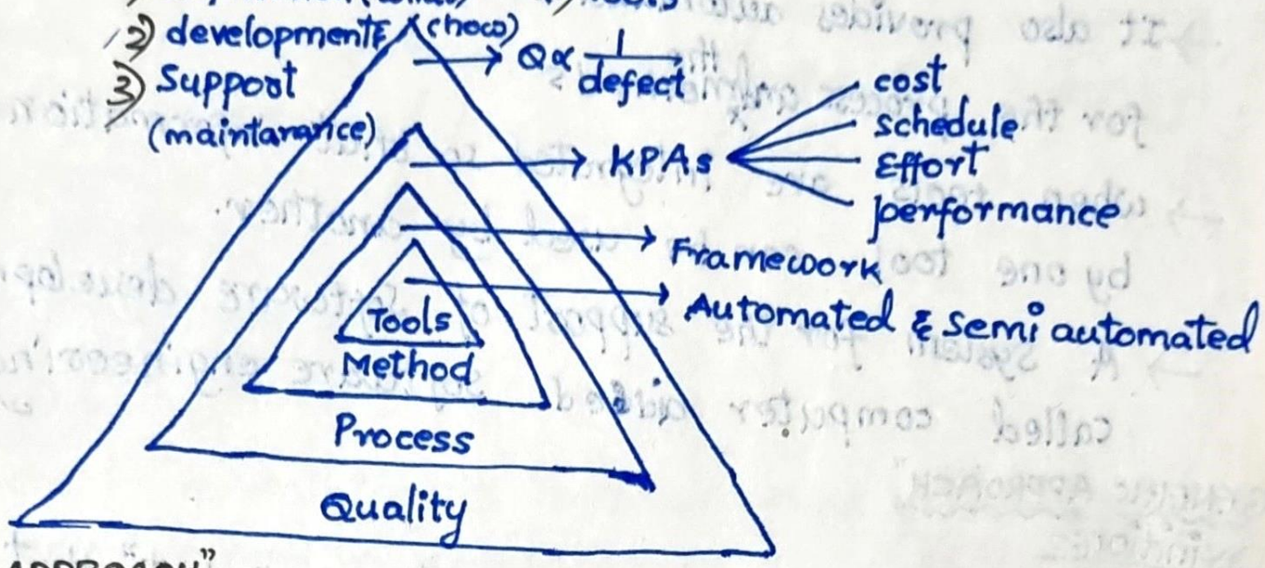
Software Engineering

Software Development approach

1. Layred Approach → 1) Quality
2. Generic Approach → 2) Process
   3) Method
   4) Tools

1) People
3) product
4) project
2) process

General
1) definition (what)
2) development (how)
3) Support (maintanance)



"LAYRED APPROACH"

Quality :-

→ Quality (layer) focus on high quality software. The quality is defined as the conformance that is explicity stated requirements in the document with respect to the behaviour of the product.

Process :- (framework) + (planning)

→ process layer concentrates on planning. planning means estimation of the software in terms of cost, schedule, effort, size, manpower, performance etc.

→ process is defined as framework for a set of KPA (key process areas) that must be established for effective delivery of sofware engineering technology.

ethod :-

→ method layes concentrates on the process model to develop the software.

→ It also provides the technical details for software development, methods in compass a broad array of the tasks that include requirement analysis, design, program construction, testing and maintaince (support) (Implementation, coding)

Tools:-

→ Tools layer concentrate on the 4th ~~generation~~ Generation technology to implement the project with minimum effort (optimized → minimum cost time and manpower).

→ It also provides automated or semi automated support for the process and the methods

→ when tools are integrated so that information created by one tool can be used by another.

→ A System for the support of software development is called computer aided software engineering (CASE).

"GENERIC APPROACH"

Definition:-

→ The definition stage concentrates on "what" that means what information is processing, what functionality and performance is desired, what system strategy is expected, what constraints are important and what validation criteria is important.

→ Analysis and requirement gathering takes places in this stage that means bussiness object, input domain and output domain are defined in this stage.

→ Validation means a system write correct or incorrect that means write conditions are injected into the software development process at entry level. And after the implementation of design, it is verified

Development:-

Development focuses on the "how" that means how data structures are defined, how many interfaces are defined, how to convert procedure into machine readable formats & how many test cases are developed to cover the systems from errors.

03/02/23

support:-

→ support concentrates on maintainance

→ After the delivery of the product to customer site when it is in the Operation, the support stage is required to maintain the software from uncovered erros, platform changes, new requirements and deteriote condition

→ support is divided into 4 types

1) correction
2) Adaption
3) Enhancement
4) prevention

correction :-

correction is maintaining the software from uncovered error It is called corrective maintainance

Adaption :-

Adoption maintains the software from platform changes (memory, processor, operating system). It is called adaptive maitainance

Enhancement :-

Enhaneement maintaing the software from new functionality changes.

prevention :-

Prevention maintains the software for frequent changes in the software(preventive maintainance or reengineering).

→ software development process

Four important entities for s/w Development (Four Ps)

1. people: (developers team)+(managers)
2. process:
   framework, planning, effort, manpower, cost, time, size
   - framework (process model)
   - effort → how much work is done in a given time
   - size → depends on line of code
3. Product: complexity + performance
   - complexity → more interfaces, connectivity b/w interfaces
   - performance → more complexity ↓ performance

→ complexity (data structures)

4 Project: (Quality)

"SOFTWARE PROCESS MODEL" (model for software development) → framework

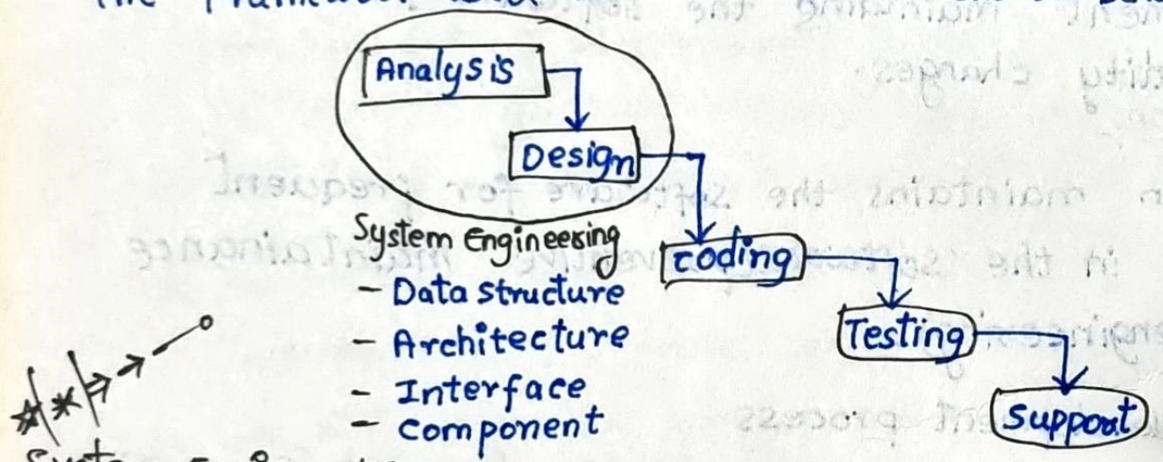
- conventional process model
  - classic Life cycle (waterfall model or Linear sequential)
  - prototype (based upon feedback not on requirements)
    - open end Prototype (evolutionary)
    - close end prototype (throw away)
  - RAD (Rapid Application Development)
- Evolutionary Process model
  - Incremental
  - spiral
  - component Based Development model
  - Agile

WaterFall Model :- (least performance model) difficult to change when requirement changes (poor)

→ This model is suitable when customer requirements are more clear and when patience level is very high.

→ Because in this model final product is available after a long schedule.

The Framework used in this model is shown below.



System Engineering

→ Software (engineering) is a subset of the system. Therefore before developing the software there is a need of analysing the application domain to finalise the business objective. (goal of the project)

(i) Analysis -

Based on the goal of the project, gather the requirements from different sources to develop the software in order to satisfy the objectives.

→ Input and output domain are defined at this stage. The output of analysis stage is software requirement specification

(SRS) document. ——— this document consists of goal of the software, functional and non functional requirements.

(ii) Design -

In this stage high level design is converted into the low level design. That means customer statements are represented into technical statements.

→ In this stage four actions are performed

a) Data structure Design - It defines the data object, attributes, relationships required in software

b) Architecture Design - It represents the high level structure of the software that means inflows and outflows are defined

c) Interface Design - It identifies the logical connections required within the software.

d) component Design - It identifies the functions required to develop the Software

(iii) coding -

this stage translates the procedural description (algorithm) into machine readable format by using 4th generation technique

(iv) Testing - (unit & system Testing) (Black & white box testing)

→ In this stage different test cases are developed and Implemented to cover the structural & functional error in the developed program (Syntax error) (logical error)

→ Structural testing covers the error after careful evaluation of the developed program (white Box Testing)

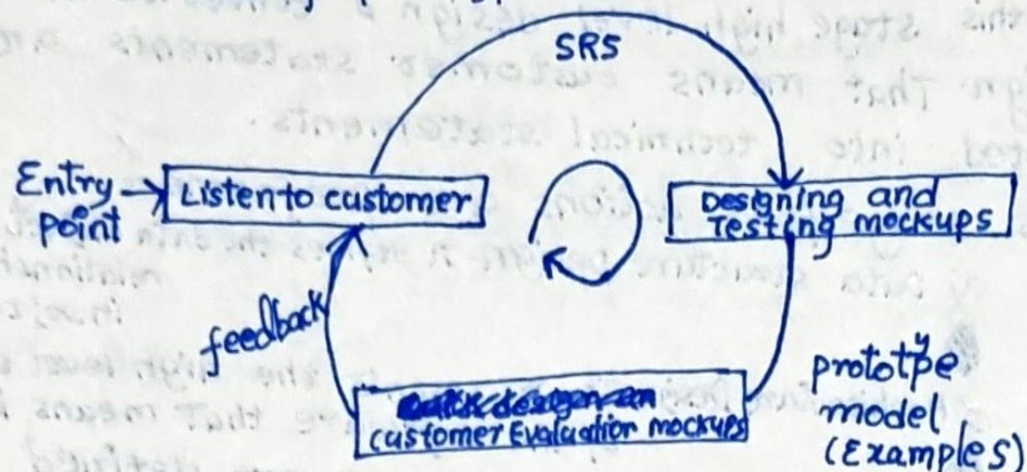
→ Functional Testing covers the errors based on the behavious of the developed program (Black Box Training)

(v) Support - maintainance, upgradation etc.

After the delivery of the software at the customers site when it is in the operational mode support phase is required to maintain the software from the uncovered errors, platform changes and requirement changes

PROTOTYPE MODEL (feedback mechanism will be there)

→ When customer requirements are not clear to finalise the SRS there is a need of prototype model



① Entry point:-

→ Listen to customers. Entry point to the prototype model is Listen to customer to gather the requirements

→ Based on the requirements the developer provide Quick design, coding and Testing operation to develop the prototype model

→ the prototype model is demonstrated to customer and their feedbacks are taken

→ Based on customers feedback the SRS document may be modified or finalized. It is an Iterative process continues up to customers satisfaction

→ After getting the positive feedback from the customer the corresponding SRS documents is finalized and it is forwarded to the design team.

It is of two types

1) Evolutionary (open end) → same design used, same process, same srs.
2) Throw away (closed end) → if difference in same design.

RAD (RAPID APPLICATION DEVELOPMENT) (coupling & cohesion)

→ This model is suitable when customer requirements are more clear but the schedule is very short

→ In this model the project is divided into different modules and assign them to different teams

→ over the short period of the time, all independent modules are available so to integrate them to generate the final softwar
→ modularity is associated with 2 different factors
1) coupling
2) cohesion

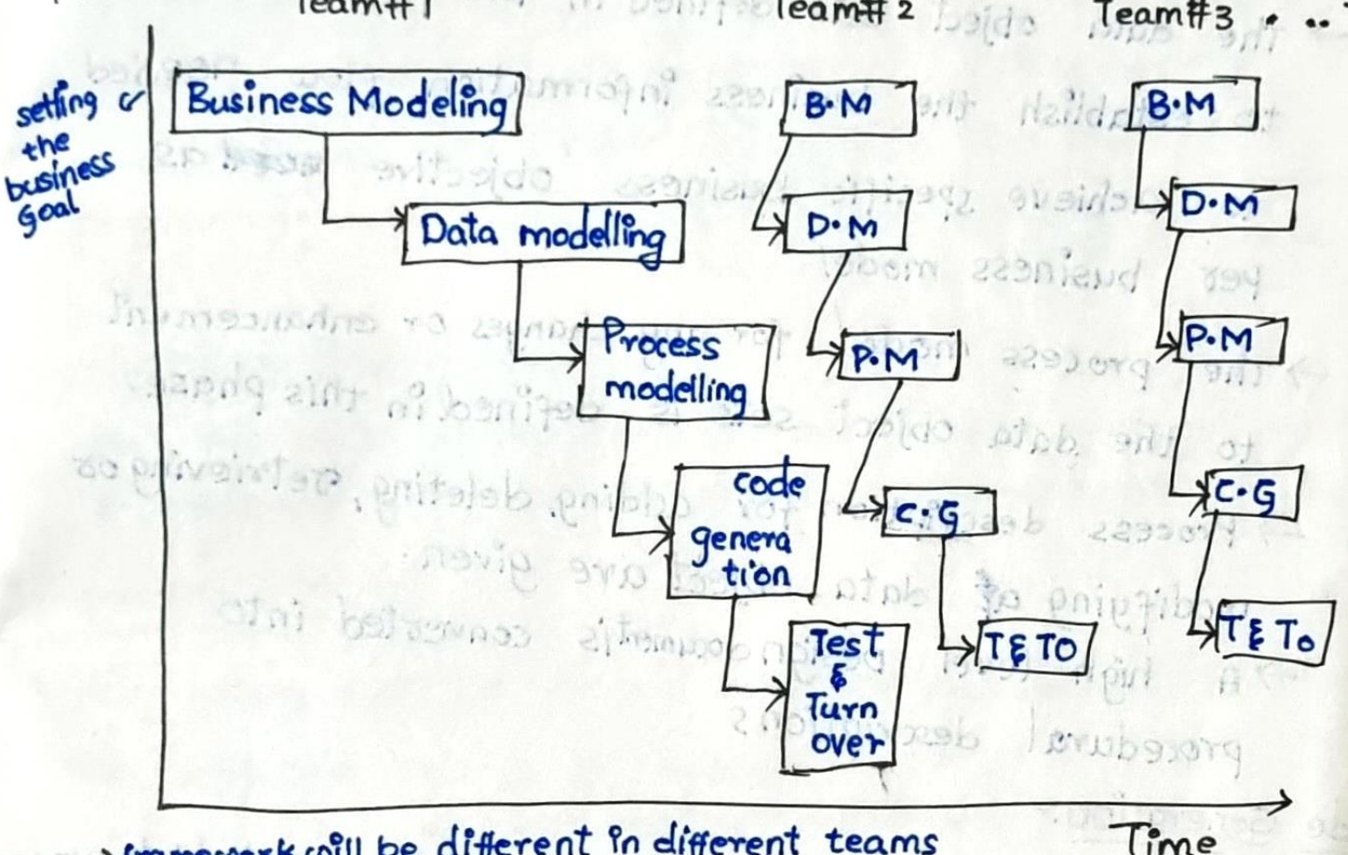
coupling :- It means the measure of "to what extent the modules are dependent on each other."

cohesion :- It means the measurement of "to what extent the modules are independent of each other."

effective modularity maintains high coupling & low cohesion

10/02/23

RAD continuation



→ framework will be different in different teams
→ efficiency will be more as same work is done in many teams

→ Business Modeling

→ requirements from the customers

→ Designed in terms of flow of information & distribution of information b/w various different business channels

→ A complete business analysis is performed to find the vital information for business.

→ Based upon the application nature levels goals are established and based on the goals of the software corresponding input & output domains are defined.

Data modelling

→ The information gathered in the business modelling acts as input for data modelling which is rieviewed & analysed to create or to form the sets of data objects vital for business

→ attributes and relation among data objects are defined here.

→ In DM high level design is created to represent the structure of software.

Process modelling

→ The data object sets defined in DM phase are converted to establish the business information flow needed to achieve specific business objective as per business model

→ The process model for any changes or enhancement to the data object sets is defined in this phase.

→ Process description for adding, deleting, retrieving or modifying a data object are given.

→ A high level design document is converted into procedural descriptions

Code Generation:-

→ In code generation algorithm is converted into program by using appriopriate background programming language

Test and Turnover

Testing & Turnover are used for various plans to implement various test cases to uncover the errors and to develop the final product.

→ All teams are maintaining the same framework to develop the modules. Therfore at the end of lifecycle over a short period of time, multiple error free modules

(components) are available.

→ Finally, all these modules are integrated to develop the final product. (Integration testing)

when each & every module is testing → unit testi
all modules are combined testing done is
Integration testing.

→ In RAD, only Integration testing is countable. Timing will be reduced. when we have less time, small team, and requirements more clear then RAD will be used.

Note:-

→ RAD projects follows Iterative and Incremental model and have small teams comprising of developers, domain experts, customer representatives and other IT resources working ~~progressively~~ progressively on the their components and prototype

(remaining models - self study)

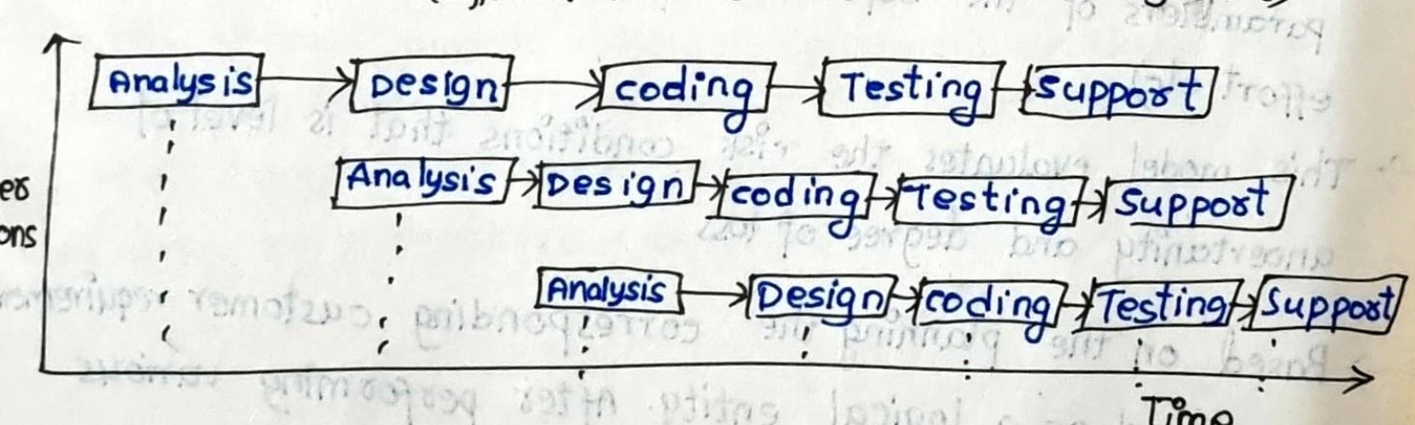
Evolutionary Process Model

→ when the customer requirements are keep on changing over a period of time, evolutionary models are used.

Incremental Model

→ This model uses the classic life cycle framework to develop the software release by release

(different versions are released time by time)



Spiral Model:

→ This model framework consists of 6 different activities

1) customer communication: (output: SRS)
2) planning: estimation
   - → cost
   - → schedule (time)
   - → size (Lines of code & delivery of source Instruction (DSI), TSI
   - → Effort
3) Risk Analysis: Risk
   - Degree of Loss
   - Level of Uncertainity
4) Engineering: Designing
   - → Data structure
   - → Architecture
   - → Interface design
   - → component design
5) construction Release: coding and Testing
6) customer Evaluation: Feedback

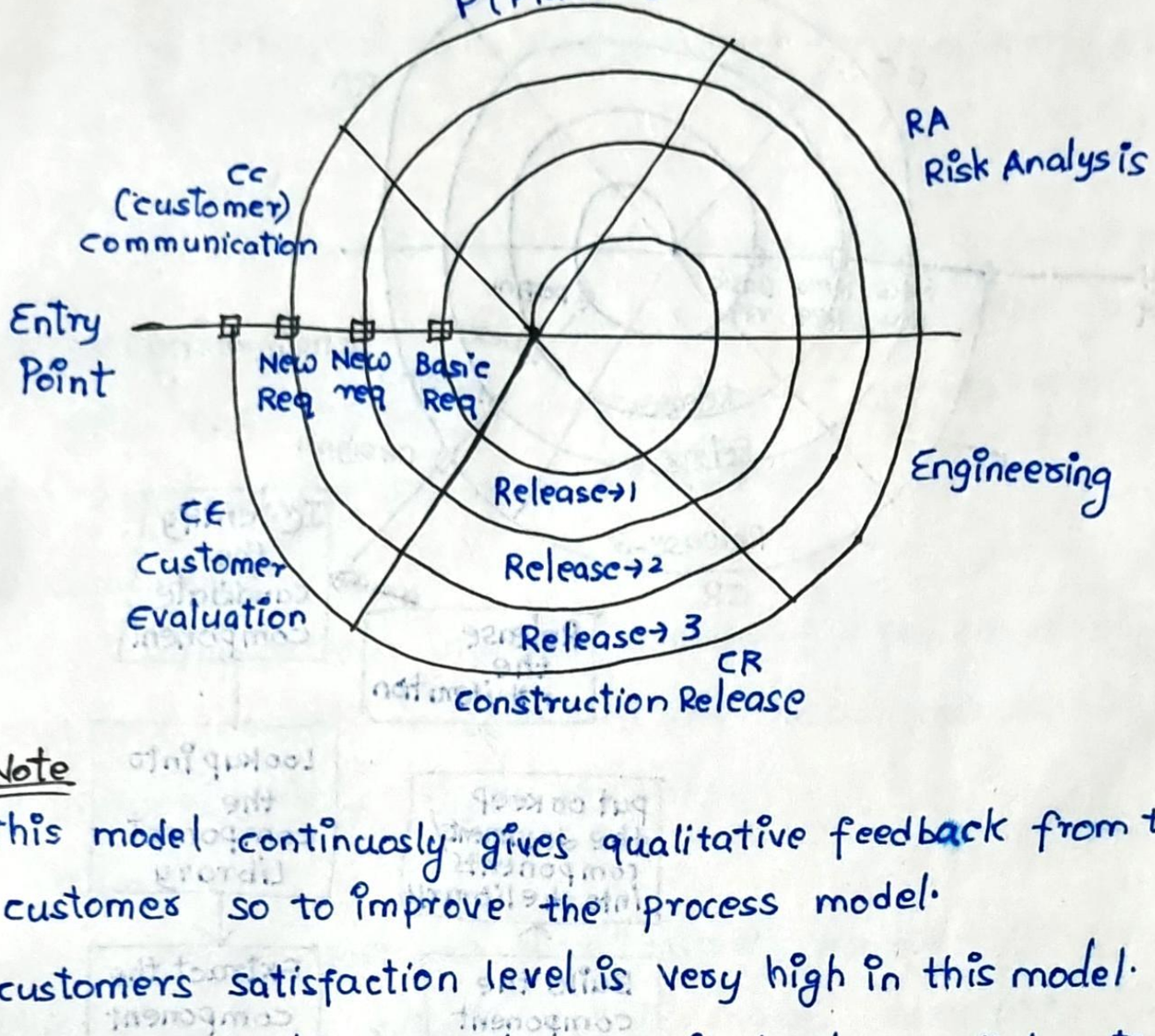
→ In this model the project estimations takes place after finalising the final document to trace the different parameters of the software (complexity, cost, productivity, effort etc)

→ This model evaluates the risk conditions that is level of uncertanity and degree of loss

→ Based on the planning the corresponding customer requirements are developed as a logical entity. After performing various test case implementation, the product is on customer site.

→ This model performs announce the quantitative feedback from the customers for the better performance of the software

fig:- win-win spiral model



→ Note

→ This model continuosly gives qualitative feedback from the customer so to improve the process model.

→ customers satisfaction level is very high in this model.

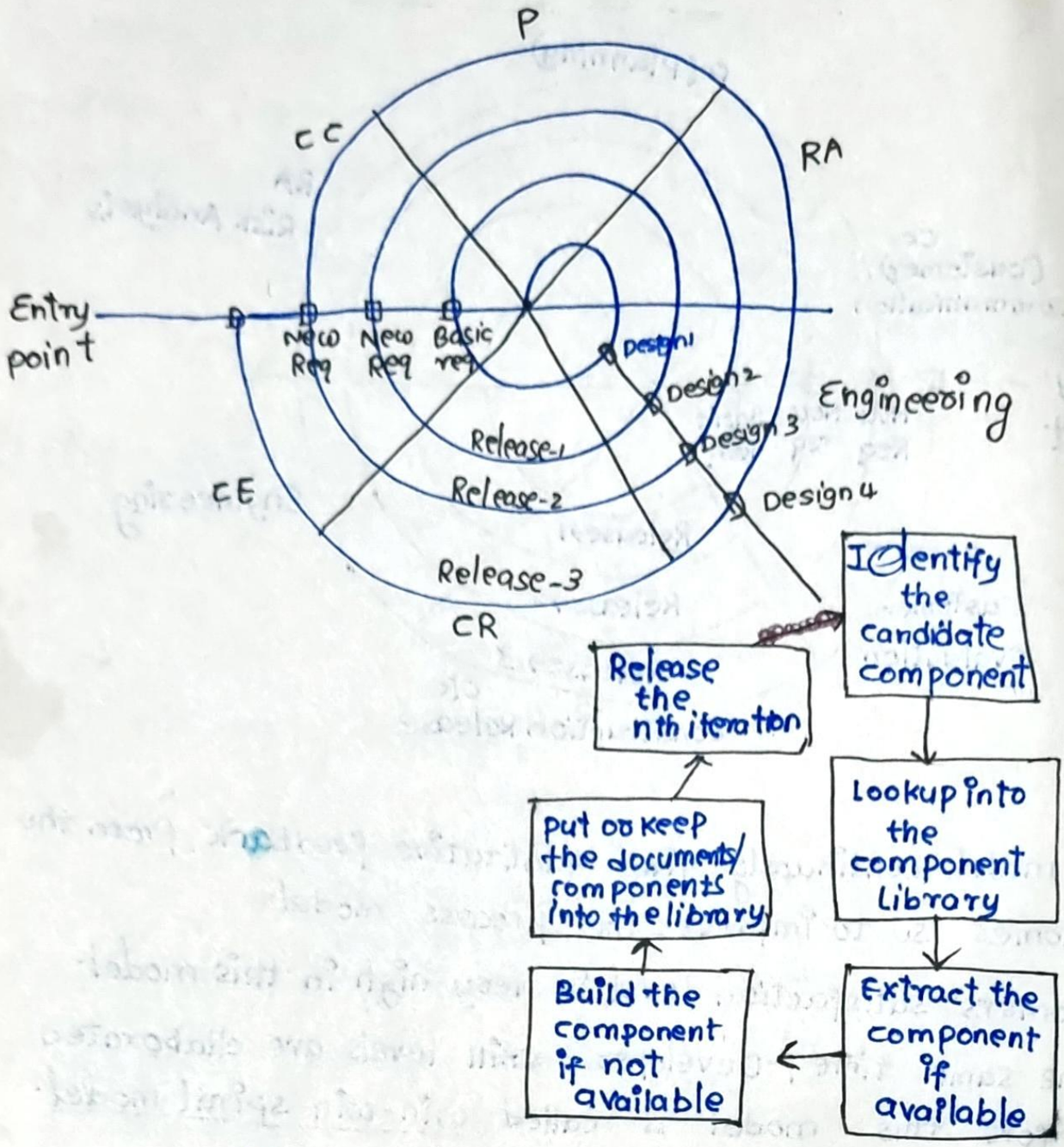
→ At the same time, Developer skill levels are ellaborated Therfore this model is called win-win spiral model.

Component Based Development Models:-

Ditto diagram of above

→ In this process models, different component modules are developed which in turn reduce the time and cost. as the component can be used in future projects as they are being developed from time to time.

BLANK



Agile (self study)

16/08/23

SOFTWARE PROJECT MANAGEMENT:- (SPM)

★ Goal:-

1) To enable a group of developers to work effectively towards the successful completion of a project.

2) Software project management (SPM) is an art and a proper way of planning and supervising software projects

3) It is a procedure of managing, allocating, & Timing resources to develop software that fulfills the ~~user~~ requirements

★ SPM complexities:-

1) Invisibility:-

→ software remains invisible, until its development is

complete & it is operational.

→ Anything that is invisible, is difficult to manage & control.

→ Invisibility of s/w make its difficult to asses th progress of the Project is a major cause for complexity of manag a software project.

2) Changebility

→ Because of s/w part of any system is easier to chan as compared to Hardware part, the software part is one that gets most frequently changed. It is specially true in the later stages of the software development.

3) Complexity:-

→ Due to the inherent complexity of the functioning of a software product in terms of the basic parts making up the software many types of risks.

4) Uniqueness:-

→ Every software project is usually associated with many unique features or situations. This makes every project much different from other projects.

5) Exactness of Solution:-

→ Requirement of Exact confirmaty of the parameters of the project introduces additional risks & contributes to the complexity of managing the software project.

6) Team oriented & Intellectual Intensive work:-

→ In a software development project, the life cycle activities not only highly Intellect intensive but its member must typically interact & review & interface with several other members, constituting another dimension of complexity of software product.

— Blank —

Project Manager's Responsibilities :- (PMR)

1) A project manager is responsible for planning, design, execution, monitoring, controlling and clogure of a project

2) SHE He is used to manage the risks and minimize the uncertainity. (He is the responsible person for everything)

3) The JOB Responsibilities of a PM ranges from invisible activities like building up of team moral to highly visible customer presentation

4) Most managers take responsibilities for -
   - Project Proposal Writing.
   - Project Estimation.
   - Scheduling.
   - Project staffing
   - Project Monitoring and control
   - Software Process tailoring.
   - Software Configuration management.
   - Risk Management, managerial report writing & presentation
   - Interaction with client.

5) Three skills that are most critical to successful project management are:-
   - Knowledge of the Project management Techniques.
   - Decision taking capabilities/ Decision making capabilit
   - Previous experience in managing similar Projects.

6) we can broadly classify project managers responsibilities into two major categories
   - project planning
   - Project monitoring and control.

17/02/22

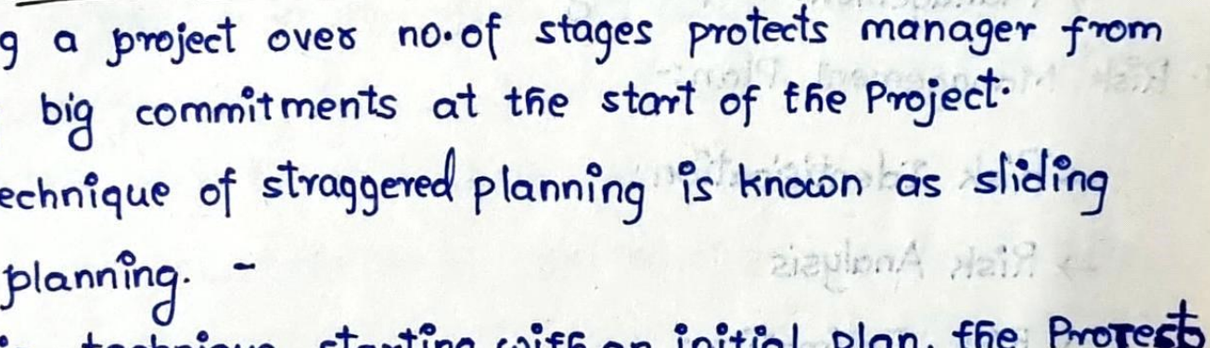
SPM (software project management):-

★ Project Planning:-

→ cost
→ Duration
→ Effort
→ Scheduling
→ Staffing
→ Risk Management
→ Miscellaneous plan (Quality Assurance plan & Configuration Management Plan)

Among all the estimation, size estimation is the first activity.

"PRECEDANCE PRIORITY ORDER AMONG PLANNING ACTIVITIES"



SLIDING WINDOW PLANNING:-

→ planning a project over no. of stages protects manager from making big commitments at the start of the Project.

→ This technique of straggered planning is known as sliding window planning.

→ In this technique starting with an initial plan, the Project is planned more accurately over a number of stages.

★ SPMP (software project management planning):-

→ The organisation of SPMP document is

1. Introduction:-
   → objective.
   → Major Functions.
   → Performance Issues.
   → Management & Technical constraints.

2. Project Estimate:-

→ Historical Data Used.

→ Estimation Techniques used.

→ Effort, Resource, cost & Duration.

3. Schedule:-

→ Work Breakdown structure.

→ Task network requirements representation.

→ Gantt chart representation.

→ pert chart representation.

4. Project Resources:-

→ Human Resources (People).

→ Hardware & Software.

→ special Resources.

5. Staff Organisation:-

→ Team structure.

→ Management reporting.

6. Risk Management Plan:-

→ Risk Identification.

→ Risk Analysis.

→ Risk Estimation.

→ Risk Recovery.

7. Project Tracking and Control Plan

→ Matrix to be tracked.

→ Tracking Plan.

→ Control Plan.

8. miscellaneous Plans

→ Process tailoring.

→ Quality Assurance Plan.

→ configuration management Plan

→ validation and verification

→ System Testing Plan

→ Delivery Installation & maintainance plan

☆ metrics for Project size Estimation

→ It helps the project manager to further predict the effort & tim which will be needed to build the Project.

→ Various measures for size estimation are:-

(1) LOC (Lines of code) → simplest & (universal) accepted measure

(2) FPs (Function Points)

(3) Number of entities in ER diagram

(4) Total no. of process in DFD (Data Flow Diagram)

LOC ⇒ cocomo model (Lines of code concept)
→ also measures lexical quantities of program

FPs
→ the no. & type of function supported by the software are utilised to find FPC (Function Point count)
→ It is Based on the idea that the software product supporting many features would certainly be of larger size than a product with smaller no. of features
→ The steps in FP analysis are:-
- count the no. of function of each proposed type
- compute the UFP (unadjusted Function Points)
- Find TDI (Total Degree of Influence)
- Compute VAF (Value Adjustment Factor).
- Find the FPC. (Function Point Count).

☆ PET (Project estimation Techniques) } (Self Study)
1) → Empirical Estimation Technique.
2) → Heuristic Techniques.
3) → Analytical Estimation Technique.

1) Based on educated guess on project parameters unknown

assumes relation among different parameters known

→ The structure of emperical model is

$$E = A + B(ev)^C$$

Effort

A, B, C → Emperically derived components

ev → estimated value

LOC:- To estimate the expected size of the software in terms of LOC there is a need of skill levels into 3 types.

1) optimistic (Sopt)
2) most likely (Sm)
3) pessimistic (Spess)

$$S = \frac{Sopt + Sm(4) + Spess}{6}$$

LOC

Eg: A company needs to develop a software for weather forecasting application. In this development the developers are came out with a range of LOC estimation as 4,600 optimistic codes, 6900 most likely codes, & 8,600 pessimistic codes.

(i) Compute ~~estimate~~ expected size of the software.

(ii) if the s/w development require 8 man per month effort then what is the productivity.

(iii) If s/w development is scheduled for 2 months then what will be the average man power (how many persons required).

(iv) If cost per man month is estimated as 500 dollars what is the total development cost?

Eg2: A company needs to develop a ~~software~~ strategy for the s/w development project for which it has 2 choices of programming language L_1 & L_2

The no. of LOC developed using L_2 is estimated to be twice the LOC required in L_1. The product will have to maintain for 5 years. the various parameters for the company shown below.

(i) man per year required to develop the software.

L_1	L_2
LOC	LOC
10K	20K

(ii) cost per year required to develop the software

L_1	L_2
10 Lakhs (1 million)	7 lakh 50,000 rupees

(iii) maintaince cost per year:-

L_1	L_2
1 Lakh	50,000

1) What will be the total cost required for development and maintainance

2) what is the LOC for L_1 in which ~~also~~ cost of the product using L_1 is equal to the cost of the product using L_2.

EP

→ A software supports with the 5 functional requirements

1) import
2) output
3) no. of inqueries (requirements)
   no. of files (logical components)
   no. of External interface (External components required for computating with the uses)

Based on the above 5 parameters range. of Estimation can calculate the expected value

$$s = \frac{sopt + 4sm + spess}{6}$$

Based on the levels of project complexity there is empirically derived constraints are maintained in the

weighing factors (simple, average, complex).

↓ 3 types

	simple	average	complex
# no. of inputs	3	4	6
# no. of outputs	4	5	7
# no. of inqueries	3	4	6
# no. of files	7	10	15
# no. of external interface	5	7	10

Based on the expected value of 5 parameters and corresponding project complexity level of the ~~project~~ weighing factor we can calculate the count values (FPC).

$$FPC = \sum_{i=1}^{5} w_{ij} \times z_{ij} \text{ weighing Factors}$$

where i = functional requirements
j = project complexity level
w = weighing factor
z = expected functional requirements value.

→ In the FP analysis, 14 parameter factors are required involved directly for size estimation

→ Based on the involvement of non functional requirements. The impact is defined.

Involvement	Impact factors
No Influence	0
if incidental	1
if moderate	2
if average	3
if significant	4
if essential	5

***

Value adjustment Factor (VAF) = $0.65 + 0.01 * \sum_{i=1}^{14} F_i$

FP = Countvalue x VAF

→ consider the software projects with numbers (weighing factors of input, output, enquiries, files and interfaces as 30(4), 25(5), 20(4), 10(10), 5(7) respectively. Assuming the complexity level is average (3). what is the FP of the project. complexity adjustment factor