18CS35

Third Semester B.E. Degree Examination, Aug./Sept.2020
Software Engineering

Time: 3 hrs. Max. Marks: 100

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

1 a. Define Software Engineering. Bring out the differences between generic and bespoke software. List Software Engineering attributes. **(10 Marks)**

b. Explain Incremental Development process model with a neat block diagram. List its benefits and problems. **(10 Marks)**

OR

2 a. Illustrate Requirement Engineering process with a neat block diagram. **(10 Marks)**

b. Explain the IEEE standard requirement document with its structure. **(10 Marks)**

Module-2

3 a. Define object orientation, list and explain the aspects of object oriented approach. **(10 Marks)**

b. List and explain the object oriented theories which supports object oriented technology. **(10 Marks)**

OR

4 a. Briefly explain Links, Associations, Ordering, Bags and Sequences with an example each. **(10 Marks)**

b. Explain Generalization and Inheritance with an example each. **(10 Marks)**

Module-3

5 a. What is system modeling? Explain the different perspective that the system model developed. **(10 Marks)**

b. Illustrate sequence diagram with an example to view patient information. **(10 Marks)**

OR

6 a. Explain Event-driven model with a state diagram of microwave oven application. **(10 Marks)**

b. Define design patterns. Briefly explain the essential elements of design patterns. **(10 Marks)**

Module-4

7 a. Discuss Test Driven Development (TDD) with its process and list its benefits. **(10 Marks)**

b. Explain software evolution process with neat block diagram. **(10 Marks)**

OR

8 a. Discuss Lehuran's laws of program evolution dynamics. **(10 Marks)**

b. Explain Reengineering process with a neat block diagram. **(10 Marks)**

Module-5

9 a. Discuss project plan. Explain the various section of project plan. **(10 Marks)**

b. With a neat diagram explain project scheduling process. **(10 Marks)**

OR

10 a. Discuss software quality and its attributes. Explain process based quality. **(10 Marks)**

b. Explain software reviews and inspections of Quality Assurance. **(10 Marks)**

Third Semester B.E. Degree Examination
Aug./Sept. 2020

Software Engineering

Staff : Saleem. B. Hebbal

Module-1

1a. Software Engineering is intended to support professional software development. Software engineering is concerned with the practicalities of developing and delivering useful software and it is an Engineering discipline

Generic products : These are stand-alone systems that are produced by a development organisation and sold on the open market to any customer who is able to buy them.

Examples : word processors, drawing packages etc

Customized (or bespoke) products : These are systems that are commissioned by a particular customer. A software Contractor develops the software especially for that customer.

Examples : Control systems for electronic devices, Air traffic control systems.

Attributes :
- ✓ Maintainability
- ✓ Dependability
- ✓ Usability
- ✓ Efficiency
- ✓ security
- ✓ Acceptability

2b. Incremental Development is based on the idea of developing an initial implementation, exposing this to user comment and evolving it through several versions, until an adequate system has been developed.

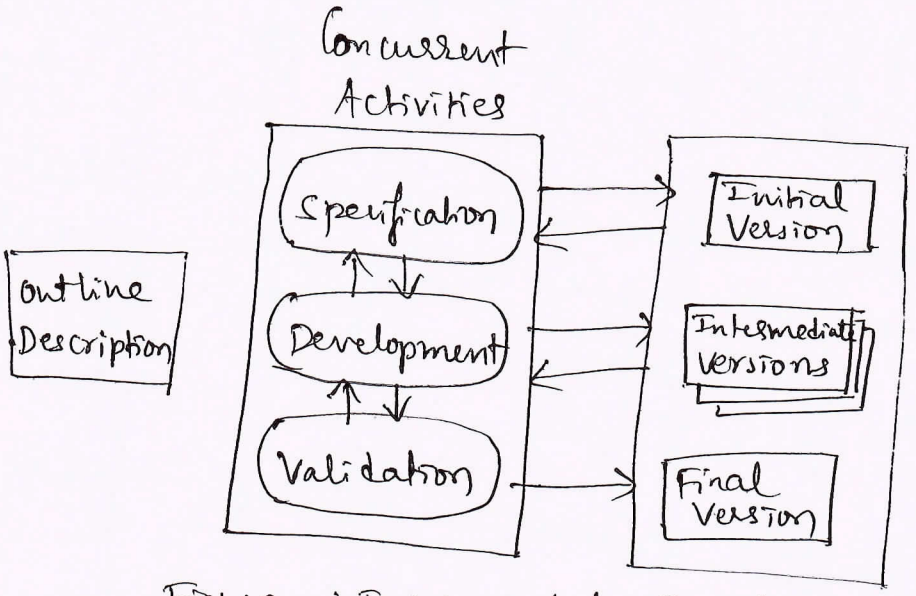


Figure : Incremental Development

Specification, Development and validation activities are interleaved rather than separate with rapid feedback across activities.

We rarely work out a complete problem solution in advance but move toward a solution in a series of steps, backtracking when we realize that we have made a mistake

Benefits: ① It is easier to get customer feedback on the development work that has been done

② More rapid delivery & deployment of useful s/w to the customer is possible.

Qa. problems :-

(1) The process is not visible. Managers need regular deliverables to measure progress. If systems are developed quickly, it is not cost-effective to produce documents that reflect every version of the system.

(2) System structure tends to degrade as new increments are added

2a. Requirement Engineering process :

→ Requirements engineering is the process of understanding and defining what services are required from the system and identifying the constraints on the system's operation and development.

→ The requirements Engineering process aims to produce an agreed requirements document that specifies a system satisfying stakeholder requirements

There are Four main activities :-

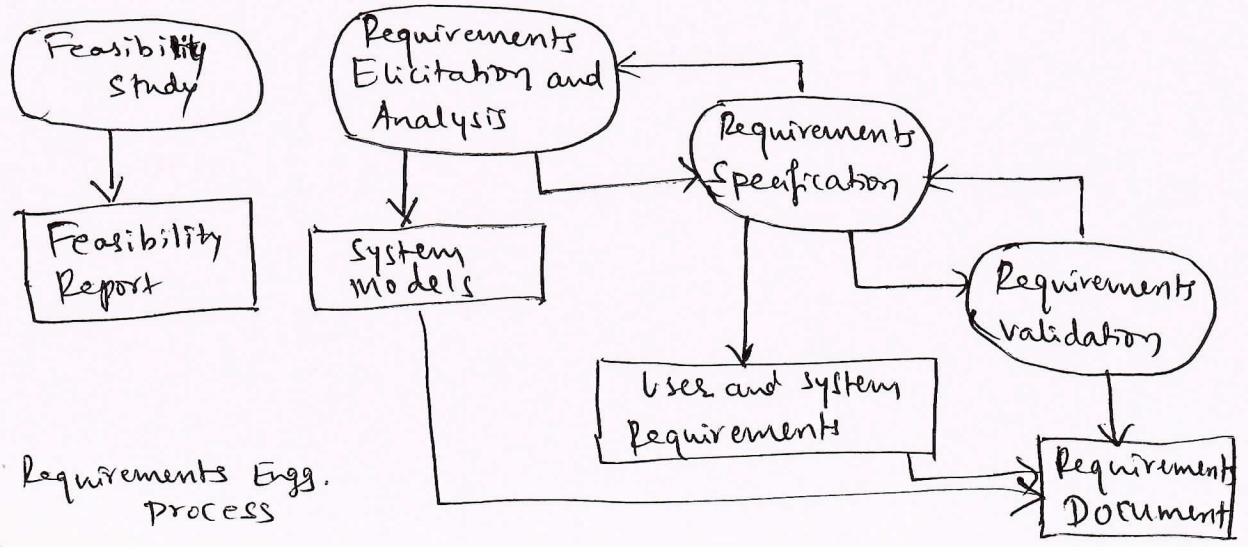


Fig: Requirements Engg. Process

1. Feasibility study : A Feasibility study should be relatively cheap and quick; Chele for budgetary constraints, Available hardware and software Technologies ; will it be a cost effective ?

2. Requirements elicitation and analysis : This is the process of deliving the system requirements through observation of existing systems, discussions with potential users and procurers.

3. Requirements specification - is the activity of translating the information gathered during the analysis activity into a document that defines a set of requirements.

4. Requirements validation: This activity checks the requirements for realism, Consistency and Completeness

2b. The software requirements document (Software requirements specification or SRS) is an official statement of what the system developers should implement.

The structure of a requirements document –

Chapter	Description
preface	This should define the expected readership of the document and describe its version history, including a rationale for the creation of a new version and a summary of the changes made in each version
Introduction	This should describe the need for the system. It should briefly describe the system's functions and explain how it will work with other systems.
Glossary	: This should define the technical terms used in the document
User requirements definition	: Here, you describe the services provided for the user. The non-functional system requirements should also be described in this section
System Architecture	: This chapter should present a high level overview of the anticipated system Architecture.

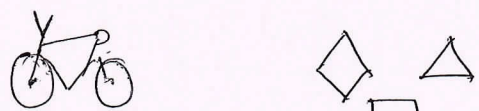
System requirements Specification	This should describe the Functional and non-functional requirements in more detail
System models	This might include graphical system models showing the relationships between the system components, the system and its environment
System Evolution	This should describe the fundamental assumptions on which the system is based and any anticipated changes due to hardware evolution, changing user needs and so on.
Appendices	These should provide detailed, specific information that is related to the application being developed; for example, hardware and database descriptions.
Index	Several indexes to the document may be included. As well as a normal alphabetic index, there may be an index of diagrams, an index of functions and so on.

3a. Object Orientation(OO) means that we organise Software as a Collection of discrete objects that incorporate both data structure and behaviour.

There are Four aspects (characteristics) required by an OO approach

1. Identity 2. classification
3. Inheritance 4. polymorphism

Identity: It means that data is quantised into discrete, distinguishable entities called objects e.g objects: personal Computer, bicycle etc
Each Object has its own inherent identity
i.e two objects are distinct even if all their attribute values are identical)

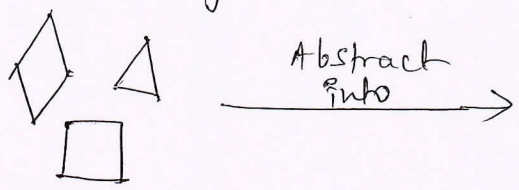


objects

Classification: It means that objects with the Same data structure (attribute) and behaviour (operations) are grouped into a class.
e.g: paragraph, monitor etc
Each object is said to be an instance of its class. Each class describes a set of individual objects.

Objects and classes

Polygon Objects



Polygon class

Attributes:
- Vertices
- Border class
- Fill color

Operations:
- Draw
- Erase
- Move

Inheritance: It is the sharing of Attributes and operations (features) among classes based on a hierarchical relationship. A super class has general information that sub classes refine and elaborate. Each subclass incorporates or inherits all the features of its super class and adds its own unique features e.g scrolling window and fixed window are sub classes of window

Polymorphism:

polymorphism means that the same operation may behave differently for different classes.
Example: move operation behaves differently for a pawn than for the queen in a chess game.

3b. List and explain object oriented themes which supports support object oriented Technologies

Ans:- Object oriented Themes

- ✓ Abstraction
- ✓ Encapsulation
- ✓ Combining data and behaviour
- ✓ Sharing
- ✓ Emphasis on the essence of an object
- ✓ Synergy

Abstraction: Abstraction lets you focus on essential aspects of an application while ignoring details i.e focusing on what an object is and does, before deciding how to implement it.

Encapsulation (information hiding):

- ✓ It separates the external aspects of an object (that are accessible to other objects) from the internal implementation details (that are hidden from other objects)
- ✓ Encapsulation has the ability to combine data structure and behaviour in a single entity

Combining data and behaviour

- ✓ Maintenance is easier
- ✓ Operator polymorphism shifts the burden of deciding what implementation to use from calling code to the class hierarchy.
- ✓ In OO system the data structure hierarchy matches the operation inheritance hierarchy

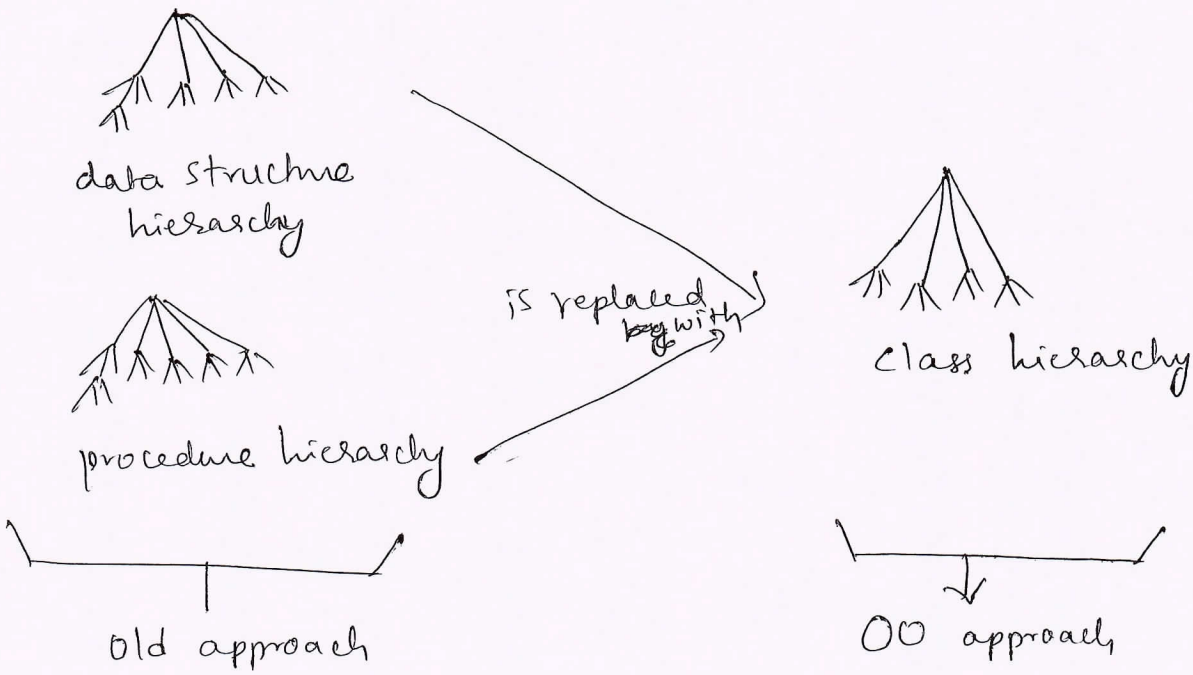


Fig: OO vs prior approach

Sharing : ✓ OO Techniques provide sharing at different levels

✓ Inheritance of both data structure and behaviour lets subclasses share common code

Emphasis on the essence of an object : OO development places a greater emphasis on data structure and a lesser emphasis on procedure structure

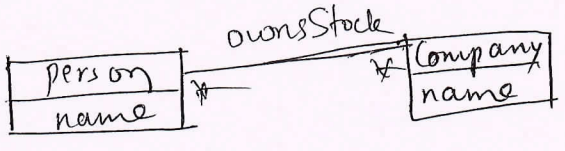
Synergy : Identity, classification, polymorphism and Inheritance — Each of these concepts can be used in isolation but together they complement each other synergistically

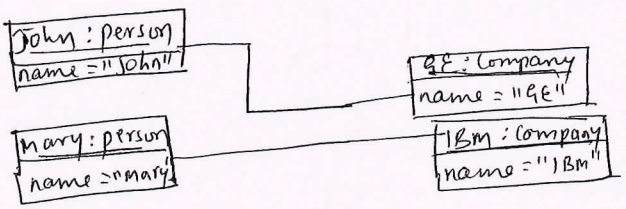
4a. Briefly explain Links, Associations, ordering, Bags and sequences with an example each

Link : A link is a physical or conceptual connection among objects. e.g Joesmith worksFor Simplex Company. Its an instance of association

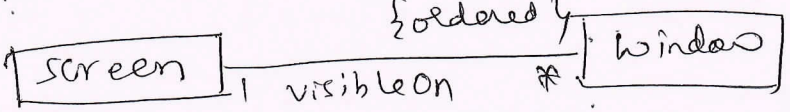
Association : It is a description of a group of Links with common structure and common semantics. Associations are inherently bidirectional

Class diagram

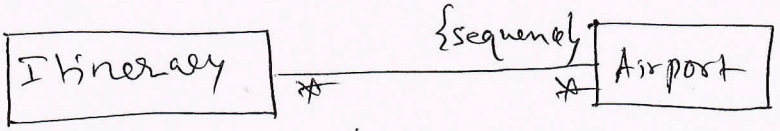




ordering : The ordering is an inherent part of association. You can indicate an ordered set of objects by writing "{ordered}" next to the appropriate association end



Bags and Sequences : A bag is a collection of elements with duplicates allowed. A sequence is an ordered collection of elements with duplicates allowed.



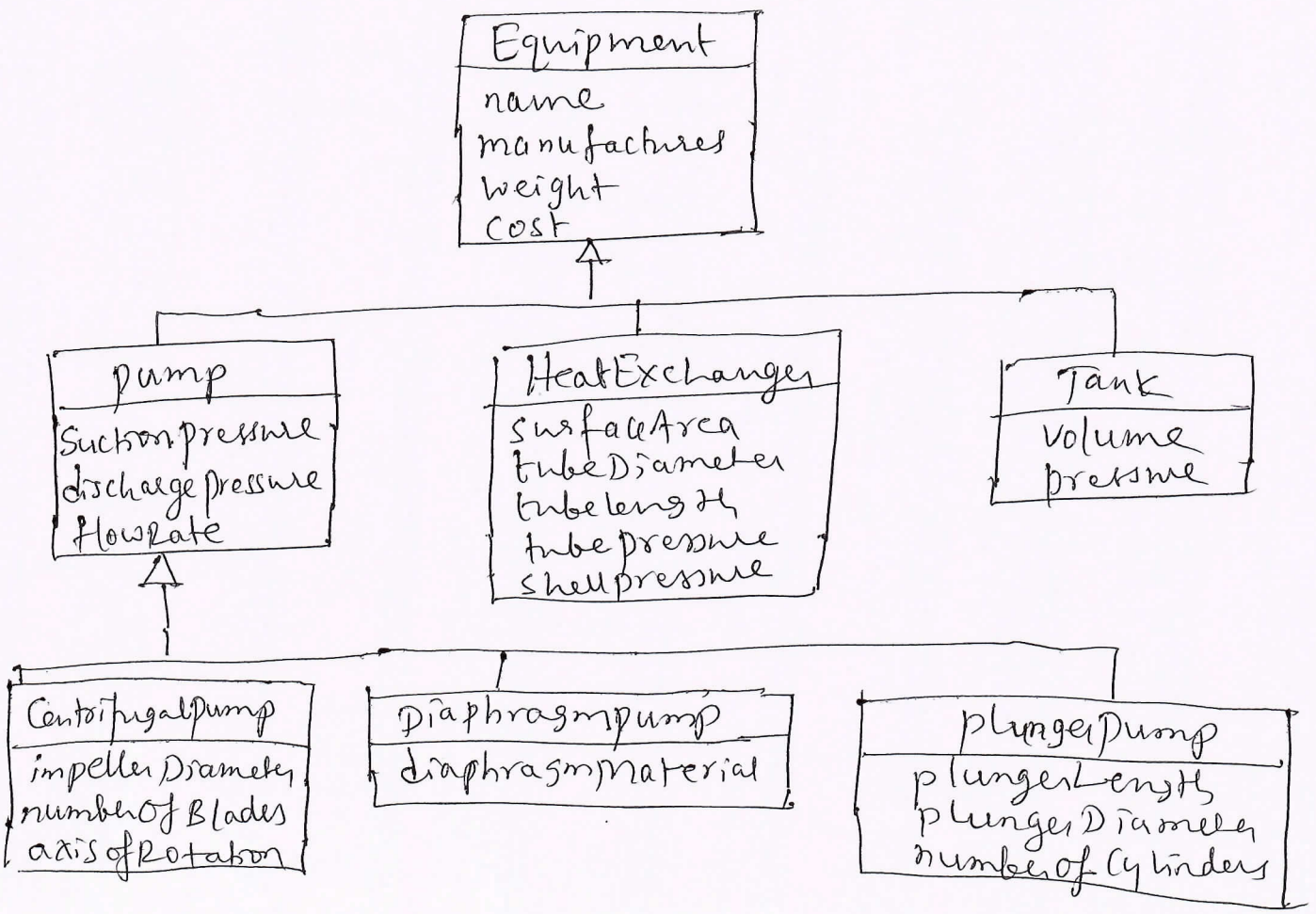
4b Explain Generalization and Inheritance with an example each

A: Generalization is the relationship between a class (superclass) and one or more variations of the class (the sub class)

Generalization organizes classes by their similarities and differences, structuring the description of objects.

Generalization is sometimes called the "is-a" relationship because each instance of a subclass is an instance of the superclass as well.

Generalization for equipment



5a What is system modeling? Explain the different perspective that the system model developed.

A: System Modeling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system

Different Models can be developed to represent the system from different perspectives
e.g

1. An External perspective, where you model the Context or environment of the system

2. An Interaction perspective where you model the interactions between a system and its environment or between the components of a system

3. A structural perspective, where you model the organization of a system or the structure of the data that is processed by the system

4. A behavioural perspective, where you model the dynamic behaviour of the system and how it responds to events

Context model : A simple context model that shows the patient information system and the other systems in its environment

The MHC - PMS system is intended to manage information about patients attending mental health clinics and the treatments that have been prescribed.

Interaction models : Modeling user interaction is important as it helps to identify user requirements.

Two related approaches to interaction modeling are:

1. Use case modeling, which is mostly used to model interactions between a system and external actors
2. Sequence diagrams, which are used to model interactions between system components, although external agents may also be included.

Structural models - display the organization of a system in terms of the components that make up that system and their relationships

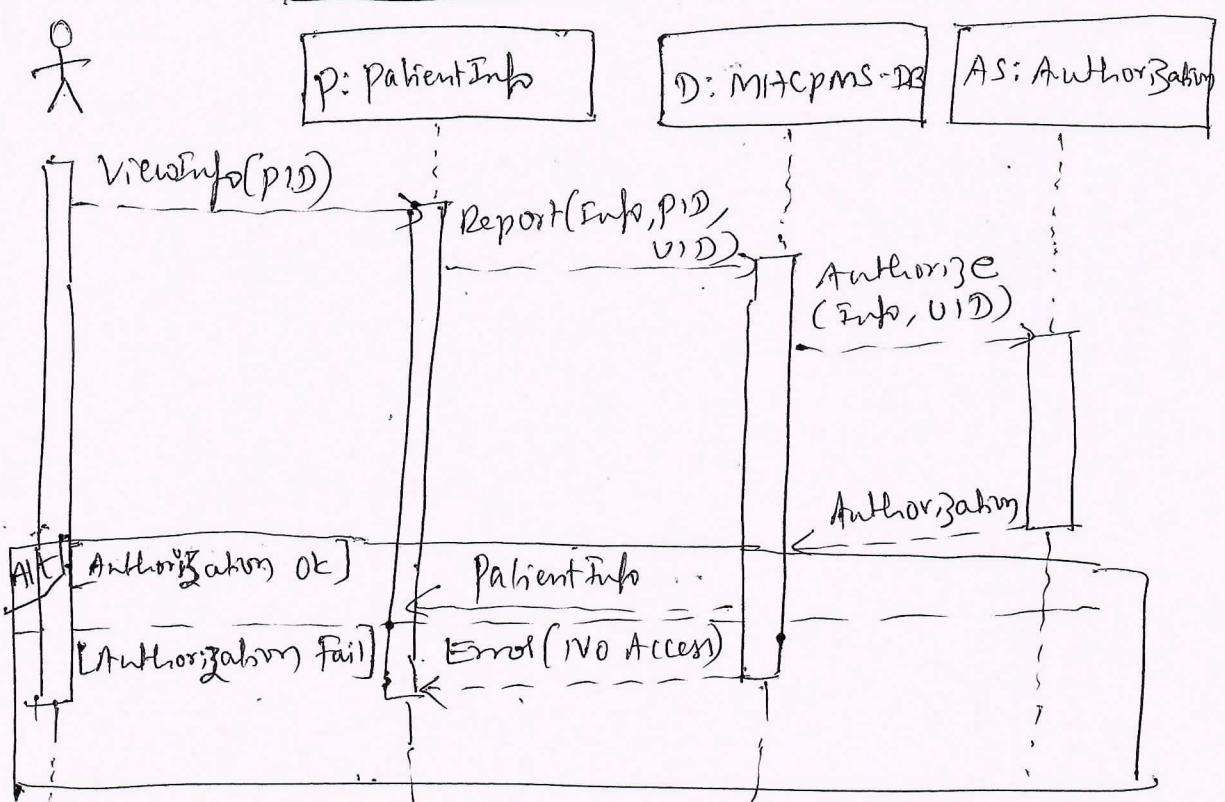
Behavioural models - Behavioral models are models of the dynamic behaviour of the system as it is executing.

5b. Illustrate sequence Diagram with an example to view patient information

A: The medical receptionist triggers the viewInfo method in an instance p of the PatientInfo object class, supplying the patient's Identifier, PID. P is a User Interface Object which is displayed as a form showing patient Information

The instance P calls the database to retrieve the information required, supplying the receptionist's identifier to allow security checking

Sequence Diagram for view patient Info



The Database checks with an authorization System that the user is authorized for this action. If authorized, the patient Information is returned, Else an Error message is returned.

6a Explain Event Driven model with a state Diagram of Microwave oven application

A! Event-driven modeling shows how a system responds to external and internal events. It is based on the assumption that a system has a finite number of states and that events may cause a transition from one state to another.

The simple microwave has a switch to select full or half power, a numeric keypad to input the cooking time, a start/stop button and an alphanumeric display

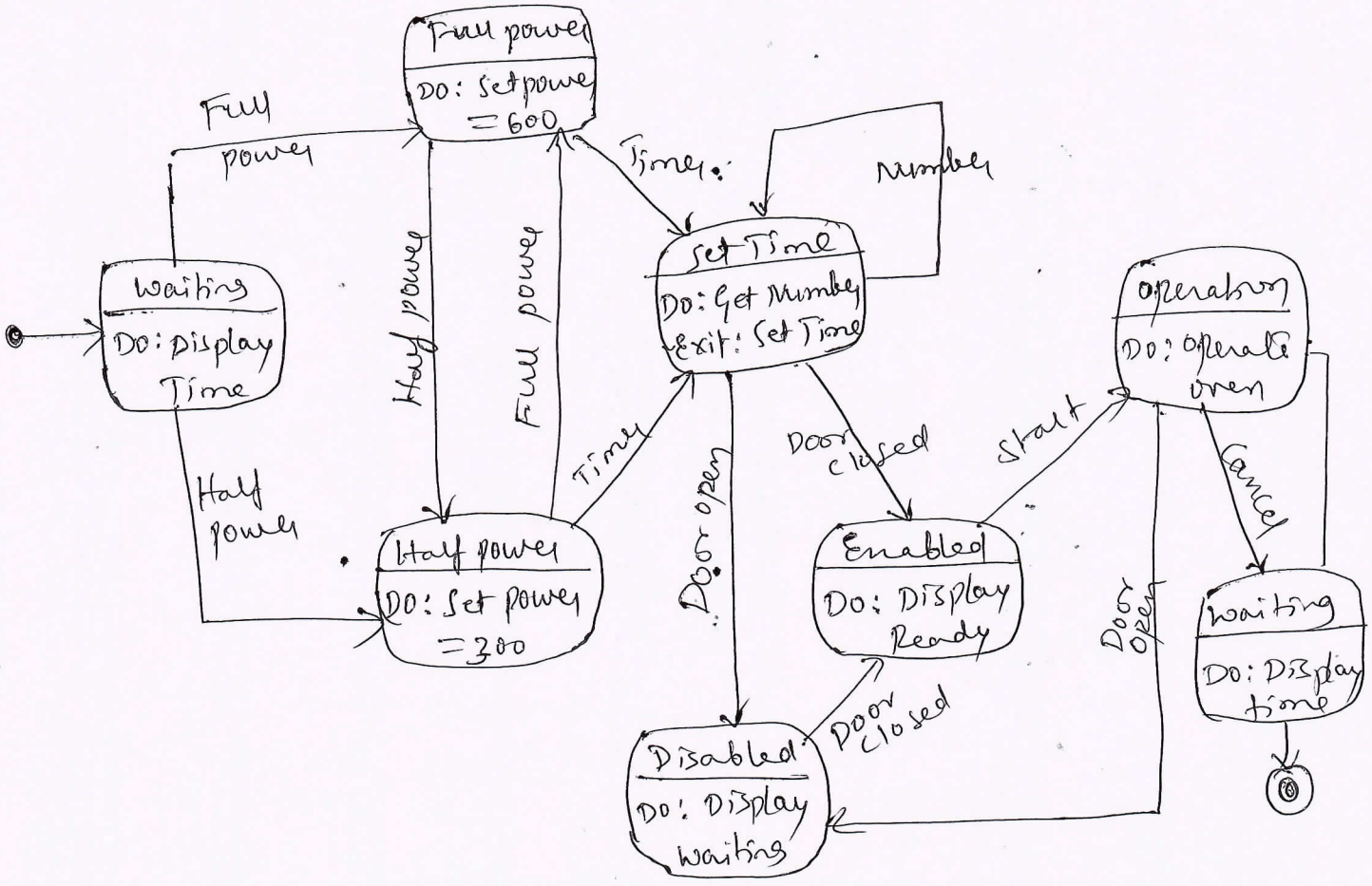


Fig: State Diagram of a Microwave Oven

6b. Define Design patterns. Briefly explain the essential elements of design patterns

A: It is description of the problem and the essence of its solution so that the solution may be reused in different settings

patterns are way of reusing the knowledge and Experience of other designers

One can think of pattern as a description of accumulated wisdom and experience, a well-tried solution to a common problem,

The Four essential elements of design patterns:

1. A name that is a meaningful reference to the pattern
2. A description of the problem area that explains when the pattern may be applied
3. A solution description of the parts of the design solution, their relationships, and their responsibilities. It is a template for a design solution. It is expressed graphically.
4. A statement of the consequences — the results and trade-offs — of applying the pattern.

Module-4

7a. Discuss Test Driven Development (TDD) with its process and list benefits

A: Test-driven development (TDD) is an approach to program development in which you interleave testing and code development

Essentially, you develop the code incrementally along with a test for that increment. You don't move on to the next increment until the code that you have developed passes its test.

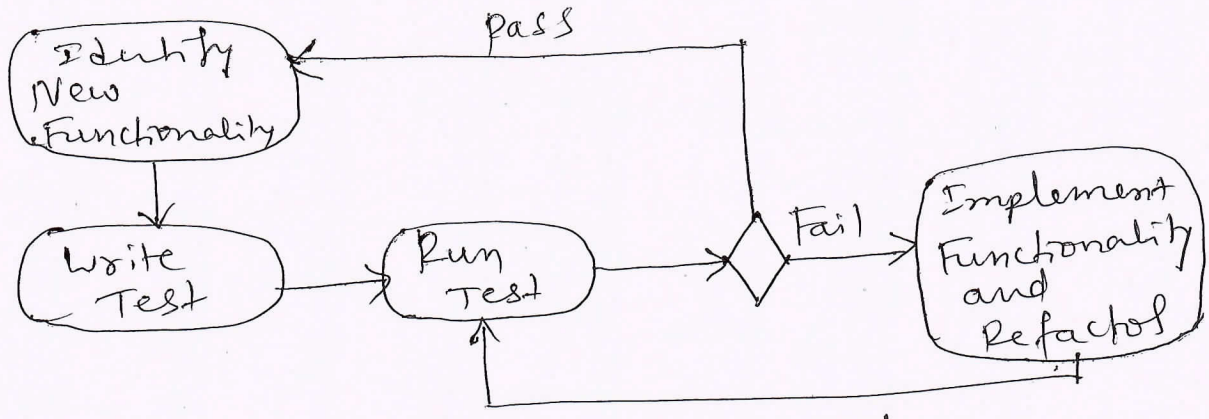


Fig: Test-Driven Development

Benefits of Test-Driven Development are:

1. Code Coverage
2. Regression Testing
3. Simplified debugging
4. System Documentation

7b. Explain software evolution process with neat block diagram

A: Software development does not stop when a system is delivered but continues throughout the lifetime of the system.

Software evolution is important because organisations have invested large amounts of money in their software and are now completely dependent on these systems.

Software evolution may be triggered by changing requirements

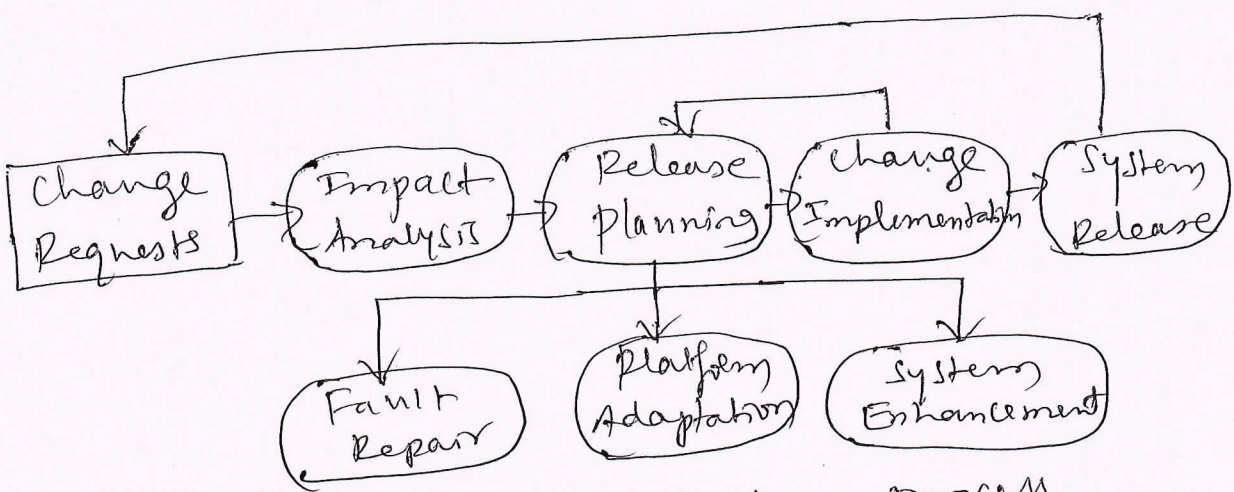


Fig: Software evolution process

The cost and impact of these changes are assessed to see how much of the system is affected by the change and how much it might cost to implement the change.

If the proposed changes are accepted, a new release of the system is planned

During release planning, all proposed changes are considered.

8a. Discuss Lehaman's Laws of program evolution dynamics

A: Program Evolution dynamics is the study of system change

Lehaman's Laws:

1. Continuing change: A program that is used in a real-world environment must necessarily change or else become progressively less useful in that environment.

2. Increasing complexity: As an evolving program changes, its structure tends to become more complex. Extra resources must be devoted to preserving and simplifying the structure

3. Large program Evolution: Program Evolution is a self-regulating process.

4. Organisational stability: Over a program's lifetime, its rate of development is approximately constant and independent of the resources devoted to system development

5. Conservation of familiarity: Over the lifetime of a system, the incremental change in each release is approximately constant.

Continuing growth: The functionality offered by systems has to continually increase to maintain user satisfaction

Declining quality: The quality of systems will decline unless they are modified to reflect changes in their operational environment.

Feedback system: Evolution processes incorporate multiagent, multiloop feedback systems and you have to treat them as feedback systems to achieve significant product improvement.

8b. Explain Reengineering process with a neat block diagram

A: To make legacy software systems easier to maintain, you can reengineer these systems to improve their structure and understandability.

Reengineering may involve redocumenting the system, refactoring the system architecture, translating programs to a modern programming language and modifying and updating the structure and values of the system's data.

There are two important benefits from reengineering rather than replacement:

1. Reduced risk 2. Reduced cost

A general model of the reengineering process:

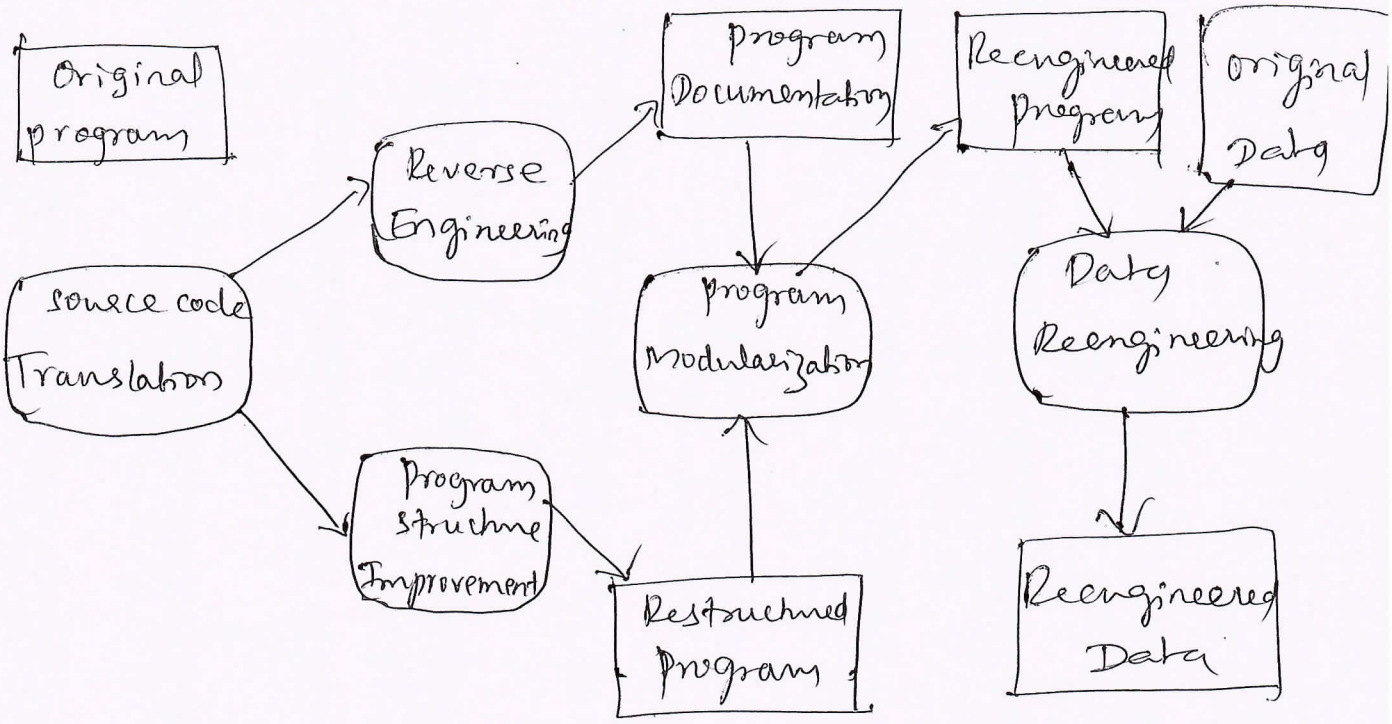


Fig: re-engineering process

Module-5

9a: Discuss project plan. Explain various sections of project plan

In a plan-driven development project, a project plan sets out the resources available to the project, the work breakdown and a schedule for carrying out the work. The plan should identify risks to the project.

Plans normally include the following sections:

1. Introduction: This briefly describes the objectives of the project and sets out the constraints (e.g budget, time etc) that affect the management of the project.

2. project organization: This describes the way in which the development team is organized, the people involved, and their roles in the team.

3. Risk analysis: This describes possible project risks, the likelihood of these risks arising and the risk reduction strategies that are proposed.

4. Hardware and software resource requirements: This specifies the hardware and support s/w required to carry out the development. If hardware has to be bought, estimates of the prices and the delivery schedule may be included

5. work breakdown: This sets out the breakdown of the project into activities and identifies the milestones and deliverables associated with each activity. Milestones are key stages in the project where progress

Can be assessed, deliverables are work products that are delivered to the customer.

6. project schedule: This shows the dependencies between activities, the estimated time required to reach each milestone and the allocation of people to activities

7. Monitoring and reporting mechanisms:

This defines the management reports that should be produced, when these should be produced, and the project monitoring Mechanisms to be used.

Supplementary plans: quality plan, validation plan, configuration management plan, maintenance plan, staff development plan

9b- With a neat diagram explain project scheduling process

A: Project scheduling is the process of deciding how the work in a project will be organized as separate tasks, and when and how these tasks will be executed.

You estimate the calendar time needed to complete each task, the effort required and who will work on the tasks that have been identified

- Estimate the resources needed to complete each task, such as the disk space required on a server, the time required on specialized hardware.

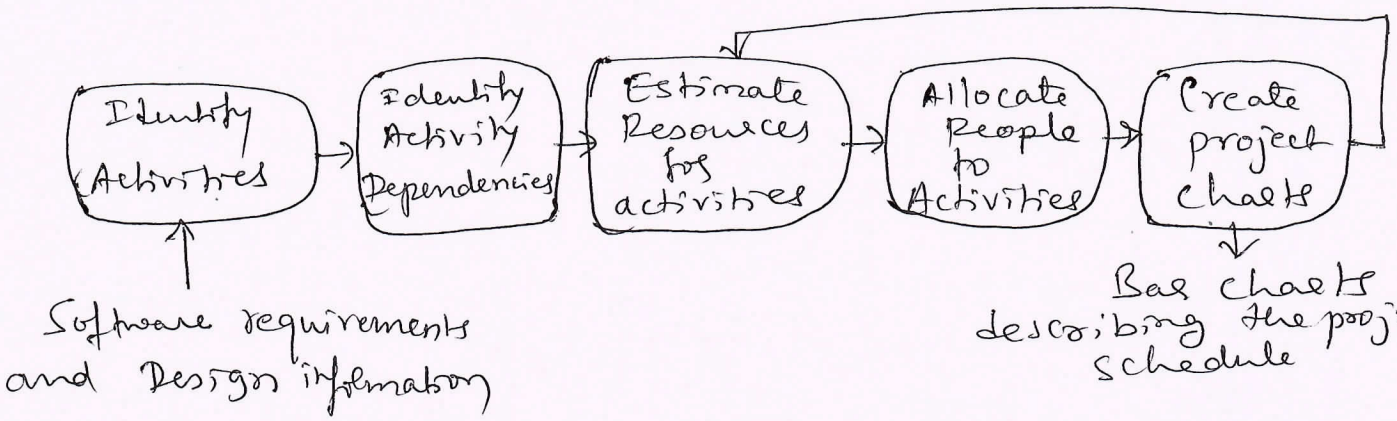


Fig: project scheduling process

Scheduling in plan-driven projects involves breaking down the total work involved in a project into separate tasks and estimating the time required to complete each task. Tasks should normally last at least a week, and no longer than 2 months

If the project is new and technically advanced, parts of it may turn out to be more difficult and take longer than originally anticipated

10. a. Discuss software quality and its attributes. Explain process based quality

A: · S/w quality is about whether the s/w functionality has been correctly implemented and depends on non-functional system attributes S/w quality Attributes are safety, understandability, portability, security, Testability, usability Reliability, Adaptability, Reusability, Efficiency

An assumption that underlies software quality management is that the quality of software is directly related to the quality of the software development process.

You measure the quality of the product and change the process until you achieve the quality level that you need. Following Figure illustrates this process process-based approach to achieving product quality.

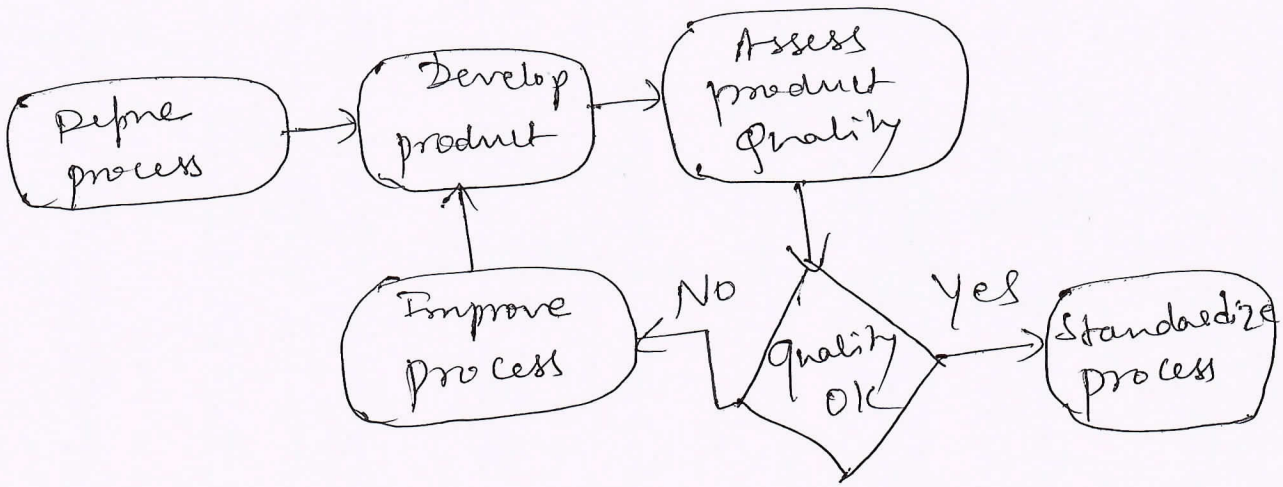


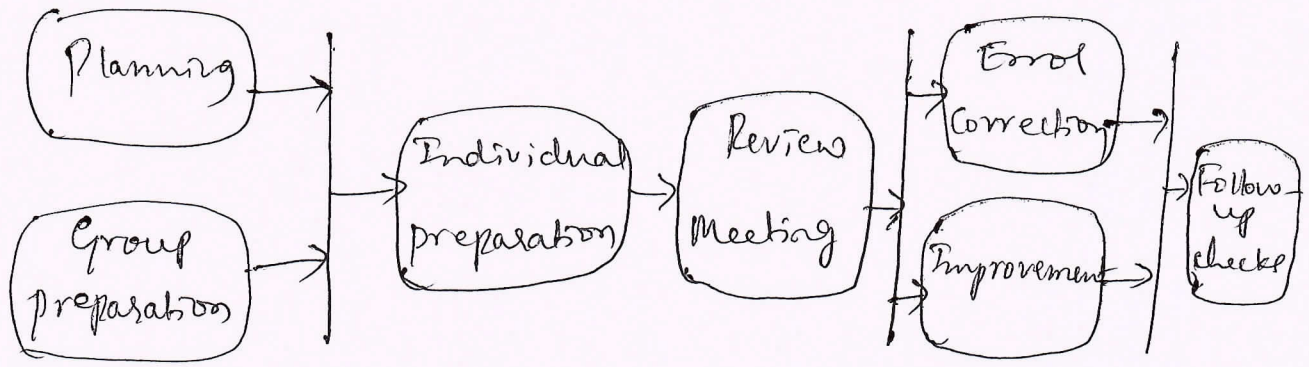
Fig: process based quality

10 b. Explain software reviews and inspections of quality Assurance

A: Reviews and Inspections are quality Assurance (QA) activities that check the quality of project deliverables. This involves examining the software, its documentation and records of the process to discover errors and omissions and to see if quality standards have been followed.

Reviews and inspections are used alongside program testing as part of the general process of software verification and validation.

The review process



The review process is normally structured into three phases: 1. pre-review Activities 2. The review meeting 3. post-review Activities

Staff.

H.O.D

HOD
Computer Science & Engineering
KLS Vishwanathrao Deshpande

Dean Academic