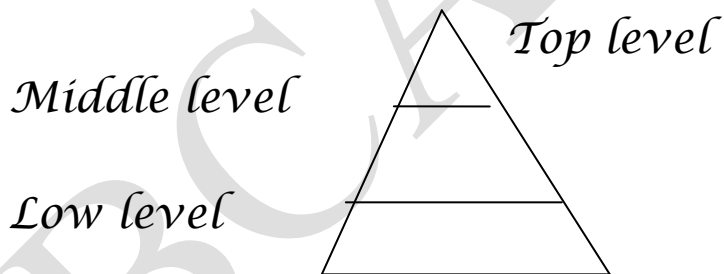# SOFTWARE ENGINEERING

Name of modules:-

1) Fundamental of S.E.
2) System analysis
3) System planning
4) System design
5) System documentation
6) Coding and programming
7) Software testing
8) Cost and time estimation
9) Software project management
10)   Software quality
11)   CASE.

Middle level

Low level

Top level

- *Set of instruction-> this is program.*
- *Set of program-> software.*

This software is a collection of computer programs, procedure, rules and associative documentation and data. Program is generally used the developer of a specific program to make a particular software.

Q) SOME CHARACTERISTICS OF SOFTWARE INCLUDES:-

1) Software is developed or engineer.
2) Most of software is custom build rather than assemble from existing component.
3) Computer program and associated documentation.
4) Easy to modified.
5) Easy to reproduce.
6) Software product may be developed for a particular customer or for the general market.

Q) DIFFERENCE BETWEEN PROGRAM AND SOFTWARE.

PROGRAM	SOFTWARE
1) Small in size.	1) Large in size.
2) Authors himself is user-soul.	2) Large number.
3) Single developer.	3) Team developer.
4) Adopt development.	4) Systematic development.
5) Lack proper interface.	5) Well define interface.
6) Large proper documentation.	6) Well documented.

DEFINITION OF SOFTWARE: - it is systematic approach to the development, operation, maintenance and retirement of software. It is the application of computer science along with mathematics and ergative science. In the current scenario the S.E has a specific importance for making particular software.

WHY SOFTWARE ENGINEERING:-

1) In the late 1960's hardware price were falling but software price rising.
2) Many software projects failed.
3) Large software project required large development loams.
4) Many software project late and over budget.
5) Complicity of software project is increased.
6) Demand for new software on the market.

WHY STUDY SOFTWARE ENGINEERING?

1) Higher productivity.
2) To acquire skills to develop large programs.
3) Ability to solve complex programming problems.
4) Learn techniques of specification design.
5) Better quality programmers.

APPLICATION OF SOFTWARE:-

1) System software.
2) Application software.
3) Engineering/scientific software.
4) Embedded software.
5) Product line software.
6) Web application software.
7) Artificial intelligence software (AI).

1) The concept analysis in the view of S.E.
2) System development life cycle (SDLC).
3) Software requirement specification (SRS).
4) Object- data base and flow base analysis.
5) Models- spiral, water fall model.

1) **THE CONCEPT ANALYSIS IN THE VIEW OF S.E:**- in the S.E the analysis phase helps to determine from the starting of the project to the end of the project. It has some specific phase. We can make particular software for real time use. The term analysis define the corresponding phase or stage by which the software developer can make a successful software. In the S.E the analyst has some specific job which is involved for making software. In a overview we can say that the software analysis is a main term through we can developed, operate and maintain a particular software. In the view of software developer as well as the user.

The software analysis we can determine by the view of two Side.

a) For the software developer.
b) For the customer (client).

SYSTEM DEVELOPMENT LIFE CYCLE (SDLC):- SDLC is an approach for making software for the developer, user and customer. SDLC focus on the internal phase to the end phase for making particular software. It generally deals with the analyst and the corresponding clients. SDLC has some specific phase. This are-

1) project identification
2) feasibility study
3) system analysis
4) system design
5) system development
6) system testing
7) system implementation
8) system maintenance
9) system documentation

1) *Project identification:* - in this phase the analyst focus the basic objective and identification need for the corresponding software. In this phase the analyst set up some meeting with the corresponding client for making the desired software.

2) *Feasibility study:* - feasibility defines in the three views for making particular software for the client.
a) Technical   b) financial   c) social feasibility.

3) *System analysis:* - analysis defines how and what type of desired software we have to make for the client. It has some pen and paper base. Exercise through which the analyst focused for there desired goals.

4) *System design:* - in this phase the analyst draw the corresponding diagrams related to the particular software. in this phase the design include in the form of flow chat, data flow diagram, ntt relationship diagram (NRD).

5) *System development:* - development refers in the form of coding, error checking and debarking for the particular software. This phase deals with the developer activity for making a successfully software.

6) *System testing:* - testing refers whatever analyst and developer done will it be correct and error free to the desired software. In the S.E there some testing technique to which we can check whether project is error free.
       Problem in the particular software. The main testing techniques are
   1) white box testing
   2) black box testing
   3) ad hope testing
   4) system testing
   5) unit testing
   6) alpha testing
   7) beta testing

7) *System implementation: -* after completing the testing phase we have to implement a particular product or system according to the customer need. In the implementation phase some design and other user activity part may be changed as per customer need.

8) *System maintenance: -* after implementation the users use the particular software to there corresponding operation to active there job. In this phase the software maintained from the user or developer side after spanning some times of use of particular software. In this phase the related hardware, software and other utilities are also maintained.

9) *System documentation: -* documentation refers the approach and guidelines for the user as well as the customer to the related software. The documentation refers some writing instruction for how to use for related hardware requirement, and also some maintains factors for the users.

SRS (SOFTWARE REQUIREMENT SPECIFICATION)-

Definition- SRS is a complete reading base documentation focus on the particular desired software to the specific client or customer. After collecting the necessary data from SDLC we have to summarize the useful and appropriate data for making desired software. SRS has some objectives which is help to the software developer as well as the customer for making a successfully software.

Characteristics of SRS-
1) complete
2) traceable
3) appropriate for the developer
4) modifiable
5) simple language
6) software requirement view

Good SRS: - SRS is a very important for fact gathering technique which include the consider delimiting with the customer gather previous information related to particular software and returned on investment the good characteristics of SRS includes the following activities:

1) It focuses on summarized from for a particular software specification.
2) The completeness deserves the related phase for making a required specification.
3) The traceable factor focus on the modification part when ever it necessary for the development of SRS.

WATERFALL MODEL: -
waterfall model linear representation of its face for developing a particular system in this model. These phases are:

1) Communication (requirement analysis, specification).
2) Planning (estimating, seducing).
3) Modeling (analysis, design).
4) Construction (code, test).
5) Deployment (delegacy, support, fid back).
    a) This phase involve the whole working process for making a particular software for desired client.
    b) Iterative waterfall model and the classical waterfall model are the two types are these model available calices. We have to follow the linear stage or phase which are available in waterfall model. But in the iterative model we can jump over from one phase to another phase.

DISADVANTAGES OF WATERFALL MODEL:-
1) In real life the project are sequential.
2) Intel finished the project the working versa is not available for the customer.

3) Hardware implementation in the middle of the project is fact.
4) In this model the risk factor is not available.

SPIRAL MODEL: - the spiral model is a step by step process it stresses on the risk factor of the SDLC.

The stages of the model are:
1) planning
2) risk analysis
3) development
4) customer assessment
   a) determine objective alternative
   b) evaluative alternative risk analysis
   c) development and product
   d) plane for next phase

From the above figure we can see that there are four parts including some loops how ever the number of loops is not fixed depending on the project. Spiral model has a special type of focus on risk factor by which it has make some difference from the other model. In this model the phases are planning, risk analysis, development and customer assessment. From the diagram we can say that first quadrant which representation and make some object and determine the analysis for a particular project. The next part deals and evaluated some alternative way by which we can achieve a simple solution. In this phase the risk analysis is also measured depending on the project.

DIFFERENCE BETWEEN SYSTEM ANALYSIS AND SYSTEM DESIGN-

SYSTEM ANALYSIS	SYSTEM DESIGN
1) System analysis is the examination of the problem.	1) System design is the creator of information system which is solution to the program.
2) It is concerned with identifying all constrains.	2) It is concerned with co erudition of the activities for a particular system goal.
3) It deals with data collection and a datelined evaluation of existing file.	3) It deals with general design specification detailed design specification I/O files and procedures. It also deals with program construction testing and user acceptance.
4) In system analysis part the main focus on data flow diagram and data dictionary.	4) It provides technical specification and reports by which problems can be trapped.

DIFFERENCE BETWEEN PROGRAM AND ENGINEERING.

PROGRAM	ENGINEERING
1) Small project. 2) You 3) Once product. 4) Cheap 5) Few sequential changes.	1) large product 2) team 3) family of product 4) costly 5) Many parallel changes.

MODULE 4

SYSTEM PLANNING

1) What is system planning?

By the planning in S.E it refers the whole internal and external working activities for making particular software. the planning involves the software developer and client base future planning from the starting to the ending phase for making a successful software.

The planning depend upon some categorize by which the software development team.

2) Data and fact gathering.

For making a particular planning the data and fact are most to important factors for set up a good plan for the development particular software.

For collecting data and fact from the real world. Some techniques are interviewing, site visit, previous software data gathering and some real world based examples software to the particular clients.

This technique are very essential and important due to original, meaningful and exact right information through which the developer (software project team) for set up there plan for making particular software.

3) planning in the view of S.E: -

In the view of S.E planning is very essential in the view of creating implementing, error checking, cost estimation schedule and maintenance for a particular software. in this part the development team and the project team set a focus for making a particular software for the desired client. The planning has some advantage for the software developer as well as the user for the particular software.

These are:
1) interviewing
2) communication
3) presentation
4) site visit

1) *Interviewing:* - it is the method by which we collect the data by a specific communication by the software developer and the customer. It is a face to face approach by which we can assume that what actually customer wants.

2) *Communication:* - it refers a technique for building a project at the start phase. Communication here means dialog between two authorities. i.e., software development team and the customer from the software company view generally we make a team for the communication purpose.

3) *Presentation:* - it can be made from the both side. i.e., software development team and customer. It is basically a technique whose we can understand that what actual they deal.

4) *Site visit:* - it is a very important technique for collecting data for a specific operation. In the site visit a development team visits the customer site and tries to understand that what actual operation they do.

MODULE 4

SYSTEM DESIGN

1) *Discuss the term of system design:* - after getting all information regarding making and implementation of software from system development life cycle or from any source which helps to build successful software. Design part in view of software is very important relating to the other issues in the software development process. Design here means that the structure and all essential steps, which is to be used for making a particular software. These are implemented with a specific diagram.

   In S.E the design part generally controlled the system analysis parts which are maintained by the system analyst. In view of S.E and information SAD, it has two types of specific design-

   a) DFD (data flow diagram).
   b) ERD (entity relationship diagram)

2) *What do you understand by design (in view of S.E):* - a model or design is an abstract representation of a set of process? Each model represents a process and data from a particular perspective model provide person information about some data and process. The genetic models do not contain process description rather it deals with product. Different processes are used to develop different part of the model.

   System design tools are basically in two types.

   a) System modifying tools (DFD, ERD).
   b) System design tools (data dictionary, process description).

   The structured analysis involves-

   a) Data flow diagram (DFD).
   b) Data dictionary.
   c) Process description.

In 1976 chain, introduced the method of data analysis in which takes please through out whole development cycle but in different degrees in detail. Usually analysis starts early by developing to level conceptual modeling with using some specific symbols from a modeling method known as entity relationship analysis. It starts when a DFD is finished.

DFD (data flow diagram) - the function or process and the data items that are exchanged between different function are represented in a diagram known as DFD. The program structure is designed from its DFD representation. A DFD shows the logical flow of data through a transaction processing system with out regard to time period when each function occurred. They are used in system development process. Symbol used in DFD:
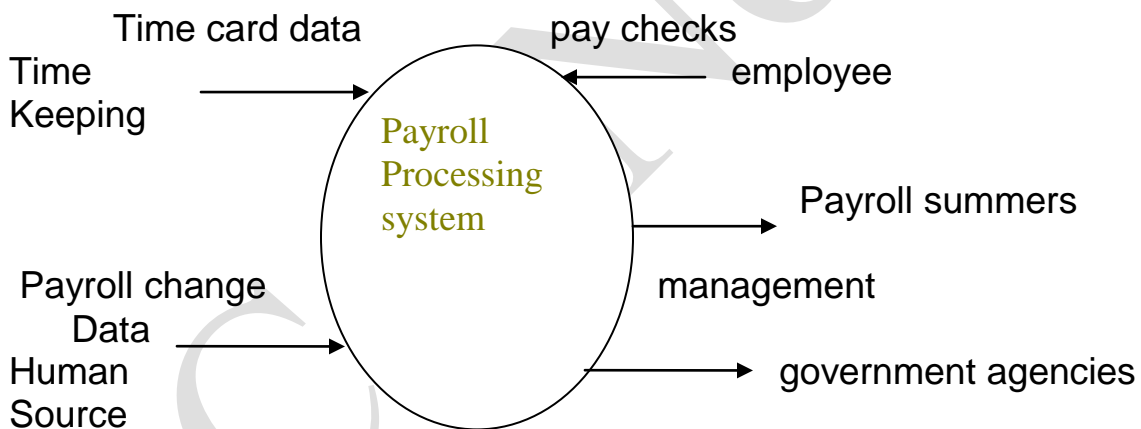
Lists ☐ external data source or data destination.

o- Internal entity or transformation process.

CONTEXT DIAGRAM: – DFD is a graphical representation of data flow. Source process and output are drowning with particular symbols. How ever the diagram close not contains any process logic or any conditional flow. When a single process connects all data source with proper arrows. The DFD is called context or zero level DFD. A context diagram is decomposed into successfully lower level in detours. These types of DFD are referred as level 1, level 2…

Physical DFD: -
the first level of details in a DFD in generally called a physical DFD. It focuses on physical entities involved in the system under study as well as the readable documents reports and other hard copy of input and output.

Logical DFD: -
it deals with participant to each bottle contain. That indicates a task the system performs. Logical DFD helps the designers to decide what system resources are available, what acuities how to protected control this system after instruction. Context level, DFD for payroll processing system.

Time card data   pay checks

Time
Keeping

(Payroll Processing system)

employee

Payroll summers

management

Payroll change
Data
Human
Source

government agencies

ERD: -

Definition: - an ERD is a graphical model of information system that describes the relationship among the system entities. ERD is a major data in our project into entities and define the relationship between the entities.

Component of ERD: - an ERD has 4 major graphical components:

1ST ENTITY: − a data entity is anything read or abstract about which we want to solve and stored in a particular project for example employee's books, computers etc.

2ND ENTITY: − a data relationship is a natural association that exists between one or more entities.

3RD ATTRIBUTE: − an attribute is a characteristic of any particular entity which we can explain in our practical file, for example if we consider the entity is an employee, then the attribute are emp_name, emp_address emp_id designation etc.

4TH CARDINALITY: − it is the specific no of occurrence from one object to another object or one entity to another entity for a particular description.

Data dictionary: - data flow analysis (DFD) consists of a 4 tools. These are:
1) data flow diagram
2) data dictionary
3) data structure diagram
4) structure chart

Data dictionary contain the information about the data of a system. I.e. the data about data or Meta data.

A data dictionary is organized into 5 sections:
a) Data elements
b) Data flows
c) Data stores
d) Process
e) External elements

Object oriented design (OOD): - in the object oriented design the analyst consider and implement the entities in a particular object. In the other design tools, the global data base concept has implemented for a particular project but in the case of OOD the main focus of haring the data through a massage passing through one to another.

In the real world scenario this type of design tools has made its own success. Due to dealing with the object and the object oriented concept in the specific design method. In the S.E the object oriented design has some specific approach as a design tool for analyst.

COUPLING AND COHESION: -

Coupling: - between two methods indicate the degree of independence between them. The degree of coupling between two modules depends on their interface complicity. It is basically determined by no of parameters that are interchange while invoking function of a particular module. Depending on their interface complicity; following five coupling can occur between two modules

data	stamp	control	common	content

Low → high

CLASSIFICATIONS OF COUPLING: -

Data coupling: - two modules are data coupled if they communicate using same data item that is passed as a parameter between two modules. For example- an integer, a float etc.

Stamp coupling: - two modules are stamp couple if they communicate using a composite data item. Such as a structure in c, as a record Pascal.

Control coupling: - this exists between to modules of data from one module is used to direct which helps to execute for another module.

Cohesion- it is a measures of the functional strength of a particular module. A module having high cohesion and low coupling is said to be functionally, independent to other module. By term functional independence we mean that a cohesion module performs a single task for a single function. We use cohesion due to following lesion
1) error ideation
2) scope of reuse
3) understandability
            Classification of cohesion-

Coinaolental
Logic
temporal
Procedural
Communicational
Sequential
function

MODULE 5

SYSTEM DOCUMENTATION

In s/w we can the basic two factors these are 4p's concept and 4w's concept
   4p's- people process, product and plan
   4w's- who, where, what, when.
 Documentation is a serial part in the S.E method of for making its own success for internal and external users. Documentation refers the whole process relating to the operation, use component Tec knowledge and other information relating to particular software.

Types of documentation: - in broad of view the documentation can be classified in two types-
   a) internal documentation
   b) external documentation
   In external documentation it can be again directed in 7 parts:
   1) management documentation
   2) system documentation
   3) operational documentation
   4) user documentation
   5) program documentation
   6) training documentation
   7) implementation documentation

   Internal documentation refers the internal work process like coding, programming, structure which helps software analyst and programmer.
   The external documentation deals with the user manual and other related information for particular plan (customer).

1) *User documentation: -* it refers the whole user details for particular software and something it refers how the are going on in a particular software.

2) *Programming: -* these types of documentation help the programmer and other system level manager to directly or indirectly involve with programming structure.

3) *System: -* the goals of to focus on the software as well as hardware component in which particular software run. It has some specific features like system overview, SRS, design, test and implement.

4) *Management: -* it is used for the upper level manager and its contains the whole work project plan including time and cost for a software.

5) *Operational: -* it maintains the internal working process including some problem solution for the customer. It is a basically a guest of how to use the software to the client.

6) *Training: -* it basically is a summarized from for training purpose and user guideline of a software. it helps for the training purpose people who wants to use a particular software in their specific application.

7) *Implementation: -* It generally helps to internal process. It describes how the software will be implementing in a particular system. This document is very important for software developer and customer of how the software implements and run for a specific purpose.

SYSTEM DOCUMENTATION: - it has the following features for its own its:

1) system overview
2) SRS
3) Specification/design/implementation
4) Test plan
5) Data dictionary
6) Acceptance of test plan.

1) *System overview:* - it is very important step for making particular software for the software developer. System overview is a part of the documentation for use specific software. it has some software as well as some hardware information which is used for the software developer.

2) *SRS:* - it is important features for the system documentation. It has some specific goals which help for making particular software.

3) *Specification/design/implementation:* - the part deals with the system analyst that describe how to use, how to implement, how to make and how to make maintain the particular software including design phase.

4) *Test plan:* - it refers the hole works must be tested with some specific steps and parameters for deciding correctness for a particular work.

5) *Data dictionary:* - it contain data about data (Meta data) this data's are useful for implementation from the coder side.

6) *Acceptance of test plan:* - this feature include in the system documentation part to know whether the design implementation and progress is accepted by designer developer and some time for user.

PRINCIPLES OF SYSTEM DOCUMENTATION: - it has

the following specific features. This are
1) availability
2) objectivity
3) cross referencing
4) easy to maintain
5) completeness

1) *Availability:* - it refers the documentation should be available for the analyst, developer and the user. In right please at right time.

2) *Objectivity:* - the objectivity must be focused and clear who are making the documentation for its specific use. Objectivity refers the original method for making particular software.

3) *Cross referencing:* - it determines for internal communication between to or more module. This feature includes the internal relational ship between the modules.

4) *Easy to maintain:* - after complete the documentation some times it refers in future it may be changed at this time the documentation should be worked properly and correctly.

5) *Completeness:* - it refers the all phase including design, coding, testing, user manual. This must be included for making the documentation successful and complete.

MODULE 6

CODING AND PROGRAMMING

The main object of coding is to implement the design with the help of some programming environment to achieve a specific task. After writing the code we have to go through the error checking part to run the code successfully. After that the code and the design part we have to match those we the original output we may get from the original code.

Coding techniques helps to increase the cast of the software in the error handing part and the implementation part.

Choice of programming language: - to achieve the particular software design output in the real world the coder has to choice. Some programming language with some specific programming environment for this we have to know very well the following objectives in the software coding part

1) Understand very well the design approach.
2) Enough knowledge of programming and environment.
3) Select a programming language to achieve a specific task for client.
4) Understand the original cost and coding part in the real world scenario.

Basically we have two types environment available in software industry

a) structured programming
b) object programming

The both programming environment has some specific norms in its own area. For example the structured programming environment deals with some specific function with its own parameter an OOP environment based on the objectives which is sharable from method to another.

Mixed language programming: - it is basically a integrated approach for the software coder to direct and indirect they can change the environment as it required at the time of operation.

The environment helps the customization for a specific software design and it is very under friendly environment for user. For example ERD these type of software like SAP Microsoft lavation. This type of software as some mixed programming language environment. Which helps the coder as well as the user?

Coding structure: - whenever the programmer writes some codes from the design view. He/she has peach a programming environment achieved the desired goal. Depending programming language and environment we have to follow norms and syntax of that programming environment coding principle are –

1) construct the algorithm
2) select the data structure that will meet the need of design,
3) understand software architecture and create a specific interface,
4) keep conditional knowledge as simple as possible,
5) create nested loop that marks their easily testable,
6) select meaningful variable and follow other local coding instruction,
7) write code that is self documenting,
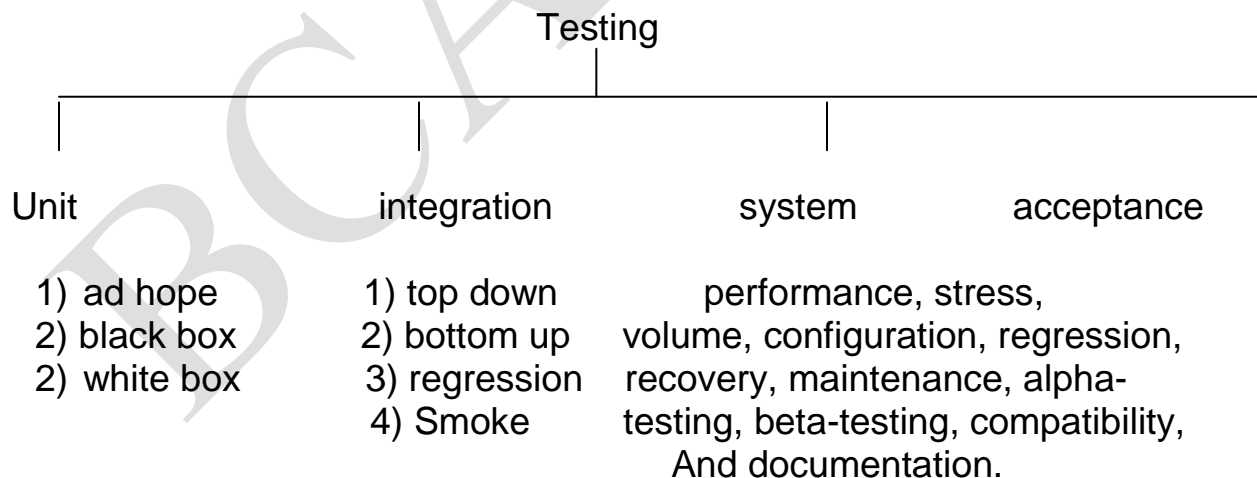8) Create a visual laired for easy understanding.

MODULE- 7

SYSTEM TESTING

Definition of testing: - according to Myers (1979) testing is technique of software to find the error or mistake in a particular software design.

The term over we can define as some unconditional or conditional, direct or indirect mistakes from our design phase to implementing phase in particular software area. In testing the main fault and failure are two main corresponding factors in testing. Fault is a condition for required achievement of a specific function where as failure is the inability to achieve a desired goal.

Software testing is a very essential part in S/E to make a product success. Its basic objective for checking all parts connecting through a specific project.

Types of testing: -

```
                              Testing
       ┌──────────────┬──────────────┬──────────────┐
      Unit        integration       system       acceptance

   1) ad hope      1) top down      performance, stress,
   2) black box    2) bottom up     volume, configuration, regression,
   2) white box    3) regression    recovery, maintenance, alpha-
                   4) Smoke          testing, beta-testing, compatibility,
                                        And documentation.
```

Unit testing: - in testing methodology first testing is unit testing. It has basically three parts a hope testing, white box testing, black box testing.

Ad hope testing: - in ad hope testing we can test whatever comes in mind for a specific testing. It is sometimes called random testing process. In this technique the scope and limitation of software is absent.

White box testing: - in white box testing the structure of the code. We can test without logic. In code segment white box testing are basically in 7 types: -

1) segment testing (each segment to control structure)
2) branch coverage or node testing (each blank)
3) basis path testing (predetermined path setting)
4) compound condition coverage (or, and)
5) data flow testing (variable data flow)
6) path testing (all paths are covered or defined)
7) Loop testing.

Black box testing: - this type of testing for unknown logic structure – in this the function oriented modules are tested; it includes the following activities -

1) error crossing
2) equivalence class classification
3) boundary value analysis
4) cause and effect graphic
5) domain testing
6) module interface testing
7) Command line testing.

Integration testing: - in the integration testing when we use more than one environment to perform software then use integration testing. In this it has basically 4 parts

1) top down testing
2) bottom up testing
3) regression testing
4) smoke testing

1) *Top down testing:* - in this test each module or class from the top level of program segment and after that we can add the first one to next and continue this process until complete module check.

2) Bottom up testing: - in this testing it is a reserve case of to down testing but a simple difference is that this does not allow to integration the tested modules from bottom up.
3) Regression testing: - this type of testing define and examine with some existing function and future achievement goal in specific software regression testing allows the software development team to ensure that the existing functions and other modules whether they can use for their coming project or not.
4) Smoke testing: - it defines the testing approach where we can implement and test each and every module and function with the existing project. This type of testing approach is basically needed. When we need to build up a long term project. Inside of project if software development team wants to test some part of code or module they may up through smoke testing.

System testing: -

1) Performance testing: - carried out to check weather system meets the non functional requirements in SRS documents.
2) Stress testing: - this type of testing help to evaluate the system performance when it is stressed for a sort period of time.
3) Volume testing: - this type of testing helps to check weather data structure, stack, array queue etc. has been classing successfully for a particular solution.
4) Configuration: - use to analysis various types of system behavior including hardware and software for a specific requirement.
5) Compatibility: - this type of testing is required when system interface with other type of system. This helps to check weather interface function performs its corresponding task or not.
6) Regression: - it is necessary when system is being tested when a system may be updated from an existing system to fix some bugs or error in that particular environment.
7) Recovery: - this helps to determine of the software or system will be failure for any résumé, then there should be a particular way to recover the data and information from previous one.

8) Maintenance: - this test the all necessary states from design to implement and determine weather in future all types of altercation will be available.
9) Documentation: - this check ensures that all types of required modules over technical manuals are existing in that particular software project.
10) Alpha testing: - it is system testing performed by development team within the developing organization.
11) Beta testing: - it is system testing performed by selected group of customer.
12) Acceptance testing: - this type of testing is done by the users to determine that software will be accepted or rejected.

Validation and verification: - validation refers to the set of authorities which ensure that software is correctly implemented for a specific function. Verification refers to a different set of authorities which ensure that software which has been built is traceable to customer requirements.

Verification: - are we making the product?

Validation: - are we making the right product.

It is basically a part of the software quantity assurance (SQA): - tested approach in the S/E-

It includes the following objectivity-

1) To ensure that product is sofa and result.
2) To ensure that product will function properly under both normal exceptional conditions.
3) To ensure that the product is what the user want.

MODULE 9

SOFTWARE QUALITY

Explain quality: - achieving a high level of product or service, the quality is the objective of this most organization. It is longer acceptable to the customer of a particular poor quality product and then it refers after they have been delivered to customer.

Software quality management can be structured in three activities: -

Quality assure: - the established next of a frame work of organizational procedures and standard which lead to high quality software.

Quality planning: - the selection appropriate procedures and standard form this frame work and adaptation for a specific software project.

Quality control: - the definition of a process which ensures that project quality procedures and standards are followed by software development team.

Quality tools: - an international standard team can be used in development of a quality management system In all industries is called ISO (international standard organization). ISO-9000 is a set of standards that can be applied to a range of organization from manufacturing to service industries ISO-9000 three interrupts ISO-9000 for software development. These tools include –
1) design control
2) product identification and traceability
3) inspection and testing
4) corrective action
5) document control
6) Service training etc.

Quality assurance: - it consist of a set of checking and reporting that perform with quality control activities of includes: -

1) correctness
2) integrity
3) maintains
4) Check for particular client.

Quality assurance activities defines frame work for achieving software quality which involves with selecting standards the should be applied for software development process or software product. The product and process standards are two vital factors for software quality assurance.

McCall's quality factors: - the factors that effect software quality can be categorized into two groups –

1) Factors that can be directly measured.
2) Factors that any measured indirectly (usability).

In each case segment the focus is to defect the quality factor in the software including program data and documents.

McCall's Richards and want user proposed a useful structure of quality factor that helps the software quality. The main three objectives are that: -

1) Operational characteristics
2) Ability to under go change.
3) Adaptability to new document.

McCall's quality factor has 3 main parts-

a) product revision
   1) maintainability
   2) flexibility
   3) testability
b) product transition
   1) portability
   2) reusability
   3) interoperation ability
c) product operation
   1) correctness
   2) integrity
   3) effectiveness
   4) reliability
   5) usability

MODULE 10

SOFTWARE PROJECT MANAGEMENT

Explain project management: - project management is the technique of maximizing the provability that a project developer its goal on time, within the budget and the good quality. Here the 4 P'S connect-people, process, product and plan are key factors for the whole project management.

A common process frame work is selected an appropriate S/E. paradigm through is applied and a set of work is done to a specific project management are –

1) Project planning
2) Project scheduling
3) Risk management
4) Managing people
5) Software cost estimation

Software project management has some specific goal to make it successful on the particular software making after 1993 the software quality management and risk management were involved with software project management.

Project control: - in software industry the main problem of making a software are –

1) estimate the cost

2) knowledgeable professional
3) delivery within time
4) meets all customer requirement
5) good quality
6) Scope for the future.

To solve this problem the software project management helps to achieve for particular software run with its own success and customer satisfaction. We know the main elements of software project management are –

Project planning: - it has some specific objectives. These are –

a) determine the requirement

b) determine the resource
c) select a particular life cycle
d) determine project strategies

Project planning is very essential from starting phase to ending phase of software.

Project scheduling: - it refers the time span or period for a specific software project. It is basically 3 components:

a) plan
b) effectiveness
c) scheduling

With help of some charts and other graphical tools helps to determine the schedule of a particular project. Here resource allocation the professional people and other related categorized are involved with project scheduling.

Risk management: - in software industry the risk term is often use the relation of a failure and successful project. In real world scenario sometimes the resource are not available as much we want for that point of view the risk term played a specific job for software developer as well as the customer.

Managing people: - it represents the professional knowledgeable person who deals a particular project as a managerial and technical view.

Software cost estimation: - it is very important for software project management to control the overall budget from company side as well as customer or client SID. Cost estimation in S/E has some specific norms and rules to estimate of particular software.

Quality management: - quality is another key factor for particular software for a development team as well as customer ideology for any industry. ISO-9000-3 look after the original quality of software in software market.

WBS (WORK BREAK DOWN STRUCTURE):

a WBS in project management and system engineering is use to define and group a project. Work elements in a way that helps organization and define that total work scope of the project.

A WBS element may product, data service or any combination. A WBS also provide the necessary from was for detailed cost estimating and control along with providing guidelines for schedule development control. Additionally the key is a dynamic tool that can be developed and received as needed by project manager. A WBS is structure which shows sol division of effort required to achieve an objective.

For example- a program a project and contract the project or contract the WBS is developed by starting with the end objectives and solve dividing into manageable components in tare-of-size, duration and responsibility.

One of the most important WBS, structure design principle is called 100% rule. The 100% rule state that WBS includes 100% of the work defined by the project scope and capture for internal and external process in terms of work to be completed including Project management.