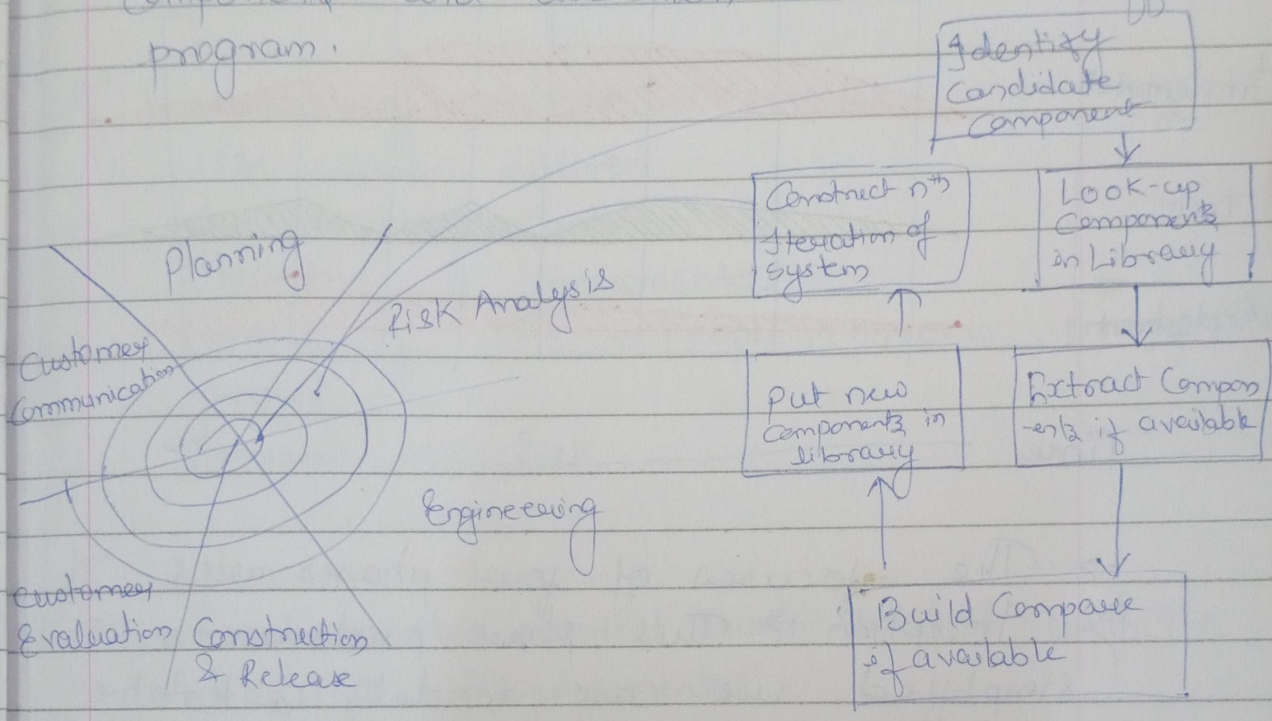
customized
← Prototype - Changes according to user involvement
Spiral - Changes according to developer without user involvement
Generic

COMPONENT ASSEMBLY MODEL →

→ It is just like prototype model, in which first a prototype is created according to the requirement of customer and sent to the user for evaluation for getting feedback.

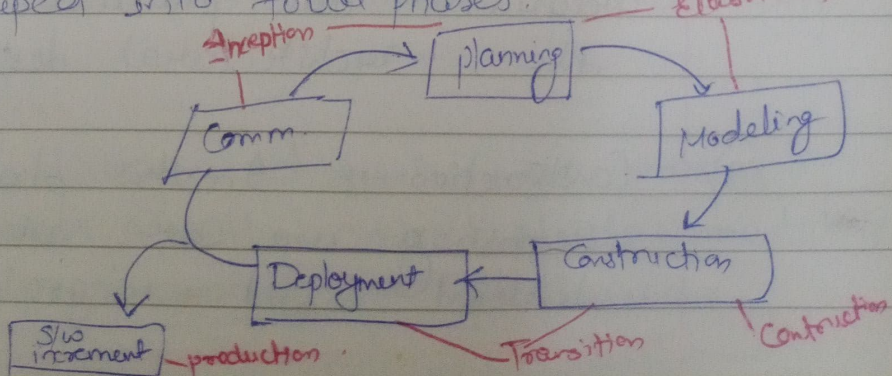
→ The developers opt for the available components and use them to make an effective program.

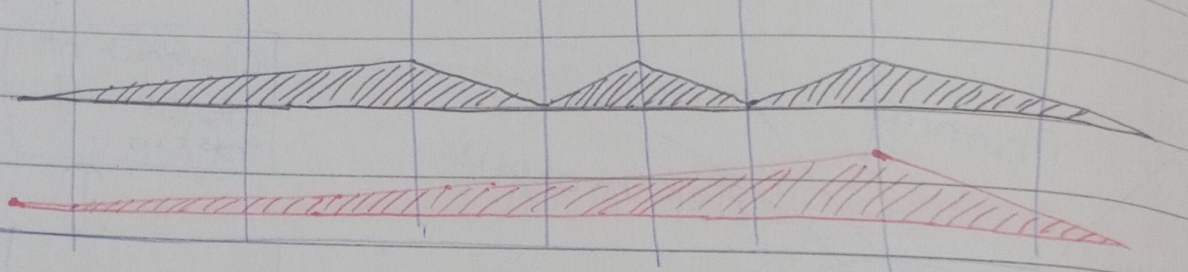


RATIONAL UNIFIED PROCESS → It consists of cycles. Each cycle concludes with a release of the system.

→ Each cycle has several iterations. The iteration are grouped into four phases:

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





→ The focuses of four phases are:

1) Inception → This phase produces a simplified use case model, a tentative software architecture and a project model plan.
eg - Use case model.

2) Elaboration → During this phase, most use cases are specified, and a UML diagrams are produced. to represents the architectural design.

3) Construction → In this phase, the remaining use cases are iteratively developed and integrated into

system.

4) Transition → During this phase, activities are performed to deploy the software system. These includes user training, beta testing, defect correction and the functional enhancement.

★ Features of RUP →
1) Use case driven
2) Architecture Centric
3) Iterative & incremental
4) Consists of 4P's - People, process, product & project.
5) Focuses on Risk.

★ Advantages →
→ Resolve the project risks related with the changing requirements.
→ Integration requires less time as it is carried out throughout the s/w development life cycle.
→ Since the components are reusable, development phase consumes less time.
→ Focuses on accurate documentation, hence can be considered as complete methodology.

★ Disadvantages →
1) Complex & disorganized development process.
2) High expensive in the context of documentation
3) Expert team members are required.

AGILE PROCESS →

→ Agile software engg combines a philosophy and a set of development guidelines.

→ Any agile software process is characterized in a through the following principles:

1) Highest priority of this process is to satisfy the customer.

2) Acceptance of changing ~~environment~~ requirement even late in development.

3) Frequently produce a working s/w in small span of time

4) Business people and developers must work together daily throughout the project.

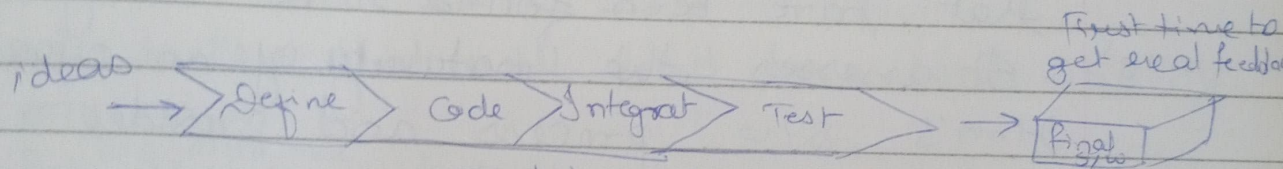
5) Build projects around motivated individuals, Give them the environment and support they need & trust them to get the job done.

6) The most efficient & effective method of conveying information to & within a development team is face to face conversation.

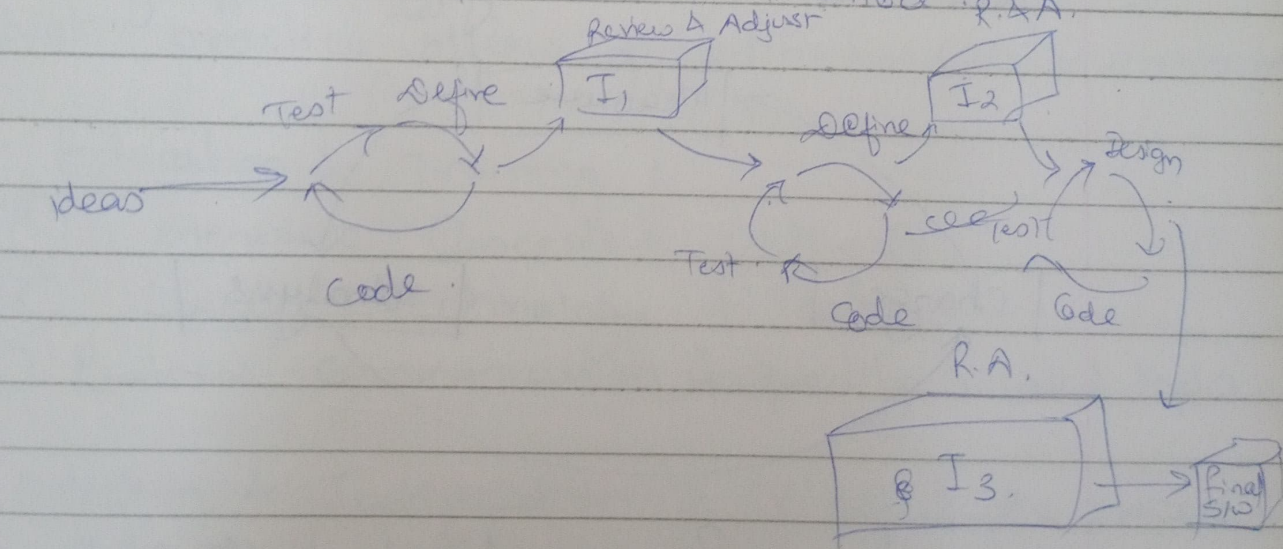
the art of work not done is essentially identified
11) The best architecture, requirements and designs emerge from self organizing teams.
12) At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly

★ Different types of agile processes →

1) Agile Modeling
2) Extreme Programming (XP)
3) Scrum
4) Agile Unified Process (AUP)
5) Kanban etc.



Traditional Method



★ SOFTWARE PROCESS CUSTOMIZATION & IMPROVEMENT →

→ The SPI (S/W process Improvement) Strategy x-forms the existing approach to S/W development into something that is

more focused, more reliable.

→ The SPI framework is as follows:

a) A set of characteristics that must be present if an effective S/W process is to be achieved.

b) A method of assessing whether those characteristics are present.

c) A mechanism for summarizing the result of any assessment.

d) A method for assisting a S/W organization in implementing those characteristics that have been found to be missing.

e) It assesses the maturity of an organization's S/W process and provides a qualitative indication of a maturity level.

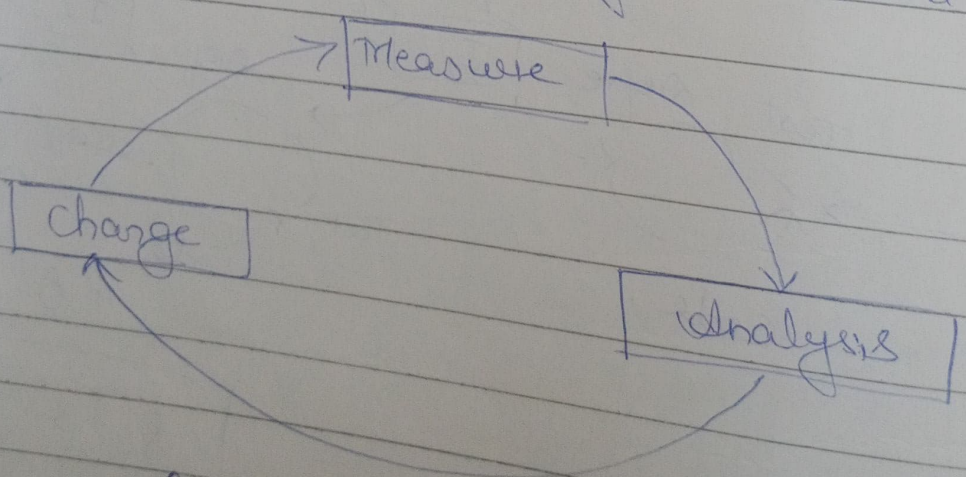


fig: Process Improvement Cycle.

☆ The five activities of SPI Process are:

1) Assessment & Difference
It examines a

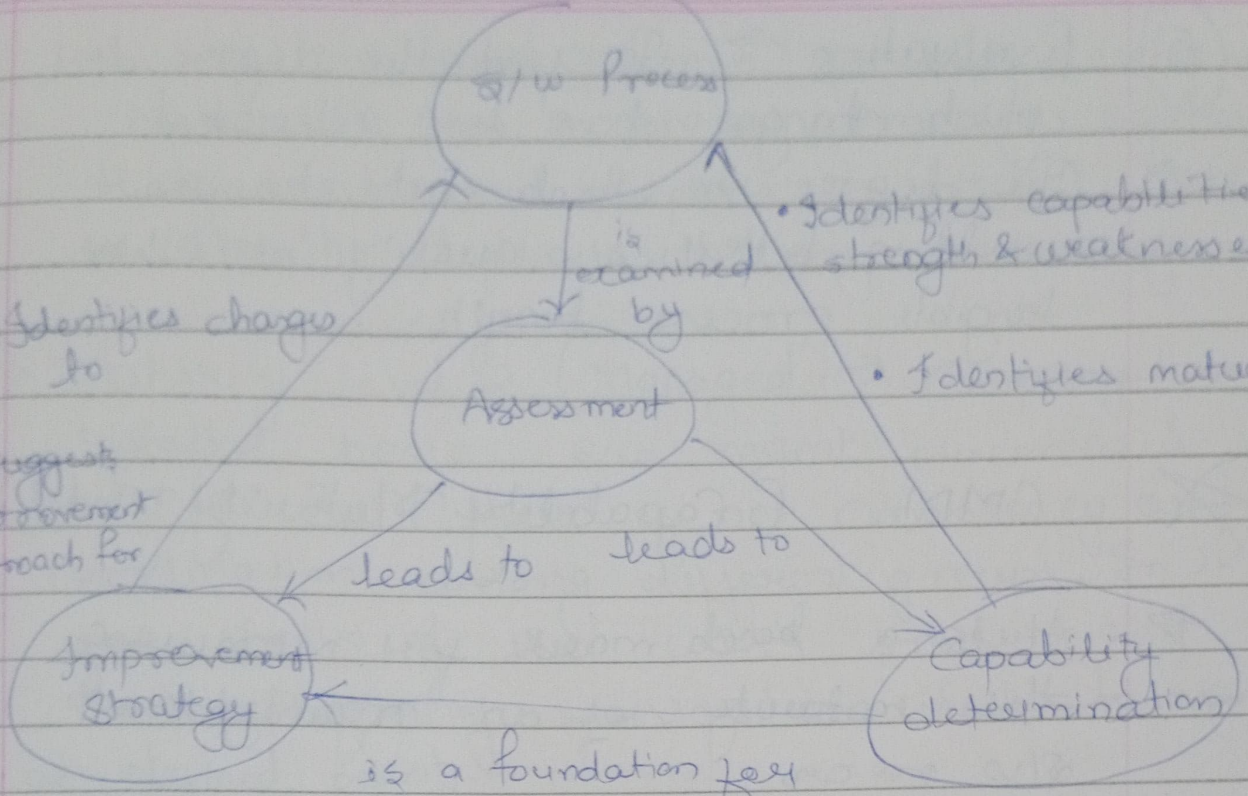


Fig → SPI framework

and tasks that will lead to a high quality process.

→ To analyse the difference between local application & best practice that offers opportunities for improvement

2) Education & Training :
(1) Generic Concepts & methods.
(2) Specific Technology & Tools.
(3) Business communication & quality related topics.

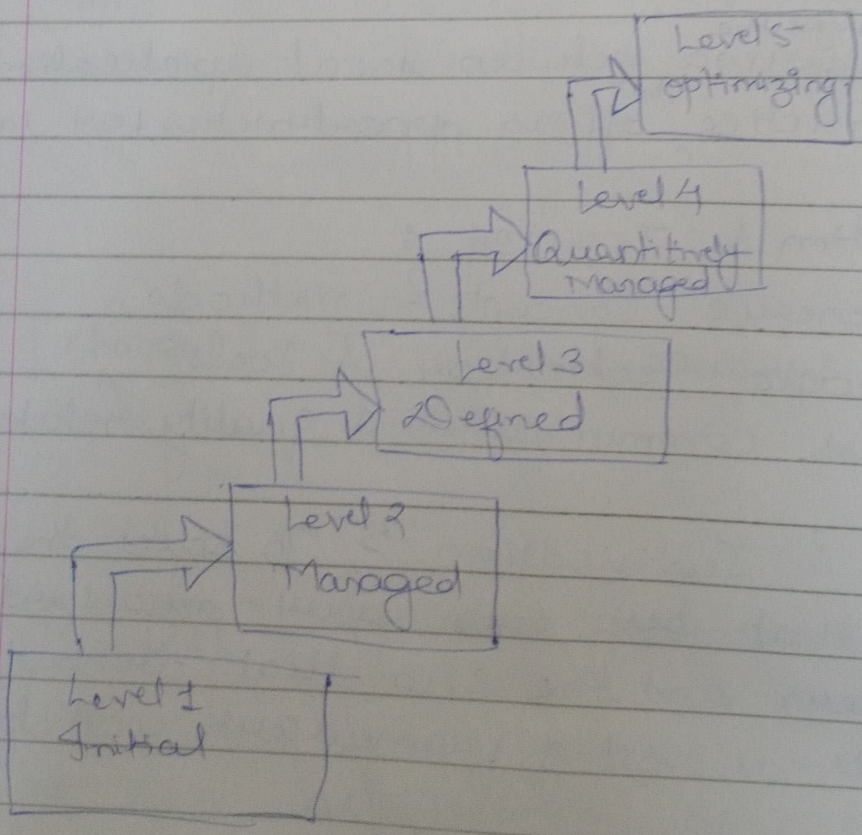
(3) Selection & Justification → To select the process models that best suits your organization, its stakeholders and the s/w that are build.
→ To decide a set of framework activity. etc.

(4) Installation / Migrat" → It is concerned with identification, application & refinement of new ways to improve and x-form s/w process.

(5) Evaluation → To find the degree to which changes have been tolerated.
→ The degree to which such changes would result in better s/w quality or other tangible process benefits.

☆ CMM (Capability Maturity Model) →

→ It is a bench mark for measuring the maturity of an organization's s/w process.
→ CMM can be used to assess an organization against a scale of five process maturity levels based on certain Key Process Area.



1) Maturity Level 1 (Initial) → Company

has no standard process for s/w development.
→ Now it has a project tracking system that enables developers to predict costs or finish dates with any accuracy.

2) Maturity Level 2 (Managed) → Company has installed basic s/w mgmt processes & controls. But there is not consistency or coordination among different groups. In it the status of the work product and the delivery of services are visible to mgmt at defined points.

3) Maturity Level 3 (Defined) → Company has applied a standard set of processes and controls for the entire organization so that developers can move b/w projects more easily and customers can begin to get consistency from different groups

4) Maturity Level 4 (Quantitatively Managed) → In

technological improv...

→ Company has accomplished all of the above & can now begin to see patterns in performance over time, so as to improved deployment and reduce defects in s/w development across the entire organization.

★ PROCESS & PRODUCT METRICS →

→ S/w metrics refers to a broad range of measurements for computer software.

→ Three main terms are used in S/w engg:

(1) measure (2) Metric (3) Indicators.

1) Measure → It provides a qualitative indication of the extent, amount, dimension, capacity or size of some attributes of product or process.

2) Metrics → It is a quantitative measure of a degree to which a system,

★ e.g of

(i) S

pla

pl

(ii) Eff

(vii)

★ e.g

(i)

(ii)

(iii),

if value is -ve that means we are behind schedule

☆ e.g of Project Metrics.

(i) Schedule variance = ((Actual Calendar days - planned Calendar days + start variance)/ planned Calender days) x 100.

(ii) Effort variance = (Actual Effort - Planned Effort)/planned Effort) x 100.

(iii) Productivity (defect fixation) = actual no. of defects fixed / actual effort spent on defect fixing

☆ e.g. of Process Metrics →

(i) Cost of Quality = [(review + testing + verification review + verificat" testing + QA + configurat" mgmt + measurement + training + rework review + rework testing)/Total effort] x 100.

(ii) cost of poor quality = (rework effort / Total effort) x 100

(iii) Review Efficiency = (no. of defects caught in review / total no. of defects caught) x 100.

☆ FUNCTION ~~POINT~~ oriented METRICS →.

Function point is a unit of measurement to express the amount of business functionality an informat" system provides to a user.

Measurement Parameter	Count		Weighing factors: simple	Avg	Complex		Sub Total
No. of User Inputs		x	3	4	6	=	
No. of User O/P		x	4	5	7	=	
No. of User inquiries		x	3	4	6	=	
No. of files		x	7	10	15	=	
No. of external interfaces		x	5	7	10	=	

Count Total →

→ The weighing factors mentioned above depends upon the fact, whether that particular entry is simple, average, or complex.

FP is calculated as:

FP = Count Total × (0.65 + 0.01 × Σ(fi))

(value added factor (VAF))

where fi (i = 1 to 14)

Σfi is a complexity adjustment factor.

→ 14 questions answered on a scale of 0 to 5

0 → no influence
1 → Incidental
2 → Moderate
3 → Average
4 → Significant
5 → Essential.

→ The following basic questions should be clearly understood by the analyst :

i) What is the problem?

ii) Why is it important to solve the problem

iii) What exactly are the data i/p to the system & what are the data o/p by the system?

iv) What are the possible procedures to solve the problem?

v) What are the complexities that might arise while solving the problem?

vi) If there are external s/w & h/w with which the developed s/w has to ~~be~~ interface, then what will be the data interchange formats?

1) FUNCTIONAL REQUIREMENT → It discuss the functionalities required by the users from the system.

→ The functional requirement of the system, should clearly describe each function which the system would support along with the corresponding i/p & o/p data set.

e.g → Business Rules, x-sactn correction, administrative functn, authentication, ~~audit~~ ~~record~~ tracking, historical data, Reporting requirement etc

2) NON-FUNCTIONAL REQUIREMENT → It deals with the characteristics of a system that cannot be expressed as function.

→ for eg. maintainability, portability, usability, maximum no. of concurrent users, timing & throughput (x-sactn per second) etc.

Difference b/w functional & non-functional requirements:

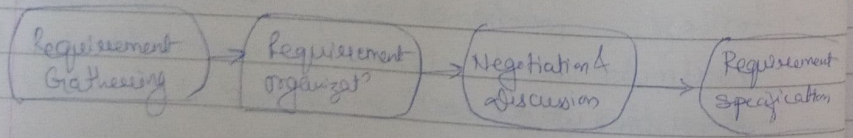
Functional Req.	Non functional req.
1) It is specified by user.	1) It is specified by technical people : e.g architect, developers etc.
2) It is the activity that system must perform.	2) It depends on the criticality or complexity of application.
3) It describes the behaviour of the system as it relates to the system's functionality	3) It defines the constraints, targets or control mechanism for the new system

☆ REQUIREMENT ELICITATION →

→ It is an active effort to extract information from stakeholders and subject matter experts.

→ Elicitation is not a step or a task we do at a point. It is a set of techniques we apply during the requirement phase.

→ Requirement gathering is more reactive, less proactive whereas requirement elicitation is more proactive and less reactive.



1) Requirement Gathering → The developers discuss with the client & end users & know their expectations from the S/w.

★ proactive – creating or controlling a situation than just responding.

★ reactive – showing a response rather than controlling it.

★ FP are used to compute Functional size measurement (FSM)

2) Organizing Requirement → The developer prioritize and arrange the requirements in order of importance, urgency & convenience.

3) Negotiation & Discussion → If the requirements are ambiguous or there are some conflicts in requirement of various stakeholders, then it can be negotiated & discussed with the stakeholders.

3) Documentation → All formal & informal, functional & non-functional requirements are documented & made available for next phase processing.

☆ REQUIREMENT ELICITATION TECHNIQUE →

1) Interviews → Interviews are strong medium to collect requirements. Organization may conduct several types of interviews such as

a) Structured Interviews → where every single information to gather is decided in advance, they follow pattern & matter of discussion firmly.

b) Non-structured Interviews → where information to gather is not decided in advance, more flexible & less biased.

c) Oral interview d) Written interviews e) one-to-one interviews

f) Group interview etc.

2) Surveys → Organization may conduct surveys among various stakeholders by querying about their expectat° & requirements from the upcoming system

3) Questionnaires → A document pre-defined set of activities objective questions and respective options is handed over to all stakeholders to answer, which are collected & compiled.

4) Task analysis → Team of engineers and developers may analyze the operat" for which the new system is required. If the client already has some s/w to perform certain operation, it is studied and requirement of proposed system are collected.

5) Domain Analysis →

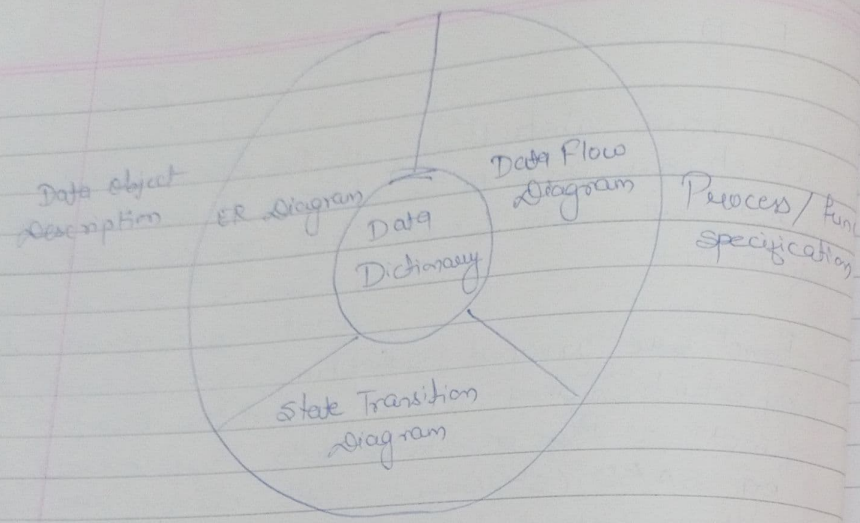
6) Brainstorming → An informal debate is held among various stakeholders and all their i/p are recorded for further requirements analysis.

ANALYSIS MODEL →

→ The intent of analysis model is to provide a description of the required informational, functional and behavioural domain for a computer based system.

7. Objectives of analysis model are:
1) To describe what the customer requires.
2) To establish a basis for the creation of a software design
3) To define a set of valid requirement after which the s/w can be built.

FOD	OOD
1) The basic abstract", which are given to the user, are real world functions.	1) The basic abstract" are not real world functions but are the data abstract" where the real world entities are represented.
2) Funct" are grouped together by which higher level funct" is obtained eg. SA/SD. (Structured analysis / Structured Design)	2) Funct" are grouped together on the basis of the data they operate since the classes are associated with their methods.
3) In this approach the state informat" is represented in centralized shared memory.	3) In it state informat" is not represented in centralized shared memory but is implemented & distributed among the objects of the system.
4) Used for Computat" sensitive applicat"	4) Used for evolving system which mimicks a business process.
5) It is decomposed in funct" / procedure level.	5) It is decomposed in class level.
6) It is Top down approach	6) It is a bottom up approach
7) It begins by considering the use case diagrams and scenarios.	7) Begins by identifying objects and classes.



☆ Analysis Modeling approaches :

1) Structured Approach : Analysis is made on data & process in which data is x-formed as separate entities.

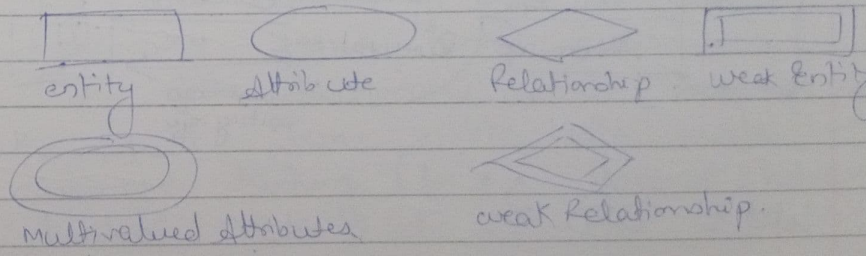
2) Object oriented approach : Analysis is made on the classes and interaction among them in order to meet the customer requirements.

Analysis & modeling

- Structured Approach
  - Data Modelling
    - E-R Diagram
  - Functional Modeling
    - DFD (Data Flow Diagram)
    - Data Dictionary
  - Behavioural Modelling
    - State chart diagram
- Object Oriented Approach
  - Unified Modeling language (UML)
    - Use case diagram
    - Class diagram
    - Activity diagram
    - Sequence diagram
    - Deployment diagram
    - Component diagram
    - State diagram.

1) Data Model → A data model is a collection of conceptual tools for describing data, data relationship, data semantics and consistency constraints. It is conceptual representation of data structures required for database and to communicate the business requirements.

(1) E-R Diagram : Tools : Smartdraw, Edraw



2) FUNCTIONAL MODEL → It gives the process perspective of the object oriented analysis model and an overview of what the system is supposed to do.

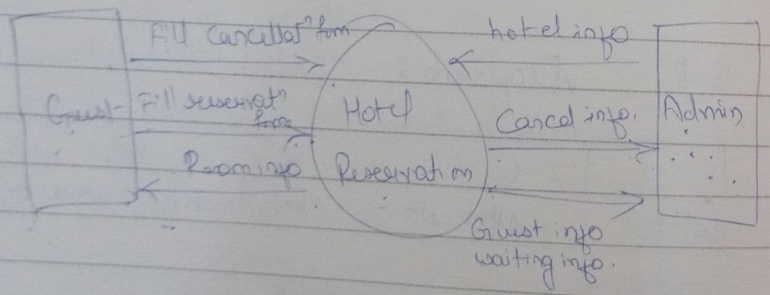
→ It depicts the functional derivat of the d values without indicating how they are derived when they are computed, or a they need to be computed.

(a) DFD (Data Flow diagram) → It is a graphical representation of a system that shows the i/p to the system, the processing upon the i/ps, the o/p of system as well as internal data stores. four main parts of DFD are:

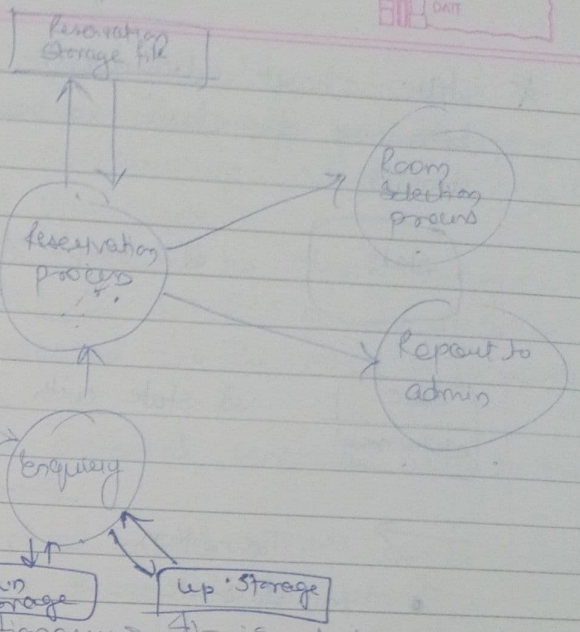
1. Process
2. Data Flow
3. Actors
4. Data stores.

other parts are:
5. Constraints
6. Control flow.

Level 0 DFD →



level 1 DFD →.



(b) Data Dictionary → It is dictionary about the data we store in database.

→ When developing a program that use the data model a data dictionary can be consulted to understand where a data item fits in the structure, what values it may contain, & basically what the data item means in real-world terms.

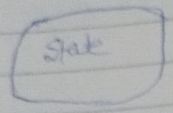
→ For eg., a bank or group of bank could model the data object involved in consumer banking for bank's programmers.

→ They data dictionary could then provide a data dictionary for a bank's programmers.

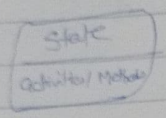
→ The data dictionary would describe each of the data item in its data model for consumer banking (for e.g. Account holder & Available credit") etc

(c) BEHAVIOURAL MODEL → It describes the overall behaviour of a system represented by state chart diagram.

A state chart diagram → It shows the behaviour of classes in response to external stimuli.



→ a simple state.

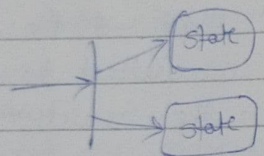


→ a state with internal activities.

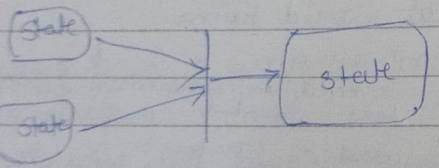
→ Transition

• initial state.

◉ final state

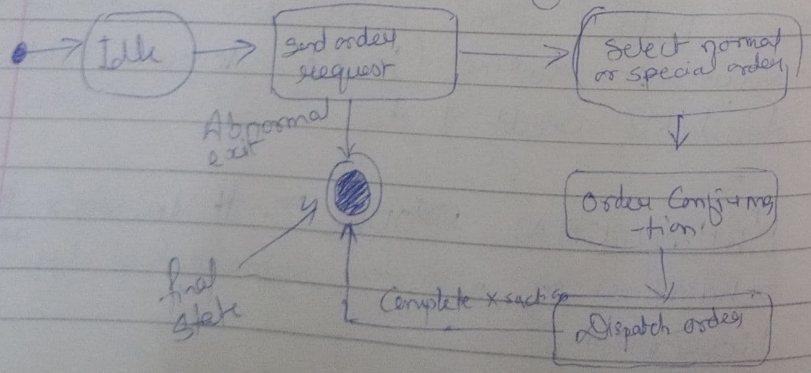


Fork



Join

State Chart diagram for order management.



☆ UML (Unified Modeling Language) →

→ It is a standard language for specifying, visualizing, constructing and documenting the artifacts of S/w system.

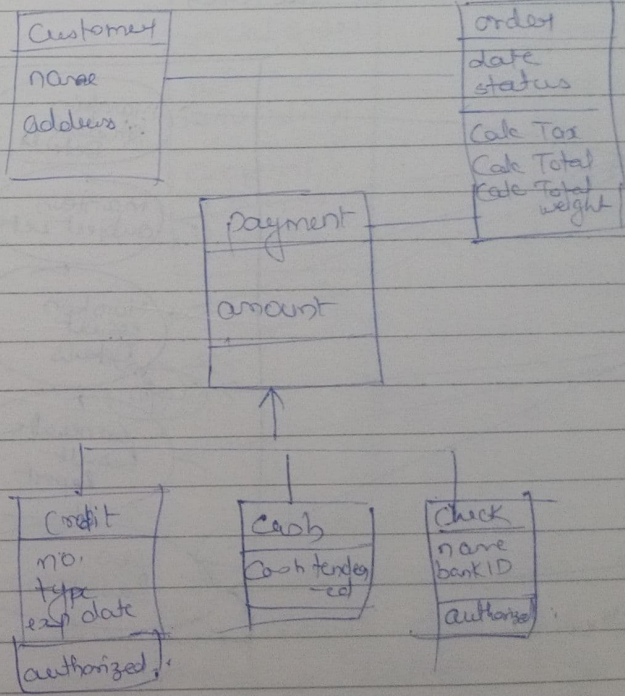
→ UML was created by Object Management Group (OMG) & UML 1.0 specification draft was proposed to the OMG in January 1997.

→ UML is a pictorial language used to make S/w blueprints.

→ The conceptual model of UML can be mastered by learning three major elements:

1) UML building block
2) Rules to connect the building blocks.
3) Common mechanisms of UML.

e.g.



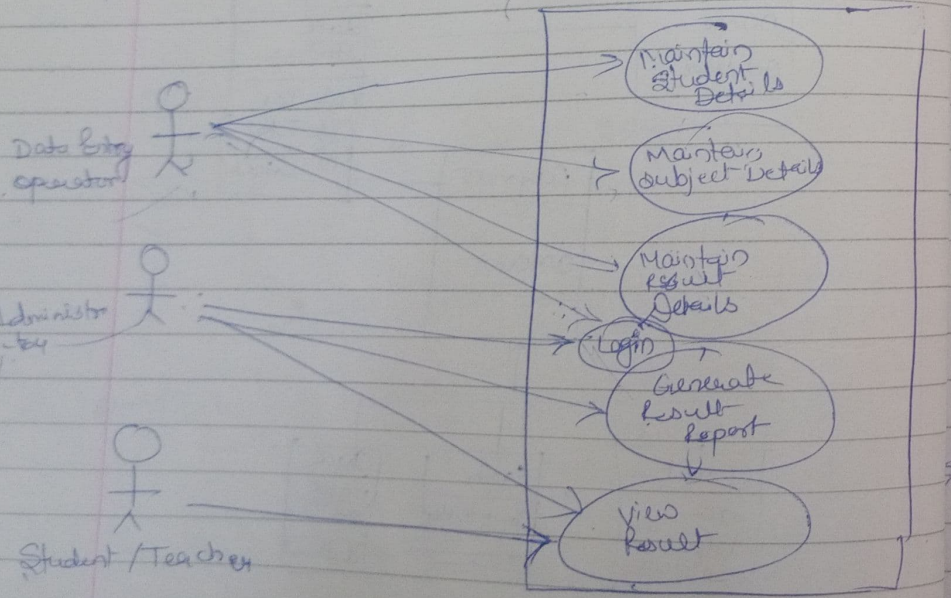
USE CASE DIAGRAM

→ In UML, a use case diagram is pictorial representation of system's user (also known as actor) and their interact with the system.

→ Use cases are a set of action, serv and func° that the system needs to perform.

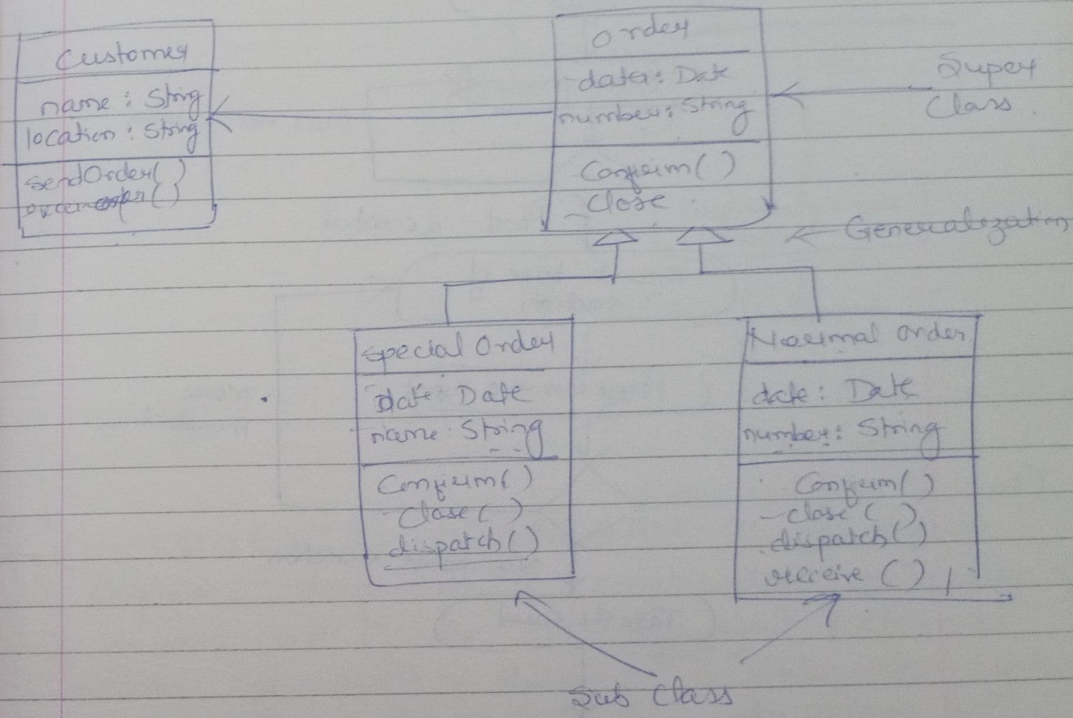
→ The guidelines to draw an efficient use case diagram are:

1) The name of use cases is very imp. The name should be selected so as to define proper functionality.
2) Identify all users.
3) Structure the use case.
4) Review & validate the user.



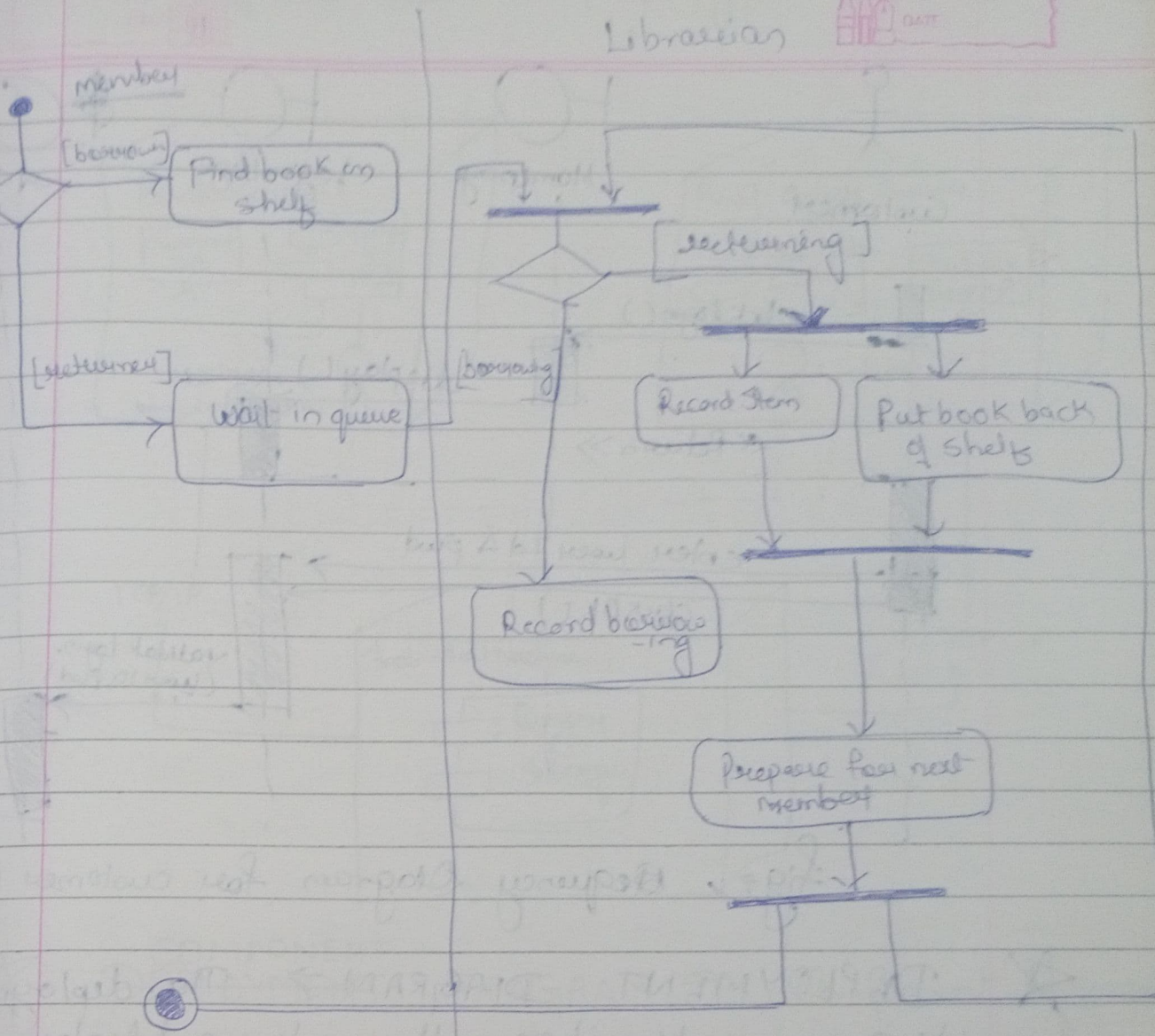
CLASS DIAGRAM →

→ It is a static diagram.
→ It represents the static view of an application
→ It describes the attributes & operations of a class & also the constraints imposed on the system.



ACTIVITY DIAGRAM →

→ It represents the dynamics of the system.
→ They are flow charts that are used to show the workflow of a system.
→ Core symbol is an activity 



SEQUENCE DIAGRAM → (These are time focused).

A system sequence diagram is a particular scenario of a use case, the events that external actors generates, & their possible inter-system events. This is also an interact[n] diagram.

They are best suited for analysis activities.

They are visual summaries of the individu use cases.

A system sequence diagram should specify the following:

1) External Actors 2) Messages (methods) invoked by these actors 3) Return value associate with previous msg 4) Indicat[n] of any loop area

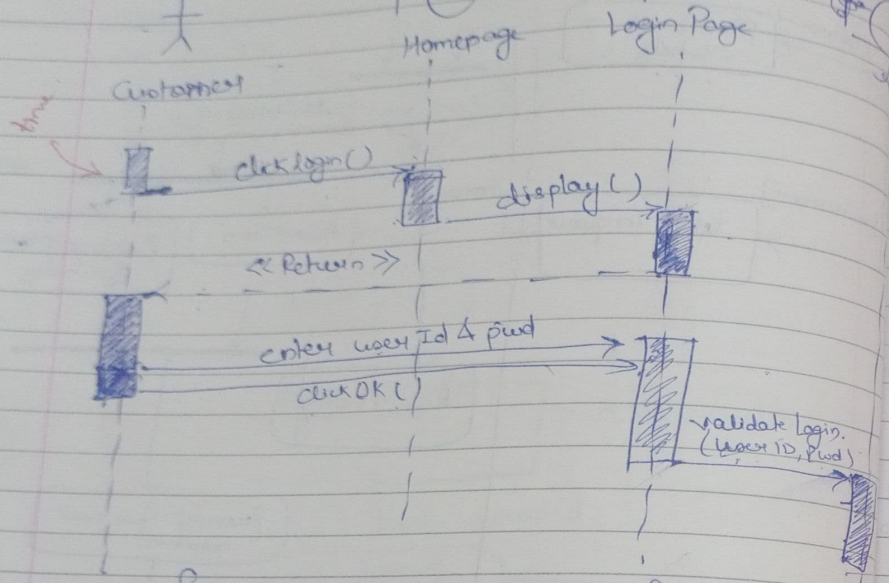


fig → Sequency Diagram for customer

☆ DEPLOYMENT DIAGRAM → The deploym diagram describes the physical deployme of informat" generated by the s/w program on h/w Components.

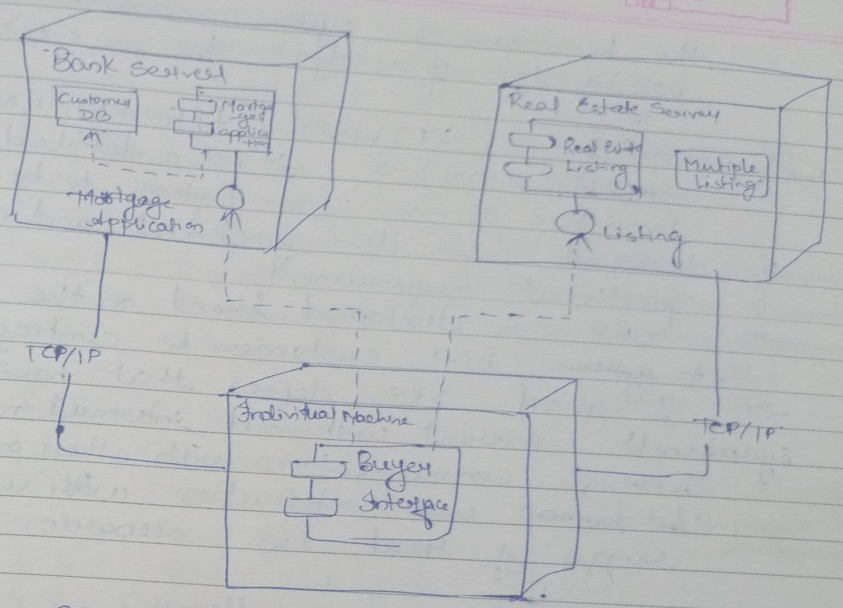
→ The three dimensional boxes, known as nodes, represent the basic s/w or h/w elements or nodes in the system.

→ Lines from node to node indicate relationship.

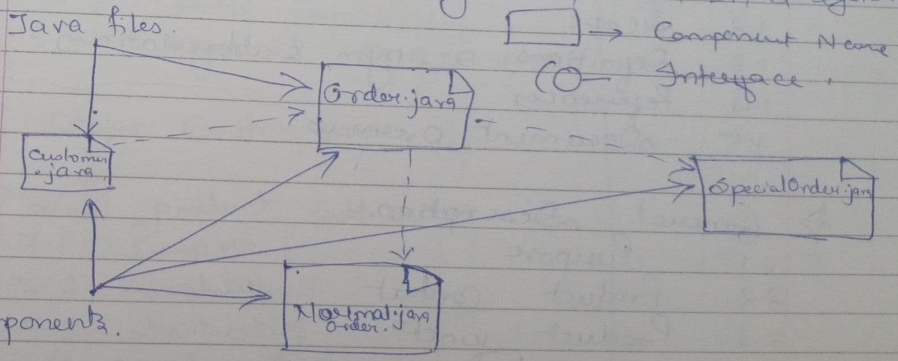
→ The purpose of this diagram are:

1. visualizat" of h/w topology.
2. To describe the h/w Components.
3. To describe the runtime processing nodes.

of the debtor's property.



☆ COMPONENT DEPLOYMENT DIAGRAM → Component diagram are used to visualize the organization & relationships among components in a system.



☆ STATE DIAGRAM → It is used to describe

the behaviour of system ... possible states of an object when an event o...

SRS :- (SOFTWARE REQUIREMENT SPECIFICATION) → It is a detailed description of a s/w system to be developed with its functional and non-functional requirements.

→ SRS is developed based on the agr... b/w customer & contractor.

→ A good SRS defines that how the s/w system will interact with all internal modu... h/w, communication with other progra... & human user interaction with wide range of real life scenario.

→ SRS template is as follows:

1) Introduction
- 1.1 Purpose
- 1.2 Scope
- 1.3 Definitions, acronym & abbreviations
- 1.4 References
- 1.5 Document overview.

2) General Description.
- 2.1 Purpose
- 2.2 Product Context
- 2.3 Product funct°
- 2.4 User characteristics
- 2.5 General Constraints.
- 2.6 Assumptions & dependencies.
- 2.7 Apportioning of requirements.

3) Product funct°
- 3.1 Purpose
- 3.2 Funct° Description
  - 3.2.1 Introduction
  - 3.2.2 Inputs
  - 3.2.3 Processing
  - 3.2.4 outputs

4) External Interfaces.
- 4.1 Purpose
- 4.2 User Interfaces
- 4.3 H/w Interfaces
- 4.4 S/w Interfaces
- 4.5 Communicat° interfaces

5) Performance requirement
- 5.1 Purpose
- 5.2 Static Capacity.
- 5.3 Dynamic performance

6) Design Constraints.

7) S/w product attributes.
- 7.1 Purpose
- 7.2 Reliability
- 7.3 Availability
- 7.4 Maintainability
- 7.5 Security
- 7.6 Portability.
- 7.7 Safety.

8) Database requirement
9) Installation requirement
10) Operational requirement
11) Documentation requirements
12) Appendices
- Appendix A - Glossary
- Appendix B - Data dictionary

stakeholders should be aligned with what the system needs to perform.

2) Consistency checks: Requirements in the document should not conflict or different description of the same function.

3) ~~Realism~~ Completeness check: The document should include all the requirements & constraints.

i) Realism check: Ensure the req. can actually be implemented using the knowledge of existing budget, scheduling, technology etc.

7 validat^n is performed by involving relevant stakeholders, other requirement sources (standards, Laws, etc) as well as external reviewers, if necessary.

Requirement Validation Techniques:
1) Test case generation.
2) Automated consistency analysis
3) Prototyping.

TRACEABILITY :→ It is a property of an element of documentation or code that indicates the degree to which it can be traced to its origin.

☆ Significance of Traceability →

1) To identify the ~~to~~ version of test case to be used.
2) To identify which test cases need to be updated.
3) To assist the debugging process so that a defect found when executing tests can be tracked back to the corresponding version of requirement.

UNIT- III
SOFTWARE DESIGN.

☆ SOFTWARE DESIGN PROCESS →.
→ The main aim of design engg is to generate a model which represents fairness and commodity.
→ S/w design is an iterative process through which requirements are translated into the blueprint for building the software.
→ The attributes of design are:
1) Functionality (2) Usability (3) Reliability
(4) Performance (5) Supportability.

☆ DESIGN CONCEPTS →. The set of fundamental design concepts are as follows:
1) Abstraction → A sequence of instructions that contain a specific & limited function to a procedural abstraction.
→ A collection of data that describes a data object is a data abstraction.

2) Architecture → The architecture of is the structure of program modules where they interact with each other in a specialized way.
→ The aim of the s/w design is to obtain an architectural framework of a system.

3) Patterns → It describes a design structure that structure solves a particular design problem in a specified content.

4) Modularity → A s/w is separately divided into name & addressable components. Sometimes they are called as modules which integrate to satisfy the problem requirement.
→ Modularity is the single attribute of a s/w that permits a problem to be managed easily.

5) Information Hiding →. Modules must be specified & designed so that the information like algorithm & data presented in a module is not accessible for other modules not requiring that information.

6) Functional Independence →. It is the concept of separation & related to the concept of

→ The functional independence is accessed by using two criteria i.e cohesion & cou

Cohesion → A cohesion module performs single task & it requires a small interac with other components in other parts of the program.

Coupling → It is an indicat" of interconn b/w modules in a structured s/w.

7) Refinement → It is a top-down desig approach and a process of elaborat".
→ A hierarchy is established by decompos a statement of funct" in a stepwise manner till the programming languag statement are reached.

8) Refactoring → It is a reorganizat" techniqu which simplifies the design of component without changing the s/w system i its functional behaviour.

9) Design classes → The model of s/w is defined as a set of design classes.
→ Every class describes the elements of problem domain and that focus on features of the problem which are visible.

Assignment - II

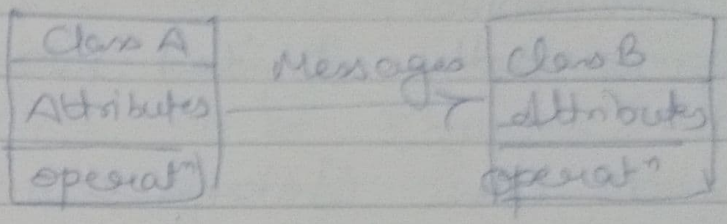
Date : 26/03/18

Q1 Define functional & Non-functional requirements
Q2 What is SRS ? Also describe its template?
Q3. Explain the analysis modeling for function oriented development?
Q4. Explain traceability ? Also describe its significance ?

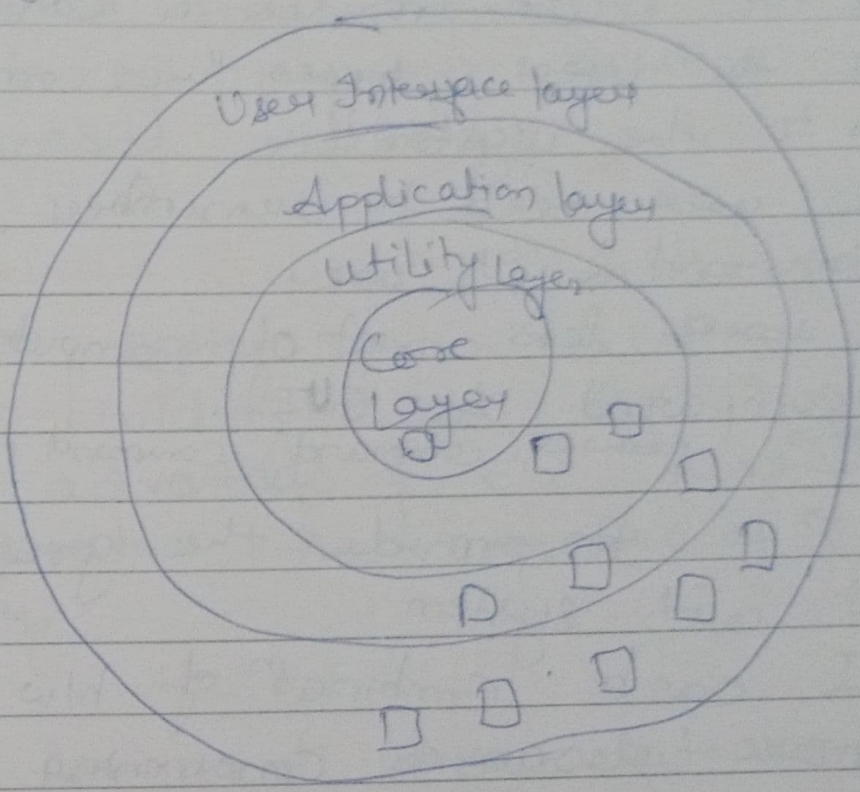
★ SOFTWARE DESIGN PRINCIPLES → (Same as concepts)
1) ~~Modularization~~

★ SOFTWARE MODELING → A s/w modelin is a computer program to build simulat or other models.
→ It is an artificial language that can be used to express informat" or knowledge or systems in a structure that is defined b a consistent set of rules.
→ The rules are used for interpretation the meaning of component in the structu
→ Modelling helps the analyst to under the functionality of the system & models are used to communicate with custom

★ ARCHITECTURAL DESIGN → The s/w architecture of a program or computing sy is the structure or structure of s/w which comprise the s/w components the properties of those components, and the relationship among the Compone



A) Layered Architecture →



★ ARCHITECTURAL VIEW →

1) Logical View (2) Process View
3) Development View (4) Physical View.

★ USER INTERFACE DESIGN → It is a front-end applicat" view to which user interacts in order to use the S/w. User can manipulate & control the S/w as well as h/w by means of user interface

of requirement specification into implementation.

→ The s/w analysis & design is the intermediate stage, which helps human-readable requirements to be transformed into actual code.

→ Component oriented s/w design has many advantages over the traditional object-oriented approaches such as:

1) Reduced time in market and the development cost by reusing existing components.
2) Increased reliability.

Software Reuse → It is the process of creating a s/w system from existing s/w rather than building s/w system from scratch.

→ The advantage of s/w reuse ~~pro~~ to facilitates the increase of productivity, quality

→ A component encapsulates functionality & ~~Component~~ behaviour of a s/w element into a reusable & self-deployment binary unit.

→ The characteristics of components are:

1) Reusability.
2) Replaceable
3) Not context specific
4) Extensible.
5) Encapsulated
6) Independent.

★ DESIGN METRICS → It is used to identify design metrics that allow the s/w engineer

nodes.

3) Connect nodes as follows:

1) If control can branch from block i to j
   Draw an arc.

2) From exit node to entry node.
   Draw an arc.

Formula to Calculate cyclomatic Complexity of a program is

$$V(G) = e - n + 2p$$

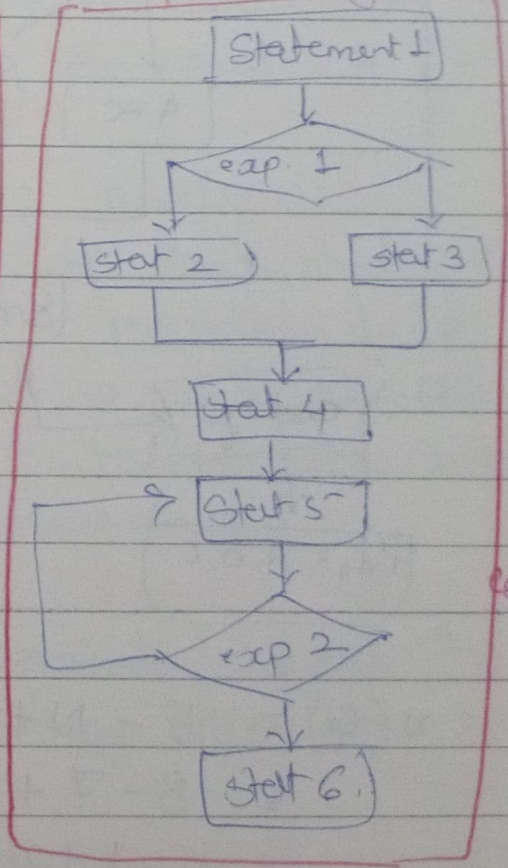
where, p is the nodes that have exit points
e is total no. of edges.
n is total no. of nodes.

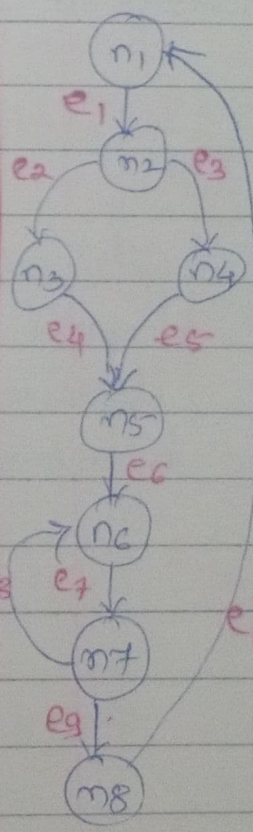
e.g.

Code

```
statement - 1
if expression 1
        statement 2
else
        statement 3
statement 4
do
        statement 5
while
        expression 2
statement 6.
```

Flow chart



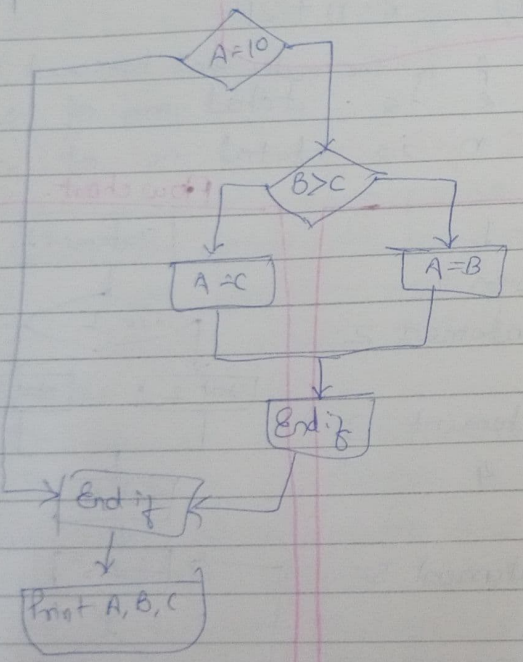


e = 10 , n = 8.

C.C. = 10 - 8 + 2

= 4. Ans

```
if A=10 then
    if B>C then
        A=B
else
    A=C
end if
end if
print A
print B
print C
```



$V(G) = E - N + 2xP$

$= 8 - 7 + 2 \times 1$

$= 1 + 3$

$= 3$ Ans.

UNIT - IV th

SOFTWARE STATIC & DYNAMIC ANALYSIS

→ Static analysis involves going through the code in order to find out any possible defect in the code.

→ Dynamic analysis involves executing the code and analyzing the o/p.

→ Advantages of static analysis:

1) It can find weaknesses in the code at the exact locat°.
2) It can be conducted by trained s/w assurance developers who fully understand the code.
3) Source code can be easily understood by other or future developers.
4) Weaknesses are found earlier in the development life cycle, reducing the cost to fix.

→ Limitation of static analysis:

1) Time consuming & conducted manually.
2) Automated tools can provide a false sense of security that everything is being addressed.
3) It does not find vulnerabilities introduced in the runtime environment.

→ Advantages of Dynamic analysis →

1) It identifies vulnerabilities in a runtime environment.
2) It allows for analysis of application for which we do not have access to the actual code.
3) It permits us to validate static code analysis findings.

(MU)

UNIT- II nd's Topic

FEASIBILITY STUDY → Feasibility is defined as the practical extent to which a project can be performed successfully.

→ To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical & workable in the software.

→ The objective of the feasibility study is to establish the reasons for developing the s/w that is acceptable to users, adaptable to change & conformable to established standards.

Types of Feasibility →

There are three types of feasibility:
① Technical Feasibility ② Economical Feasibility ③ Operational feasibility.

① Technical Feasibility → It assess the current resources (such as h/w & s/w) and technology, which are required to accomplish user requirements in the s/w within the allocated time & budget. The technical feasibility performs following tasks:

a) Analyzes the technical skills & capabilities of the s/w development team members.
b) Determines whether the relevant technology is stable and established.
c) Assured that the technology chosen for s/w development has a large no of users so that they can be consulted when problems arise & or improvements are required.

② Economical feasibility → It determines whether the required s/w is capable of generating financial gains for an organization.

→ It involves the cost incurred on the s/w development team, estimated cost of h/w & s/w, cost of performing feasibility study & so on.

→ A s/w is said to be economically feasible if it focuses on the issues listed below:

a) Cost incurred on s/w development to produce long-term gains for an organization.
b) Cost required to conduct full s/w investigat°
c) Cost of h/w, s/w, development team & training.

③ Operational Feasibility: It assesses the extent to which the required s/w performs a series of steps to solve business problem & user requirements.

→ This feasibility is dependent on human resources (s/w development team) & involves visualizing whether the s/w will operate after it is developed & be operative once it is developed & be operative once it is installed.

→ It also performs following tasks:

a) Determines whether the problems anticipated in user requirements are of high priority.

b) Determines whether the sol" suggested by the s/w development team is acceptable.
c) Analyses whether users will adapt to a new s/w
d) Determines whether the organizat" is satisfied by the alternative sol" proposed by the s/w development team.

★ Feasibility study Process ⇒
1) Inputmat" assessments
2) Inputmat" Collection
3) Report writing
4) General Inputmat"
5) Mgmt summary
6) Environment etc.

★ TRACEABILITY ⇒ It is a property of an element of documentation or Code that ~~eterne~~ indicates the degree to which it can be traced to its origin or "reason for being".

⇒ The significance of traceability is:
a) To identify the apt version of test cases to be used
b) To identify which ~~can~~ test case can be reused or need to be updated.
c) To assist the debugging process so that a defects found when executing tests can be tracked back to the corresponding version of requirements.

⇒ e.g. of traceability include:
a) External source to System requirements.
b) System requirement to s/w requirement.
c) s/w requirement to high level design.
d) Detailed design to code.
e) s/w requirement to test case.

⇒ The o/p of requirement traceability is represented through a traceability matrix.

Business Req.	Business Req/ Use case	Functional Req. ID.	Functional Req/Use case	Priority	Test Case ID.
~~Business~~ Req 1.	Reservation Module.	FR 1.	One way Ticket booking	High	TC# 001 TC# 002
		FR 2	Round way ~~Booking~~	High	TC# 003 TC# 004
		FR 3.	Multicity Ticket Booking	High.	TC# 005 TC# 006
Bus Req 2	Payment mode	FR_4	By Credit card	High	
		FR_5	By Debit card	high.	
		FR_6	By Reward pt.	Medium	

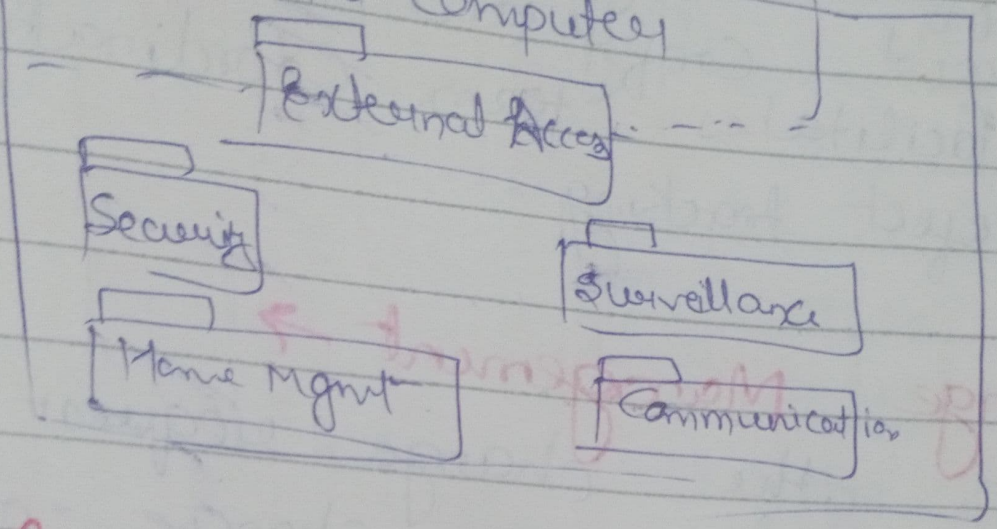


fig → Deployment diagram.

SOFTWARE CONFIGURATION MANAGEMENT →

→ It consists of standard processes & techniques often used by organizat" to manage the changes introduced to its software product.

→ It helps in identifying individual elements, Configurat", tracking changes, version selection, control & baselining.

→ SCM is also known as software Control mgmt.
→ SCM deals with different technical difficulties of a project plan.
→ SCM helps to eliminate the confusion caused by miscommunicat" among team members.
→ SCM system controls basic components such as s/w objects, program code, test data, test o/p, design document and users manual.
→ It has following advantages:
a) Reduced redundant work.
b) Effective mgmt of simultaneous updates
c) Avoid Configurat" related problems.
d) Facilitates team coordination.
e) Defect tracking

★ Change Management →

→ In it the change request, change report, & engg change order sequence can be achieved in an agile manner that is acceptable for most webapp development projects.
→ Each change should be categorized into one of four classes:

class 1 → a content or funct" change that corrects an error or enhances local content or functionality.

class 2 → a content or funct" that change that has an impact other content objects or functional components.

Configuration Management cycle

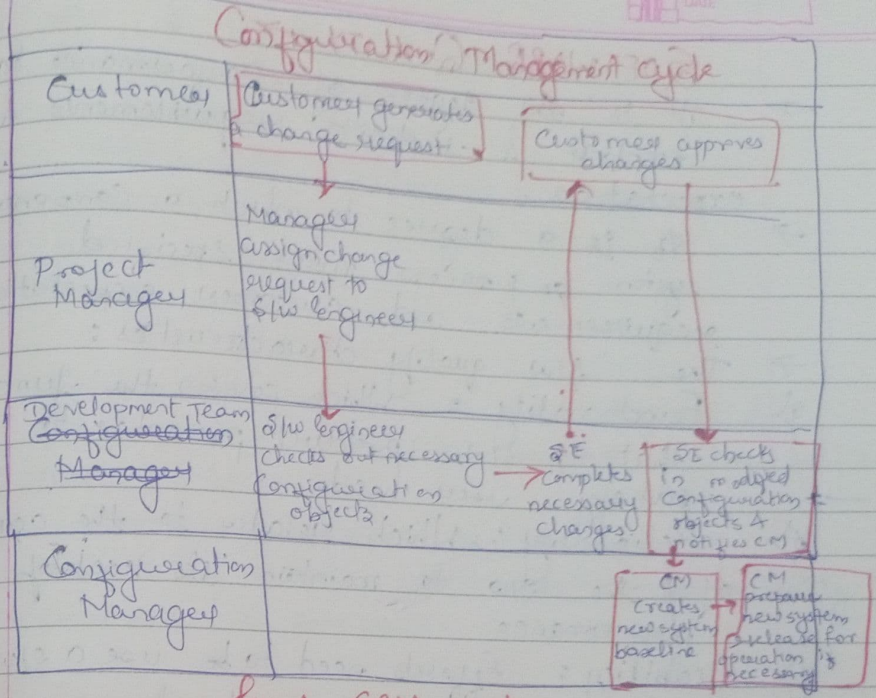


fig: SCM cycle.

class 3 → a content or function change that has a broad impact across a webapp

class 4 → a major design change that will be immediately noticeable to one or more categories of users.

[Refer fig 22.7 Managing changes for web app.]

★ Elements of Configuration mgmt are:

a) Configurat" identificat" b) Configuration Control. (each change to an instance of product is known, authorized & unchanged)
c) Configuration Accounting (it keeps track of the status of each instance of product)
a) Configurat" identificat" → It provides method for specifically & uniquely identifying each instance.

SOFTWARE QUALITY →

→ It is the degree of conformance to explicit & implicit requirements & expectations.

→ It is a degree to which a component, system or process meets specified requirements and customer expectations.

→ The s/w quality characteristics:

a) Functionality: Which covers the function that a s/w product provides to satisfy users needs.

b) Reliability: Which relates to the capability of the s/w to maintain its level of performance.

c) Usability: Efforts need to be use a s/w.

d) Efficiency: Resources used when a s/w is executed.

e) Maintability: Efforts needed to make changes to s/w.

f) Portability: Availability of the s/w to be x-ferred to a different environment.

APPROACHES FOR SOFTWARE QUALITY ASSURANCE →

→ SQA is a set of activities for ensuring quality in s/w engineering processes.

→ It includes following activities:

1) Process definition & implementation
2) Auditing
3) Training.

→ SQA process includes:

a) S/w Development Methodology
b) Project Management
c) Configuration Mgmt
d) Requirements Development / Management.
e) Estimation
f) S/w Design
g) Testing etc.

→ Once the process have been defined & implemented, Quality assurance has following responsibilities:

a) Identifying weakness in the processes
b) Correcting those weaknesses to continually improve the processes.

A. Software Quality Control → It is a set of activities for ensuring quality in s/w products, s/w quality control is limited to the Review / Testing phases of SDLC & the goal is to ensure that the products meet.

→ requires more experience, better guidance & Creativity.
or
e.g. Compilers

3. Embedded → It requires highest level of complexity, creativity & experience.
→ Large team.
→ developers are sufficiently experienced.
→ Creative to develop complex models.

☆ Types of COCOMO Models →

1. Basic COCOMO Model
2. Intermediate COCOMO Model
3. Detailed COCOMO Model.

→ Basic COCOMO can be used for quick & slightly rough calculation of S/W costs.
→ Intermediate COCOMO takes cost drivers into account & detailed COCOMO additionally accounts for the influence of individual project phases. It refines estimate by using a set of 15 cost drivers.

Formula:

Effort Applied (E) = $a\,(KLOC)^b$ (thousands of lines of code)

KLOC → Kilo-lines of code.
E → Efforts in staff months
a & b, c & d are coefficients or constants

Development Time (D) = $c\,(\text{effort applied})^d$

People required (P) = E/D.