CBCS SCHEME

USN | | | | | | | | | | |

18CS35

Third Semester B.E. Degree Examination, Jan./Feb. 2021
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. What are the attributes of good software? Explain the key challenges faced in software engineering. **(08 Marks)**

b. With a neat diagram, explain the waterfall model of software development process. **(06 Marks)**

c. Describe the general model of software design process. **(06 Marks)**

OR

2 a. Define and differentiate functional and non-functional requirements. **(06 Marks)**

b. What is requirements specification? Explain various ways of writing system requirements. **(08 Marks)**

c. What is ethnography? How ethnography is effective in discovering the types of requirements? **(06 Marks)**

Module-2

3 a. What is OO development? Explain object oriented themes briefly. **(08 Marks)**

b. What are links and associations? Write and explain UML notation for links and association with an example. **(06 Marks)**

c. Describe generalization and inheritance with an example. **(06 Marks)**

OR

4 a. What is object orientation? What are the important characteristics of OO approach? Explain. **(08 Marks)**

b. Define model. Describe the relationship among three models. **(08 Marks)**

c. With the help of a sample class model explain multiplicity and Association and names. **(04 Marks)**

Module-3

5 a. Draw and explain a contest model for patient information system. **(06 Marks)**

b. With a diagram, explain the phases in the Rational Unified Process (RUP). **(06 Marks)**

c. With the help of a neat state diagram, illustrate the working of a microwave oven. **(08 Marks)**

OR

6 a. What is model driven engineering? State the three types of abstract system model produced with a neat diagram. **(08 Marks)**

b. What are the activities to the carried out in object oriented design process of a system? How the objects are identified? **(08 Marks)**

c. What is open source development? Explain general models of open source licensing. **(04 Marks)**

Module-4

a. What is list driven development? With a neat diagram, explain test driven development process. **(08 Marks)**

b. With neat diagram, explain six stages of acceptance testing process. **(08 Marks)**

c. What are the different types of interfaces to be tested during component testing? Explain. **(04 Marks)**

OR

a. Write and explain Lehman's laws related to system charge. **(08 Marks)**

b. What is software maintenance? Draw the general model of reengineering process and explain. **(08 Marks)**

c. What are the strategic options involved in legacy system management? Discuss. **(04 Marks)**

Module-5

a. For the set of tasks shown below draw the activity bar chart for the project scheduling.

Task	Duration (Days)	Dependencies
T_1	10	-
T_2	15	
T_3	15	T_1 (M1)
T_4	10	-
T_5	10	T_2, T_4 (M3)
T_6	5	T_1, T_2 (M4)
T_7	20	T_1 (M1)
T_8	25	T_4(M2)
T_9	15	T_3, T_6 (M5)
T_{10}	15	T_7, T_8 (M6)
T_{11}	10	T_9 (M7)
T_{12}	10	T_{10}, T_{11} (M8)

(08 Marks)

b. Write and explain the factors affecting software pricing. **(05 Marks)**

c. Explain briefly the algorithm cost modeling and write the difficulties. **(07 Marks)**

OR

a. With a diagram, explain the phase involved in software review process. **(08 Marks)**

b. Explain briefly the key stages in the process of product measurement. **(08 Marks)**

c. Write any four product and process standards. **(04 Marks)**

* * * * *

Third Semester B.E Degree Examination
Jan/Feb 2021

Subject : Software Engineering - 18CS35
Faculty : Prof. Saleem. Hebbal
Max. Marks: 100

Solutions

Module 1

Q:- N: 1 a) What are the attributes of good software? Explain the key challenges faced in software engineering

A:- Major Good Attributes :
a) maintainable b) dependable c) usability (8M)
d) Effiency e) Acceptability - Explain - 4M

challenges :-

1. coping with increasing Diversity : The challenge here is to develop techniques for building dependable s/w that is flexible enough to cope with the heterogeneity - 4M

2. Demand for reduced delivery Time :- need to reduce the delivery time of software

3. Developing trustworthy software - it is essential that we can trust the software. Make sure that malicious users cannot attack our software and that information security is maintained

1b : water fall model of software development process 6M

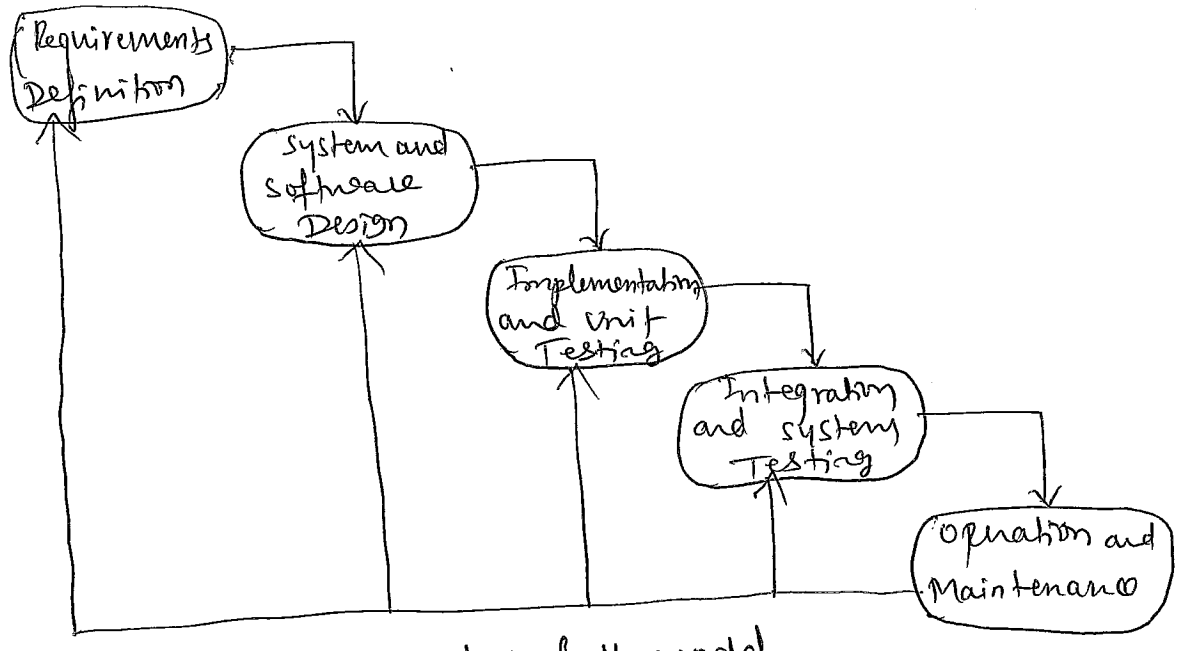


Fig: 1b : water fall model

- ✓ A software process model is a simplified representation of a software process.
- ✓ The waterfall model is consistent with other engineering process models and documentation is produced at each phase
- ✓ This makes the process visible so that managers can monitor progress against the development plan
- ✓ Its major problem is the inflexible partitioning of the project into distinct stages.
- ✓ In principle, the waterfall model should only be used when the requirements are well understood and unlikely to change radically during System development.

1C General model of software design process 6m

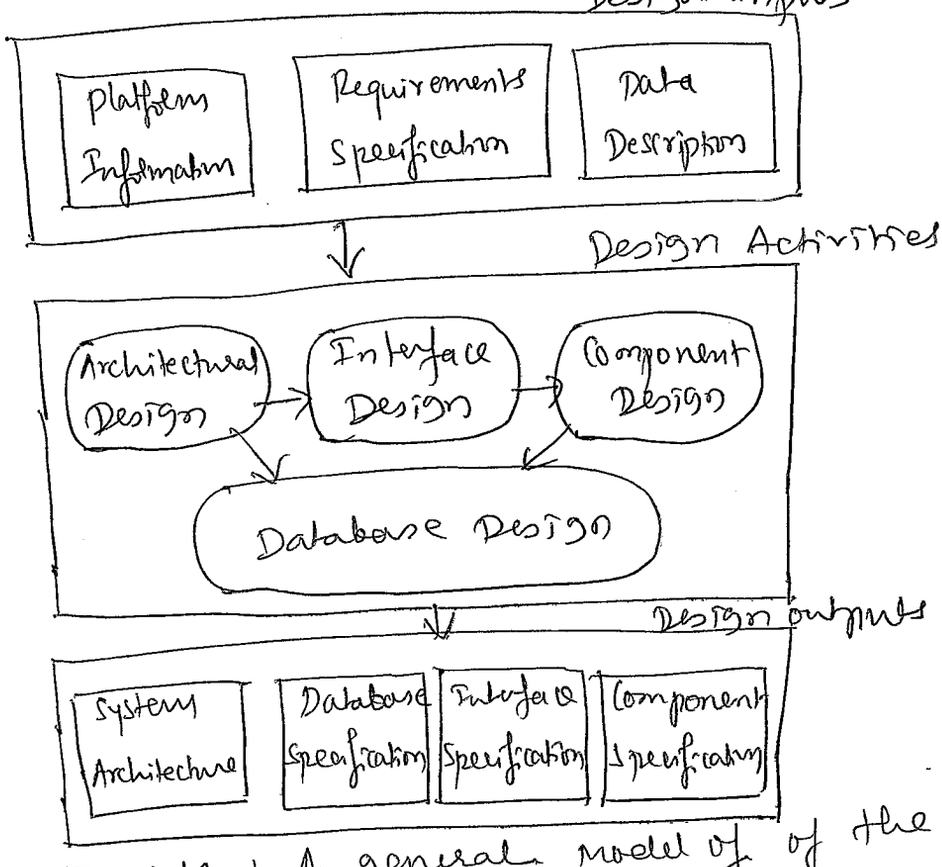


Fig: 1C.1 A general model of of the Design process

✓ A Software Design is a description of the structure of the software to be implemented, the data models and structures used by the system, the interfaces between system components and sometimes the algorithms used.

✓ Designers do not arrive at a finished design immediately but develop the design iteratively.

✓ Fig 1C.1 is an abstract model of this process showing the inputs to the design process, process activities and the documents produced as outputs from this process

2a. Define Functional & non-functional requirements

Functional requirements: These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations

Non-functional requirements: These are constraints on the services of functions offered by the system. They include timing constraints, constraints on the development process and constraints imposed by standards.

The functional requirements specification of a system should be both complete and consistent.
Completeness means that all services required by the user should be defined. Consistency means that requirement should not have contradictory definitions.

1. product requirements
   - ✓ Usability Requirements
   - ✓ Efficiency Requirements — performance requirements / space "
   - ✓ Dependability "
   - ✓ security "

2. organisational Requirements
   - ✓ Environmental Requirements
   - ✓ Operational "
   - ✓ Development "

3. External Requirements:
   - ✓ Regulatory Requirements
   - ✓ Ethical "
   - ✓ Legislative — Accounting Requirements / Safety/Security Requirements

2b Requirements specification is the process of writing down the user and system requirements in a requirements document. 8m

Ways of writing a system requirements specification:

- ✓ Natural language sentences
- ✓ Structured natural language
- ✓ Design description languages
- ✓ Graphical Notations
- ✓ Mathematical specifications

Natural language sentences:- The requirements are written using numbered sentences in natural language. Each sentence should express one requirement

Structured natural language: The requirements are written in natural language on a standard form or template. Each field provides information about an aspect of the requirement.

Design Description Languages: This approach uses a language like a programming language, but with more abstract features to specify the requirements by defining an operational model

Graphical Notations: Graphical models supplemented by text annotations are used to define the functional requirements for the system.

Mathematical Specifications : These notations are based on mathematical concepts such as finite-state machines of sets.

2c. Ethnography is an observational technique 6m that can be used to understand operational processes and help derive support requirements for these processes. An analyst immerses himself or herself in the working environment where the system will be used.

The day-to-day work is observed and notes made of actual tasks

The value of ethnography is that it helps discover implicit system requirements that reflect the actual ways that people work.

Ethnography is particularly effective for discovering two types of requirements :-

1. Requirements that are derived from the way in which people actually work
2. Requirements that are derived from co-operation and awareness of other people's activities.

module - 2

3a :. OO development := 8m

Object oriented Analysis (OOA)
+
Object-oriented Design (OOD)
+
Object oriented Programming (OOP)

OO development refers to the s/w life cycle : Analysis, Design and Implementation. The essence of OO Development is the identification and organization of application concepts

OO development is fundamentally a way of thinking and not a programming technique.

Object oriented themes :-

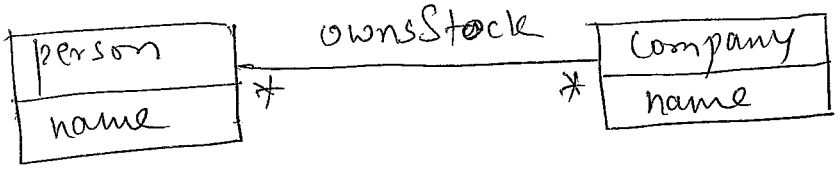
1. Abstraction 2. Encapsulation (information hiding)
3. Combining data and behaviour
4. Sharing 5. Emphasis on the essence of an object
6. Synergy

3b :- Links and Associations :- are the means for establishing relationships among objects and classes.

Link :- is a physical or conceptual connection among objects. e.g Joe smith worksFor Simplex Company.

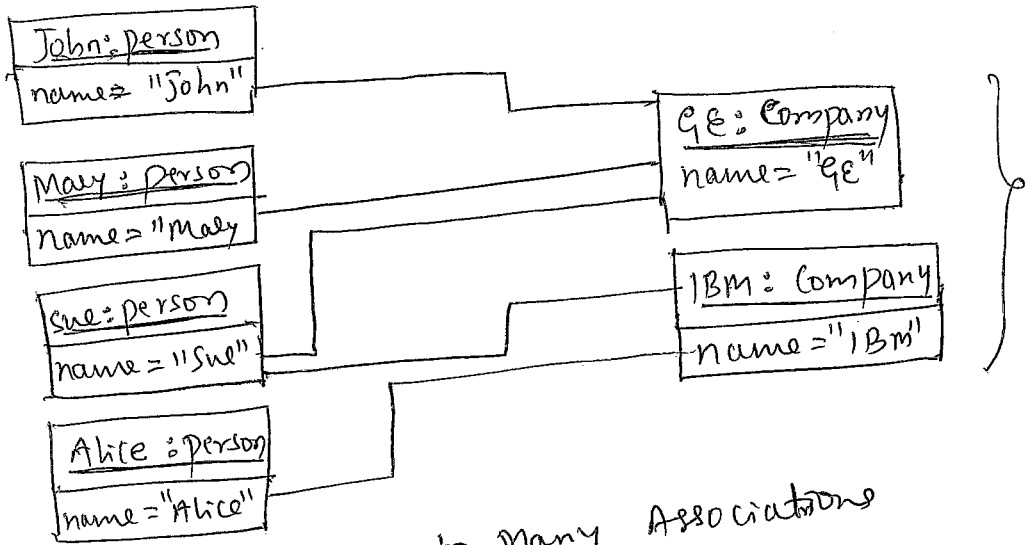
Association :- is a description of a group of links with common structure and common semantics as in the class Diagram.

For Example, a person worksFor a company.



} class Diagram

6m



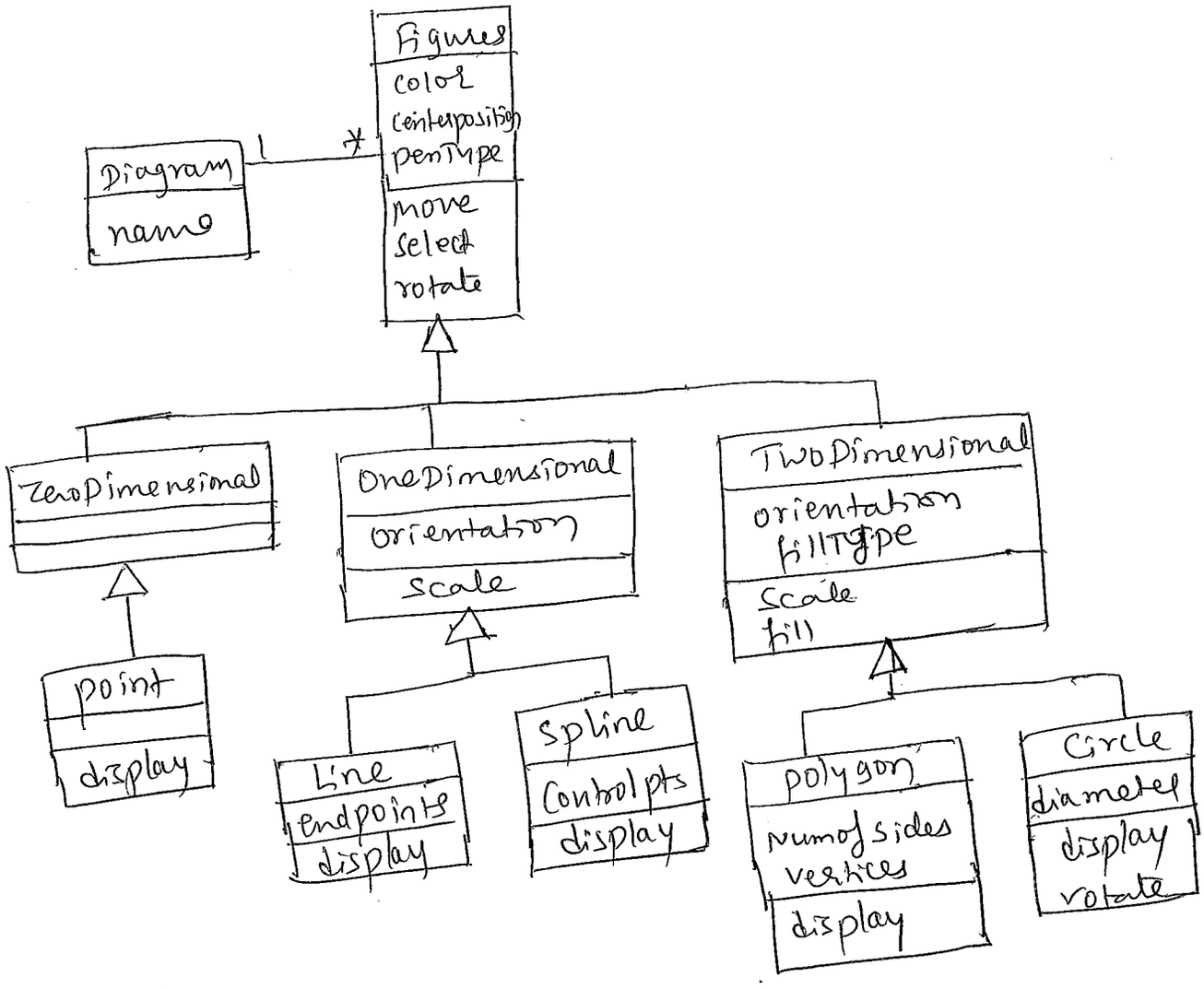
} Object Diagram

Fig 3b: Many to many Associations

3c. Generalization and Inheritance

- Generalization is the relationship between a class (superclass) and one or more variations of the class (subclass)
- Generalization organises 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.
- Inheritance is a mechanism for sharing attributes, operations and associations via generalization / specialization relationship

Inheritance for graphic figures



4a. Object orientation means that we organise Software as a collection of discrete objects that incorporate both data structures and behaviour (8m)

Four aspects required by an OO approach

- ✓ Identity
- ✓ classification
- ✓ Inheritance
- ✓ polymorphism

4b: A model is an abstraction of something for the purpose of understanding it before building it.

Three models: 8m

class model: represents the static, structural, "data" aspects of a system. It describe the structure of objects in a system — their identity, their relationships to other objects, their attributes and their operations.

State model: represents the temporal, behavioural, "control" aspects of a system. State model describes those aspects of objects concerned with time and the sequencing of operations

Interaction model: — represents the collaboration of individual objects, the — interaction aspects of a system. Interaction model describes interactions between objects — how individual objects collaborate to achieve the behaviour of the system as a whole

Relationship among the models:

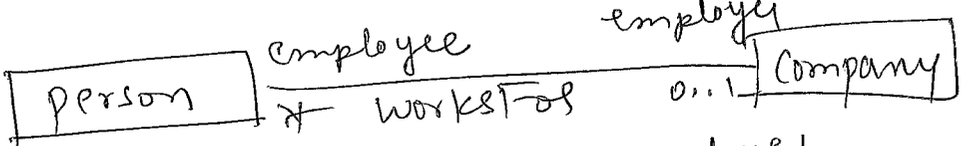
Each model describes one aspect of the system but contains references to the other models. The class model describes data structure on which the state and interaction models operate. The operations in the class model correspond to events and actions.

The state model describes the control structure of objects. It shows decisions that depend on object values and causes actions that change object values and state. The interaction model focuses on the exchanges between objects and provides a holistic overview of the operation of a system.

4c. Multiplicity : specifies the number of instances of one class that may relate to single instance of an associated class.
Multiplicity constrains the number of related objects

Association End names

- ✓ Association end names are especially convenient for traversing associations
- ✓ Association end names are necessary for Associations between two objects the same class.
- ✓ use of Association end names are optional but it is often easier and less confusing to assign association end Names



employee
Joe Doe
Mary B
Jean

employer
simplex
simplex
United widges

Sample class model

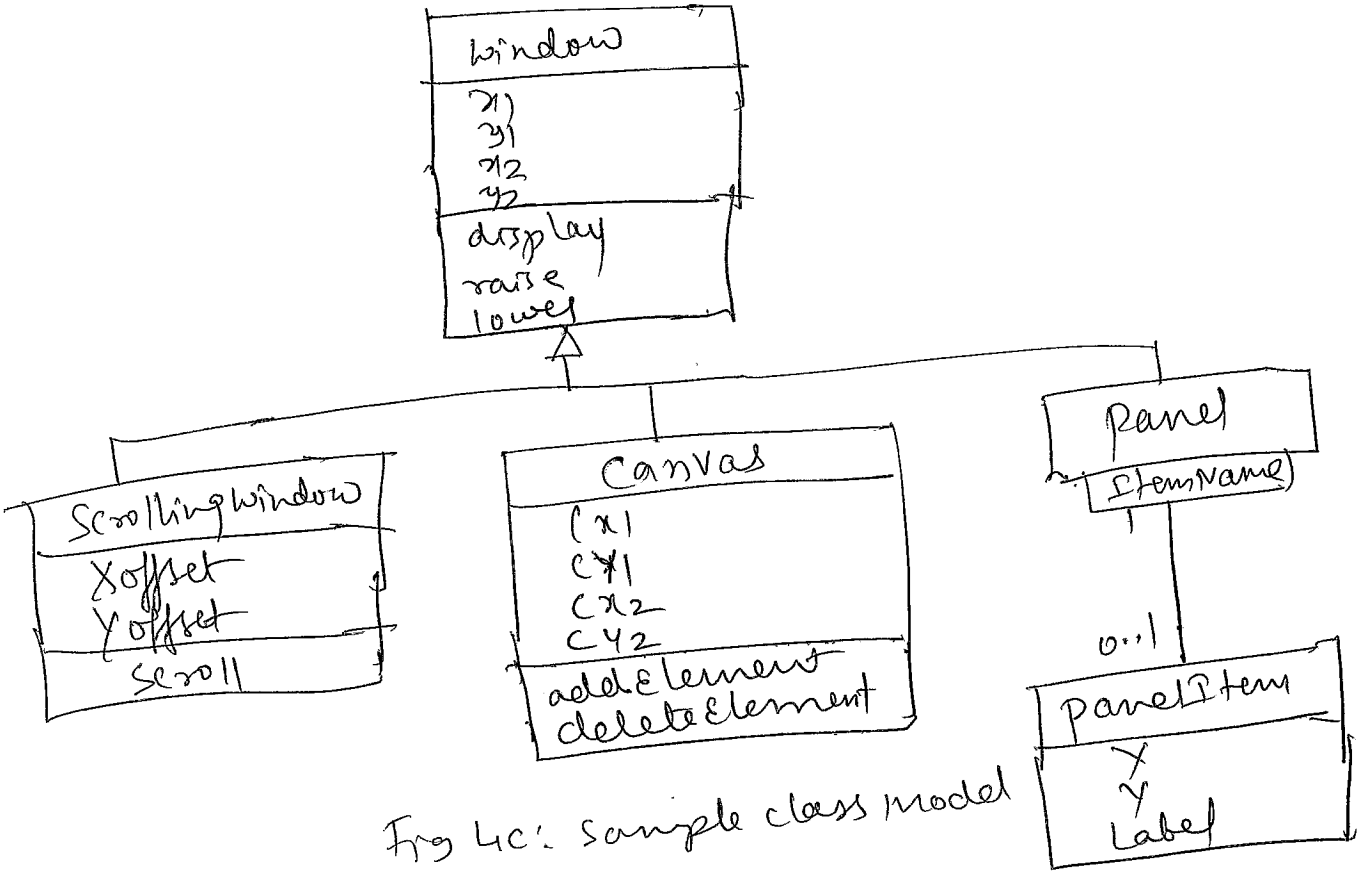


Fig 4c: Sample class model

Module 3

5a : Context model for patient information System – MHC-PMS – 6M

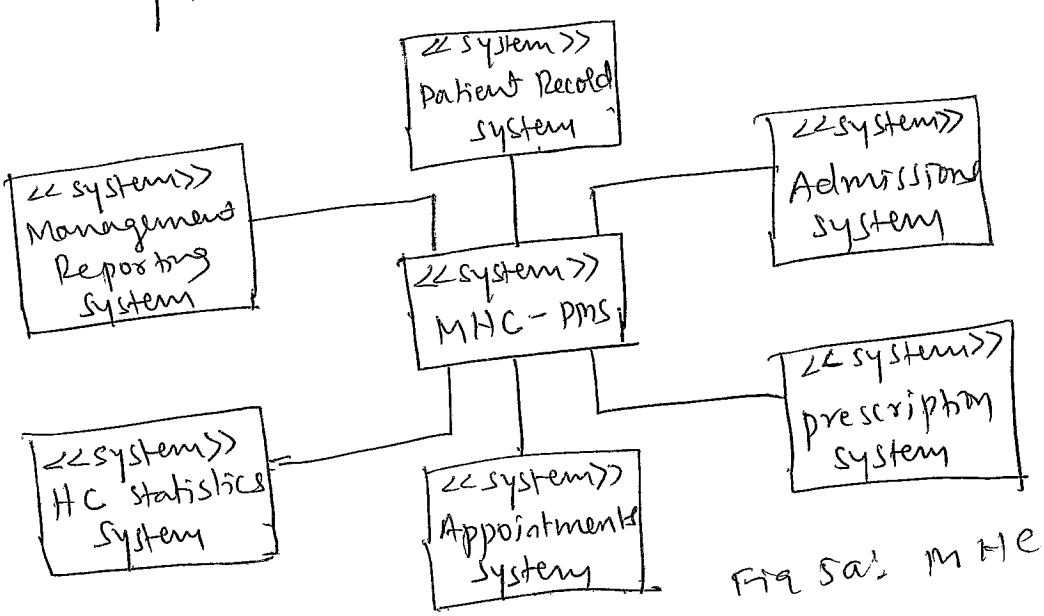


Fig 5a: MHC-PMS

This system is intended to manage information about patients attending mental health clinics and the treatments that

have been prescribed. The figure 5a is a simple context model that shows the patient information system and the other systems in its environment. You can see that the MHC-PMS is connected to an appointments system and a more general patient record system with which it shares data.

5b: Phases in Rational Unified process 6M

The Rational Unified process is an example of a modern process model that has been derived from work on the UML and the associated Unified software development process. It is a good example of a hybrid process model.

There are Four phases

1. Inception 2. Elaboration
3. Construction 4. Transition

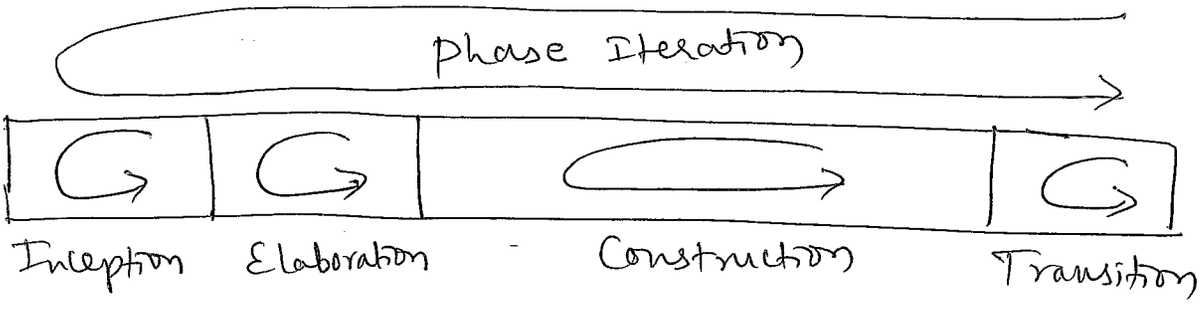
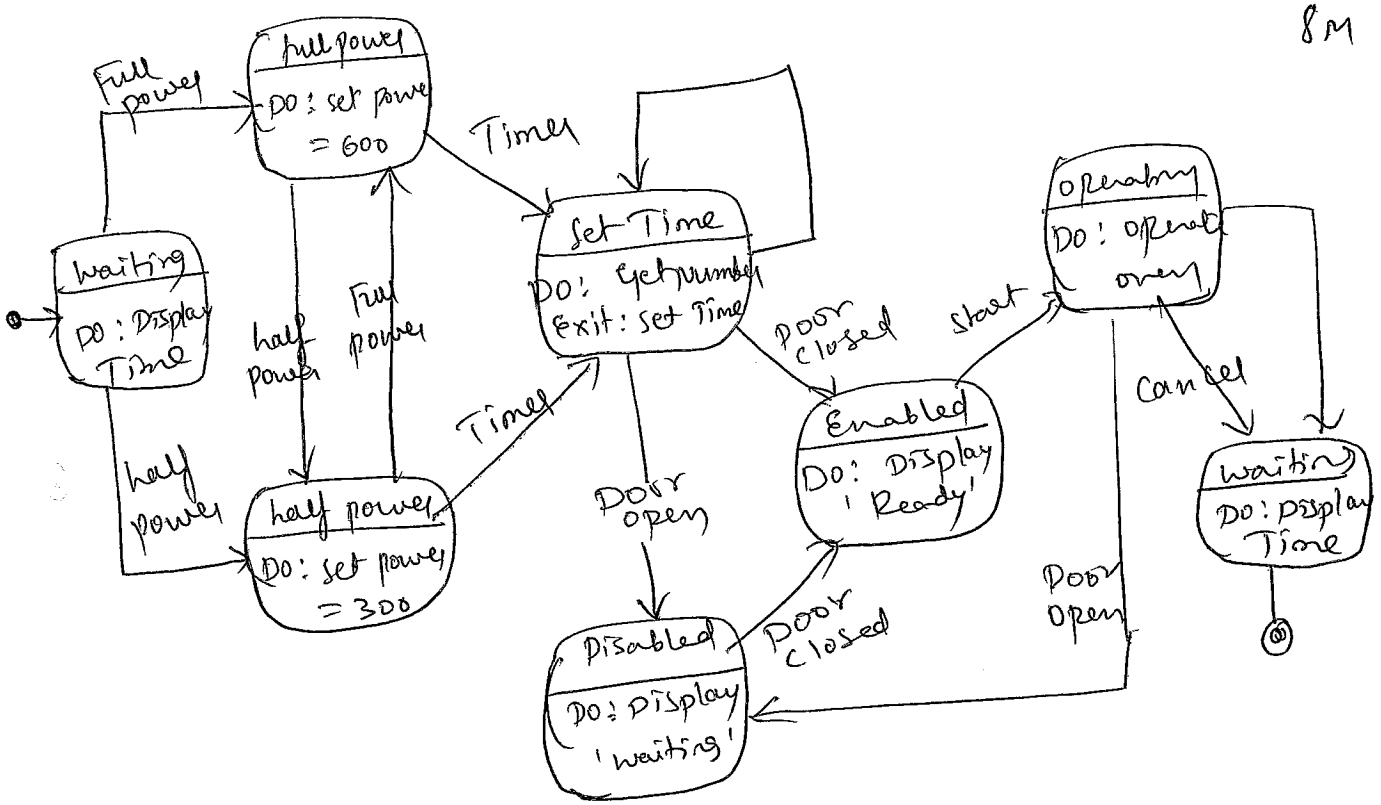


Fig.5b:- Phases in the Rational Unified process

5c state diagram of - microwave oven

8M



The UML notation lets you indicate the activity that takes place in a state.

It is assumed that the sequence of actions in using the microwave oven is

1. Select the power level
2. Input the cooking Time using a numeric keypad
3. press start and the food is cooked for a given time

6a : Model driven engineering (MDE) is an approach to software development where models rather than programs are the principal outputs of the development process. The programs that execute on a h/w/s/w platform are then generated automatically s/m from the models.

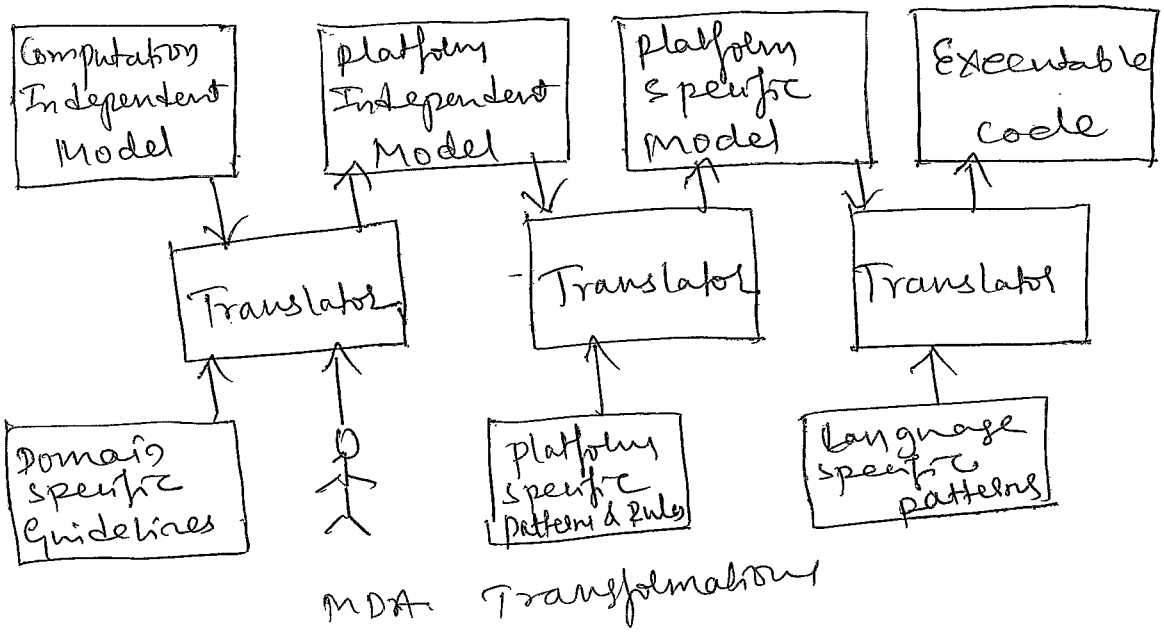
Model-driven engineering has its roots in model-driven Architecture (MDA)

Three types of Abstract system model:

1. A Computation independent model (CIM) that models the important abstractions used in the system. CIMs are sometimes called domain models.

2. A platform independent model (PIM) that models the operation of the system without reference to its implementation. The PIM is usually described using UML models that show the static system structure and how it responds to external and internal events

3. platform specific models (PSM) which are transformations of the platform-independent model with a separate PSM for each application platform. In principle, there may be layers of PSM, with each layer adding some platform-specific detail.



6b. 1. Understand and define the context and the external interactions with the system
2. Design the system Architecture
3. Identify the principal objects in the system
4. Develop design models
5. Specify interfaces

how to identify object classes ?

1. Use a gramatical analysis of a natural language description of the system to be constructed Objects and attributes are nouns : operations or services are verbs

2. Use tangible entities in the appn domain such as aircraft, roles such as manager or doctor, events such as requests, interactions such as meetings, locations such as offices, organizational units such as companies and so on.

3. Use a scenario-based analysis where various scenario of system use are identified and analyzed in turn. As each scenario is analysed, the team responsible for the analysis must identify the required objects, attributes and operations.

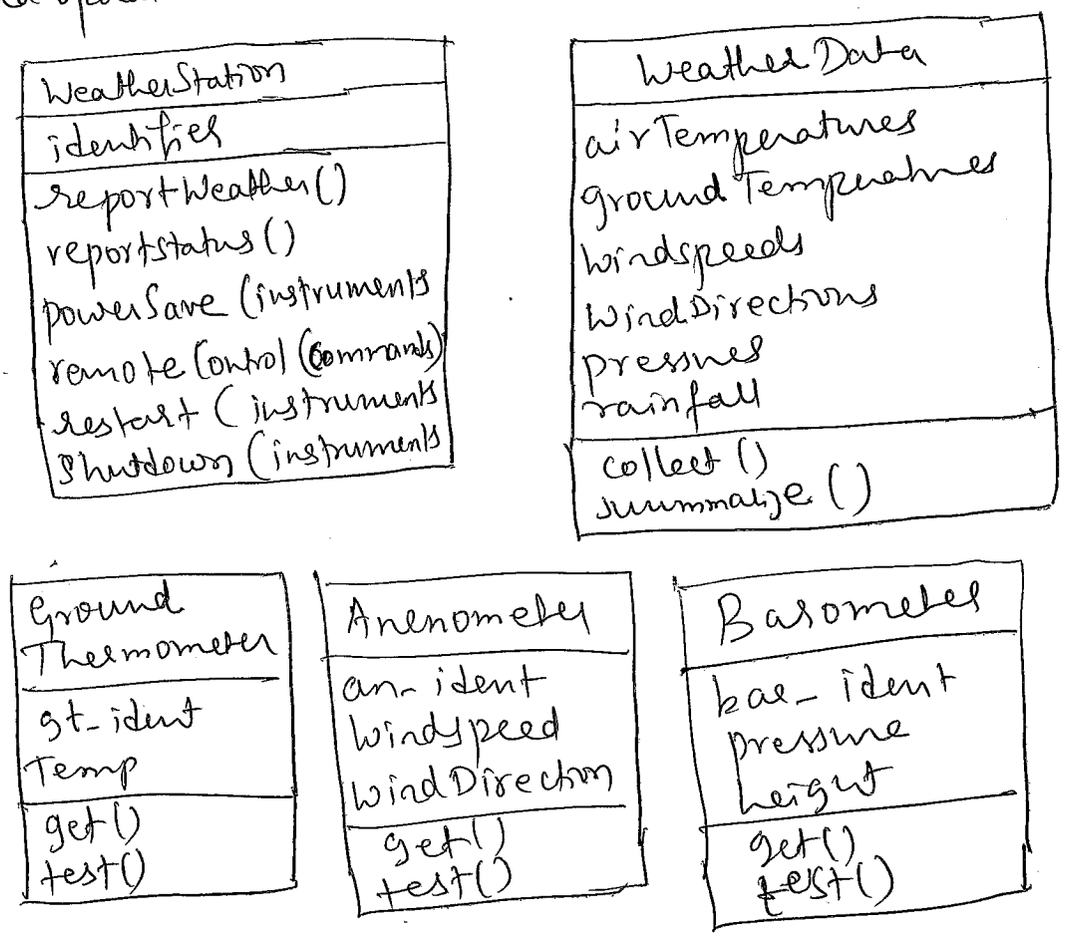


Fig:6a : Weather station objects

4m

6c. Open source development is an approach to software development in which the source code of a s/w system is published and volunteers are invited to participate in the development process.

General models of:

1. GNU General public license (GPL) - Reciprocal
2. GNU Lesser General public License (LGPL)
3. Berkley standard Distribution (BSD).

Module 4

— 8M

7a: Test Driven development

Test Driven development (TDD) is an approach to program development in which you interleave testing and code development

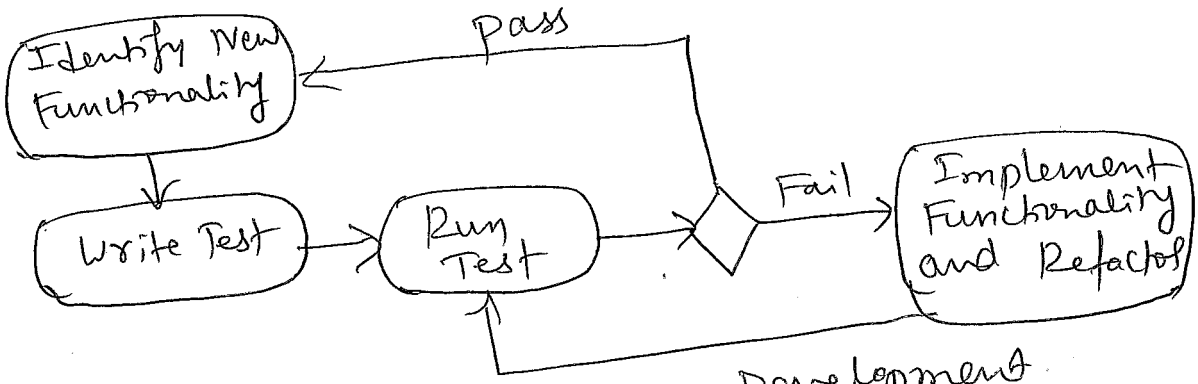


Fig 7a: Test Driven Development

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

7b Six stages of acceptance testing process — 08M

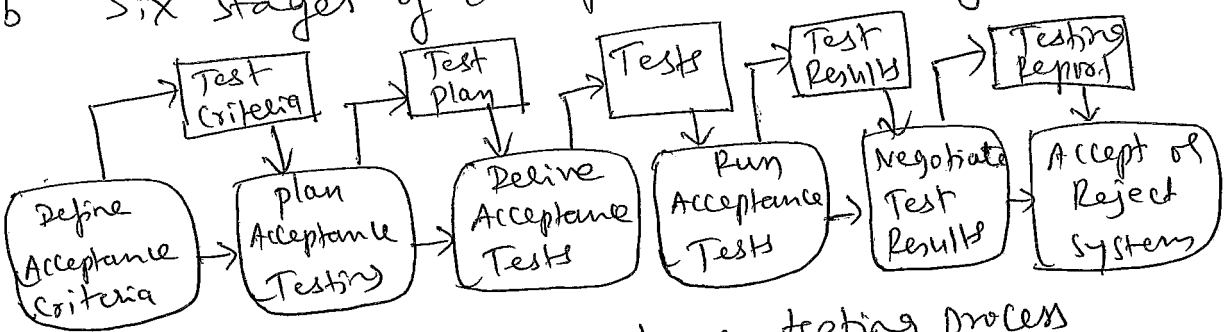


Fig: 7b: The acceptance testing process

1. Define acceptance criteria
2. Plan acceptance testing
3. Derive acceptance tests
4. Run acceptance tests
5. Negotiate test results
6. Reject/accept system

7c. Different types of interfaces to be Tested during Component testing – 4M

A! 1. Parameter interfaces :- These are interfaces in which data or sometimes function references are passed from one component to another. Methods in an object have a parameter interface.

2. Shared memory interfaces :- These are interfaces in which a block of memory is shared between components.

3. Procedural interfaces :- These are interfaces in which one component encapsulates a set of procedures that can be called by other components

4. Message passing interfaces :- These are interfaces in which one component requests a service from another component by passing a message to it.

8m

8a: Lehman's Law

- ✓ Continuing change
- ✓ Increasing Complexity
- ✓ Large program evolution
- ✓ organisational stability
- ✓ Conservation of familiarity
- ✓ Continuing growth
- ✓ Declining quality
- ✓ Feedback System

8b- Software maintenance – is the general process of changing a system after it has been delivered. – 8m

General model of re-Engineering process

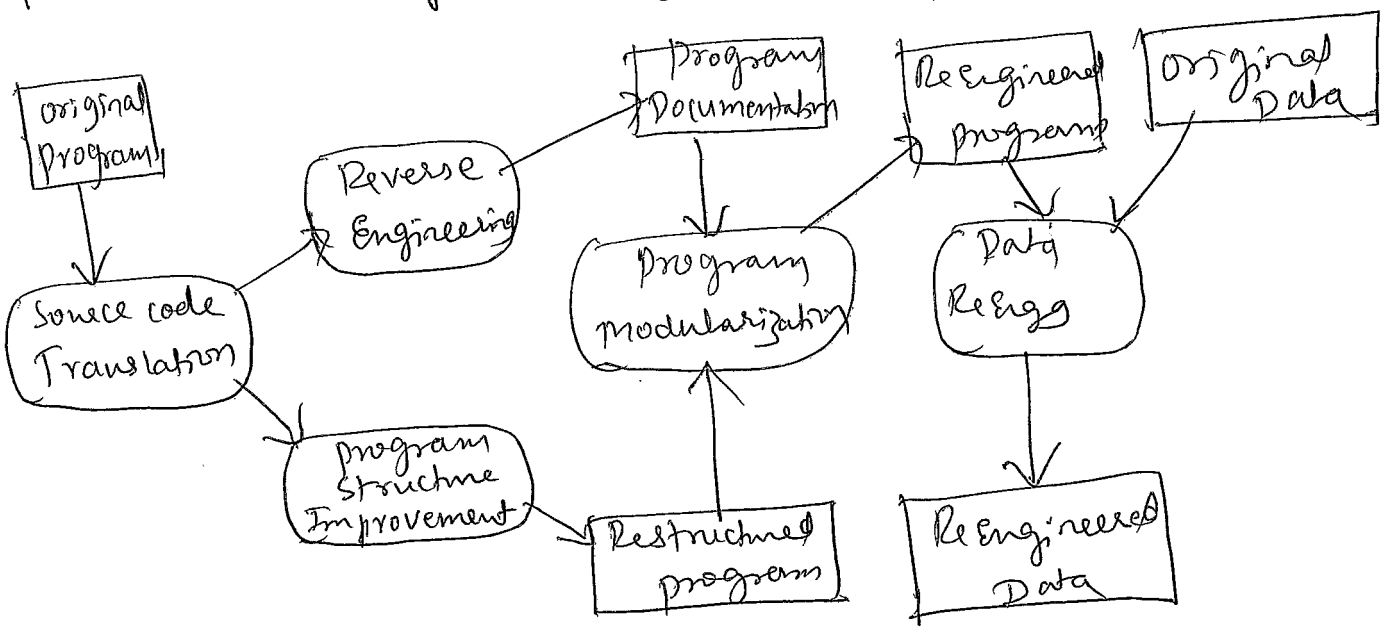
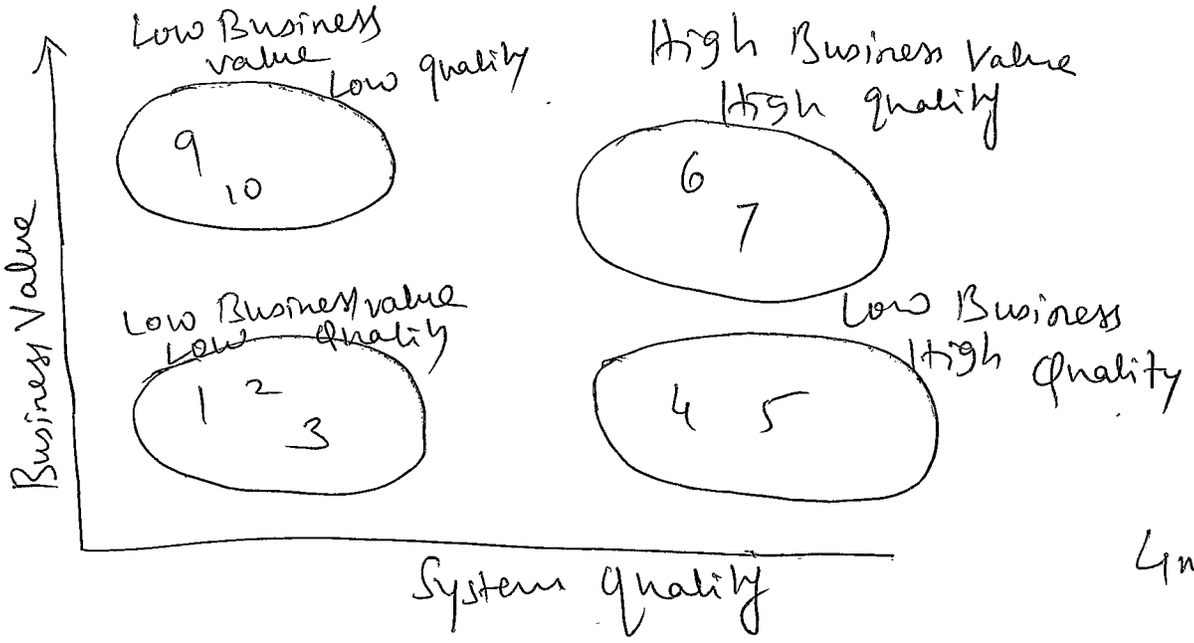


Fig 8b: reengineering process

8c:. Strategic options involved in legacy system management.

1. Scrap the system completely
2. Leave the system unchanged and continue with regular maintenance
3. Reengineer the system to improve its maintainability
4. Replace all or part of the system with a new system.



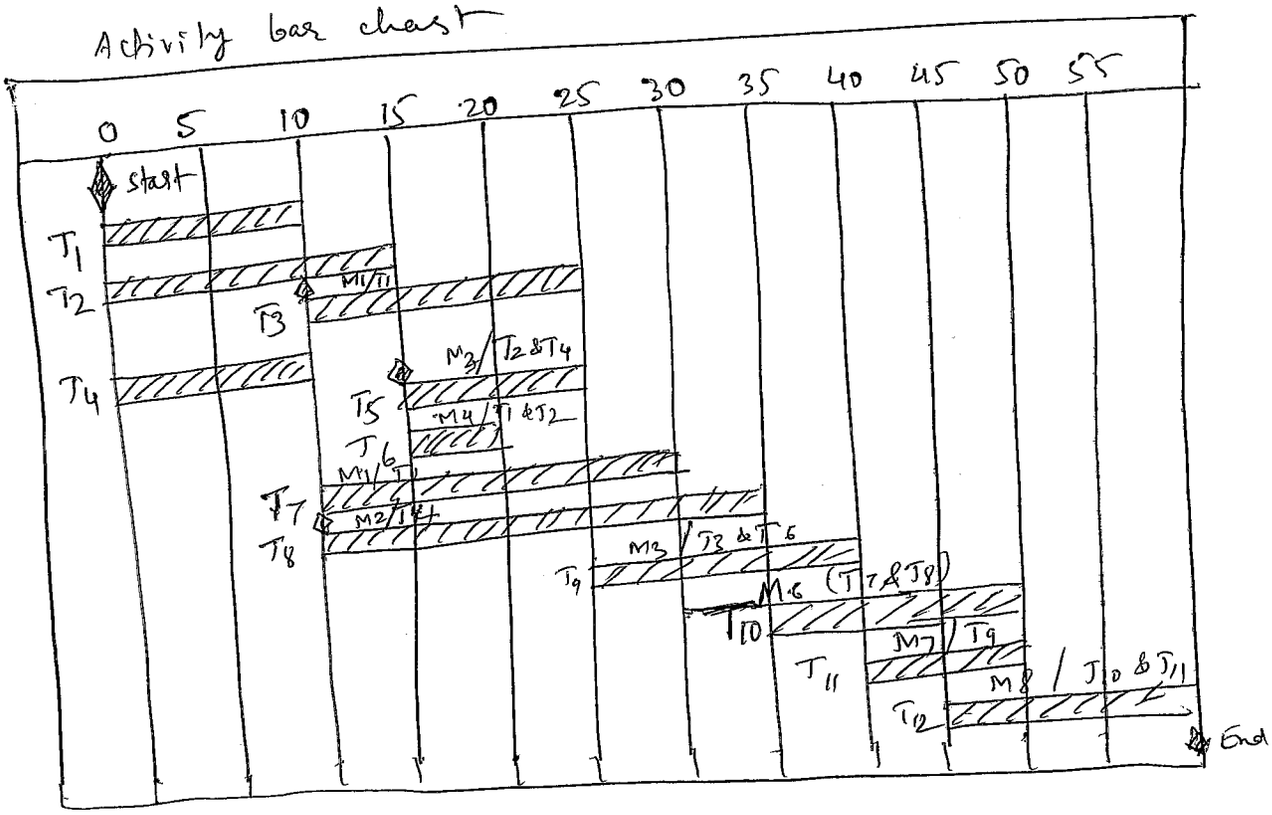
4m

Fig: 8C: An Example of a legacy system assessment

Assume that an organisation has 10 legacy systems. You should assess the quality and the business value of each of these systems. You may then create a chart showing relative business value and system quality

8m

9a. Activity bar chart



9b. factors affecting s/w pricing —

- Market opportunity : 5m
- Cost estimate uncertainty
- Contractual terms
- Requirements volatility
- Financial health

9c Algorithmic cost modeling

- This uses a mathematical formula to predict project costs based on estimates of the project size : the type of s/w being developed ; and other team, process and product factors.
- An algorithmic cost model can be built by analyzing the costs and attributes of completed projects and finding the closest - fit formula to actual experience
- Algorithmic cost models are primarily used to make estimates of s/w development costs. Algorithmic models for estimating effort in a s/w project are mostly based on a simple formula:

$$\text{Effort} = A \times \text{Size}^{B} \times M$$

A is a constant factor which depends on local organisation practices and the type of software that is developed.

- Size may be either an assessment of the code size of the software or functionality estimate expressed in function of application points.
- The value of exponent B usually lies between 1 and 1.5
- M is a multiplier made by combining process, product and development attributes such as the dependability requirements for the software and the experience of the development team.

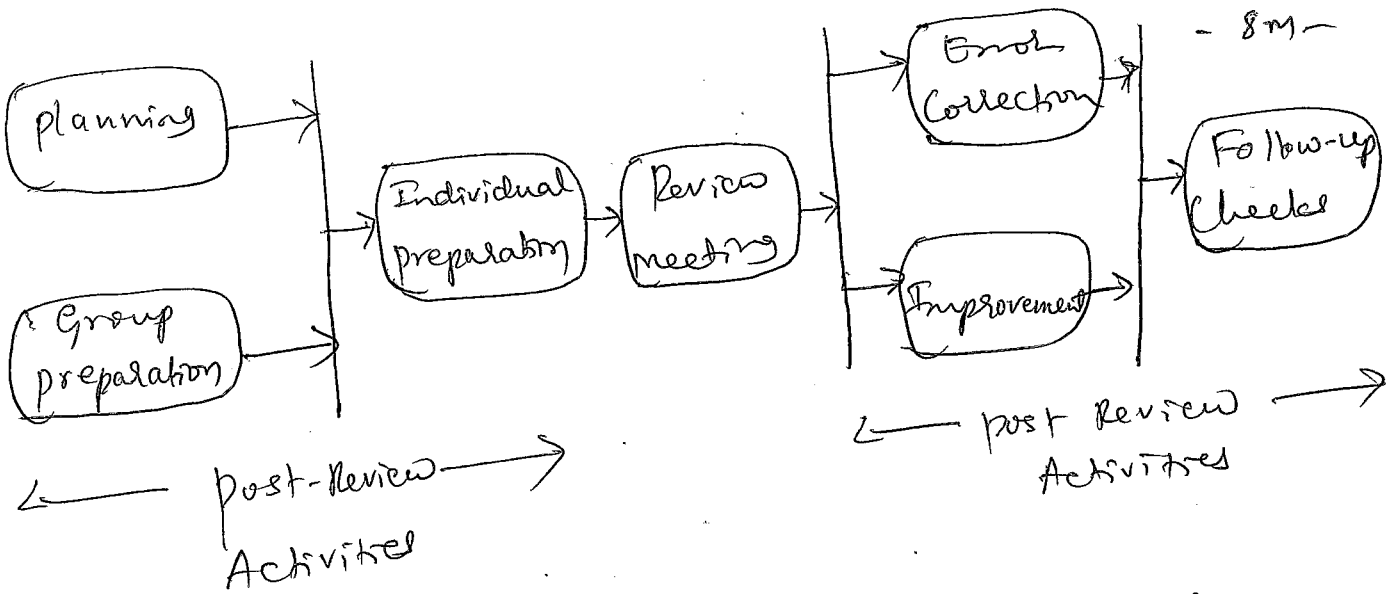
problems / difficulties

1. It is often difficult to estimate size at an early stage in a project, when only the specification is available. Function-point and application point estimates are easier to produce than estimates of code size but are still often inaccurate
2. The estimates of the factors contributing to B and M are subjective. Estimates vary from one person to another, depending on their background and experience of the type of system that is being developed.

10a. Phases involved in s/w review process

Although there are many variations in the details of reviews, the review process is normally structured into three phases.

Fig 10.a S/w Review process



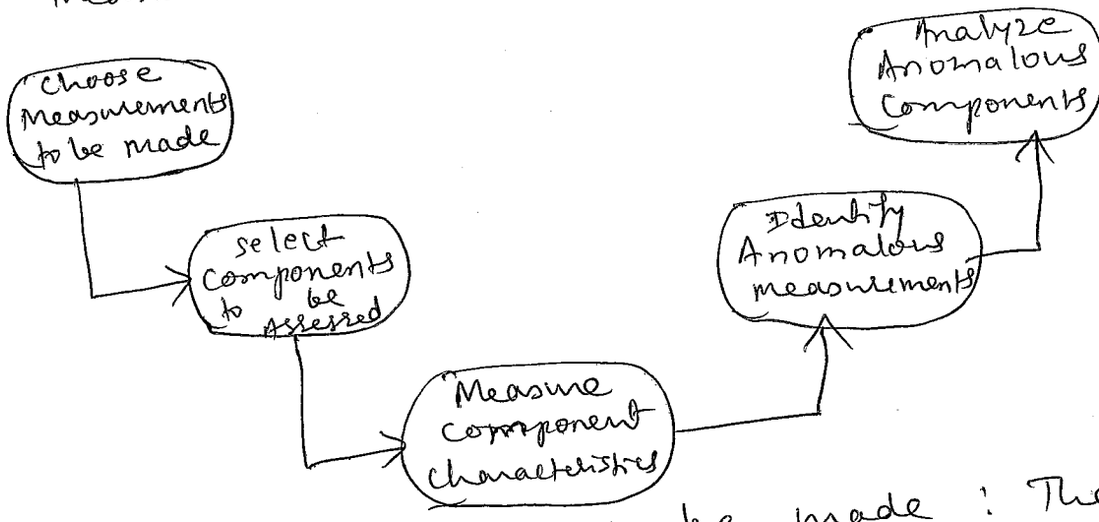
1. Pre-review Activities:- These are preparatory activities that are essential for the review to be effective. Typically, pre-review activities are concerned with review planning and review preparation.

Review planning involves setup up a review team, arranging a time and place for the review, and distributing the documents to be reviewed.

2. The review meeting:- During the review meeting, an author of the document or program being reviewed should walk through the document with review team. One team member should chair the review and another should formally record all review decisions and actions to be taken.

3. Post review activities:- After a review meeting has finished, the issues and problems raised during the review must be addressed. They may involve fixing s/w bugs, refactoring s/w so that it conforms to quality standards.

10b: Key stages in the process of product measurement -8M



1. Choose measurements to be made: The Questions that the measurement is intended to answer should be formulated and the measurements required to answer these questions defined.

2. Select Components to be assessed :- You may not need to assess metric values for all of the components in a software system. Sometimes, you can select a representative selection of components for measurement, allowing you to make an overall assessment of system quality

3. Measurement Component Characteristics :- The selected components are measured and the associated metric values computed.

4. Identify anomalous measurements: After the component measurements have been made, you then compare them with each other and to previous measurements that have been recorded in a measurement database.

5. Analyse anomalous components :- When you have identified components that have anomalous values for your chosen metrics, you should examine them to decide whether or not these anomalous metric values mean that the quality of the component is compromised.

10.c. Four product and process standards

Static product metrics
- Fan-in / Fan out
- Length of code
- Cyclomatic Complexity
- Length of identifiers

Product standards
1. Design review forms
2. Method header format
3. Java programs style
4. project plan format

Process standards
1. Design review conduct
2. Version Release process
3. change control process
4. Test recording process

Staff :

HOD
Computer Science & Engineering
KLS Vishwanathrao Deshpande
Institute of Technology, Haliyal.

Dean Academic