Page NO.

1. 1 to 33 – Mid Term
2. 34 to 81- Final Term

MD IFTAKHAR KABIR SAKUR

25th BATCH

COMPUTER AND COMMUNICATION ENGINEERING

International Islamic University Chittagong

COURSE CODE: CCE-3609

COURSE TITLE: Software Engineering

COURSE TEACHER:

Md Jiabul Hoque

Assistant Professor
Computer and Communication Engineering

CCF-3609

(Software Engineering)

pdf-1

Confidentiality → গোপনীয়তা

Competence → যোগ্যতা এবং নির্ভুলতা

Intellectual property → Laws

Computer misuse → ~~Technical skills~~ Game হ্যাক, virus

Question:-

→ Software vs program

→ Importance of S.E

→ ~~Attribution~~ Characteristic of good software

→ Types of application

→ Ethics of SEngineer

PdF-2

Classical waterfall Model:-

Feasibility:- সম্ভাব্যতা যাচাই

Requirement:- কী দরকার, কী নাই। না থাকলে বাদ

Design:- Design করা

Coding & maintanence (unit test):- Code লেখা। যেভাবে একটা Language এ লেখা হয়।
এরপর test করা

Integration & system:-
সকল ফাইলগুলো এক করা

Maintenance:-

Testing:
Alpha test:- Done by developed team

Beta :- Done by customers

Acceptance :- গ্রাহক কি গ্রহণ করে।

Maintanence! -

Corrective! - কোনো error থাকলে Correct করা।

perfective! - Customer এর request মতো কাজ করা।

Adaptive! - New environment or working in PC or not. (Also depend on OS)

Advantages of CWM! -

Disadvantages of CMW! -

Iterative waterfall Model:-

Spiral Model:- উদ্দেশ্য, বিকল্প সমাধান

Risk handling, per phase এ Analysis করতে হবে

Objectives determination --

কি কি দরকার হবে। কি করতে হবে সে সম্পর্কে information

Identify & resolve risks:-

সকল সমস্যার সমাধান দেখা হবে এবং সেরা সমাধান adopt

develop next version:-

update আনা

Review & Plan for next phase:-

পরবর্তী ধাপে কি কি পরিবর্তন হবে তার জন্য প্ল্যান করা।

Risk Handling in spiral:-

Risk অনেক বেশি হয়ে যেতে পারে। Customer এর ৬

Incremental PM:-

Successive version model হিসেবেও পরিচিত। Sample or demo দেওয়া হয় প্রথমে তারপর প্রয়োজনমতো change করা হয়।

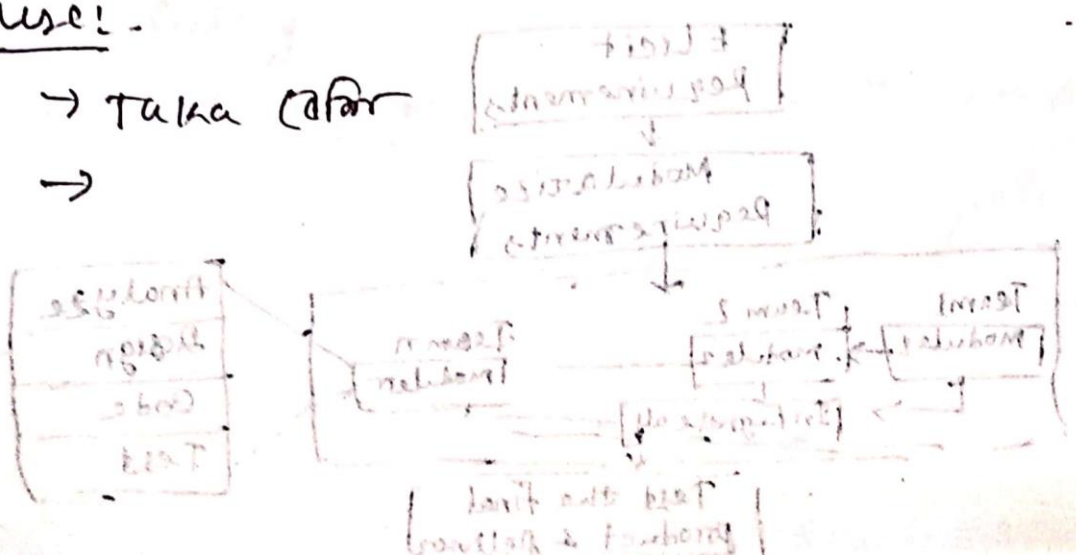
I] Single:- At a time একটা

Parallel:- At a time অনেকগুলো team wise

Use:-

→ Taka বেশি

→



RAD:-

প্রতিটি module বিভিন্ন দলে সুষ্ঠুভাবে কাজ করা হয়।

4 basic phase:-

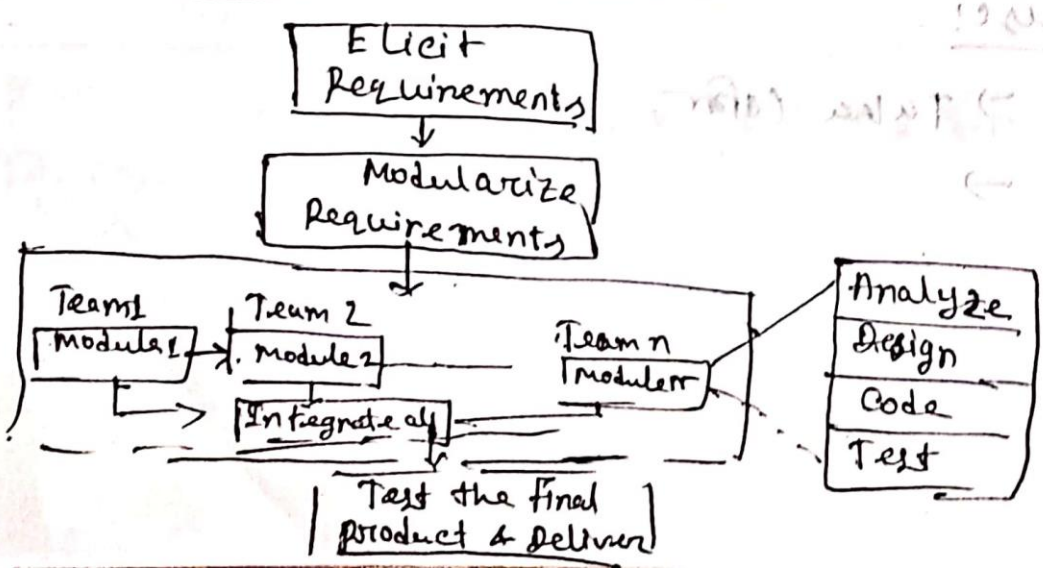
1] Requirement planning:- brainstorming task ~~storming~~ analysis, FAST(Facilitated Application Development Technique)

2] User description:- user কি চায়

3] Construction:- কোন কিছু পরিবর্তন, পরিবর্ধন এগুলো লাগলে। এই জন্য prototype build করা হয়।

4] Cutover:- ~~সব পছন্দের নাই তা kick out~~। সকল subparts একত্রিত করা হয় এবং test করা হয়।

~~RAD vs Traditional~~ SDLC:-



Agile Model:- (চমৎকার মডেল)

নির্দিষ্ট time এর মধ্যে করতে হবে। বার বার করতে হবে।

পুরো software তৈরি করে একবার deliver দিতে হয় না।

SDLC Model:-

→ Crystal

→

Agile Model এর Combination:-

→ Requirement gathering

→ " Analysis

→ Design

→ Coding

→ Unit testing

→

XP:- Agile Model এর framework।
Software quality improve করে। এই Model খালো Good practice খালো programmer রা choose করে।

Code Review:- Detect error and correct errors. Pair programming suggest করে।

Testing:- Test করা হয়। (TDD suggest by XP) যাতে errors remove করে reliability improve করে।

Incremental Development:-
Customer Feedback পাওয়া যায় কাল সহজেই পরবর্তী ধাপে ও যাওয়া যায়।

Simplicity:- Simplicity এর কারণে খালো quality code হয় test ও debug ও করা যায়।

Design:- খালো Software এর জন্য খালো design। খালো design ও করা হয় daily.

Integration testing:-
বিভিন্ন functionalities এর bugs identify করে।

Basic principles of XP:-

(পড়ে আছে)

Application of XP:-

→ ছোট project এ

→ যেকোনো নতুন tech বা Research project এ use হয়।

SDLC V- Model:-

V Shaped procedure

Require Analy.

Sys. Deg.

Arc. Design

Acc. Testing

Sys Testing

Integra...

(পড়ে আছে)

Verification!- সকল requirement আছে কি না?
বানানোর আগে কাজ করবে কি না
দেখা (coding এর আগে check)

Validation!- কাজ করার পর soft. আসলে কাজ
করে কি না (coding এর পর checking)

Testing:-

Unit test!- eliminating bugs; একেবারে project
এর শুরুর দিকে করা হয়।

~~Integrate test~~:-

Integration Testing!- Unit test এর পর করা হয়।
এই test বিভিন্ন module কাজ করবে কি না তা
test করে।

System Testing:-

Functional, non-functional requirement test করে;

User Acceptance Testing (UNT):

User এর মন মতো হয়েছে কি না তা test
করে।

Why preferred ? (V - Model)

When to use?

Comparison of different

project ও choose ও rules:-

Disadvantages of RAD:-

→ Highly skilled professional লাগে।

→ খুব: ব্যবহারযোগ্য Components না থাকলে failed হতে পারে project.

→ Team Leader কে Always close থাকতে হয় time to time কাজ এর জন্য

→ যে system modularize করা যায় না তার জন্য অনুপযুক্ত

→ Customer কেও খুব কাছ থেকে Attach থাকতে হয়

→ ছোট project এ use হয় না।

Advantage:-

→ একই Component বারবার ব্যবহার সময় কমায়।

→ Customer হতে feedback কুড়ার stage এ সবসময় থাকে

→ Fewer developers are required, so costs are reduced

→ কম সময়ে ভালো টুলস এর ব্যবহার করলে অনেক product বানায়।

→ প্রতিটি stage এর কার্যক্রম মাপা যায়

Application:-

→ This model should be used for a system with known requiring short development time.

→ ~~Project~~ যে project modularized করা যায় ~~তার জন্য~~ এবং reuse করা যায় components তে তার জন্য ভালো।

→ ছোট পরিবর্তন ছাড়া new system develop এ ব্যবহার করা হয়।

→ Skilled person থাকলে use করা

→ Budget বেশি এবং automated tools & techniques থাকলে use করা।

RAD vs Traditional SDLC

RAD	Trad. SDLC
1] Stage গুলো তালোভাবে সংজ্ঞায়িত না	1] তালো ভাবে সংজ্ঞায়িত
2] প্রতিটি stage এ review করার মাধ্যমে approach is iterative	2] Application Development এর জন্য futuristic, কলের পদ্ধতি অনুসরণ করে
3] RAD tools & techniques enable faster & easier prototypes	3] prototype অনেক কঠিন এবং অনেক সময় লাগে তৈরিতে
4] সকল কিছু আগে থেকে জানা লাগে না।	4] project শুরুর আগে সবই জানতে হয়
5] পরিবর্তন সহজ	5] পরিবর্তন কঠিন
6] Customer Feedback বেশি থাকে	6] Limited থাকে
7] কম documentation	7] documentation বেশি
8] ছোট Team এ ভাগ করা যায়।	8] বড় team ~~আবার~~ বিভিন্ন উন্নয়ন stage বানায়
9] একই জিনিসের বারবার ব্যবহার সময় কম লাগায়।	9] powerful & efficient tools Highly skilled professional লাগে

Agile Development Models:

সময় নির্দিষ্ট, ছোট ছোট করে deliver করা।

In the mid 1990 এ proposed হয়।

→ Request এর উপর depend করে project change এ ব্যবহৃত হয়।

Agile SDLC Models:-

eg. → Crystal
→ Feature-Driven Development
→ Scrum
→ Extreme programming (XP)

ছোট করে কাজ করা হয়। সহজে manage করা যায়। কয়েক সপ্তাহ এর মধ্যে তৈরি করা হয়। এক সময়ে একটি iteration, বড় plan করা হয় না।

Steps involve in agile SDLC:

→ Requirements gathering
→ " Analysis
→ Design
→ Coding
→ Unit testing → Acceptance testing

principle of Agile model:-

1] Individuals & interactions over processes & tools :-

Customers ও team member দের মধ্যে Communication এবং Collaboration বেশি হয়। Face to Face Communication বেশি হয় process ও documentation এর জন্য।

2] Working soft. over comprehensive documentation:-

Software তৈরি করা প্রাথমিক focus থাকে। Documentation Important হলেও এতোটা গুরুত্ব দেওয়া হয় না।

3] Customer collaboration over contact negotiation:-

Customer daily কাজের রিপোর্ট ধরে ও Feedback পান। কারণ এখানে Customer এর প্রয়োজনীয়তাকে গুরুত্ব দেওয়া হয়।

4] Responding to change over following a plan:-

যেকোনো সময় পরিবর্তন করা সম্ভব।

5] pair programming :- দুইজন programmer একই station একসাথে করে। একজন কোড করে অন্যজন review করে।

6] Development team:-

ছোট হয়। ৫-৯ জন Developer থাকতে পারে। সবার face to face collaborate করে।

Advantage:-

→ Pair programming এর জন্য code ভালো হয়।
→ project deadline এর আগেই হয়।
→ যেকোনো পরিবর্তন সহজ।

Disadvantage:-

→ কম documentation এর কারণে team member রা confused হতে পারে।
→ কম documentation এর ফলে maintunence এর সময় developer রা ঝামেলায় পড়ে।

→ Customer এর trust বাড়ে
→ সহজেই পরিবর্তন সহনশীল
→ দ্রুতই Feedback পাওয়া যায়
→ Interaction to high priority দেয়
→ ভালো design

→ বড় software এর ক্ষেত্রে কুরতে effort দেওয়া লাগে
→ Code focused, documentation না
→ Depends on customer
→ বড় organizations এ face to face meeting করা
→ Senior programmer রা ও decision নিতে পারেন।

Basic principles of XP:-

Coding:- Normal code হচ্ছে different। এখানে diagram use করা হয় যা পরবর্তীতে code এ Convert করা হয়।

Testing:- Testing ছাড়া কোনো সুযোগ নেই।

Listening:- Good quality এর Soft. এর জন্য Customer এর কথা শোনাটাও important।

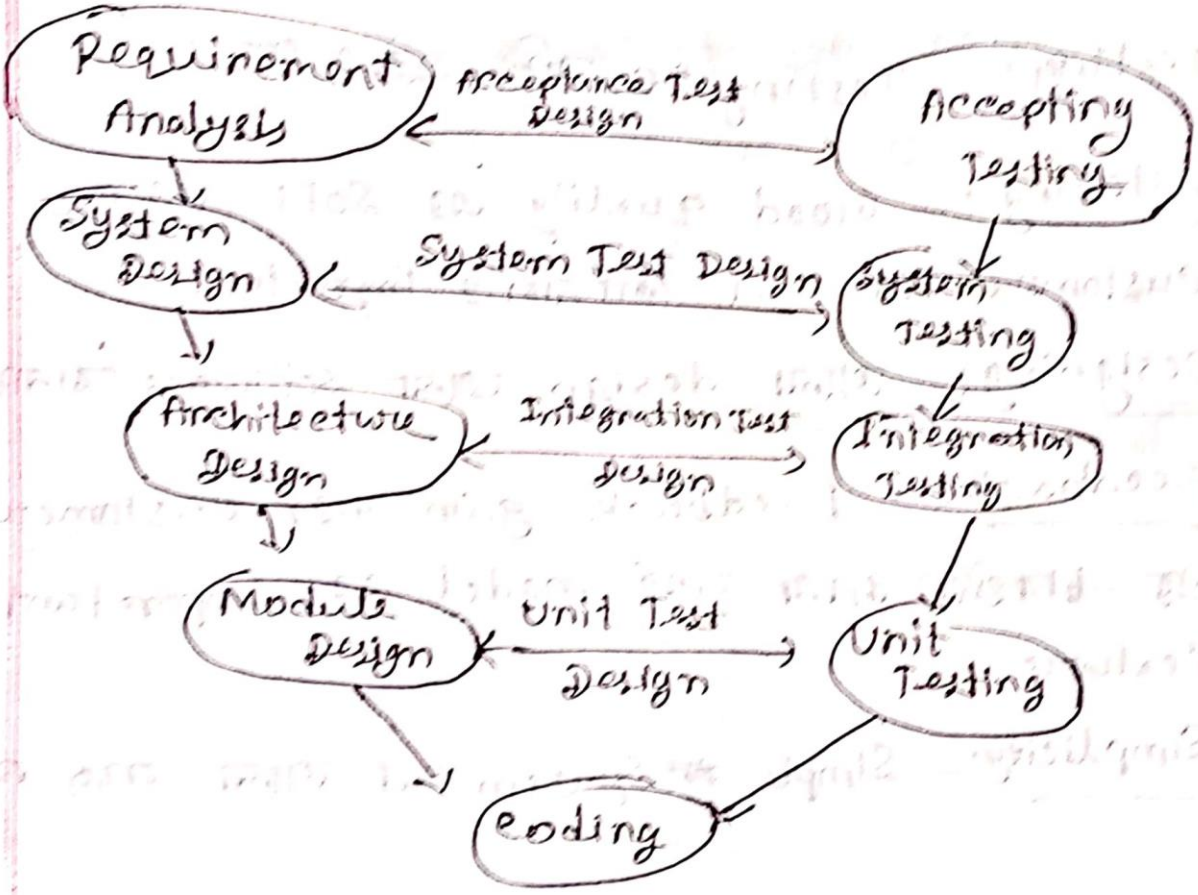
Designing:- ভালো design, ভালো software বানায়।

Feedback:- Feedback gain করা customer এর চাহিদা জানা এই model এর important feature.

Simplicity:- Simple ও System যা ভালো কাজ করে।

SDLC V-MODEL

Process executes in a sequential manner in V-shape. known as verification & validation model.



Design phase

Requirement Analysis:- কি বা কি ধরনের (All kinds of needs)

System Design:- Complete Hardware & software setup for the development

Architectural Design:- বিভিন্ন ধাপে (module) এ ধাপ করা হয়। ~~Ex~~ ~~data transfer~~

Module design:- Small Modules এ ভাগ করা হয়। Module গুলো LLD (Low Level Designed) এ design করা হয়।

Principles of V-Model

Large to small:- প্রতিটি phase এ user এর চাহিদা পূরন করে। যেহেতু অনেক ধাপ পার হয়ে আসে তাই এরা খুবই ডেভালপড Stage (refined) হিসেবে থাকে

Data/process Integrity:- Data এবং process খুবই সুরক্ষিত। প্রতিটি Stage এই ~~একটির~~ গুরুত্ব প্রদান.

Cross referencing:- কোনো specific information কে refer করে। Requirement এর তার testing - Z

এখানে cross ref.

Tangible Documentation:-

প্রতি project এ documentation দরকার। Support team ও project Development team উভয়ের ব্যবহার করা উচিত।

When to use V-Model:-

→ Requirement fixed
→ Technical resource ও expert available থাকলে।

Advantages:-

→ Highly Disciplined
→ ছোট project -এ
→ আঁকারন ও সহজ
→ Verification ও validation এ focus করে
→ project management track করতে পারে।

Drawbacks:-

→ বড় risk
→ complex ও object এর জন্য ভালো না
→ Contains clear না থাকলে পরিবর্তনের শঙ্কা থাকলে ব্যবহৃত হয় না
→ Phase Iteration হয় না
→ সমসাময়িক event সহজে handle করে না।

Classical water-fall Model:

সকল Model এই Model এর উপর নির্ভরশীল। এটা আদর্শ Model। এটা কোনো project ব্যবহার না করাই ভালো যেহেতু এতে error Correction এর কোনো option নেই।

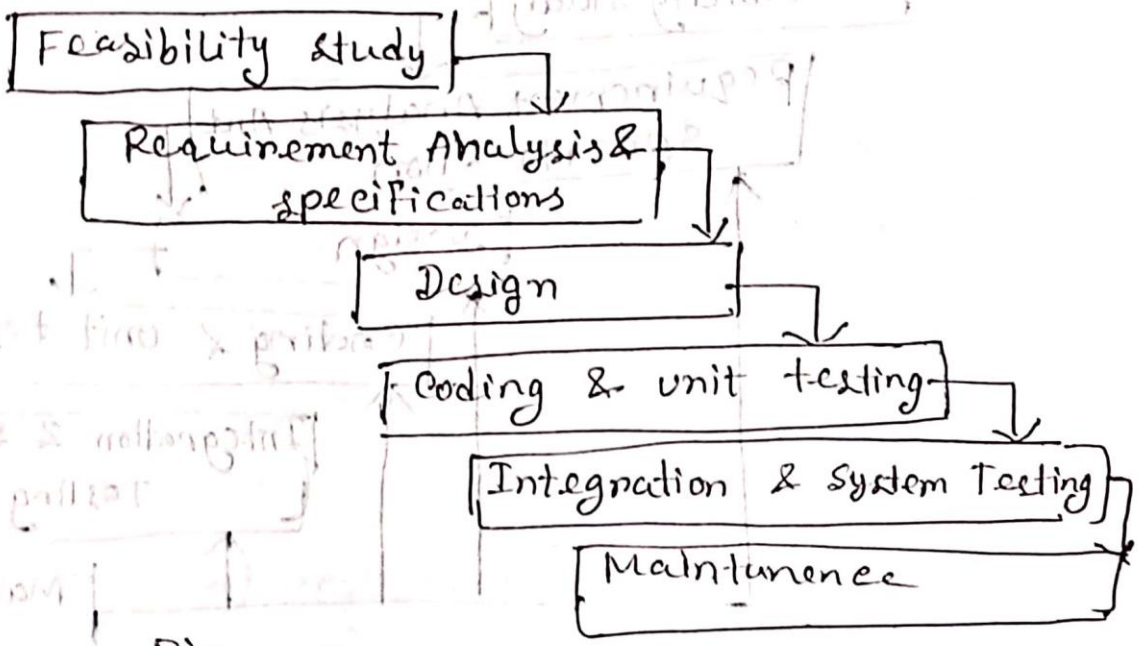


Fig:- CWM

Iterative Waterfall Model:

-সবচেয়ে বেশি ব্যবহৃত model। সহজে ব্যবহারযোগ্য এবং বুঝতে সহজ। তবে বড় project এ অব্যবহারযোগ্য। কম বাজেট বড় project এ এই model এ Risk বেশি থাকে।

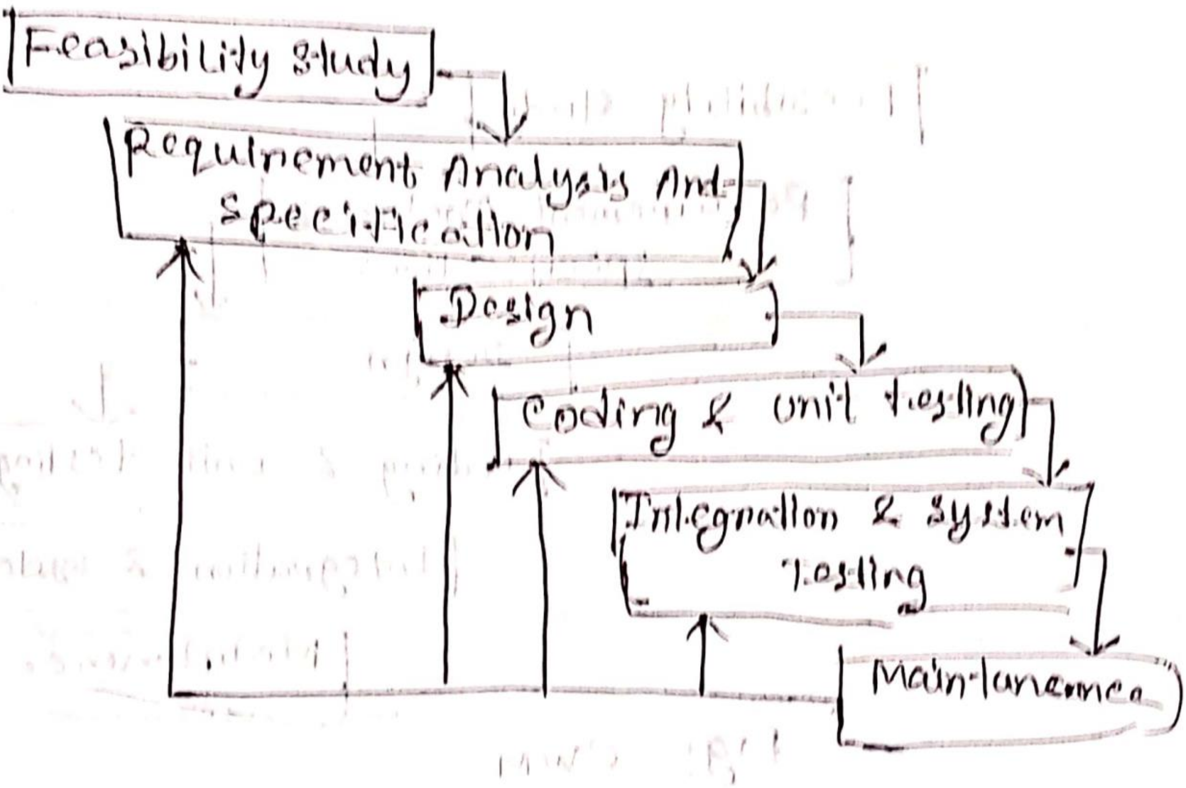


Fig:- IWM

prototyping model

Suitable for project এ যে customer এর কিছু না technical solution জানা থাকে project এর ক্ষেত্রে risk বেশি থাকে। User interface part এ এখন বেশি popular.

Evolutionary model

বড় project এর জন্য suitable। ছোট ছোট module এ ভাগ করা যায়। Object oriented Development project এর জন্য বিশেষ উপযোগী। adopt হয় যখন customer এই ছোট ছোট module গুলো দেখে বুঝতে পারেন।

Spiral Model

It is a Meta-Model। Flexibility ও Risk handling এর জন্য বিখ্যাত। বড় Software ও Challenging Software এর জন্য ব্যবহার হয়। Model টি অন্যান্য model হতে তৈরি করি।

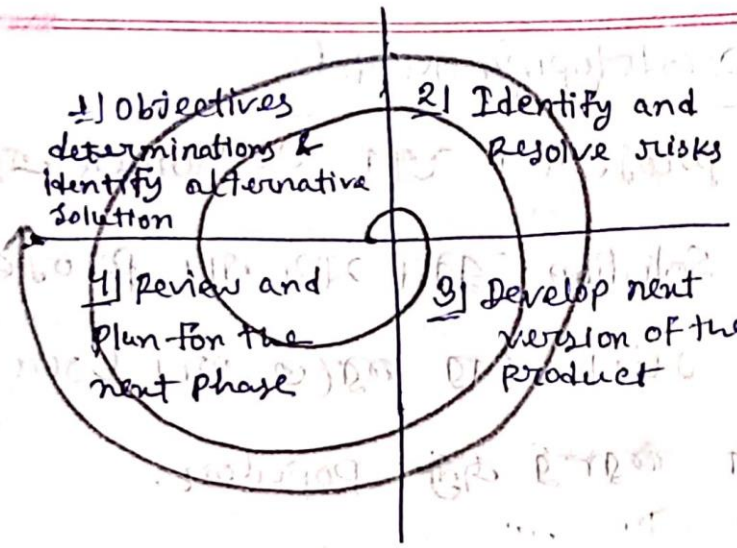


Fig: SM

User interface Design :-

Front-end application view

- → Attractive
- → Simple to use
- → Responsive in short time
- → Clear to understand
- → Consistant on all interface screens.

Two types of UI :-

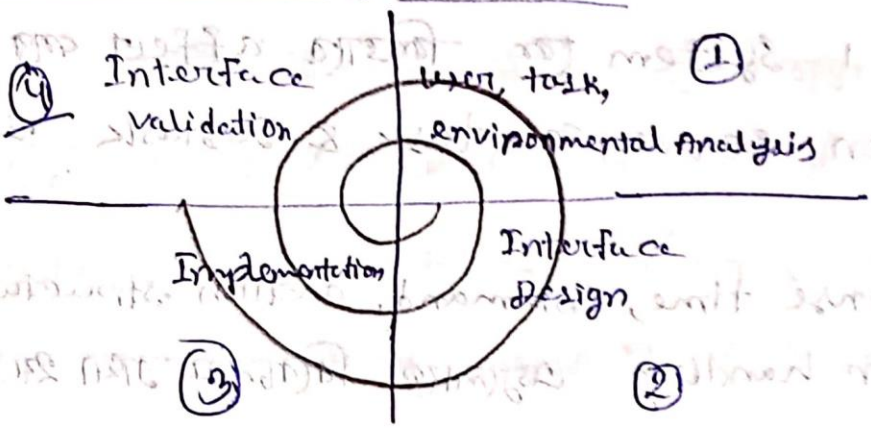
Command Line Interface :-

Command prompt ত Command দিলে system change হয়। তবে user কে syntax মনে রাখতে হবে

GUI :- ☒ Combination of both hw & sw.

Simple interactive interface to interact with the system.

User Interface Design process :



① profile of users. Who will use the system. Based on user profile of users are made. per category the requirements are gathered. And based on this requirements the developer develop the interface.

The analysis of the user environment focuses on physical work environment.

→ Physically কোথায় located আছে

→ ~~User for Interface এর~~ →

→ user এর অবস্থান কেমন হবে।

→ Space, light, constraints কেমন হবে

→ পারিপার্শ্বে কারা মানুষ কেমন হবে।

(2) user তার নিজের task কে perform করে। System কে কিভাবে affect করে। Action sequence, task & subtask specify করে।

Response time, Command, action structure, error handling এগুলোকে বিবেচনায় রাখা হয়।

(3) prototype তৈরির সাথে সাথে implementation activity শুরু হয়।

window, menu, device interaction, error messages, Command এগুলোকে বিবেচনায় রাখা হয়।

(4) Testing the interface. Perform tasks correctly or not? Should achieve all users requirements. Easy to use & learn. user should accept this too.

Golden rule For designing interface :-

→ User এর জন্য সহজভাবে তৈরি করা যাতে সহজেই enter ও exit করতে পারে।

→ Key board, Mouse, touch screen এসব দিয়েও যেনো সহজে ব্যবহার করতে পারে তার জন্য এখানে এসব feature রাখতে হবে

→ Undo operation আনতে হবে। এবং কাজের আগের অন্যকাজ পূর্বের কাজ হারিয়েই যেনো করতে পারে।

→ Highly skilled or advanced user যেনো interface change করতে পারে তা option আনতে হবে। যাতে user একই জিনিস দেখতে দেখতে Bored না হয়।

→ Backended things should be hidden. so that the user don't see it.

→ Objects যা interface এ আসে তা user যেনো edit করতে পারে।

Reduce the user's memory-load:-

→ পূর্বের কাজ যেনো user মাথায় না রেখে সেভাবে design করা। যাতে user কে কয়েকবার জানতে না হয়।
→ Alternate default রাখতে হবে। user যদি চায় তবে নিজের মতো ও দিতে পারে।
→ Keyboard shortcut থাকতে হবে
→ Interface should be established based on real world.
→ Should'd be organized hierarchically.

Conceptual Design of the System

→ Customer যেন সহজ language
→ Full explaination থাকতে হবে
→ Functionality describe করতে হবে
→ Independent of implementation
→ যা চাওয়ার তা যেনো সামঞ্জস্য হতে হবে।

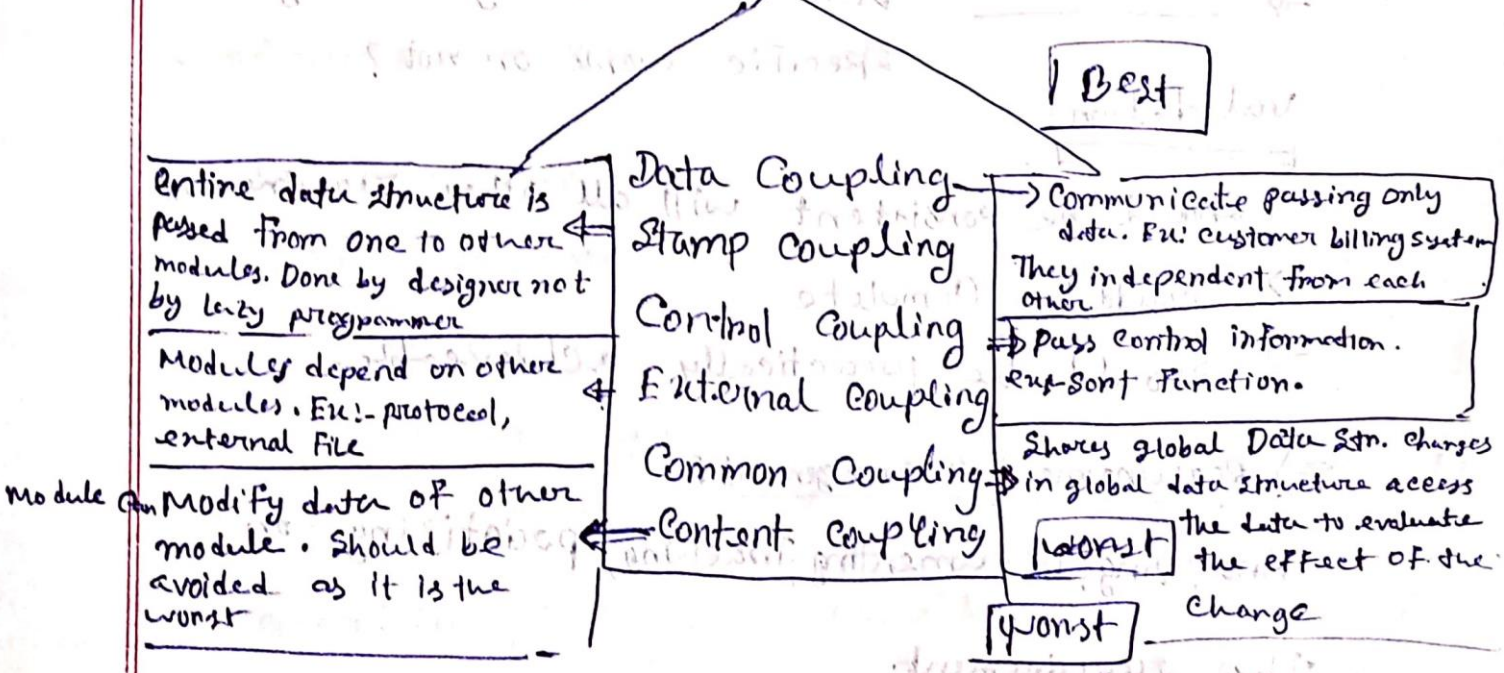
The.

Technical design of the system:-

→ Hardware
→ Software components
→ Software Architecture
→ Network Architecture
→ Data structure and flow of data
→ I/O components
→ Shows interface.

Coupling

→ Interdependency between the modules.



PdF-3

Requirement Engineering (defining, Documenting, maintaining the requirements)

→ Requirements elicitation :-

Interview, brainstorming, Task Analysis, prototyping etc. Does not form formal model. widens knowledge of Analyst & thus helps in providing input to next stage.

⇒ Requirement

→ verification :- Software correctly working for the specific work or not?

Validation :-

→ Should be consistent will all other requirement
→ Should be Complete
→ Should be practically achieveable

⇒ Requirement management :-

Analyzing, Documenting, Tracking, priotizing on the requirements

Software Requirements: (3 types):-

1] Functional Requirements:-

All the functionalities must be in there. They are stated as a part of input of the system.

Example:- In a Hospital Management System, a doctor should be able to retrieve all the information of his patients.

2] Non-functional Requirements:-

portability, Security, Maintainability, Reliability, Scalability, performance, Reusability, Flexibility.

→ Interface constraints
→ Security, response time, storage space
→ Economic

3] Domain Requirement:-

A school management system where all information are gathered. The functionality of being able to access the list of faculty, student.

Requirement Elicitation

-গতি, .color ও (বক্তি)।

Requirement:-

→ Applied Place সমস্যাকে চিহ্নিত করা।

→ Customer এর সমস্যা বুঝতে হবে

→ Detail investigation

→ Constrans for System Development.

Methods:-

→ Interviews → open questions

→ Brainstorming

→ FAST (Facilitated Application Specification Technique) → Team oriented approach :-
- environment surround the system
- Produced by the system
- Used by the system

→ Quality Function Deployment (QFD)

→ Case Approach

→ ① Normal Requirement : (Entry of marks, calculation)

② Expected " (& protection from threat)

③ Exciting " (Beyond customers expectation)

→ **Actor:-** External agent .to provide better requirements.

→ **use cases:-** All the possible ways to use system.

→ **Use case Diagram:-** Graphics of what happens when an actor interacts.

CCE-3602
(Software Engineering)

Requirement Engineering process:-

Defining, documenting & maintaining the requirements.

REP consists of:

(1) Requirements elicitation (তথ্য সংগ্রহ করা)

(2) Requirements specification

(3) Requirements verification & validation

(4) Requirements management.

(1) Requirement Elicitation:-

→ Used to gain knowledge about project domain & requirements. The various sources of domain knowledge include customers, business manuals, existing software of same type, standards & other stakeholders of the project.

The techniques for requirements elicitation include interviews, brainstorming, task analysis, prototyping etc.

Elicitation does not produce formal models of the requirement understood.

2] Requirements specification:-

Used to produce formal software requirement models.

Functional, Non-functional requirements and the constraints are specified by this models. During specification more knowledge might be needed, which can trigger the elicitation process.

Model used at this stage include ER diagrams, data flow diagrams (DFDs), function decomposition diagrams (FDDs) etc.

3] Requirement verification & validation:-

Verification:- Ensures that the software correctly implements in the function or not.

Validation:- Different set of tasks. If requirements are not validated, errors in the requirements definitions would propagate.

The main steps for these are :-

→ NO two requirement should conflict with each other

→ The requirements should be completed in every sense.

→ Should be partially achievable

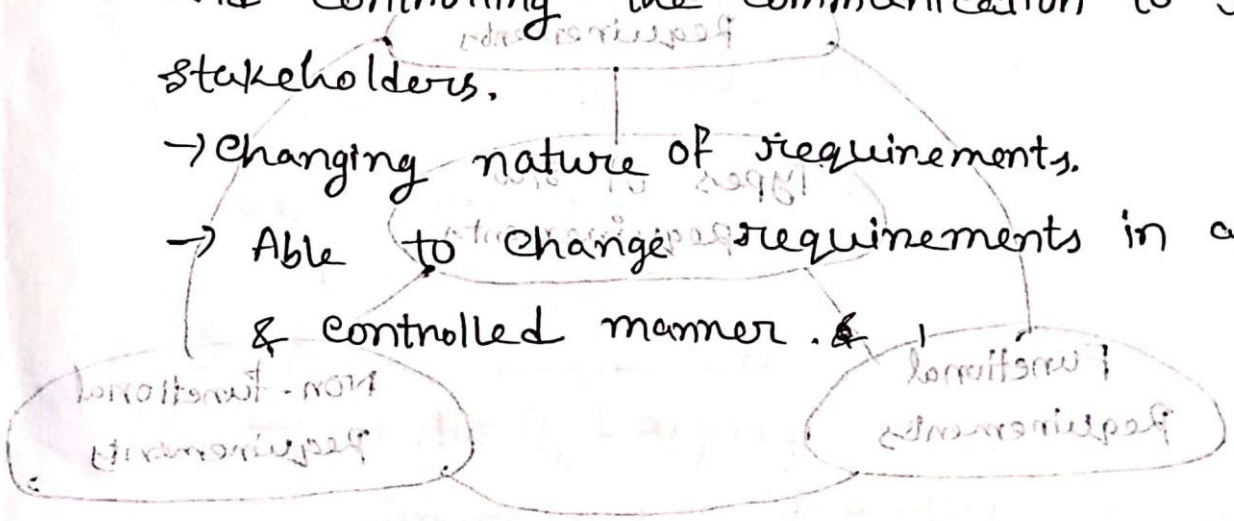
Reviews, Buddy checks, Making Test cases etc are the methods used for this.

4) Requirement Management :-

Process of analyzing, documenting, tracking, prioritizing & agreeing on the requirement and controlling the communication to relevant stakeholders.

→ Changing nature of requirements.

→ Able to change requirements in a systematic & controlled manner.



Prev | Next

Classification of S/W requirement

According to IEE 729:-

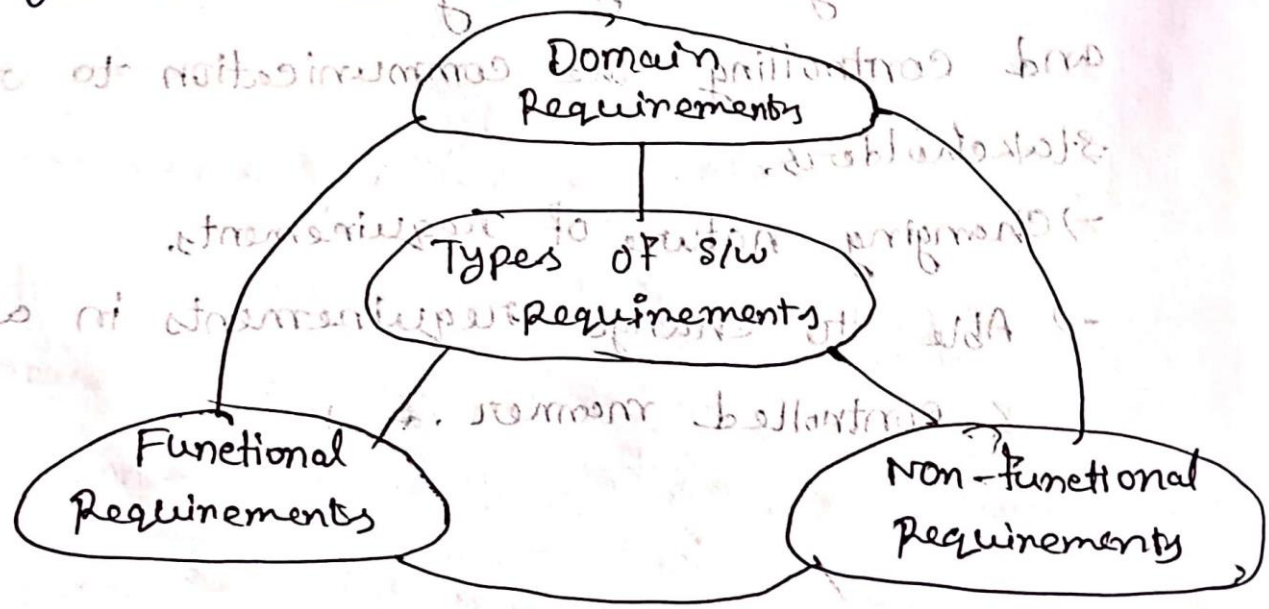
⇒ Condition or capability by a user is needed to solve problem or achieve an objective.

⇒ Condition or capability must be met or possessed by a system or system component to satisfy contract

⇒ A documented representation of condition, capability in 1 & 2.

Next

Types of Softwares



Functional Requirements:- (Specify the feature of software)

End user (সর্বশেষ ব্যবহারকারী) এর demand অনুযায়ী Basic facilities system এ offer করা।

সকল functionalities আগে থেকে contract এ অন্তর্ভুক্ত হতে হবে।

এগুলো Input হিসেবে দিতে হবে এবং expected output পাওয়া সাপেক্ষে এমন হতে হবে।

User দ্বারা recommended এবং কোন বেটা final প্রজেক্টে দেখতে পারবে না Non-functional R এর মতো।

Example:- Hospital Management system where doctor should be able retrieve info of the patients.

Many ways of doing:-

→ Natural Language

→ Structured or formatted Language

→ Formal specification language with proper syntax.

Non Functional Requirements:- (Specify quality attributes)

System must meet the quality constraints according to the project contract.

The priority might varies from project to project. They are also called as Non-behavioral requirements.

Eg They deal with:

(i) Portability
(ii) Scalability
(iii) Security
(iv) Maintainability
(v) Reliability
(vi) performance
(vii) Reusability
(viii) Flexibility

NFR's different Types:-

(1) Interface constraints
(2) performance constraints (response time, security, Storage space etc)
(3) Operation constraints
(4) Life cycle constraints (Maintainability, portability)
(5) Economic constraints.

Domain Requirement:-

DR is specific to a particular domain or industry.

For example:- Medical software system might be that it must be compliant with the Health Insurance portability & Accountability Act. (HIPAA).

It can be functional or Non-functional

Requirement of Elicitation

Difficult, error-prone, communication intensive software development. Can be successful with effective customer developer partnership.

Activities of RE:-

(1) Knowledge about the overall area where it is going to be implemented

(2) The details of the precise customer problem where the system are going to be applied must be understood.

(3) Interaction ~~with~~ of system with external requirements.

(4) The constraints for system development.

Methods:- (IBFSU)

(1) Interviews.

(2) Brainstorming.

(3) Facilitated Application Specification Technique (FAST).

(4) Quality Function Deployment (QFD)

(5) Use case Approach.

(1) Interview:-

Customer's expectation. Expertise of different sites are selected for this.

(i) pre-ended interviews (Free questions are asked)

(ii) Structured interview (proper questions are asked).

~~(iii) Every idea is documented~~

(V) Generate lots of

2] Brainstorming sessions:-

→ A group technique

→ Generates a lot of new ideas.

→ A highly trained facilitator is needed.

→ Every idea is documented

→ A document is prepared which consists all the requirements.

3] FAST:-

works as a bridge between developers & users. A team oriented requirement gathering

* Each attendee are asked to make lists which consists

→ part of environment that surround the systems

→ produced by the system

→ used by the system.

4] Quality Function Deployment:-

Customer's satisfaction is of prime concern.

3. types:-

1] Normal requirements:-

Objective & goals of the s/w by customer.

Ex:- Grading system. Input number/marks getting the grade with calculated value

2] Expected Requirements:-

Must be needed Pectures.

Ex:- Third party/ unauthorize access.

3] Exciting Requirements:-

Beyond expectation & makes user happy.

Ex:- Third party interruption? Shut down entire system.

Major steps of this procedure:-

=> Identifying all stakeholders.

=> Making a list of all needs

=> Value indicating degree of importance is assigned to each requirement.

=> Final requirement category:-

(i) It is possible

(ii) Should be cancelled & the reason.

(iii) Can't be achieved & should be dropped off.

5] Use case Approach:-

A Technique combines text, pictures etc.

Describe 'what' insted of 'how'

Three major things:-

1] Actor:- Might be a person, machine etc.

An external agent.

2 types:-

1] primary Actors:-

Requirements assistant from the system.

2] secondary:- Their system needs assistance from.

2] Use cases:-

Sequence of interactions between actors & system. Capture who (actors) & what (interactions) with the system.

3] Use case Diagram:-

Graphically shows what happens when actor Interact

→ Stick Figure is used as actors

→ Oval is used to represent use case

→ A line is used to show relationship.

VVIM prev

Challenges in eliciting Requirements:

Eliciting first step. requirement gather error

1] Understanding large & complex system requirements is difficult:-

→ word 'large' represent 2 aspects:

(i) Large constraints in security

(ii) A large number of function to be implemented.

2] Undefined System Boundaries:

User unrelated, unnecessary function include important extremely large system budget

3] Customers Stackeholders are not clear about their needs:-

→ very basic idea about their needs but haven't planned much.

4) Conflicting requirements are there:-
(Two different stakeholder's want might conflict).

5) Changing requirement is another issue:-
(users want to change)

6) partitioning the system suitability to reduce complexity:-
(partitioned to make by different teams. Must ensure these don't get overlapped).

7) validating & tracing requirement:-
Entity name must be the same.
'Student' or 'students' can't be used use the same name.

8) Identifying critical requirements:-
Must be implemented Feature should be prioritized First so that crucial one

Can be implemented first.

9] ~~Reso~~ Resolving the 'to be determined' part of the requirements:-

To be determined are requirements which are yet to be resolved in the future. The number of such requirement should be kept low.

10] proper documentation, proper meeting time, and budget constraints:-

proper documentation is a challange as it might be changed also budget handling is also a challange too.

VVIm prior

SRS

Software Requirement Specification.

A description of a S/W system to be developed.

why? (Save time, punctuality, How to Complete...)

Characteristic: (CCCURVMT)

→ Correct
→ Complete
→ Consistent
→ Unambiguous.
→ Ranked for requirement and/or stability
→ verifiable
→ Modifiable
→ Traceable

SRS

Project Plan:- MeetUpMate (Better read from P Sir's pdf)

1] Introduction :-

The document lays out a project plan for the development of the "MeetUpMate" open source repository system by Rahim.

——— current & future developers of the project ---. The plan will include a summary of a system functionality, the scope of the project from the perspective of the 'MeetUpMate' team, scheduling & delivery estimates, project risks & how those risks will be mitigated. ---- will be recorded throughout the project.

2] Overview:-

In today's world ---- stress in life, so many gadgets but can't reduce stress ---. So my Application 'MeeturMate' is made. Where you will meet similar passion person ... Can also share videos ---- chat & have some ~~some~~ fun.

2.1: Customers:- Anyone can use.

2.2:- Functionality:-

→ Already existing account can register.

→ NO account make one.

→ Share videos/ images.

2.3] Platform:- web-based, mobile app, softw.

2.4] Development Responsibility:-

1] Fahim: developing software & I am developing.

3] Goals & scopes:-

(Almost similar to Functionality)

→ Account register

→ Create it If none

→ Sharing Photos / Videos

4] Deliverable: (I'll deliver during development)

→ Feature Specification

→ product Design

→ Test Plan

5] Risk Management :-

5.1 :- Risk Identification :-

→ people already using FB : So why people will join?

5.2] Risk Mitigation :-

→ In FB they don't meet with the common interest person

→ people can solve mysteries / cases --

→ =-

6] Scheduling : Estimates :-

Milestone	Description	Release Date	Release Iteration
M1	→ App view & Design	→ Oct-5, 2022	R1
M2	→ DB of my App	→ Oct-17, 2022	R1
M3	→ Integrating views	→ Nov-12, 2022	R1
M4	→ Testing initial release	→ Nov-30, 2022,	R2

7] Technical process :-

→ Front end :- HTML, CSS, PHP

→ Back-end → PHP, MySQL

→ Android App :- Java on SDK

The blog is contributed by Mahdia & Fahim

Quality Characteristics of good SRS

1] Correctness :- It is covers all the requirements of users / expectation

2] Completeness :- Indicated all requirement including number of page's. Where all functional & non-functional requirements are applied.

3] Consistency :- If there is no conflict with any requirement.

4] Unambiguousness :-

→ Called when SRS has 1 interpretation

→ use ER Diagram, Buddy checks etc

5] Ranking for importance & stability :-

An identifier mark can be used to indicate the rank or stability.

6] Modifiability :-

→ changes in the system

→ should be properly indexed & cross-referenced

7] Verifiability:-

→ If every requirement matched or not.

→ System must be user friendly. If not should be avoided.

8] Traceability:-

→ Should be able to trace test cases.

→ "" to "" design Component & code segment.

9] Design Independence:-

→ SRS should not include any implementation details.

10] Testability:-

→ SRS should be written in a way so that it is easy to generate test cases and test plans from SRS.

11] Understandable by the customer:-

→ user might be able to understand normal user but not the expert in computer science.

→ So, formal notations, symbols should be

avoided. And language should be easy & clear.

12] Right level of abstraction:-
IF SRS is in requirement phase
→ Requirement details are needed
→ For Feasibility ⇒ Fewer details can be used.
→ The level of abstraction varies.

Segment - 05

Project Management

Good management can't guarantee project success. However, bad management usually results in project failure.

(*) Success criteria for project management:- (might varied from project)
→ Deliver software to the customer ~~in the~~ at the agreed time
→ Overall cost in between budget.
→ Deliver S/W with what customer wanted
→ Maintaining happy & well-functioning ~~Dede~~

development team.

Very Important

Differences between other Engineering & S/w engineering :/ Intangibility :-

1) The product is intangible (ধোরে দেখা যায় না)

=) Construction of a bridge / building can bee seen through eyes, so we can get the problem easily & can fix it.

=) But S/w development is different. It is hard to find error. As it can't be touched or seen. So, need to rely on others to produce evidence that they can use to review the progress of the work.

(2) Large S/w project are often 'one-off' project :-

Large S/w projects are different. Even highly skilled manager can't track the problem. Often the experience of him does not ~~work~~ work to find the errors.

(3) S/w processes are variable & Org. specific:-

⇒ In other types Engineering there are specific methods. There any specific method can be adopted to construct these.

=) But in S/w Engineering there is no specific structure or method. It varies from project to project. As some projects are school system, some are related to banking.

As a result the late project submission, over budget, behind schedule, Software systems are often new & technically innovative.

S/W Project Manager Job:-

varies from place to place, project to project.

1] project planning (প্রজেক্ট ম্যানেজার responsible থাকে কোন রিসোর্স, Time schedule, progress monitoring ...)

2] ~~Rep~~

2] Reporting (customer কে progress সম্পর্কে report করা, Detailed project report লেখা।

3] Risk management (project affect করতে পারে এমন। Monitor risks & take action).

4] People Management (Have to choose people, form a team to get better performance)

5] Proposal writing (First stage of S/W project is this to win a contract. Include cost, schedule, justifies. It's a skill can gain by experience)

Risk Management :-

Anticipating risk which might affect project schedule or the quality of the S/W being developed. And then taking action to avoid these risks.

3 risks :-

1] project risks (Affect project schedule or resources.
Ex! Loss of an experience designer.

2] product risks (Affect the quality or performance

3] Business Risks (Affect: Organization developing/ purchasing a s/w.
Ex! Competitor launched a new product)

→ Algo if an experienced person leaves it is a risk. And it falls into Business risk.

Risk management process :-

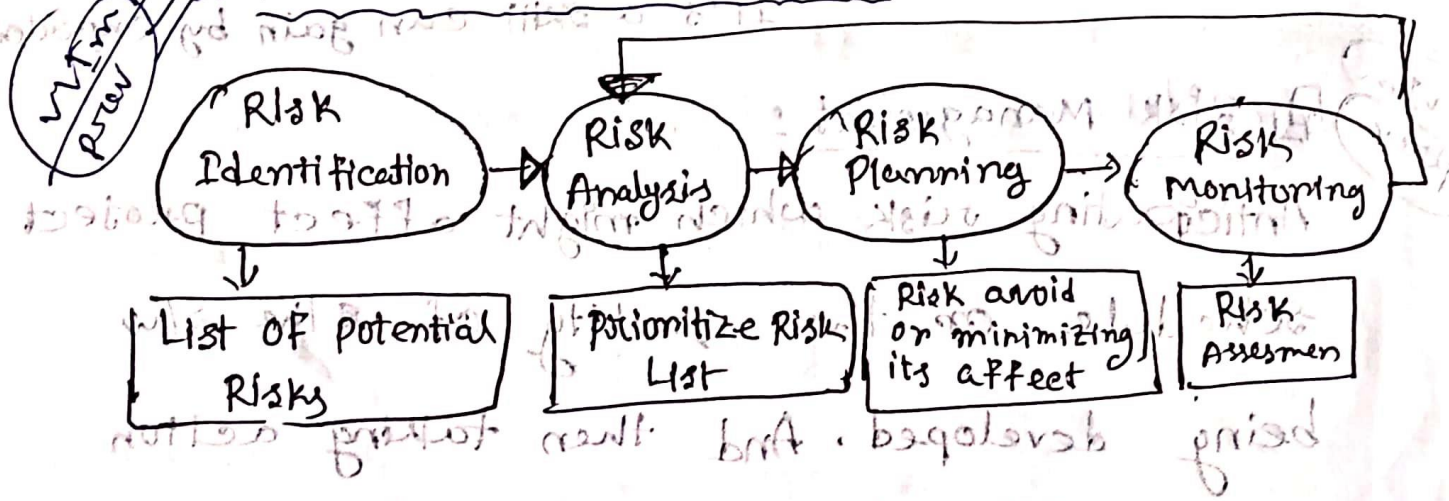


Fig :- Risk Management process.

Risk Identification:-

Can be a team work or project Manager can do it alone.

6 types:-

1] Technology risks:- S/w or H/w will be used to develop the system

2] People risks:- people who are with the development team.

3] Organizational risk:- organizational environment where S/w is being developed.

4] Tools risks:- S/w tools which are used to develop.

5] Requirement change:- customer & process of managing

6] Estimation Risk:-

→ The time needed to develop S/w is underestimated

→ The rate of detect repair is "

→ The size " S/w is "

Risk Analysis:-

Identification first then this step. Have to be done with experience.

1] Risk might be assessed as ~~very low~~

→ very low (<10%)

→ Low (10-25%)

→ Moderate (25-50%)

→ High (50 - 75%)

→ Very High (>75%)

2] The effect:-

(i) Catastrophic (Threaten the survival of the project)

(ii) Serious (would cause major delays)

(iii) Tolerable / Insignificant

Risk Table:-

Risk	probability →	Effects
Organization financial prblm force reductions in the project budget (7)	Low	Catastrophic
It is impossible to ~~recruit~~ recruit staff with the skills for the proj. (3)	High	Catastrophic
Key staff is ill during important time of proj (4)	Moderate	Serious
S/w components needs to be repained before they are used (2)	Moderate	Serious
Software tools can't be integrated (9)	High	Tolerable
Customer fails to understand the impact of requirement changes (11)	Moderate	Tolerable
Code generated by code generation tools is inefficient (8)	Moderate	Insignificant

3] Risk planning :-

After prioritizing risks the key risks have been identified.

Risk	Strategy
Organizational Financial problems	Showing senior management that how project is helping to grow business so cut-off budget will not be cost-effective.
Recruitment problems	Alert customer to potential difficulties & about delays.
Staff Illness	Reorganize team so that people understand each other's job
Defective components	Replace defective component with new one.
Requirement changes	Derive info to asses requirement changes.
Org. restructuring	prepare document showing how it is contributing good to the business
Database performance	Buy Higher DB.
Underestimated Development Time	Buy new components & use a program generator.

There are 3 categories for strategies:-

1] Avoidance strategy:-

probability of the risk that will arise will reduce.

→ Defective components.

2] Minimization strategy:- The impact of the risk will reduce

Ex:- Staff illness

3] Contigency plan:-

You are prepared for the worst & have a strategy in place to ~~det~~ deal with it.

Risk monitoring:-

Regularly need to change; Also need to think about whether or not the effects of the risk have changed.

Risk Type	potential Indicators
Technology	Late delivery of H/w component, many reported tech problems.
people	poor staff morale; poor relationship
Organizational	Lack of action by senior management
Requirements	Changes in requirement, Customer complaints.

[4] Managing people:-

Good software engineers are not necessarily good people manager. As a project manager, should be aware of the potential problems of people management and should try to develop people management skills.

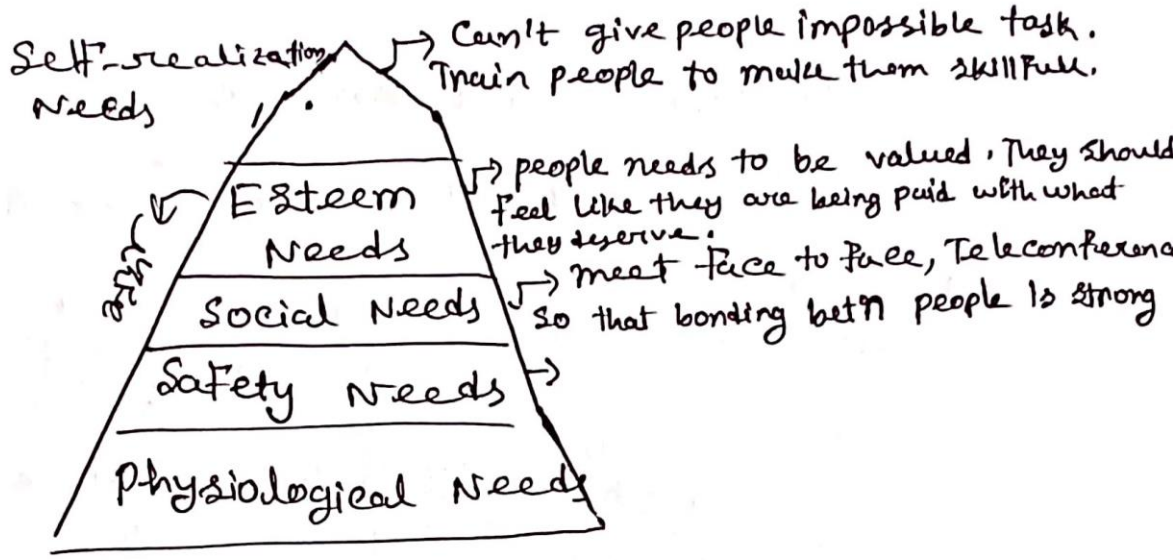
1] Consistency:- In a project team everyone's work should be highlighted or respected so that none feel they are undervalues.

2] Respect:- All members of the team should be given an opportunities. If some people don't fit into a team then they can't continue. But no need to jump at early stage.

3] Inclusion :- Everyone's proposal is considered

4] Honesty:- Always have to honest about myself.

(i) Motivating people



-> Human needs Hierarchy

5] Teamwork!-

A group should not be having more than 10 members.

to] The benefits of creative a cohesive group are!-

1] The group can establish its own quality standard

2] Individuals learn from & support each other

3] Knowledge is shared

4] Refactoring & continual improvement is encouraged.

Segment-3

Project planning

3 Stages in a project life cycle!-

(1) At the proposal stage (Bidding time. Can you do it or not)

(2) During the project start-up phase!-
(who will work, How team will be formed, How resources will be allocated. More info than proposal stage).

(3) Periodically throughout the project-
In the light of experience you created the plan. Better the plan will reduce time & cost.

In proposal stage no need to have full plan. After winning need to replan.

During bidding need to work with budget.

Three main parameters to compute the cost of S/w Development project:-

(1) Effort cost (The cost of paying S/w engineers & managers);

(2) H/w & S/w cost including maintenance. (Normally use commodity Hardware, relatively cheap. But S/w cost can be significant as we use licenced S/w).

(3) Travel & Training cost: (Low cost)

As. (E-meeting & other S/w users can reduce this cost.

Software pricing

→ S/w pricing is the cost of development plus profit for the developer.

Factors:-

→ Market Opportunity:-

A development org. may want a low price S/w. A low profit on one project may give the org. the other projects, which may profit later.

-2] Cost estimate uncertainty:-

→ If organization don't know the actual price then they might increase the price & above its normal profit.

Contractual Terms:-

If user want the source code & developer agree then the price be less than if the source code is handed over.

Requirement volatility:-

If requirement is changed, An org. might lower the price to wint contract.

(After winning the contract high prices can be charged.

Financial Health:- Developers in financial problem might lower the prize. cause cash flow is more important than profit in difficult economic times.

2] Plan Driven Development

Development process is planned in detail.

1] project plans:-

Schedule to work. The plan should identify risks to the project and the S/w under development.

1] Introduction

2] project organization (The people & their role)

3] Risk Analysis.

4] H/w & S/w ~~req~~ resource requirements.

5] Work breakdown (Breakdown the project)

6] project Schedule (Time needed to reach milestone)

7] Monitoring & reporing mechanisms.

(ii) The planning process:-

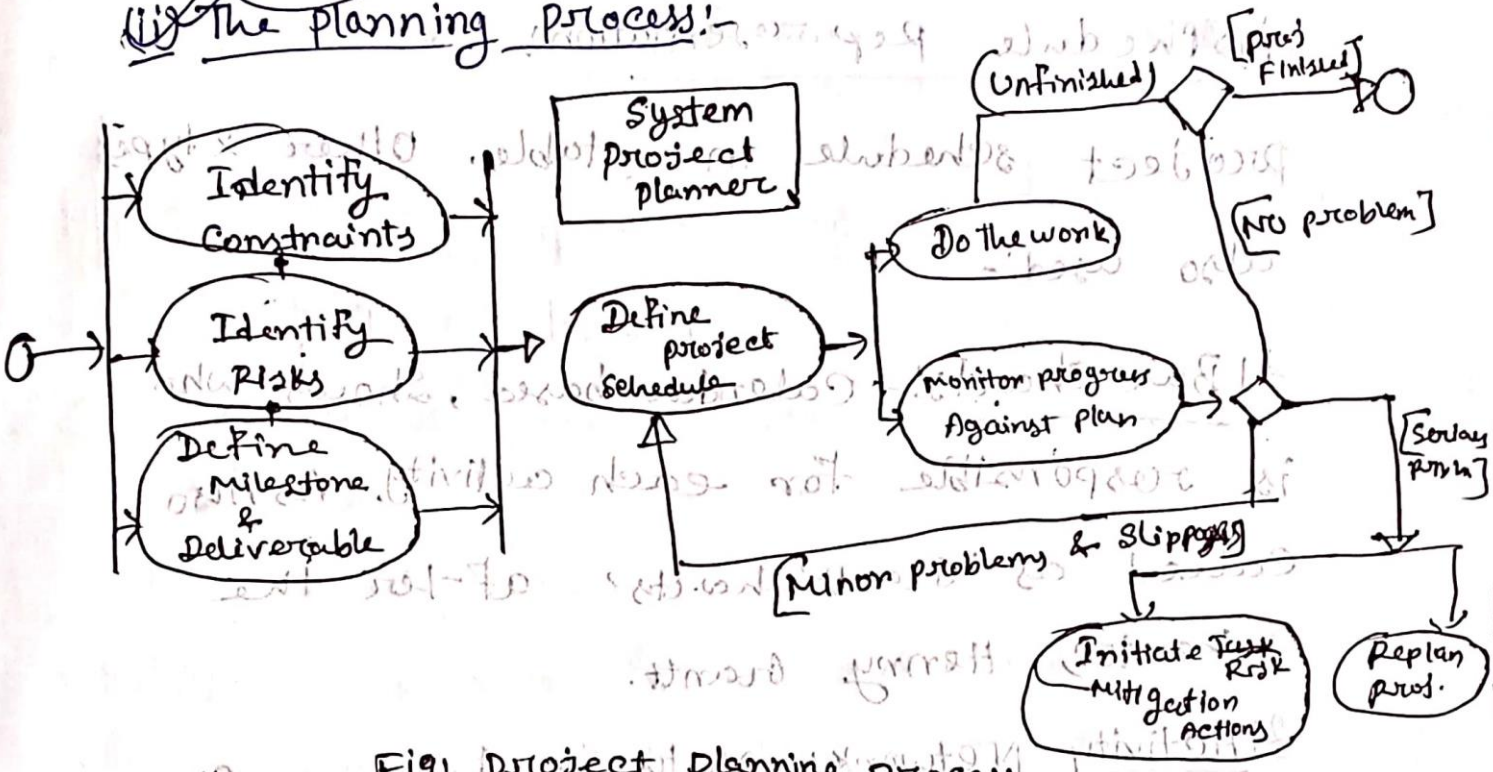


Fig:- project planning process

(iii) project scheduling:-

How the project will organized. And ~~to~~ when which tasks will be executed. Who & when will work. How ~~as~~ many people in a team etc are ~~the~~ in schedule.

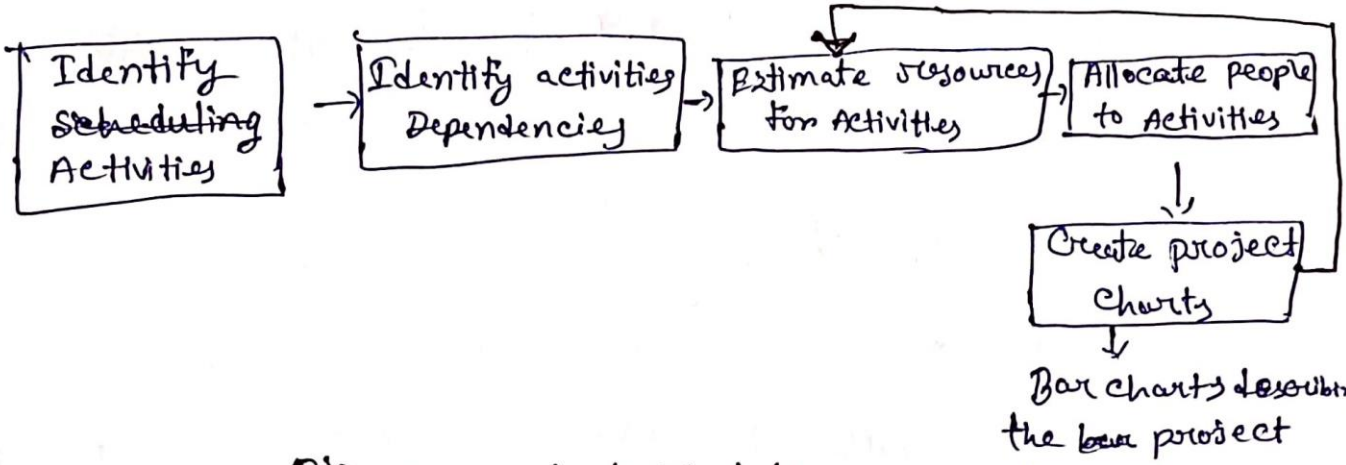


Fig:- The project scheduling process.

A. Schedule Representation:-

project schedule in a table. Other 2 types also used:-

1. **Bar Charts:-** Calender based, shows who is responsible for each activity. Also called as 'Gantt Charts' after the inventor, Henry Gantt.
2. **Activity Network:-** Network diagram. Shows dependencies between the different activities.

B. Milestone:- production of a project deliverable.

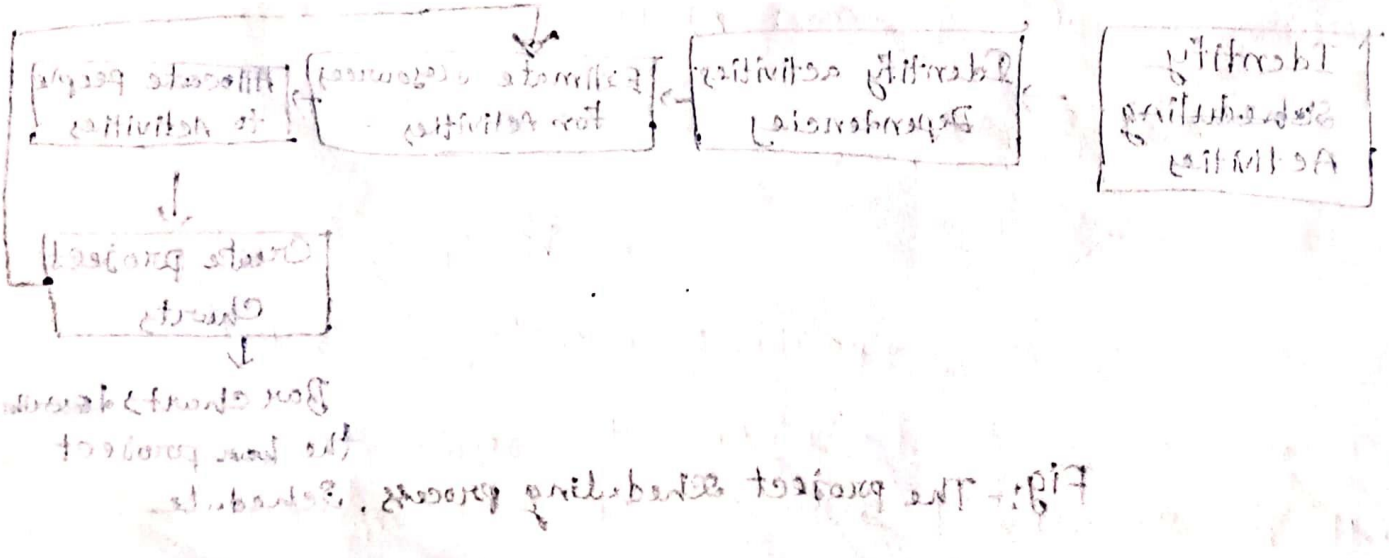


Fig:- The project scheduling process.

Segment-6
(Software Testing)

→ After building a S/w we test it with artificial data to check errors, anomalies, information about the program's non-functional attributes.

Testing process has 2 distinct goal:-

① If S/w meets its requirement or not. For generic S/w products it means that there should be tests for all System Features. plus combination of this features.

② Discover situation in which the S/w behaves incorrectly.

"Testing can only show presence of errors, not their absence."

Model of the S/w testing process:-

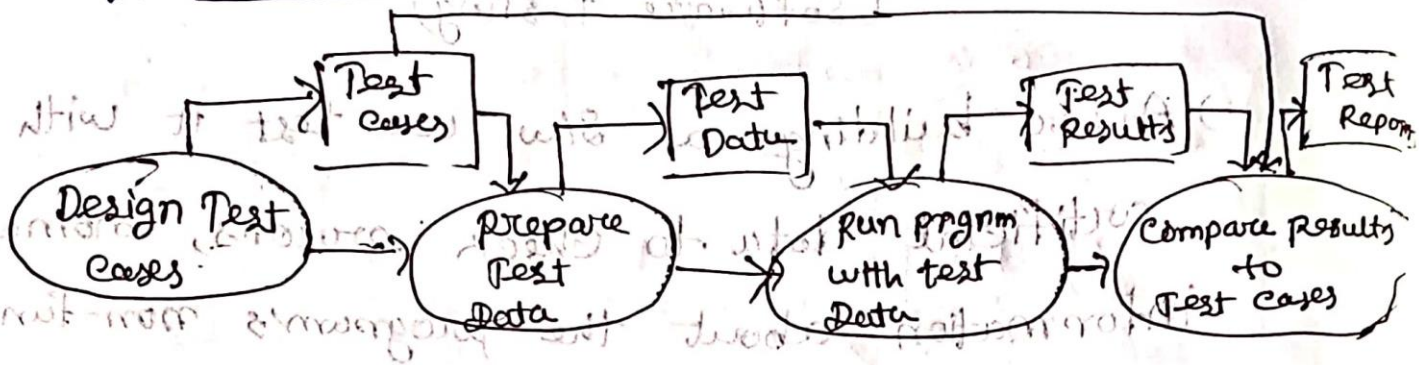


Fig: Mode of S/w Testing process

Commercial S/w system has to go through 3 stages:-

1) Development Testing:-

System designers & programmers are likely to be involved in the testing process.

2) Release Testing:-

The aim is to check if the system meets all requirements.

3) User Testing:-

- ~~Accepted~~
- Acceptance Testing:-

one type of user testing where the customer use & give Feedback if the

system need anything more or not.

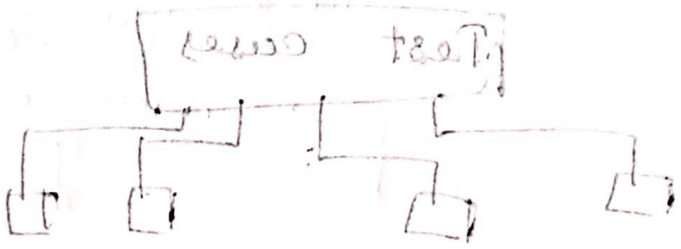
4] Development Testing :-

ज्ञि system develop करतु समय ये testing येत.

1] Unit Testing :- Individual pg unit or object classes are tested.

2] Component Testing :- Several individual units are integrated to create composite components.

3] System Testing :- All of the components are tested.



Choosing Unit Test Cases:-

→ The test cases should show that when used as expected & you are testing does what it is supposed to do.

→ If there are defects in the component, these should be revealed by Test cases.

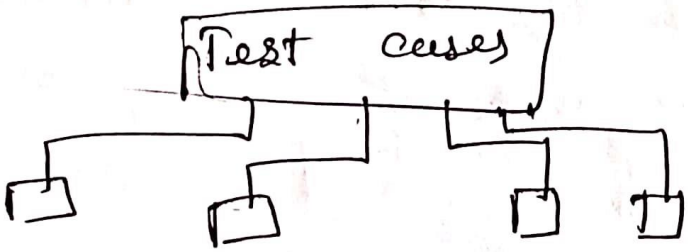
Choosing Test cases:- 2 steps:-

(i) Partition Testing

(ii) Guideline Based Testing

Component Testing:-

Several interacting objects make this.



System Testing

Completely new component which need is a new version need to test the integrated system.

It obviously overlaps with component testing but there are two important differences.

1] During system testing :-

New component which is separately developed may get integrated with the system. And then the complete system is tested.

2] Component dev. by diffent team members :-

To get a good result some companies involve a separate testing who have no involvement in dev. or design.

VVIm prev

TDD (Test Driven development)

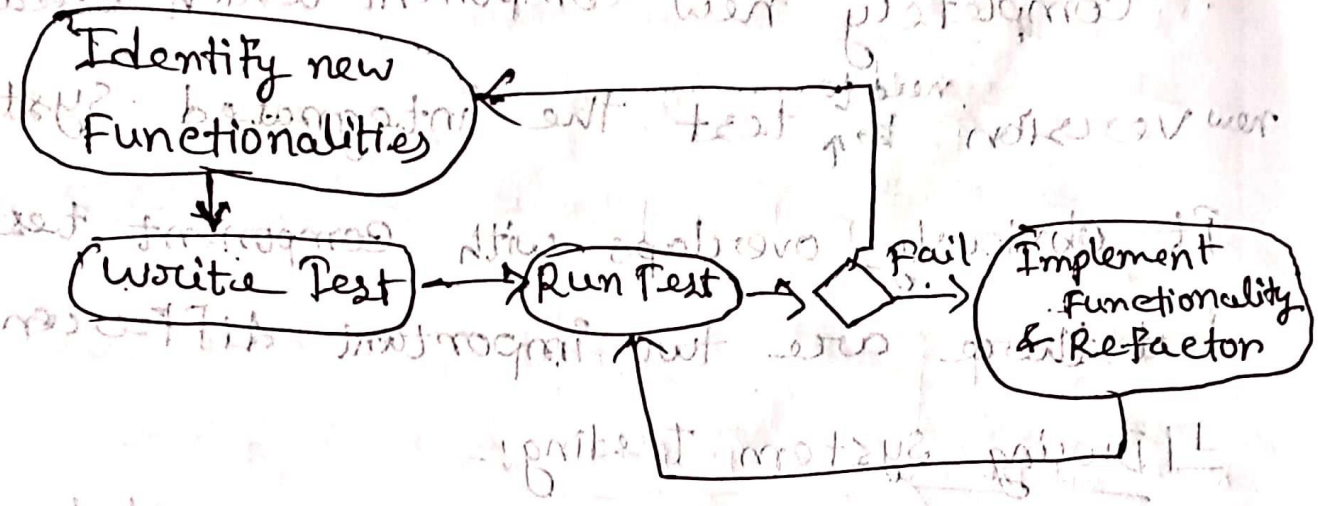


Fig: TDD

→ Interleave Testing & Code Dev.

→ You can't move to the next test cases until you pass the previous one

→ part of agile mode (Extreme prog.)

Fundamental process:

①

② Automated Testing Environment: Junit Environment.

Benefits of TDD:-

1] Code coverage (প্রতি segment এ একটি Test case তৈরি করা। যাতে system কি মতো কাজ করে কিনা দেখা যায়। এতে খুব ভালো system test হয়)

2] Regression Testing (progrnam Develop এর সময়ে সময়ে Test করা। যাতে system এ bug আছে কিনা ধরা যায়)

3] Simplified debugging (Test Fails. use debugger)

4] System documentation

Release Testing :-

A process of testing a particular release of a system for the use of outside of dev team. Mainly for customers & users.

2 Important things:-

1] Those who are not in main team will build this.

2] The main thing is to check if all require-

ments are matched.

It is usually a black-box testing.

Also calls Functional testing.

(1) Requirement based Testing:- (requirement should be testable)

(2) Scenario Testing

In which way system might be used

(Story telling)

(3) Performance Testing:-

(when completely implemented)

~~Unit~~ User Testing

→ Alpha Testing:- users test S/W in the development place

→ Beta Testing:- Release of S/W is made.

→ Acceptance Testing:- Customer decide to Keep or give

Therefore the figure is give below:-

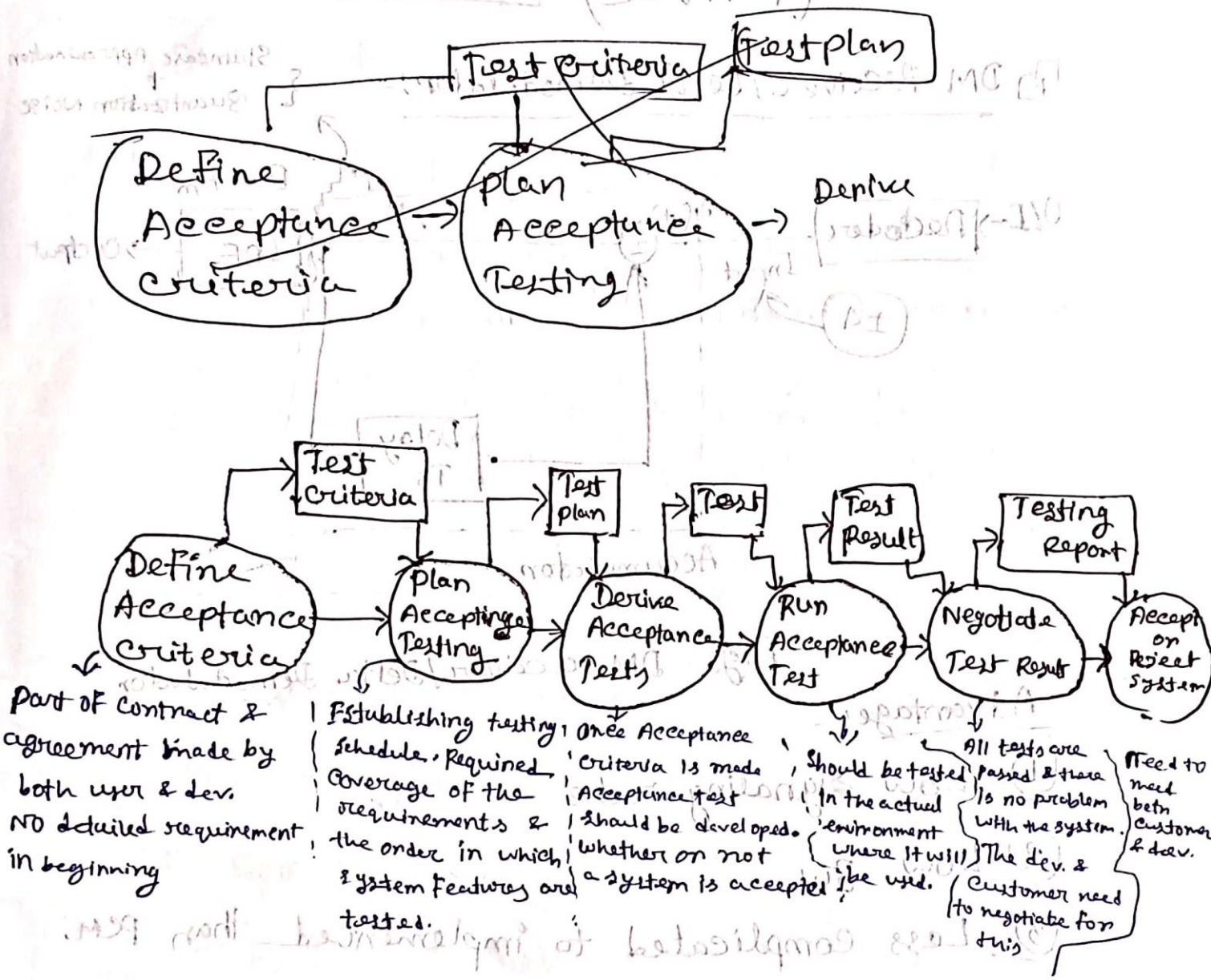


Fig:- 6 Stages of Acceptance Testing process.