OOSE - Object Oriented Software Engineering.

UNIT - I - Software process and Agile Development.

Introduction to Software Engineering:- - Ivar Jacobson

Software Engineering

* The Software is a collection of Integrated program
* Instructions and code written by developers
* computer program and Related documents such as Requirements, design model and user manual. [features, functionalities & better performance]

Engineering :-

Engineering is the application of scientific and practical knowledge to Invent, design, build, maintain and improve frameWork processes etc..,.

Definition of Software Engineering:-

* Software Engineering is a discipline in which theories, methods and tools are applied to develop professional software product.
* In Software Engineering the systematic and organized approach is adopted.

The definition of software Engineering is based on two terms,

* Discipline * product.

Defining Software:-

Software product may be:-

1. Generic - Developed to be sold to a range of different customers
2. Custom - Developed for a single customer according to their specification.

Software characteristic:-

* Software development is a logical activity and it is important to under the basic characteristic of software.
  * Software is engineered, not manufactured
  * Software does not wear out → Not damaged.
  * Most software is custom built rather than being assembled from components.

1) Software is engineered, not manufactured:-
   * Software development and hardware development are two different activities
   * A good design is backbone of for both the activities
   * Quality problems that occurs in hardware manufacturing phase can not be removed easily. On the other hand, during software development process such problem can be rectified.

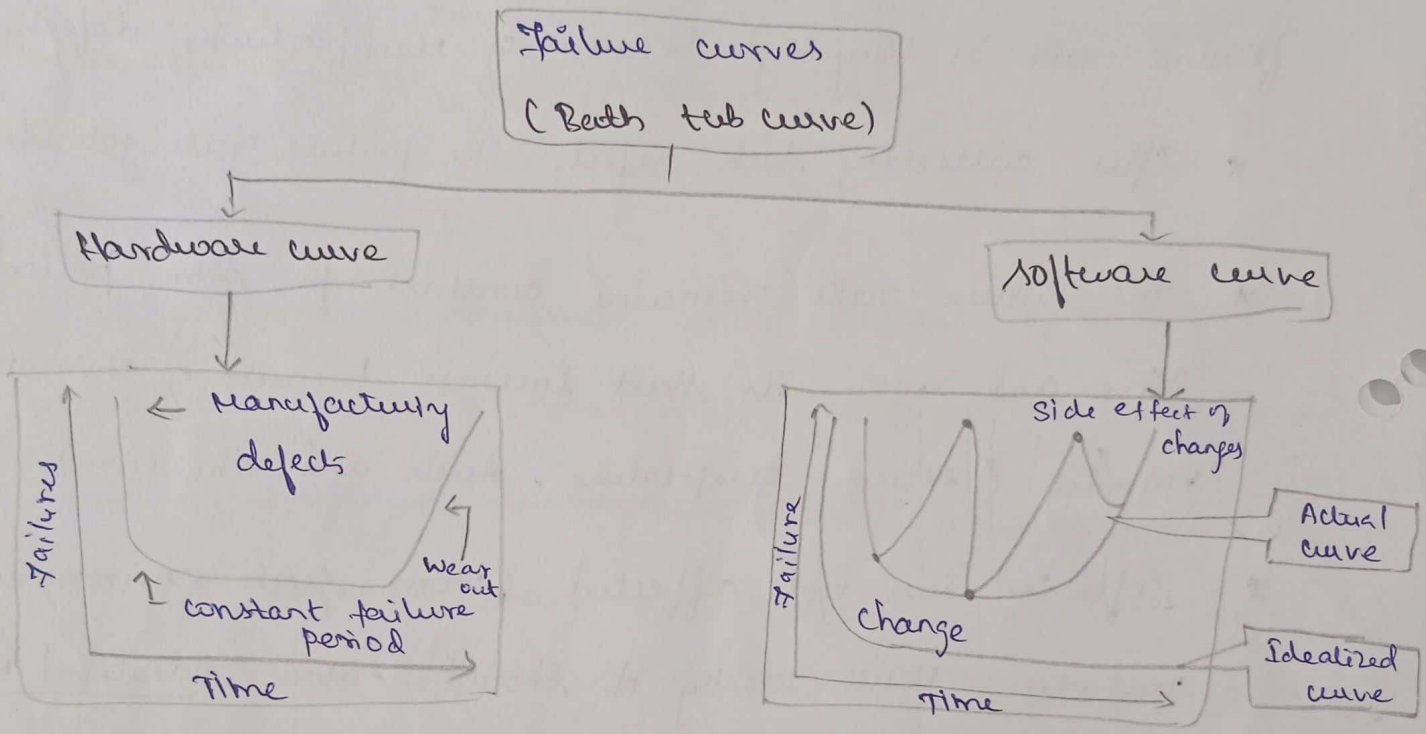
In both the activities, developers are responsible for producing qualitative product.

2) Software does not wear out:-

* In early stage of hardware development process the failure rate is very high because of manufacturing defects.
* After correcting such defects the failure rate gets reduced.
* The failure rate remains constant for some period of time and again its start increase because of environment maladies (extreme temperature, dusts and vibrations).
* Software does not affected from such environment maladies. Hence ideally it should have "idealized curve".
* But due to some undiscovered errors the failure rate is high and drop down as soon as error get corrected.
* Failure rating of software the "actual curve" is shown below,
* Failure curve looks like a spike. → pointed piece of material.
* Another issue with software is that there are no spare parts for software.
* If hardware components wear out it can be replaced

by another components but it is not possible in case of software.

* Software maintanance is more difficult than the hardware maintenance.



Failure curves for hardware and software.

Most software is custom built rather than being Assembled from components:-

* While developing any hardware product firstly the circuit design with desired functioning properties is created.

* The Required hardware components such as ICs, capacitor, and regulars are assembled according to the design, but this is not done while developing software product.

* cost of the software is custom built.
* We look for Reusability of software components
* It is practised to reuse algorithms and data structures.
* Today software industry is trying to make library of reusable components.

categories of Software

Based on a complex growth of software it can be classified into following categories,

1) System software - It is collection of program written to service other program. Some system software are compiler editor and assembler.

   purpose - To establish communication with hardware.

2) Application software - Standalone program that are developed for specific business needs.

3) Engineering / Scientific software - It has been characterized by "number crunching" algorithms.

4) Embedded software - program that can reside within product or system.

* Software can be used to implement and control features and functions for the end-user and for the system itself.

5) Web applications - It consists of various web pages that can be retrieved by a browser.

* The web pages can be developed using programming language like java, PERL, CGI, HTML, DHTML.

6) Artificial Intelligence software - Based on knowledge based expert systems. This software is useful in robotics, expert system, image and voice recognition, Artifical neural network, theorem proving and game playing.

Goals and objectives of software:

While developing software following are common objectives.

1) Satisfy user Requirement
2) High reliability
3) Low maintenance costs
4) Delivery on time
5) Low production costs
6) High performance
7) Ease of reuse.

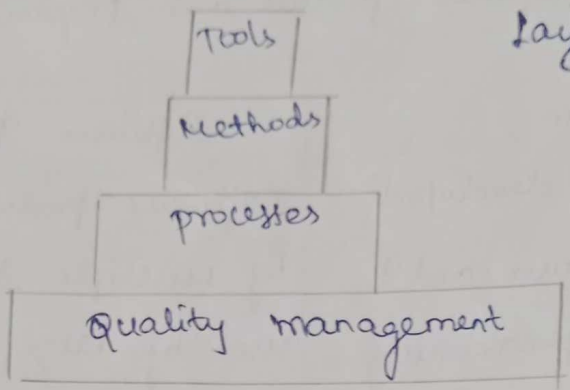
Difference between software product and program.

S.No	Program	Software Product
1.	Programs are developed by individual user and it is used for personal use	Software product are developed by multiple user and its used by large number of people or customer.
2.	Small in size and limited functionality	It consists of multiple program codes, related document such as SRS, design documents, user manual, test case and so on.
3.	Only one person uses the program, Hence there is lack of user Interface	Good graphical user Interface is most required by any software product.
4.	Developed by programmer	Developed by software Engineering.
5)	Ex: program of sorting n elements	Ex - A word processing software.

Layered Technology:- - (AU - May 22 *)

* software Engineering is a layered Technology.
* Various layer on which the technology based are quality focus layer, process layer, methods layer, tools layer.
* quality management is backbone of software Engineering technology.

Layered Technology.



* process layer is a foundation of Software Engineering.
* In method layer is actual method of Implementations is carried out with help of requirement analysis, designing, coding using desired program constructs and testing.
* Software tools are used to bring automation in software development process.

2) Software process (* Important Question)

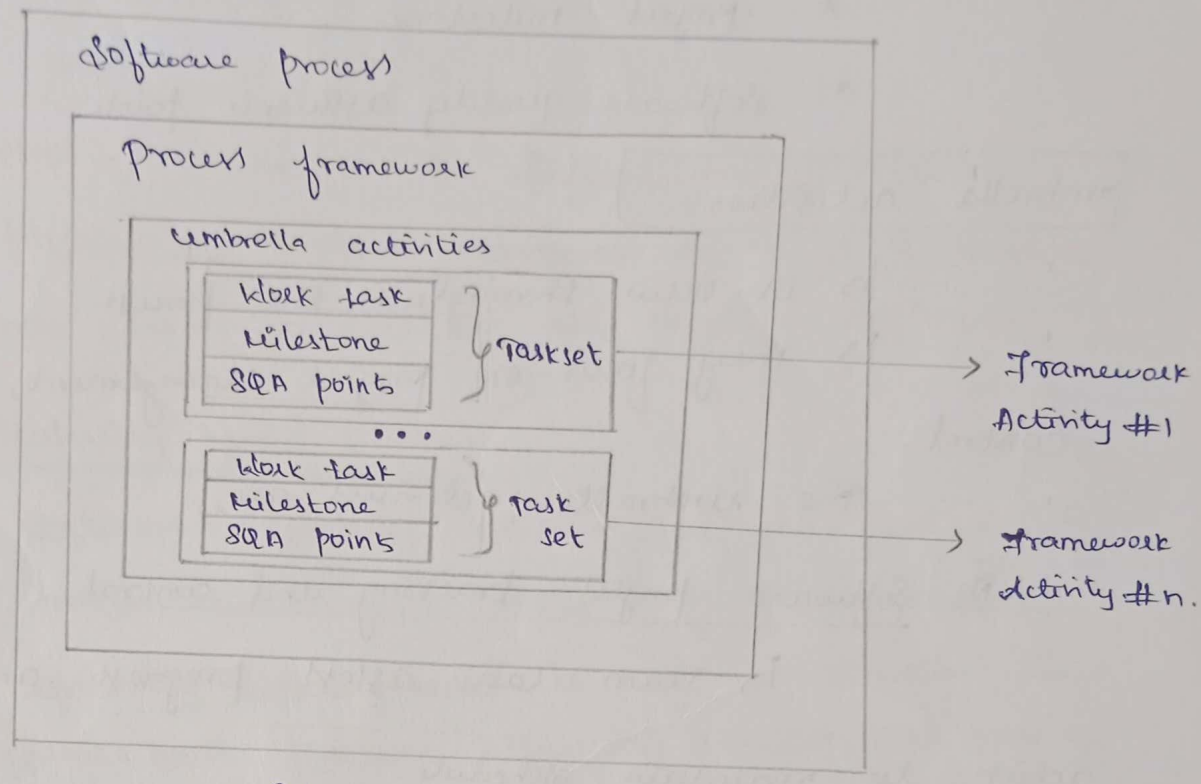
* Software process can be defined as the structured set of activities that are required to develop the software system.

The fundamental activities are

* Specification
* Design and Implementation
* Validation
* Evolution.

Common process framework:

The software process is characterized by process framework activities, task sets and umbrella activities.



Software process framework.

process framework activities:-

* Communication - communicating customer requirement gathering.
* planning - Work plan, describes technical risks, list resource requirement, work product produced and work schedule
* Modeling ↳ Analysis of Requirement and Design.
* Construction - code generation, Testing
* Deployment - The software delivered for customer evaluation and feedback is obtained.

Task sets - The actual work done in order to achieve the software objective.

- It is used to adapt the framework activities and proj team Requirement using

* collection of Software Engineering Work tasks
* project milestone
* Software quality assurance points

umbrella activities

↳ It occur throughout the process.
↳ They focus on project management, tracking and control.

The umbrella activities are,

1) Software project tracking and control
   ↳ Team can assess progress and take corrective action to maintain Schedule.
2) Risk management - May affect project outcomes or quality can be analyzed.
3) Software quality assurance - To maintain Software quality.
4) Formal technical Reviews - To assess engineering Work products to uncover and remove error before they propagate to next activity.
5) Software configuration management - Managing of configuration process when any changes in the Software occur.

6) Work product preparation and production.

The activities to create models, documents, logs, forms and lists are carried out.

7) Reusability Management - It defines criteria for work product reuse.

8) Measurement - process can be defined and collected.

- project and product measures are used to assists the software team in delivering the required software.

Capability Maturity Model (CMM)

* The Software Engineering Institute (SEI) has developed a comprehensive process meta-model emphasizing process maturity.

* The capability maturity model is used in assessing how well an organization's processes allow to complete and manage new software projects.

Various process maturity levels are:

Level 1 - Initial

Level 2 - Repeatable

Level 3 - Defined

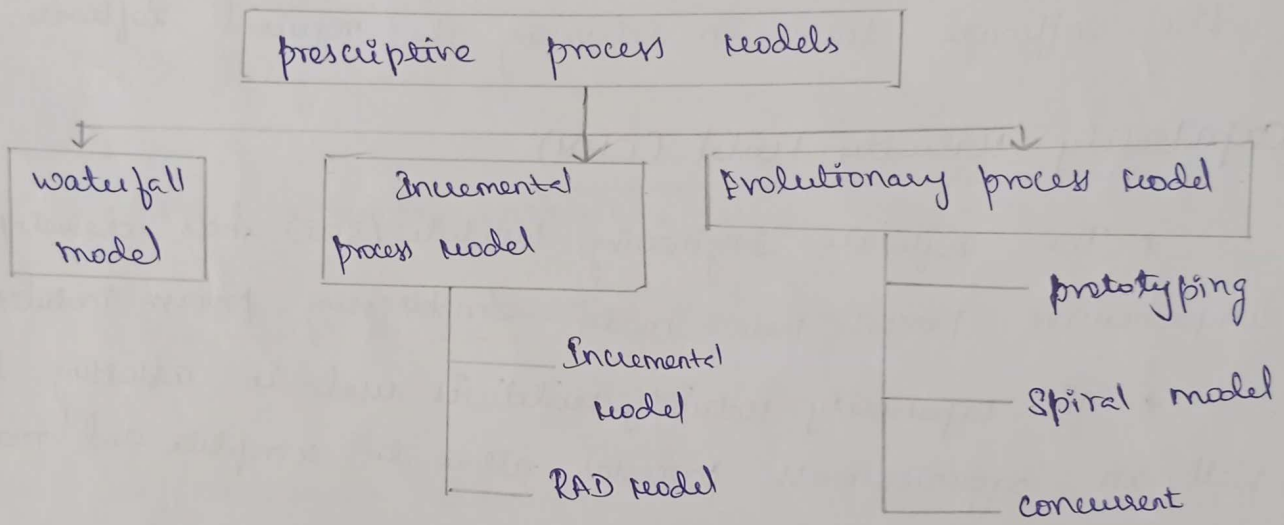
Level 4 - Managed

Level 5 - Optimizing

Thus CMM is used for improving the software project

③ Prescriptive process models.

* **Process model** - It can be defined as the abstract representation of process.
* The software process model is known as Software Development life cycle (SDLC) model.
* These models are called (prescriptive process model) because they are following some rules for correct usage.



Need for process model.

* Each team member will understand - what is the next activity and how to do it.
* process model will be consists of definite entry and exit criteria for each phase.

Waterfall model.

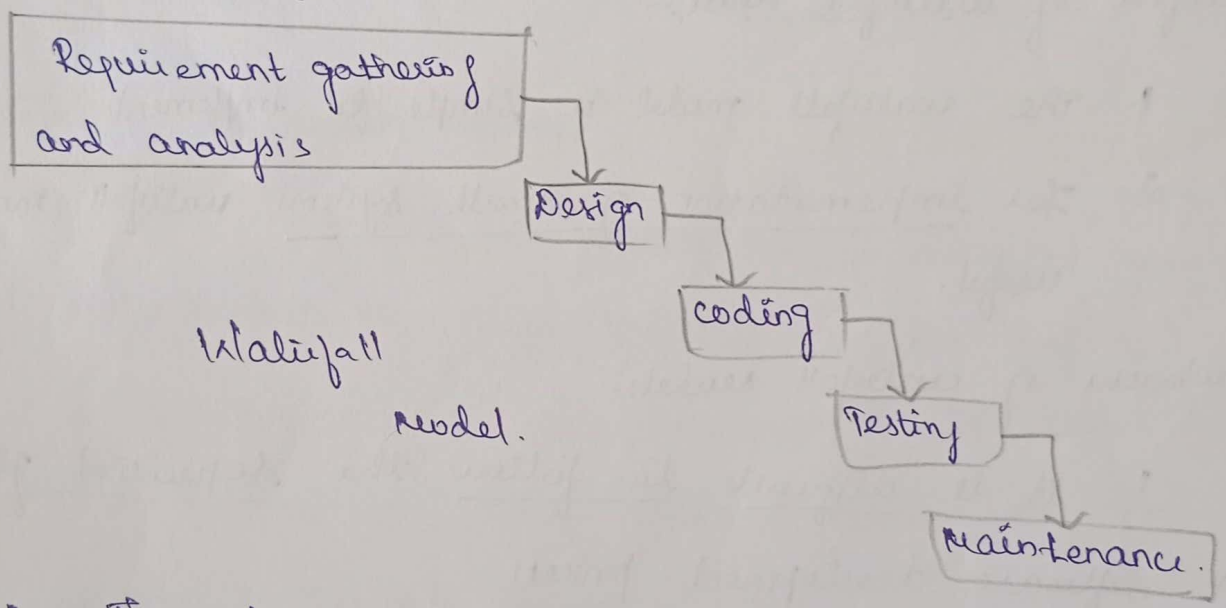
↳ It is also known as "linear - sequential model' or) classic life cycle model.

Waterfall model:-

* The software development start with requirements gathering phase. Then progresses through analysis, design, coding, testing and maintenance.

* The basic requirements of the system must be understood by software engineer, who'll called Analyst.

* The information domain, function, behavioural requirements of the system are,



Waterfall model.

* The design is an intermediate step between requirements analysis and coding.

Design focuses on program attributes such as -

* Data Structure
* Software Architecture
* Interface representation
* Algorithmic details

* coding is a step in which design is translated into the machine - readable form.
* Testing begins when coding is done
* Focus is on logical Internals of the software
* The purpose of testing is to uncover errors, fix the bugs and meet the customer Requirements.
* Maintenance is the longest life cycle phase.

Benefits of waterfall Model:-

1. The waterfall model is simple to implement
2. For implementation of small systems waterfall model is useful.

Drawbacks of waterfall model:-

1. It is difficult to follow the sequential flow in software development process.
2. The customer can see the working model of the project only at the end.
3. Linear nature of waterfall model induces blocking state, because certain tasks may be dependent on some previous task.

Incremental process Model:-

The initial model with limited functionality is developed created for user's understanding about the software product *

Incremental Model:-

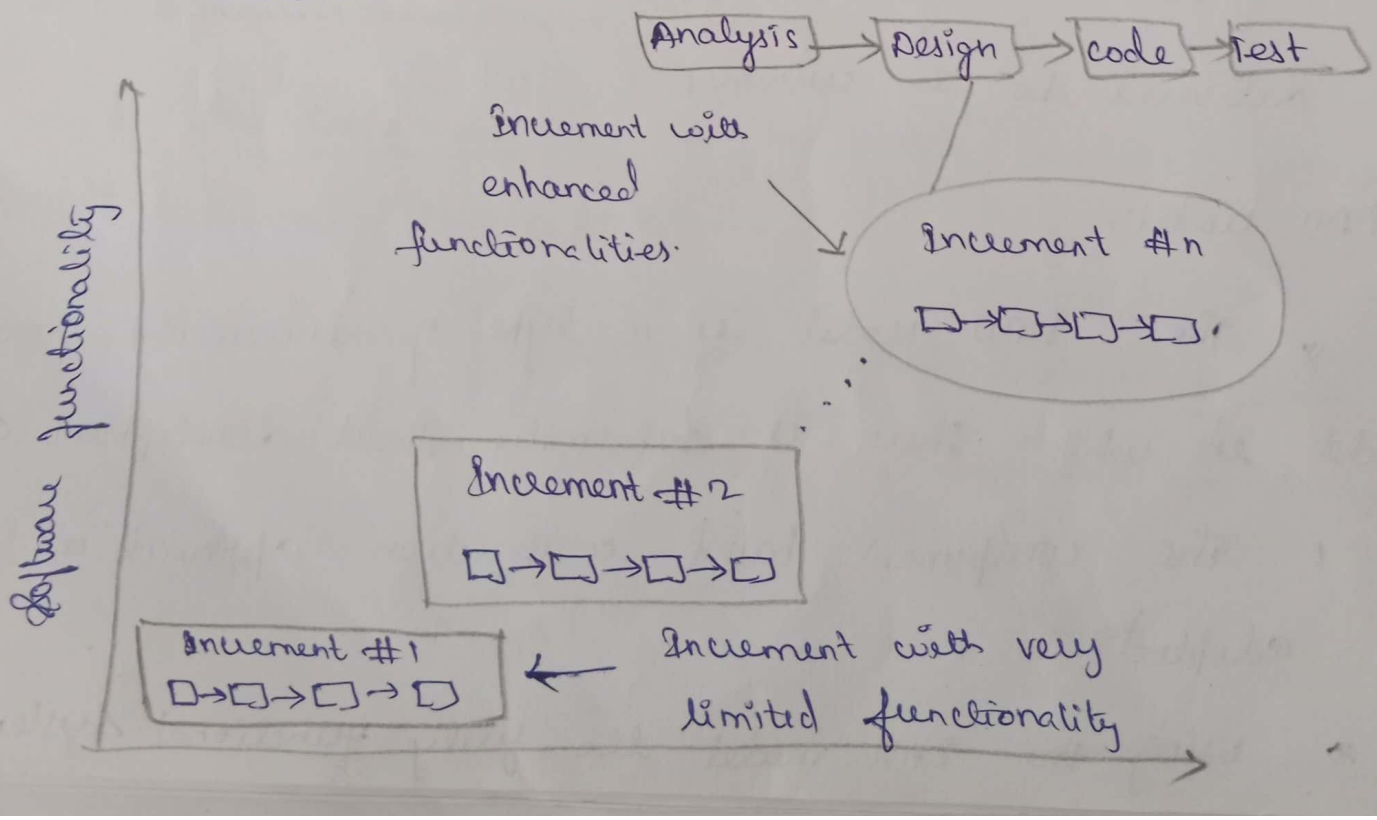
* The Incremental model has same phases that are in waterfall model. (Iterative in nature)

* The Incremental model has following phases.

1. Analysis
2. Design
3. code
4. Test

* Delivers series of release to the customer. These release are called increments.

* More and more functionality is associated with each increment.



The first increment is called core product.

* The word processing software package can be considered as an example of incremental model.

When to choose it?

1. When requirement are reasonably well-defined
2. When overall scope of the development effort suggests a purely linear effort.
3. When limited set of software functionality needed quickly.

Merits of Incremental Model:-

1. The incremental model can be adopted when there are less number of people involved in the project
2. Technical risk can be managed with each increment
3. For a very small span, atleast core product can be delivered to the customer.

RAD Model:-

* The RAD model is a type of incremental process model in which there is extremely short development cycle.
* The component based construction approach is adopted
* Using the RAD model the fully functional systems can

developed within 60 to 90 days.

* Various phases in RAD are Requirement Gathering, Analysis and planning, Design, Build or Construction and finally Deployment.

* In the requirement gathering phase the developers communicate with the users of the systems and understand the business process.

* During analysis and planning phase, the analysis on the gathered requirements.

* During the design phase various model are created. Those models are Business model, data model and process model.

* The build is an activity in which using the existing software components and automatic code generation tool the implementation code is created for software system.

* Finally the deployment of all the software components (created by various teams working on the projects) is carried out.

Drawbacks of rapid application development:-

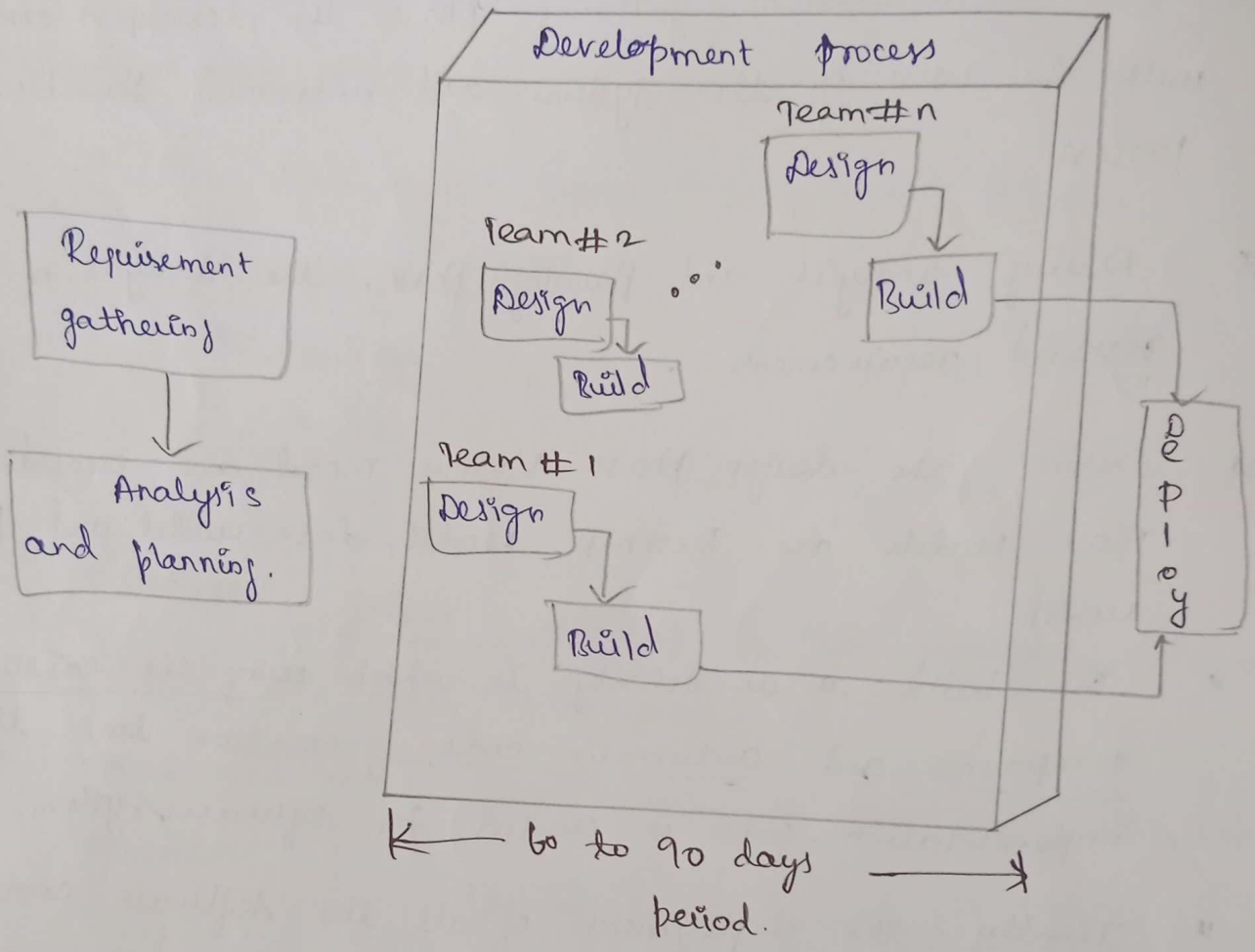
1. It requires multiple teams or large numbers of people to work on the scalable projects.

2. This model requires heavily committed developers and customers.

3) If commitment is lacking then RAD project will fail

3) The project using RAD model requires heavy resources.

4) The project using RAD model find it difficult to a new technologies.



Evolutionary process model:-

Evolutionary process model is a Iterative model.

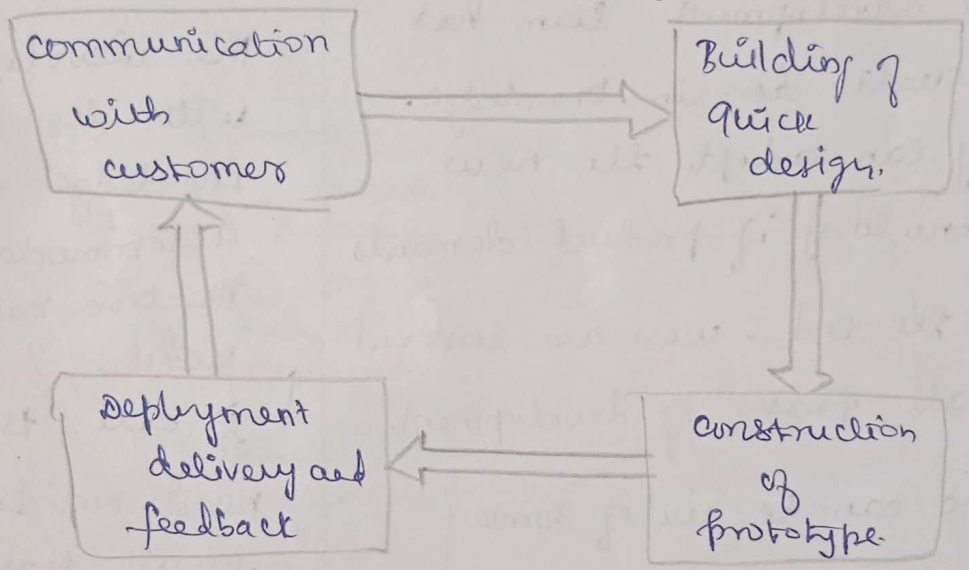
prototyping:-

* In prototyping model initially the requirement gathering is done.

* Developer and customer defines overall objectives

* quick design is prepared. This design represents what will be visible to user in input and output format
* from the quick design a prototype is prepared.
* customer or user evaluates the prototype in order to refine the requirements
* Important to identify the software requirements

Prototyping.



Drawbacks of prototyping:-

* In the 1st version itself, customer often wants "few fixes" rather than rebuilding of the system
* New system maintains high level of quality
* Sometimes developer may take implementation compromises to get prototype working quickly.

comparison between prototyping and incremental process model

S.No	prototyping	Incremental process model
1.	Some requirements are gathered initially, but there may be change in requirement when the working prototype is shown to the customer	The requirements are precisely defined and there is no confusion about the final product of the software.
2.	The development team has adequate domain knowledge. they can adopt the new technology if product demands	The development team with less domain knowledge can be accommodated due to iterative nature of this model.
3.	All the end-users are involved in all phases of development	All the end-users need not be involved in all phases of development
4.	There can be use of some reusable software components in project development process.	4) There is no use of reusable components in development process.

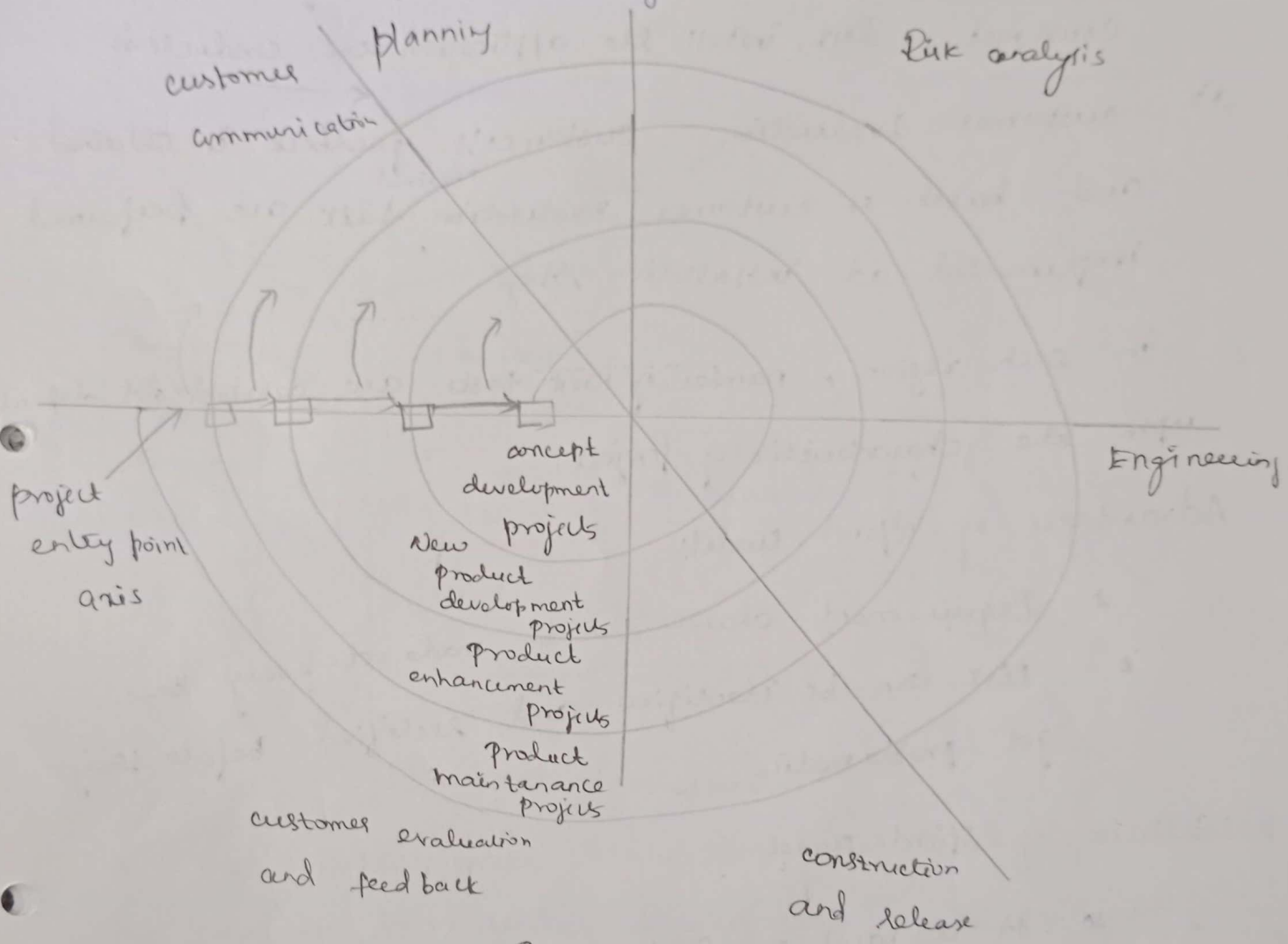
Spiral Model:-

* This model possess the iterative nature of prototyping model and controlled and systematic approach of the linear sequential model.

* The spiral model is divided into a number of framework activities. These framework activities are denoted by task region.

Spiral model is invented by Dr. Barry Boehm in 1988

* There are six task regions.



Spiral model.

(i) customer communication - TO establish customer communication

(ii) planning - All planning activities are carried out to define resources time line and other project related activities.

(iii) Risk analysis - The task Required to calculate technical and management risks are carried out.

(iv) Engineering - In this task Region, task required to build one or more Representation of application are carried out.

(v) construct and release - All the necessary task required to construct, test, install the application are conducted.

(vi) customer Evaluation - Customer's feedback is obtained and based on customer Evaluation task are performed implemented at installation stage

In each region, number of work tasks are carried out depending upon the characteristic of project.

Advantages of Spiral model:

* Requirement changes can be made at every stage.
* Risk can be identified and rectified before they get problematic

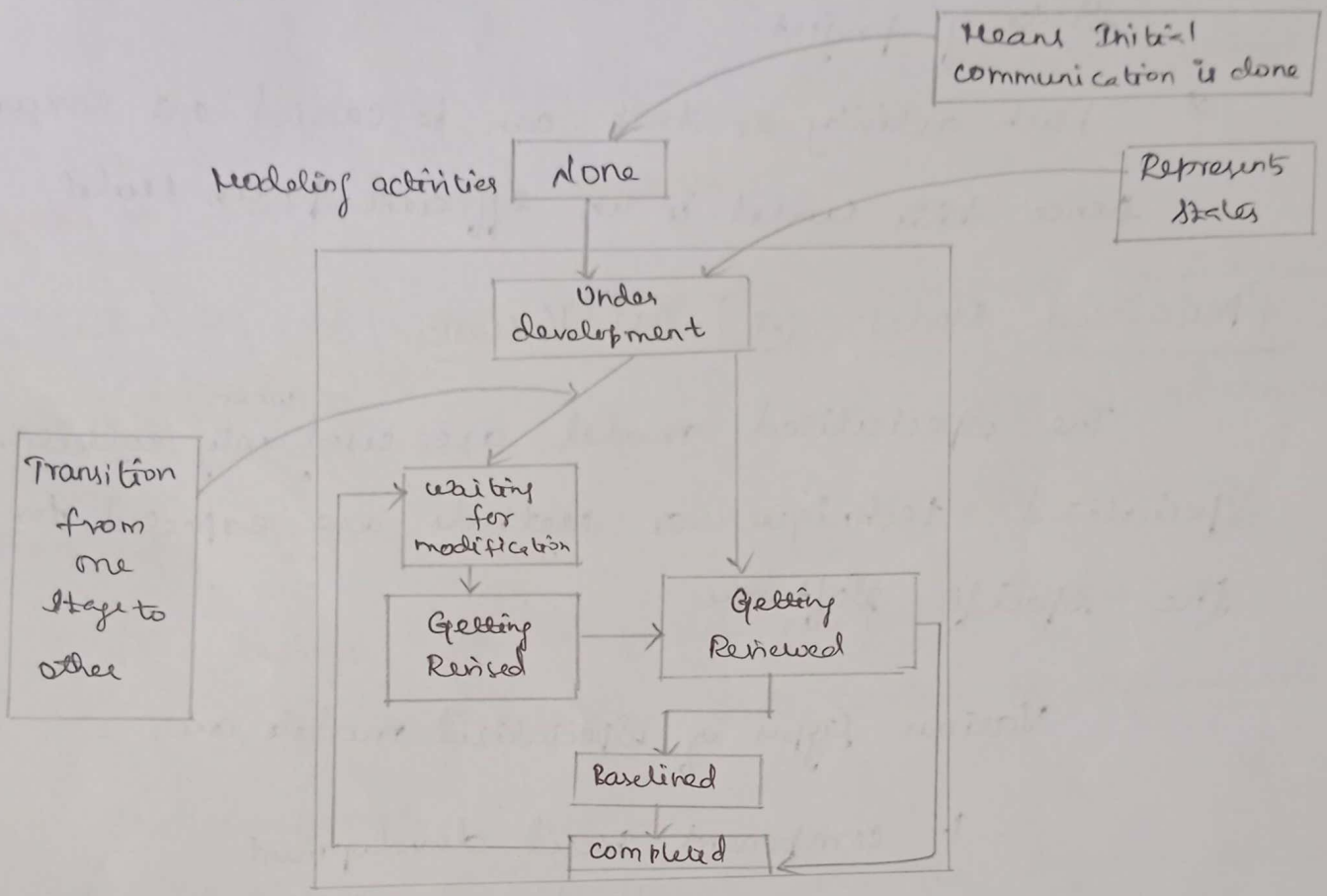
Drawback of Spiral model:

* It is based on customer communication.
* It demands considerable risk assessment

Concurrent Development model:-

* It is also called as Concurrent Engineering
* In this model, the framework activities or software development task are represented as States.

The modeling or designing phase of software development can be in one of the state like under development, waiting for modification, under revision or under review and so on



Concurrent development model.

* These states make transitions. That is during modeling, the transition from under development state to waiting for modification state occurs.

* It defines the series of events due to which the transition from one stage to other state occurs. This is called triggering.

Adv:-

* All types of software development can be done.
* This model provides accurate picture of current state of project
* Each activity or task can be carried out concurrently.

Hence this model is an efficient process model.

Specialized model:- (4) AU 16 marks.

The Specialized model are used when only collection of Specialized technique or methods are expected for developi the specific software.

Various types of Specialized models are,

1. component based development
2. Formal methods model
3. Aspect oriented software development.

① Component based development:-

* The commercial off-shelves components that are developed by the vendors are used during the software built.

* These components have specialized targeted functionalities and well defined Interface.

* The component based development model makes use of various characteristic of spiral model.
* Before beginning the modelling and construction activity of software development the candidate components must be searched and analyzed.

Following steps are applied for component based development:

* Identify the component based products and analyze them for fitting in the existing application domain.
* Analyze the component integration issues
* Design the software architecture to accommodate the components
* Integrates the components into the software architecture
* Conduct comprehensive testing for the developed software.

* Software Reusability is the major advantage of component based development.
* The reusability reduces the development cycle time and overall cost.

2) Formal Method Model:-

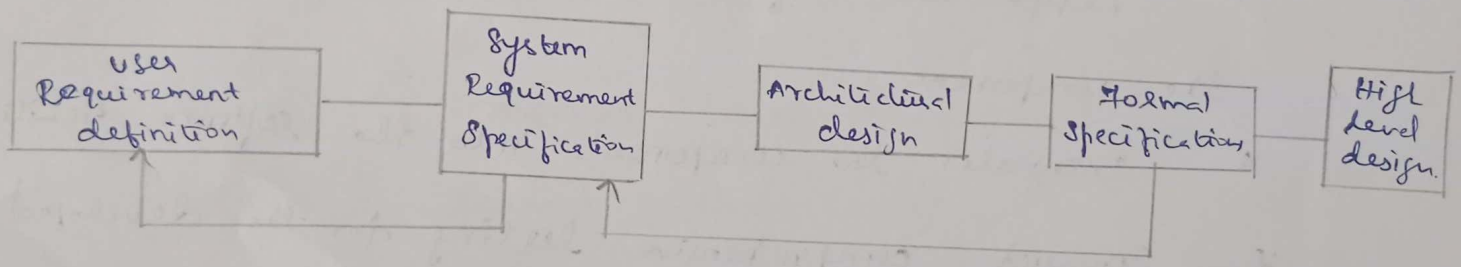
* This model consists of the set of activities in which the formal mathematical specification is used.

* The software engineer specify, develop and test the compute based system using the mathematical notation.

* Cleanroom software Engineering make use of the formal method approach.

* Advantages - To overcome many problems that we encounter in traditional software process model.

* Ambiguity, Incompleteness and inconsistency.



Activities for formal methods model.

- * offers defect-free software

Drawbacks:

* This formal method is time consuming and expensive
* The developer needs the strong mathematical background or some extensive training.
* This model is chosen for development then the communication with customer becomes very difficult.

3) Aspect oriented Software Development:-

* In traditional Software development process, the system is decomposed into multiple units of primary functionality.

* The coding process for realizing a concern become very critical.

Aspect oriented Software development focuses on the identification, specification and representation of cross-cutting concerns and their modularization into separate functional units.

Aspectual Requirements defines those cross-cutting concerns that have impact on the software architecture.

Aspect Aspect oriented Software Development is referred as Aspect oriented programming.

* It is new software paradigm and is not matured enough. It will adopt the characteristic of both the Spiral model and concurrent process model.

---

(4) Introduction to Agility:-

The Agile Manifesto is also called the Manifesto for Agile Software development. a formal declaration of four key values and 12 principle to guide an iterative and people centric approach to Software Development.

* The Agile methods were developed to overcome the weakness of conventional Software Engineering.

The Agile manifesto is represented by,

Individual and Interactions	over	process and tools
working software	over	comprehensive documentation
customer collaboration	over	contract negotiation
Responding to change	over	following plan.

Agile manifesto.

Agile process:-

* In 1980's, the heavy weight, plan based Software development approach was as to develop any software product.
* This approach was rigid. (ie) If requirements get changed, then rework as essential
* In 1990's, New methods were proposed are known as Agile process.

* The Agile processes are the light weight methods are people-based rather than the plan based methods.

* The Agile team from the development team to focus on software itself rather than design and

Documentation.

* The Agile process believes in iterative method.
* The aim is to deliver the working model of software quickly to the customer.

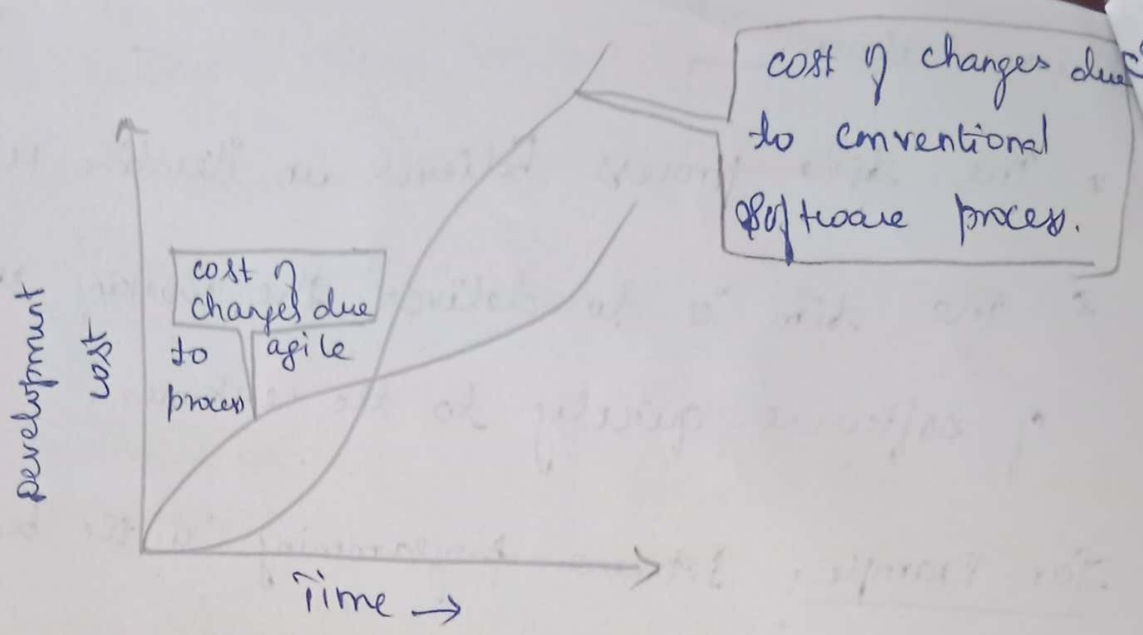
For Example: Extreme programming is the best known of Agile process.

Conventional Software Development Methodology:

* As the software project makes the progress, the cost of the changes increases non-linearly
* It is easy to accommodate changes during the requirement gathering stage.
* As the stage to accommodate the changes - usage scenario are modified, list of functions can be extended. or written specification be edited.

Agile Methodology:

* If the incremental delivery is combined with agile practices such as continous unit testing and pair programming then the cost of changes can be controlled.
* The software development approach has a strong influence on the development cost due to changes suggested.



pros:-

1. Efficient delivery - Delivering the small increments of working software quickly.
2. Flexibility - Agile software development methodology allows for changes and adaptions to be made throughout the development process.
3. Continous Improvement - Encourage continous reflection are improvements in the software.
4. Transparency - Daily stand-up meeting makes all the team members and stakeholders aware of the project's progress challenges and decisions.
5. Reduced Risk - To identify and mitigate risks early on. When everyone aware of the project's challenges they can work together to develop solutions.

Cons:

1. Need for Disciplined Team
2. More Collaboration and communication
3. Lack of Documentation.

Agile principles (refer page no 1-35)

Extreme programming :-

The set of five values that serve as a basis for the work performed in xp.

1. Communication - The effective communication must be established between software developers and stakeholders in order to convey the important concepts and to get the important feedback.

2. Simplicity - xp focuses on the current needs instead of future needs to incorporate in design.

3. Feedback - The feedback for the software product can be obtained from the developers of the software, customers and other software team members.

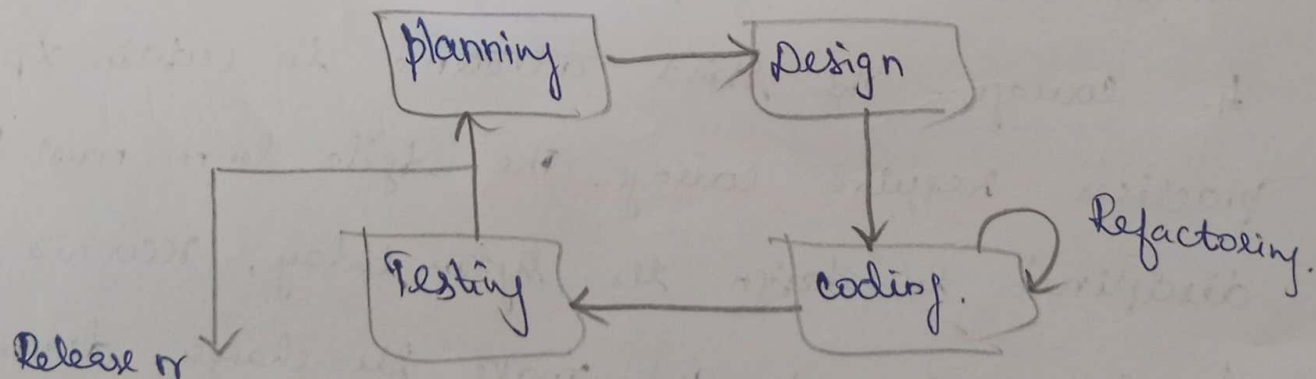
4. Courage - The strict adherence to certain xp practices require courage. The Agile team must be disciplined to design the system today, recognize the function requirement and make the changes dramatically as per the demand.

5. Respect - By following the above states xp values agile team can win the respect of stakeholders.

process:-

The extreme programming process is explained as follows

* The developer and customer together prepare a story-card in which customer needs are mentioned.
* The developer team then aims to implements the scenario in the story-card.
* The development team break the total work in small tasks.
* The customer prioities the stories for implementation.
* Unimplemented stories have to be discarded.
* For accommodating new changes, new story-card mus be developed.
* Evaluate the system along with the customer



xp process.

Various rules and practices used in extreme programming are,

Planning – user Story - cards
Release planning
Small releases
Iterative process
Stand up meeting.

Designing – Simple design
Spike Solutions
Refactoring.

Coding – customer availability
paired programming
collective code ownership.

Testing – unit testing
continous Integration
No overtime.

Example – Banking software checking balance, depositing and withdrawing.

Industrial xp

* The Industrial xp (IXP) can be defined as the organic evolution of xp.
* It is customer centric.

* It has expanded role for customers and advanced technical practice.

Various new practices that are appended to xp to create Exp are follows.

1. Readiness Assessment: Following Issues are assessed,

   1) proper environment must be available to support xp

   2) Team should contain appropriate and skilled stakeholder.

   3) The organization should support quality programs and continous improvements.

   4) The organizational culture should support new values of agile team.

2. project community:-

   * Skilled and efficient people must be chosen as the agile team members for the succes of the project.

   * It is referred as community. The project community consists of technologies, customers and other stakeholders. who play the vital role for the succes of the project.

3. project chartering:

   project chartering means assessing the justification for the project as a business application.

* The Ixp team asess whether the project satisfies the goals and objectives of the organization.

4) Test Driven Management:

For assessing the state of the project and its programming industrial xp needs some measurable criteria.

5) Retro Spectives:

* after delivering the software increment, the specialized review conducted which is called as retrospective.

* TO Improve the industrial xp process.

6) continous learning:

The team members are inspired and encourage to new methods and techniques that can improve the quality of the product.