prototype-oriented, stimulation-oriented, or another type of approach. An essential element of the model is that each period of the spiral is completed by a review that includes all the products developed during that cycle, including plans for the next cycle. The spiral model works for development as well as enhancement projects.

Advantages:

1. Risk Handling: Risk Handling is one of the important advantages of spiral model, it is best development model to follow due to risk analysis and risk handling at every phase.
2. It is good for large and complex projects.
3. Flexibility in requirements: Change requests in the requirements at later phase can be incorporated

accurately by using this model.

4. It is good for customer satisfaction. We can involve customers in the development of products at early phase of software development. Also, the software is produced early in the software life cycle.

Disadvantages:

1. It is not suitable for smaller projects as it is expensive.
2. It is much more complex than other SDLC models. Process is complex.
3. End of the project may not be known early.
4. It is not suitable for low risk projects.
5. Too much dependable on Risk Analysis and requires highly specific expertise.

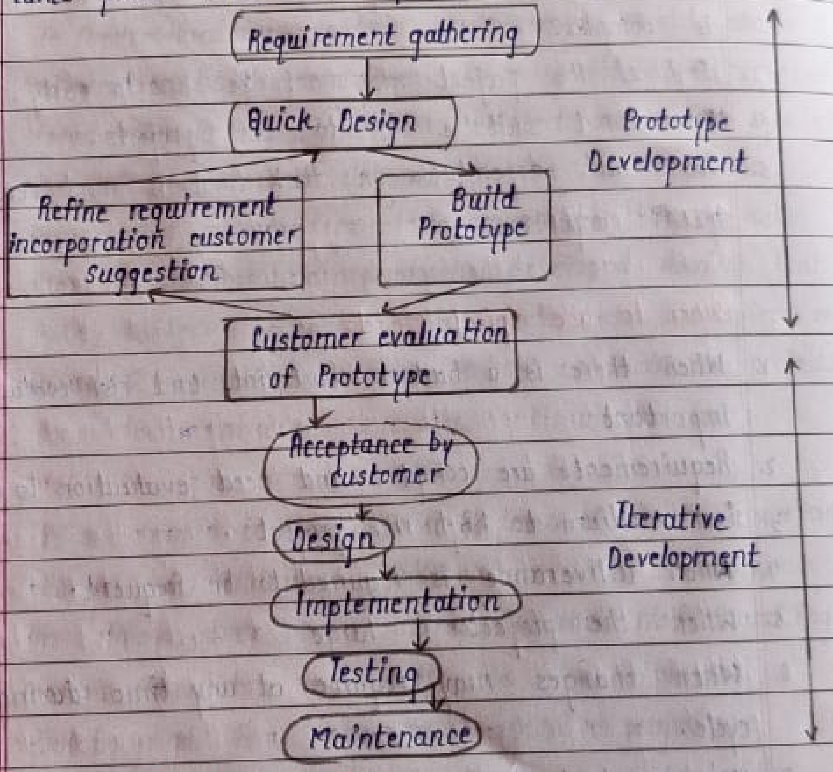
When to use Spiral Model?

1. When there is a budget constraint and risk evaluation is important.
2. Requirements are complex and need evaluation to get clarity.
3. For medium to high-risk projects.
4. When deliverance is required to be frequent.
5. When the project is large.
6. When changes may require at any time during development cycle.
7. New product line which should be released in phases to get enough customer feedback.

Date :
Page No. :

PROTOTYPING MODEL:

Prototyping Model is a software development model in which prototype is built, tested and reworked until an acceptable prototype is achieved. It also creates base to produce the final system or software. It works best in scenarios where the projects requirements are not known in ideal. It is an iterative, trial and error method which takes place between developer and client.



Prototype Model

Date :
Page No. :

The Protype Model has following six SDLC phases :

1. Requirement Gathering and Analysis : A prototyping model starts with requirement analysis. In this phase, the requirements of the system are defined in detail. During the process, the users of the system are Interviewed to know what is their expectation from the system.

2. Quick Design : The second phase is preliminary design or a quick design. In this stage, a simple design of the system is created, however, it is not a complete design. It gives a Brief idea of the system to the user. The quick design helps in developing the prototype.

3. Build a Prototype : In this phase, an actual prototype is designed based on the information gathered from quick design. It is a small working model of the required system.

4. Initial user evaluation : In this stage the proposed system is presented to the client for an initial evaluation. It helps to find out the strength and weakness of the working model. Comments and suggestions are collected from the customer and provided to the developer.

5. Refining prototype : If the user is not happy with the current prototype, you need to refine the prototype according to the user's feedback and suggestions. The changes accepted are again incorporated in the new prototype developed and the cycle repeats until the customer expectations are met.

Date :
Page No. :

6. Implement product and Maintain: Once the final system is developed based on final prototype, it is thoroughly tested and deployed to production. The system undergoes routine maintenance for minimizing downtime and prevent large scale failures.

Advantages:

1. The customers get to see the partial product early in the life cycle. This ensures a greater level of customer satisfaction and comfort.
2. New requirements can be easily accomodated as there is scope for refinement.
3. Missing functionalities can be easily figured out.
4. Flexibility in design.

Disadvantages:

1. Poor documentation due to continuously changing customer requirements.
2. It is very difficult for developers to accomodate all the changes demanded by the customer.
3. The client may loose interest in the final product when he/she is not happy with the initial prototype.
4. Prototyping is a slow and time consuming process.
5. Prototyping tools are expensive.

Date :
Page No. :

When to use Prototyping:

1. The Prototyping Model should be used when the requirements of the product are not clearly understood or are unstable
2. If the requirements are changing quickly.
3. When the desired system needs to have alot of interaction with the end-users.
4. This model can be successfully used for developing user interfaces, high technology software-intensive systems, and systems with complex algorithms and interfaces.

Date :
Page No. :

6. Why there is a need of Requirement Analysis in the initial stage of a software development?

Requirement Analysis also known as Requirement Engineering is the process of defining user expectations for a new software being built or modified. Requirement Analysis encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product or project, taking account of possibly conflicting requirements of various stake holders, analyzing, documenting, validating and managing software or system requirement.

Requirement Analysis is the first stage in the software development process. Analysis of the software is critical to the success or failure of a system or software project. Even a single miscommunicated task could cost a lot both in terms of budget and time.

Date :
Page No. :

7. Write short notes on :

(i) Elements of a System :

There are 6 main elements of a System. These are : Hardware, Software, People, Procedures , Data, Connectivity.

Hardware : The physical components of a computer constitute its Hardware. These include keyboard, mouse, monitor and processor. Hardware consists of input and output devices that make a complete computer system. Examples of input devices are keyboard, optical scanner, mouse and joystick which are used to feed data into the computer. Output devices such as monitor and printer are media to get output from the computer.

Software: A set of programs that form an interface between the hardware and the user are reffered to as Software. The software gives instructions to the hardware to tell it how to function. It works by gathering, organizing and manipulating data and then carrying out instructions. When you use a computer, everything you do is done by the software inside.

People : The people who design and operate the software & hardware are the most important part of any information system. They determine whether or not an information system is going to succeed or fail.

The following types of people interact with a computer system.

a) System Analysts: People who design the operation and

Date :
Page No. :

processing of the system.
System Programmers: People who write codes and programs to implement the working of the system.
System Operators: People who operate the system and use it for different purposes.
Procedures: It is a step by step series of instructions to perform a specific function and achieve desired output.
Data: The facts and figures that are fed into a computer for further processing are called data. Data is raw until the computer system interprets it using machine language, stores it in memory, classifies it for processing and produces result in conformance with the instructions given to it. Processed and useful data is called information which is used for decision making.
Connectivity: When two or more computers are connected to each other, they can share information and resources such as sharing of files, sharing of printer, sharing of facilities like internet etc. This sharing is possible using wires, cables, satellite, infrared, bluetooth, microwave transmission etc.

Date :
Page No. :

ii) Functional and Non-functional Requirements :

Requirement Analysis is very critical process that enables the success of a system or software project to be accessed. Requirements are generally divided into two types: Functional and Non-functional requirements.

Functional Requirements : These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Non-functional Requirements : These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to another. They are called non-behavioral requirements. They deal with issues like : portability, security, maintainability, reliability, scalability, performance, reusability, flexibility.

DIFFERENCE BETWEEN FUNCTIONAL AND NON-FUNCTIONAL REQs

	Functional Requirements	Non-functional Requirements
1	A functional req. defines a	A non-functional req. defines.

	system or its components	the quality attribute of a software system.
2.	It specifies "What should the software system do?"	It places constraints on "How should the software system fulfill the functional req's".
3.	Functional req. is specified by User.	These are specified by technical people eg, architect, technical leaders & software developers.
4.	These reqs are mandatory.	They are not mandatory.
5.	They are easy to define.	They are hard to define.
6.	Helps you verify the functionality of the software.	Helps to verify the performance of the software
7.	There is functional testing such as API, testing, system integration etc.	There is non-functional testing such as usability, performance, stress, security etc
8.	Example: Authentication of user whenever he/she logs into the system.	Emails should be sent within a latency of no greater than 12 hours from such an activity.

Date :
Page No. :

Requirement Modelling:

Requirement modelling in software engineering is essentially the planning stage of a software application or system. Requirement modelling is the process used in software development projects where requirements and solutions constantly evolve through collaborative efforts and teamwork. By using this method of cross-functional and, self-organizing teams, you can ensure that your team meets the exact needs of the stakeholders.

The main aim of requirements modelling is to support the end goals of software development. It also aims to achieve these objectives:

1. Identify and establish the best practices required to create an effective model
2. Outline the ways you intend to put said practices into action.
3. Always have alternatives to improve the overall modelling approach.

[illegible]

[illegible]

Date :
Page No. :

8- Explain in detail Requirement Engineering Framework ?

Requirement Engineering provides the appropriate mechanism for understanding what the customer wants, analyzing need, accessing feasibility, negotiating a reasonable solution, specifying the solution unambiguously, validating the specification, and managing the requirements as they are transformed into an operational system.

Requirement Engineering process is accomplished through the execution of 7 distinct functions / tasks.

1. Inception : Inception is a task where the requirement engineering asks a set of questions to establish a software process.

- In this task, it understands the problem and evaluates with the proper solution.
- It collaborates with the relationship between the customer and the developer.
- The developer and customer decide the overall scope and nature of the question.

2. Elicitation : Elicitation means to find the requirements from anybody. The requirements are difficult because the following problems occur in elicitation :

Problem of scope : The customer gives unnecessary technical detail rather than clarity of the overall system objective.

Problem of understanding : Poor understanding between the customer and developer regarding various aspects of the

Date :
Page No. :

project like capability, limitation of computing environment.
Problem of volatility: In this problem, the reqs change from time to time and it is difficult while developing the project.

3. Elaboration: In this task, the information taken from user during inception & elication are expanded & refined. Its main task is to develop pure model of software using functions, feature & constraints of a software.

4. Negotiation: In this task, a software engineer decides how will the project be achieved with limited business resources. To create rough guesses of development and access the impact of the requirement on the project cost & delivery time.

5. Specification: Outcome of negotiation task of REF is given to specification task, which gives final document or final work of the product in the form of Software Requirement Specification (SRS). The reqs are formalized in both graphical & textual formats.

6. Validation: When the final software is developed, the output of the specification task of REF moves to the next stage/ task i.e, validation, which examines the specification of the software i.e, whether software is working as per needs of the customer/client.

7. Req. Management: It is the process of analyzing, documenting, tracking, prioritizing and agreeing on the requirement & controlling the communication to relevant stakeholders.

This stage takes care of the changing nature of requiremen
It should be ensured that the SRS is as modifiable as
possible so as to incorporate changes in reqs specified b
end users at later stages too. Being able to modify the
software as per reqs in a systematic & controlled manne
is an extremely important part of RE process.

9. Explain the concept of Requirement Elicitation Process
and what are its methods?

Requirement Elicitation Process: It is the process to find
out the requirements for an intended software system
by communicating with client and others who have a
stake in the software development system. Requiremen
Elicitation process can be depicted using the d

Requirement Gathering

Requirement Organization

Negotiations & Discussion

Requirement Specification.

1. Requirement Gathering: The developers discuss with the clie
& end users and know their expectations from the softwar
2. Organizing Requirements: The developers prioritize and arra
the requirements in order of importance, urgency &

Date :
Page No. :

convenience.

Negotiation & Discussion: If requirements are ambiguous or their are some conflicts in requirements of various stakeholders, if they are, it is then negotiated and discussed with all stakeholders. Reqs may then be prioritized and reasonably compromised. The reqs come from various stakeholders to remove ambiguity and conflicts, they are discussed for clarity and correctness.

Unrealistic reqs are compromised reasonably.

Requirement Specification: All formal and informal, functional & non-functional reqs are documented & made available for next phase processing.

Requirement Elicitation methods:

There are a number of requirement elicitation methods. Few of them are;

1. Interviews
2. Brainstorming Sessions.
3. Facilitated Application Specification Technique (FAST)
4. Quality Function Deployment (QFD)
5. Use Case Approach

The success of an elicitation technique used depends on the maturity of the analyst, developers, users, and the customer involved.

1. Interviews: Objective of conducting an interview is to understand the customers expectations from the software. It is impossible to interview every stakeholder hence

Date :
Page No. :

representatives from groups are selected based on their expertise and credibility. Interviews may be open-ended or structured.

In Open-ended interviews there is no pre-set agenda. Context free questions may be asked to understand the problem.

In structured interview, agenda of fairly open questions is prepared. Sometimes a proper questionnaire is designed for the interview.

2. Brainstorming Sessions : It is a group technique which is intended to generate lots of new ideas hence providing a platform to share views.

A highly trained facilitator is required to handle group bias and group conflicts.

Every Idea is documented so that everyone can see it.

Finally a document is prepared which consists of the list of reqs and their priority if possible.

3. Facilitated Application Specification : Its objective is to bridge the expectation gap - difference between what the developers think they are supposed to build and what the customers think they are going to get.

A team oriented approach is developed for requirements gathering. Each attendee is asked to make a list of objects that are:

* Part of the environment that surrounds the system.

Date :
Page No. :

2. Produced by the system
3. Used by the system.

Each participant prepares his/her list, different lists are then combined, redundant entries are eliminated, team is divided into smaller sub-teams to develop mini-specifications and finally a draft of specifications is written down using all the inputs from the meeting.

4. Quality Function Deployment (QFD): QFD is a method to transform qualitative user demands into quantitative parameters, to deploy the functions forming quality, and to deploy methods for achieving the design quality into subsystems and component parts, and ultimately to specific elements of the manufacturing process.

(Use cases: They describe the sequence of interactions Between actors and the system. They capture who(actors) do what(interaction) with the system. A complete set of use cases specifies all possible ways to use the system.)

5. Use Case Approach: This technique combines text and pictures to provide a better understanding of the requirements. They use cases to describe the 'what' of a system and not 'how'. Hence they only give a functional view of the system. The components of the use case design include three major things - Actor, use cases, use case diag.

Actor: It is the external agent that lies outside the system but interacts with it in some way. An actor may

Date :
Page No. :

be a person, machine etc. It is represented as a stick figure. Actors can be primary actors or secondary actors
Primary actors:- It requires assistance from the system to achieve a goal.
Secondary actors: It is an actor from which the system needs assisstance.
Use case diagram: A use case diagram graphically represents what happens when an actor interacts with a system. It captures the functional aspect of the system.

- A stick figure is used to represent an actor.
- An oval is used to represent a use case.
- A line is used to represent a relationship between an actor and a use case.