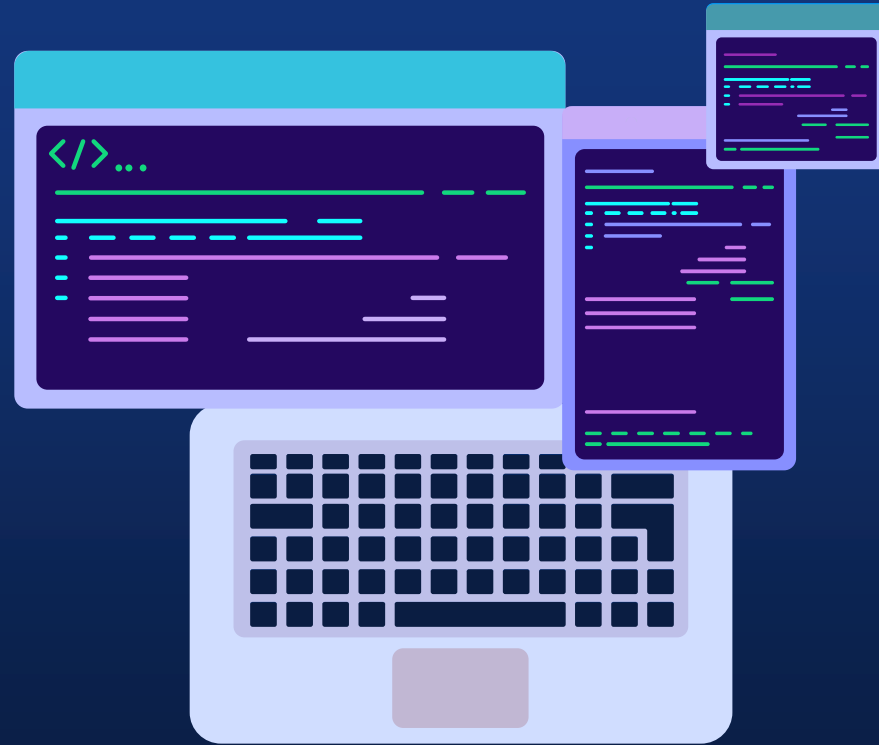


SOFTWARE ENGINEERING





Course Objectives

- To learn and understand the principles of Software Engineering.
- To be acquainted with methods of capturing, specifying, visualizing and analyzing software requirements.
- To apply design and testing principles to software project development.
- To understand project management through life cycle of the project.





Course Outcomes:

At the end of the course students will be able to-

- CO1: Analyze software requirements and formulate design solution for a software.
- CO2: Design applicable solutions in one or more application domains using software engineering approaches that integrate ethical, social, legal and economic concerns.
- CO3: Apply new software models, techniques and technologies to bring out innovative and novelistic solutions for the growth of the society in all aspects and evolving into their continuous professional development.
- CO4: Model and design User interface and component-level.
- CO5: Identify and handle risk management and software configuration management.
- CO6: Utilize knowledge of software testing approaches, approaches to verification and validation.
- CO7: Construct software of high quality – software that is reliable, and that is reasonably easy to understand, modify and maintain efficient, reliable, robust and cost-effective software solutions.



Examination Scheme and Marks:

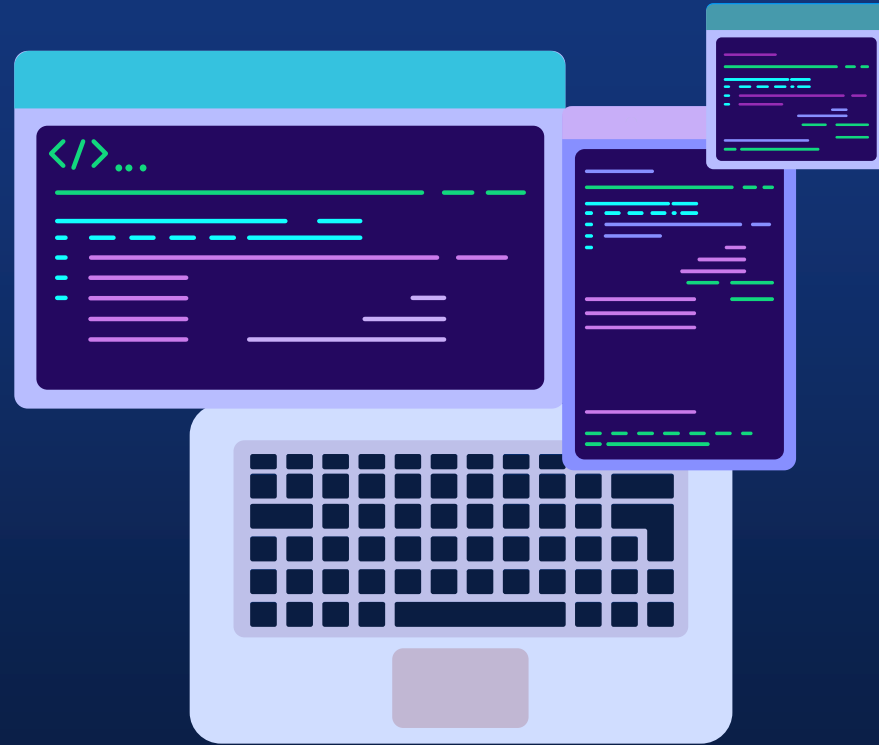
Mid-Semester(TH): 30 Marks

End-Semester(TH): 70 Marks



UNIT I

Introduction to Software Engineering and Software Process Models





Unit I Contents

- Software Engineering Fundamentals: Introduction to software engineering, The Nature of Software, Defining Software, Software Engineering Practice.
- Software Process: A Generic Process Model, defining a Framework Activity, Identifying a Task Set, Process Patterns, Process Assessment and Improvement, Prescriptive Process Models, The Waterfall Model, Incremental Process Models, Evolutionary Process Models, Concurrent Models, A Final Word on Evolutionary Processes.
- Unified Process, Agile software development: Agile methods, plan driven and agile development.





THE NATURE OF SOFTWARE

What is it?

Computer software is the product that software professionals build and then support over the long term. It encompasses programs that execute within a computer of any size and architecture, content that is presented as the computer programs execute, and descriptive information in both hard copy and virtual forms that encompass virtually any electronic media.


Who does it?

Software engineers build and support software, and virtually everyone in the industrialized world uses it either directly or indirectly.





Software is:

- (1) instructions (computer programs) that when executed provide desired features, function, and performance;
 - (2) data structures that enable the programs to adequately manipulate information, and
 - (3) descriptive information in both hard copy and virtual forms that describes the operation and use of the programs.
- 



Engineering:

Engineering is the application of **scientific** and **practical** knowledge to **invent, design, build, maintain,** and **improve frameworks, processes, etc.**





Software Application Domains

SYSTEM SOFTWARE

Compilers, editors, and file management utilities

APPLICATION SOFTWARE

Microsoft Office

ENGINEERING/SCIENTIFIC SOFTWARE

MATLAB, AUTOCAD

EMBEDDED SOFTWARE

washing machines, satellites, microwaves

PRODUCT-LINE SOFTWARE

WEB/MOBILE APPLICATIONS

World Wide Web

ARTIFICIAL INTELLIGENCE SOFTWARE

Google Cloud Machine Learning Engine, Amazon Alexa





The Changing Nature of Software

- Web-Apps
- Mobile Applications
- Cloud Computing
- Product Line Software



SOFTWARE ENGINEERING

Software engineering encompasses a process, a collection of methods (practice) and an array of tools that allow professionals to build high-quality computer software.



SOFTWARE ENGINEERING

- (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
- (2) The study of approaches as in (1).



SOFTWARE ENGINEERING

The IEEE definition:

- Software Engineering: The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
- Engineering approach to develop software.
Building Construction Analogy.
- Systematic collection of past experience:
Techniques,
Methodologies,
Guidelines.

Why is Software Engineering required?

Software Engineering is required due to the following reasons:

- To manage Large software
- For more Scalability
- Cost Management
- To manage the dynamic nature of software
- For better quality Management



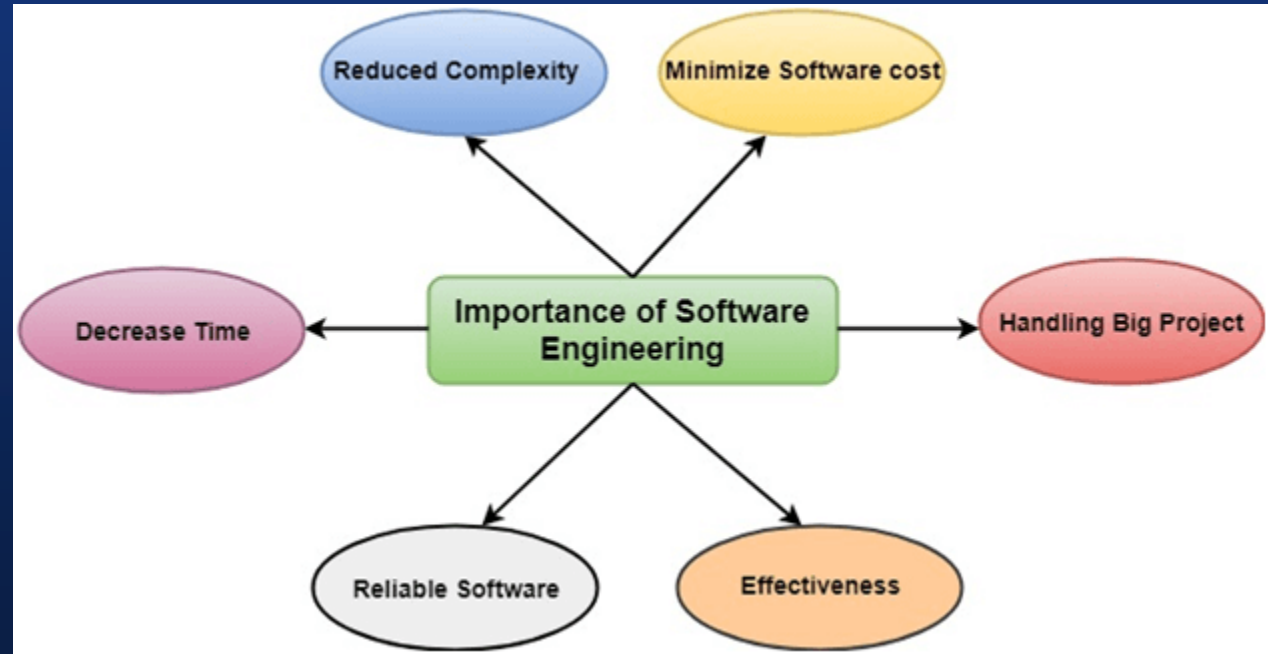
Need of Software Engineering

The necessity of software engineering appears because of a higher rate of progress in user requirements and the environment on which the program is working.

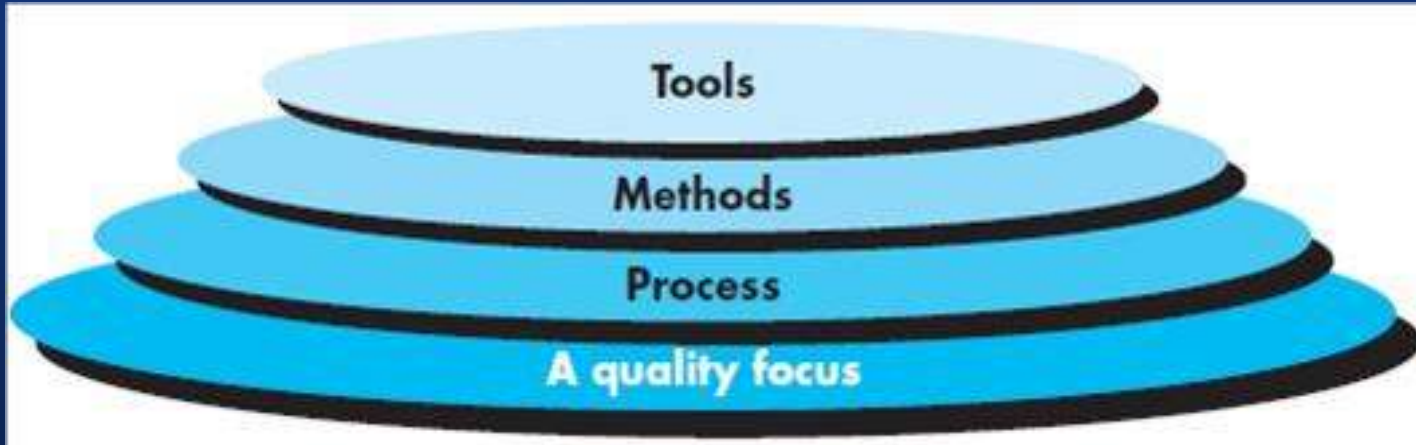
- **Huge Programming**
- **Adaptability**
- **Cost**
- **Dynamic Nature**
- **Quality Management**



Importance of Software Engineering



Software engineering is a layered technology.



Software engineering is a layered technology.

- Any engineering approach (including software engineering) must rest on an organizational commitment to quality.
- TQM, Six Sigma or any similar philosophies foster a continuous process improvement culture, and it is this culture that ultimately leads to the development of increasingly more effective approaches to software engineering.
- The bedrock that supports software engineering is a quality focus

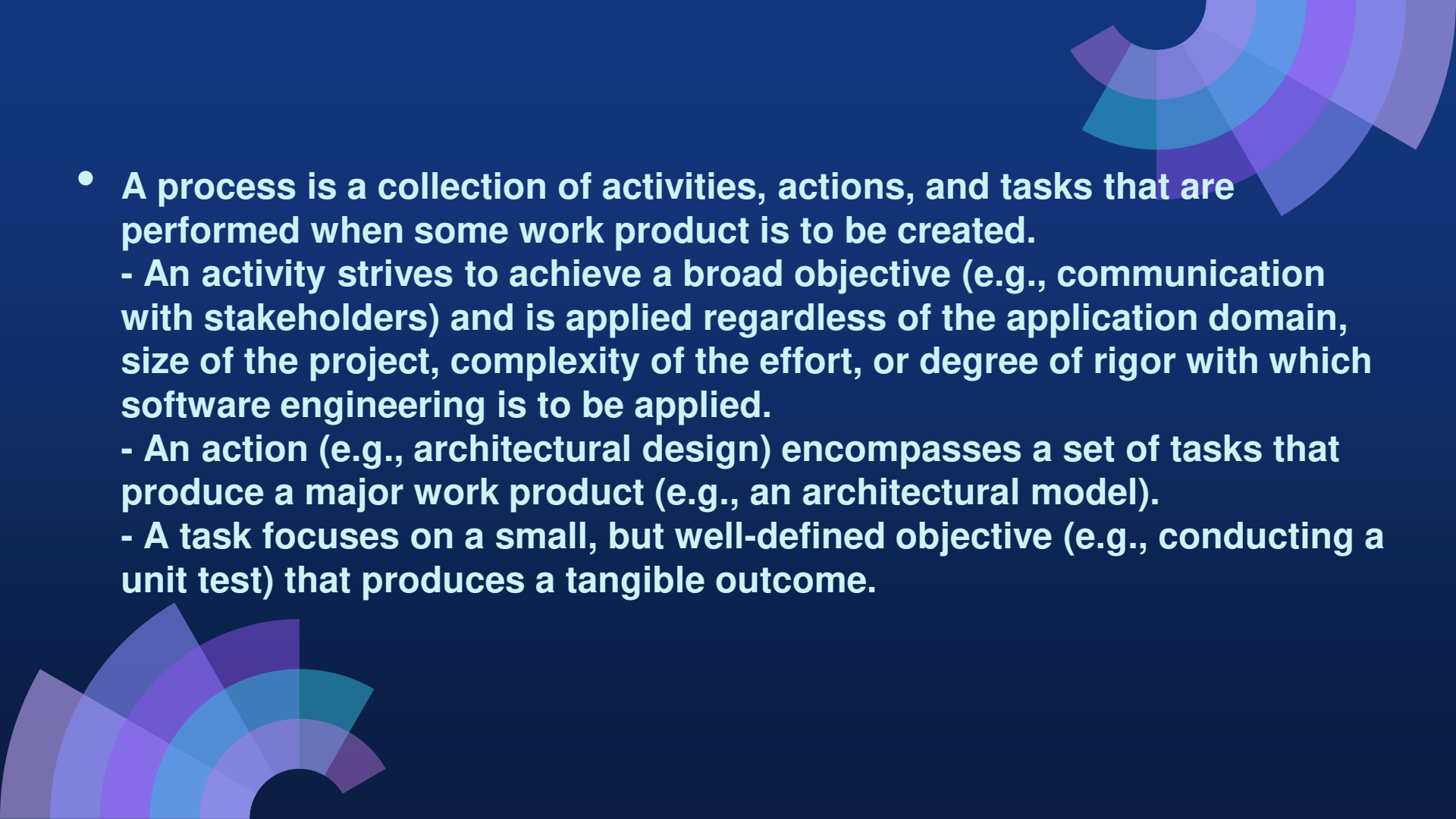
Software engineering is a layered technology.



- Foundation layer is Process Layer
- Definition of Process : a series of actions or steps taken in order to achieve a particular end.
- Process Model : A framework i.e. responsible for milestones, quality, work products
- Methods: Technical — “how to” s. Methods encompass a broad array of tasks that include communication, requirements analysis, design modelling, program construction, testing, and support.
- Tools : Automated or simulated support for s/w development. When tools are integrated so that information created by one tool can be used by another, a system for the support of software development, called computer-aided software engineering, is established.



THE SOFTWARE PROCESS

A process is not a rigid prescription
for how to build computer software.

- 
- **A process is a collection of activities, actions, and tasks that are performed when some work product is to be created.**
 - **An activity strives to achieve a broad objective (e.g., communication with stakeholders) and is applied regardless of the application domain, size of the project, complexity of the effort, or degree of rigor with which software engineering is to be applied.**
 - **An action (e.g., architectural design) encompasses a set of tasks that produce a major work product (e.g., an architectural model).**
 - **A task focuses on a small, but well-defined objective (e.g., conducting a unit test) that produces a tangible outcome.**

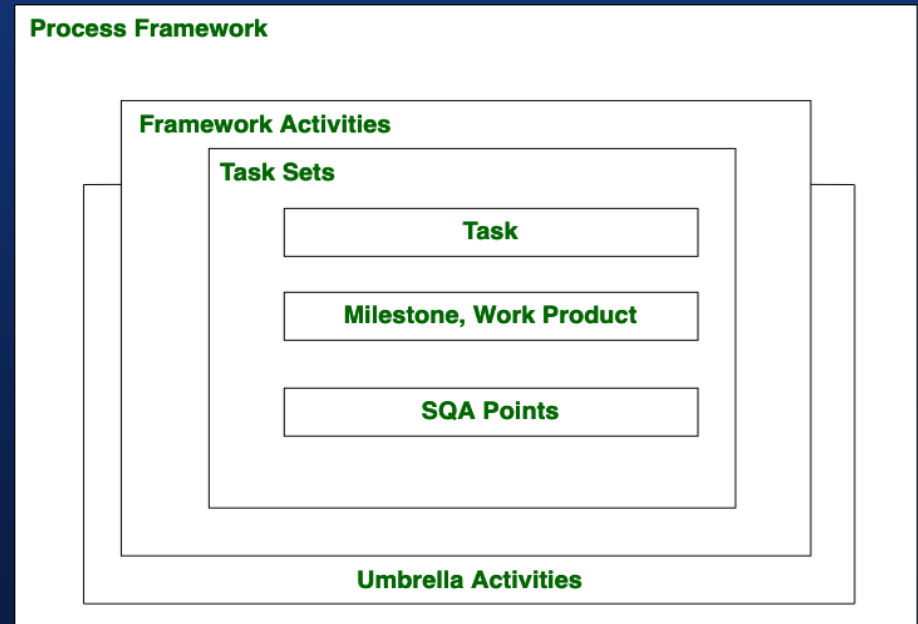
- 
- A process is not a rigid prescription for how to build computer software.
 - The intent is always to deliver software in a timely manner and with sufficient quality to satisfy those who have sponsored its creation and those who will use it.
 - Includes umbrella activities which protect and control the process.
 - Process Framework has 5 activities:
 1. Communication
 2. Planning
 3. Modeling
 4. Construction
 5. Deployment
- 




The Process Framework




- The process of framework defines a small set of activities that are applicable to all types of projects.
- The software process framework is a collection of task sets.
- Task sets consist of a collection of small work tasks, project milestones, work productivity and software quality assurance points.



Software Process Framework




A generic process framework encompasses five activities which are given below one by one:

- 1. Communication:** This framework activity involves heavy communication and collaboration with the customer (and other stakeholders) and encompasses requirements gathering and other related activities.
 - 2. Planning:** This activity establishes a plan for the software engineering work that follows. It describes the technical tasks to be conducted, the risks that are likely, the resources that will be required, the work products to be produced and a work schedule.
 - 3. Modeling:** This activity encompasses the creation of models that allow the developer and the customer to better understand software requirements and the design that will achieve those requirements.
 - 4. Construction:** This activity combines code generation (either manual or automated) and the testing that is required to uncover errors in the code.
 - 5. Deployment:** The software is delivered to the customer who evaluates the delivered product and provides feedback based on evaluation.
- 

Umbrella activities include

1. Software project tracking and control
2. Risk management
3. Software Quality Assurance (SQA)
4. Formal Technical Reviews (FTR)
5. Measurement
6. Software Configuration Management (SCM)
7. Reusability management
8. Work product preparation and production






1. Software project tracking and control- In this activity, the developing team accesses project plan and compares it with the predefined schedule. If these project plans do not match with the predefined schedule, then the required actions are taken to maintain the schedule.


2. Risk management- Risk is an event that may or may not occur. If the event occurs, then it causes some unwanted outcome. Hence, proper risk management is required.

3. Software Quality Assurance (SQA)- SQA is the planned and systematic pattern of activities which are required to give a guarantee of software quality.

For example, during the software development meetings are conducted at every stage of development to find out the defects and suggest improvements to produce good quality software.

4. Formal Technical Reviews (FTR)- FTR is a meeting conducted by the technical staff. The motive of the meeting is to detect quality problems and suggest improvements. The technical person focuses on the quality of the software from the customer point of view.






5. Measurement- Measurement consists of the effort required to measure the software. The software cannot be measured directly. It is measured by direct and indirect measures. Direct measures like cost, lines of code, size of software etc. Indirect measures such as quality of software which is measured by some other factor. Hence, it is an indirect measure of software.

6. Software Configuration Management (SCM)- It manages the effect of change throughout the software process.

7. Reusability management- It defines the criteria for reuse the product. The quality of software is good when the components of the software are developed for certain application and are useful for developing other applications.

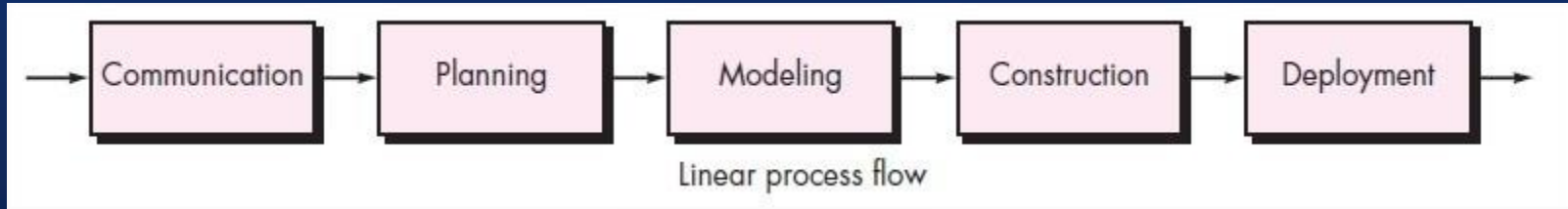
8. Work product preparation and production- It consists of the activities that are needed to create the documents, forms, lists, logs and user manuals for developing a software.



Process Flow:

1. Linear Process Flow

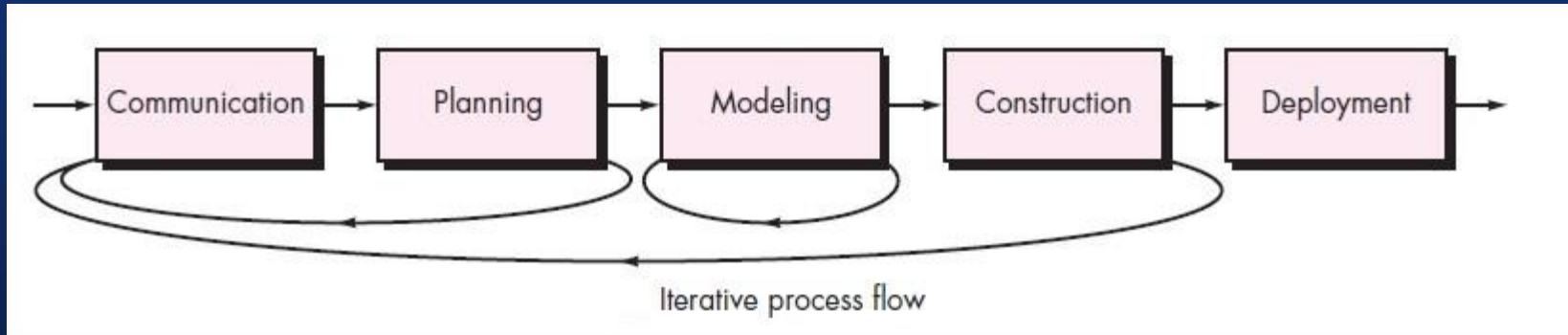
A **linear process flow** executes each of the five framework activities in sequence, beginning with communication and culminating with deployment.



Process Flow:

1. Iterative Process Flow

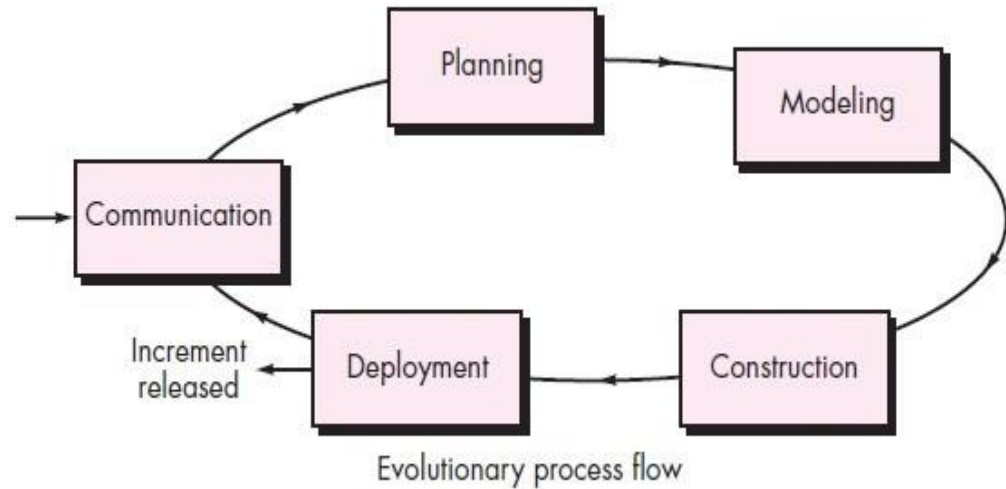
An **iterative process flow** repeats one or more of the activities before proceeding to the next.



Process Flow:

1. Evolutionary Process Flow

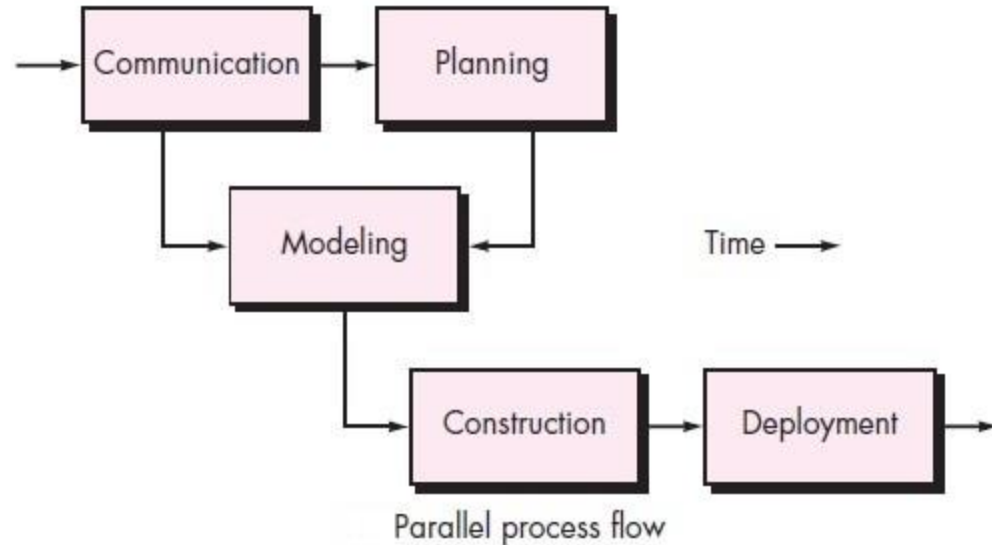
An **evolutionary process flow** executes the activities in a “circular” manner.



Process Flow:

1. Parallel Process Flow

A **parallel process flow** executes one or more activities in parallel with other activities .



The Process Model

The goal of a software process model is to provide guidance for controlling and coordinating the tasks to achieve the end product and objectives as effectively as possible.




The Process Model

- A software process model is an abstraction of the software development process.
- The models specify the stages and order of a process. So, think of this as a representation of the **order of activities** of the process and the **sequence** in which they are performed.
- **A model will define the following:**
 1. The tasks to be performed
 2. The input and output of each task
 3. The pre and post conditions for each task
 4. The flow and sequence of each task

Factors in choosing a software process



If you know your requirements well, it will be easier to select a model that best matches your needs. You need to keep the following factors in mind when selecting your software process model:

1. **Project requirements**
 2. **Project size**
 3. **Project complexity**
 4. **Cost of delay**
 5. **Customer involvement**
 6. **Familiarity with technology**
 7. **Project resources**
- 



PROCESS MODEL

PROCESS MODEL



Generic Process Model

Prescriptive Process Models

Evolutionary Process Models

1. The Waterfall Model
2. Incremental Process model
3. RAD model

1. The prototyping model
2. The spiral model
3. Concurrent development model



Generic process model

The Generic process model is **an abstraction of the software development process**. It specifies the stages and order of a process.

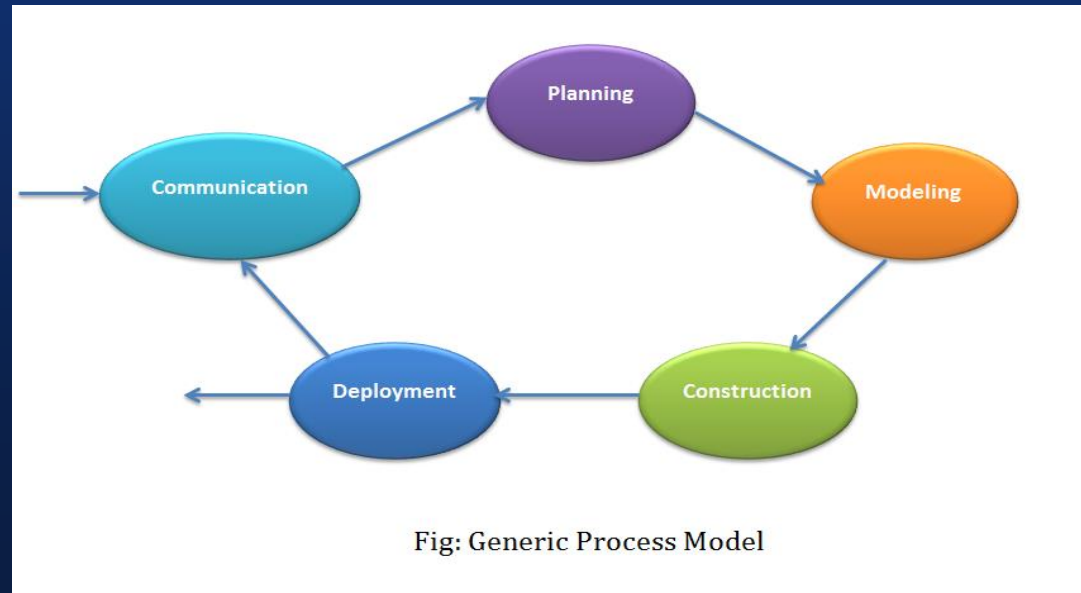


Fig: Generic Process Model


Prescriptive Process Models



The following framework activities are carried out irrespective of the process model chosen by the organization.

1. Communication
2. Planning
3. Modeling
4. Construction
5. Deployment

The name 'prescriptive' is given because the model prescribes a set of activities, actions, tasks, quality assurance and change the mechanism for every project.



Prescriptive Process Models: 1. The Waterfall Model

- The waterfall model is also called as '**Linear sequential model**' or '**Classic life cycle model**'.
- In this model, each phase is fully completed before the beginning of the next phase.
- This model is used for the small projects.
- In this model, feedback is taken after each phase to ensure that the project is on the right path.
- Testing part starts only after the development is complete.

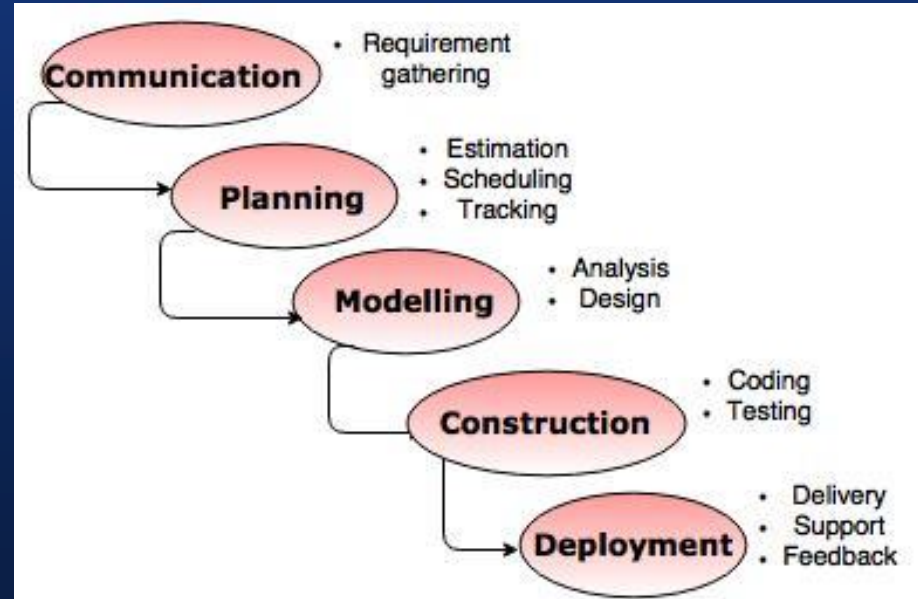


Fig. - The Waterfall model

Prescriptive Process Models: 1. The Waterfall Model

-> Advantages of waterfall model

1. The waterfall model is simple and easy to understand, implement, and use.
2. All the requirements are known at the beginning of the project, hence it is easy to manage.
3. It avoids overlapping of phases because each phase is completed at once.
4. This model works for small projects because the requirements are understood very well.
5. This model is preferred for those projects where the quality is more important as compared to the cost of the project.

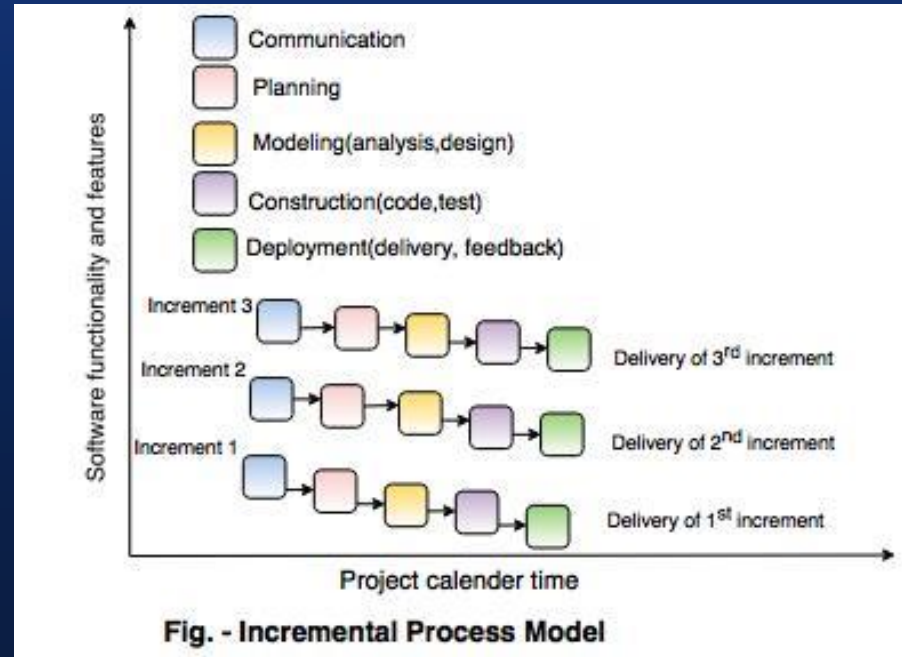
-> Disadvantages of the waterfall model

1. This model is not good for complex and object oriented projects.
2. It is a poor model for long projects.
3. The problems with this model are uncovered, until the software testing.
4. The amount of risk is high.

Prescriptive Process Models: 2. Incremental Process model

- The incremental model combines the elements of waterfall model and they are applied in an iterative fashion.
- The first increment in this model is generally a core product.
- Each increment builds the product and submits it to the customer for any suggested modifications.
- The next increment implements on the customer's suggestions and add additional requirements in the previous increment.
- This process is repeated until the product is finished.

For example,
the word-processing software is developed using the incremental model.



Prescriptive Process Models: 2. Incremental Process model

Advantages of incremental model

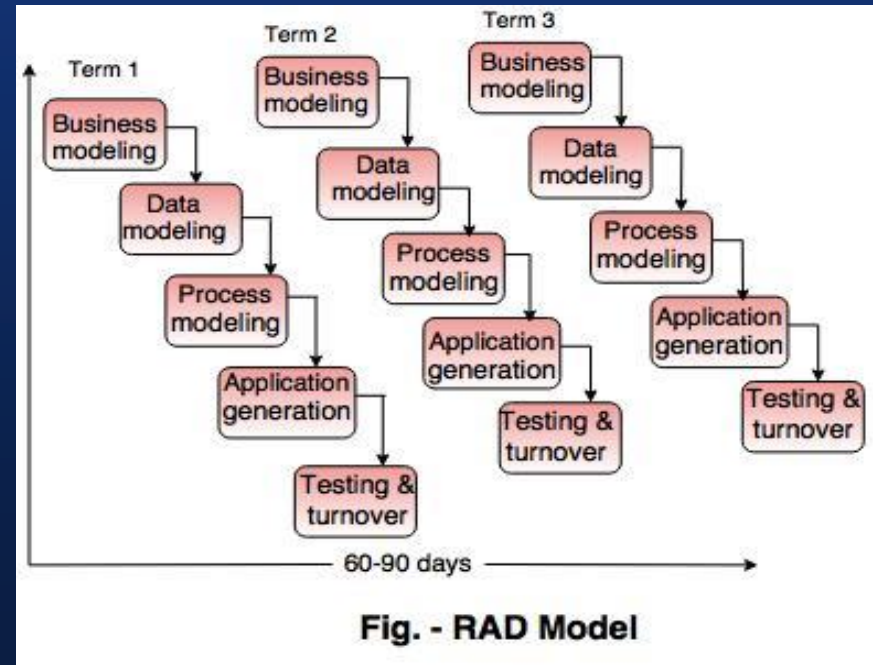
- This model is flexible because the cost of development is low and initial product delivery is faster.
- It is easier to test and debug during the smaller iteration.
- The working software generates quickly and early during the software life cycle.
- The customers can respond to its functionalities after every increment.

Disadvantages of the incremental model

- The cost of the final product may cross the cost estimated initially.
- This model requires a very clear and complete planning.
- The planning of design is required before the whole system is broken into small increments.
- The demands of customer for the additional functionalities after every increment causes problem during the system architecture.

Prescriptive Process Models: 3. RAD model

- RAD is a Rapid Application Development model.
- Using the RAD model, software product is developed in a short period of time.
- The initial activity starts with the communication between customer and developer.
- Planning depends upon the initial requirements and then the requirements are divided into groups.
- Planning is more important to work together on different modules.





Evolutionary Process Models

The Generic process model is **an abstraction of the software development process**. It specifies the stages and order of a process.

- Evolutionary models are iterative type models.
- They allow to develop more complete versions of the software.

Following are the evolutionary process models.

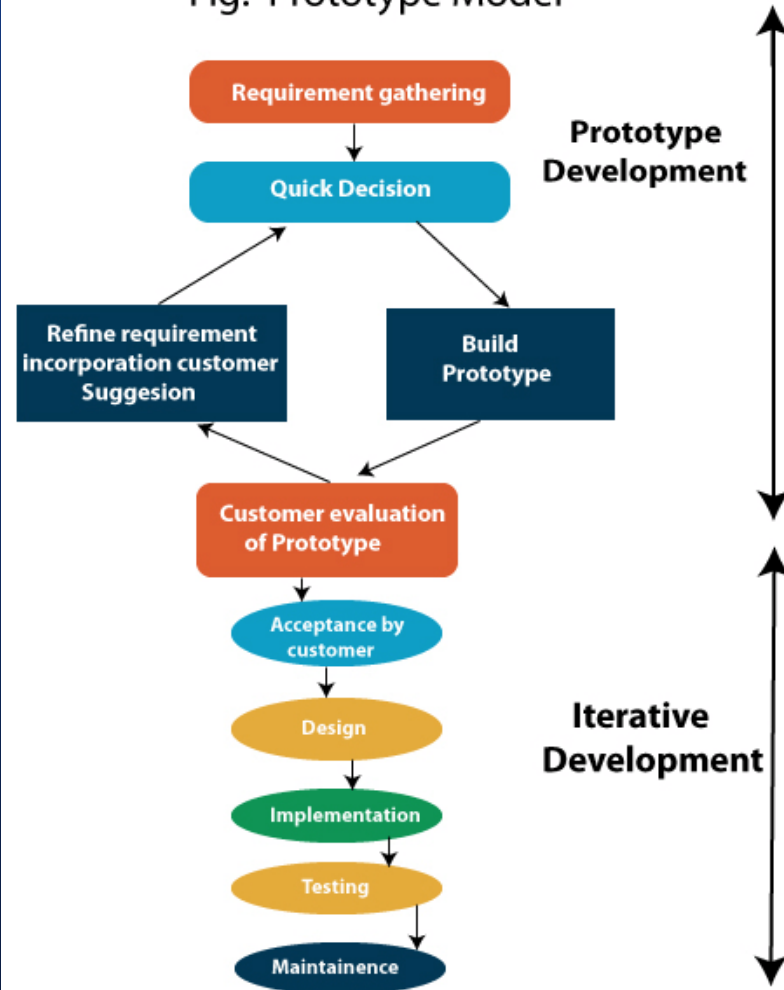
1. The prototyping model
2. The spiral model
3. Concurrent development model



Evolutionary Process Models: 1. The prototyping model

1. The prototype model requires that before carrying out the development of actual software, a working prototype of the system should be built.
2. A prototype is a toy implementation of the system.
3. A prototype usually turns out to be a very crude version of the actual system, possibly exhibiting limited functional capabilities, low reliability, and inefficient performance as compared to actual software.

Fig: Prototype Model



Advantage of Prototype Model:

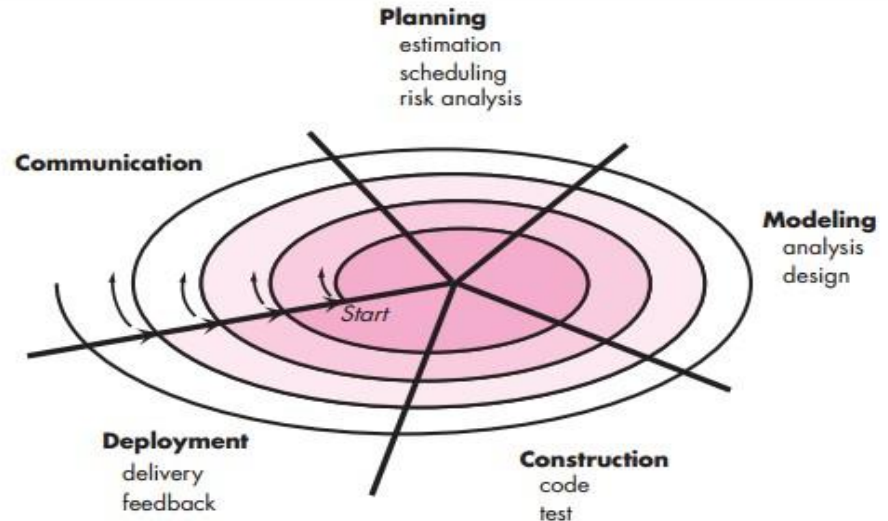
1. Reduce the risk of incorrect user requirement
2. Good where requirement are changing/uncommitted
3. Regular visible process aids management
4. Support early product marketing
5. Reduce Maintenance cost.
6. Errors can be detected much earlier as the system is made side by side.

Disadvantage of Prototype Model

1. An unstable/badly implemented prototype often becomes the final product.
2. Require extensive customer collaboration
 - Costs customer money
 - Needs committed customer
 - Difficult to finish if customer withdraw
 - May be too customer specific, no broad market
3. Difficult to know how long the project will last.
4. Easy to fall back into the code and fix without proper requirement analysis, design, customer evaluation, and feedback.
5. Prototyping tools are expensive.
6. Special tools & techniques are required to build a prototype.
7. It is a time-consuming process.

Evolutionary Process Models: 2. The Spiral model

1. Spiral model is a risk driven process model.
2. It is used for generating the software projects.
3. In spiral model, an alternate solution is provided if the risk is found in the risk analysis, then alternate solutions are suggested and implemented.
4. It is a combination of prototype and sequential model or waterfall model.
5. In one iteration all activities are done, for large project's the output is small.



Advantages of Spiral Model:

Below are some advantages of the Spiral Model.

1. **Risk Handling:** The projects with many unknown risks that occur as the development proceeds, in that case, Spiral Model is the best development model to follow due to the risk analysis and risk handling at every phase.
2. **Good for large projects:** It is recommended to use the Spiral Model in large and complex projects.
3. **Flexibility in Requirements:** Change requests in the Requirements at later phase can be incorporated accurately by using this model.
4. **Customer Satisfaction:** Customer can see the development of the product at the early phase of the software development and thus, they habituated with the system by using it before completion of the total product.

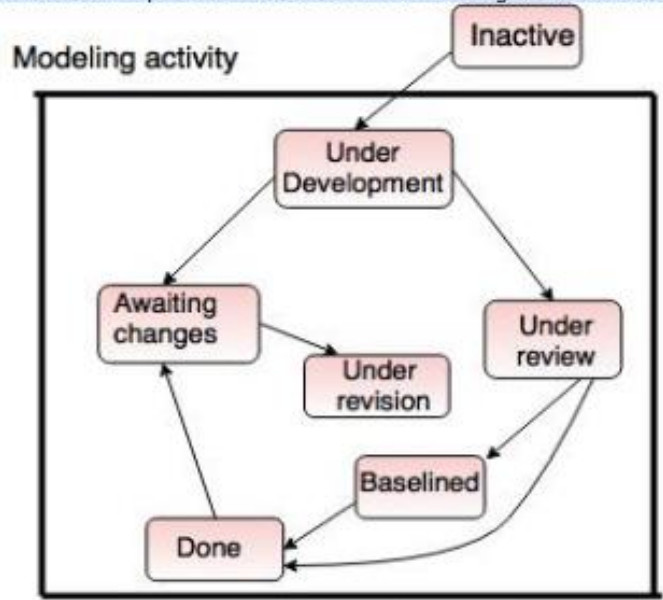
Disadvantages of Spiral Model:

Below are some main disadvantages of the spiral model.

1. **Complex:** The Spiral Model is much more complex than other SDLC models.
2. **Expensive:** Spiral Model is not suitable for small projects as it is expensive.
3. **Too much dependability on Risk Analysis:** The successful completion of the project is very much dependent on Risk Analysis. Without very highly experienced experts, it is going to be a failure to develop a project using this model.
4. **Difficulty in time management:** As the number of phases is unknown at the start of the project, so time estimation is very difficult.

Evolutionary Process Models: 3. The Concurrent model

1. The concurrent development model is called as concurrent model.
2. The communication activity has completed in the first iteration and exits in the awaiting changes state.
3. The modeling activity completed its initial communication and then go to the underdevelopment state.
4. If the customer specifies the change in the requirement, then the modeling activity moves from the under development state into the awaiting change state.
5. The concurrent process model activities moving from one state to another state




 → In above figure each block represents the state of software engineering activity

Fig. - One element of the concurrent process model

Advantages of the concurrent development model

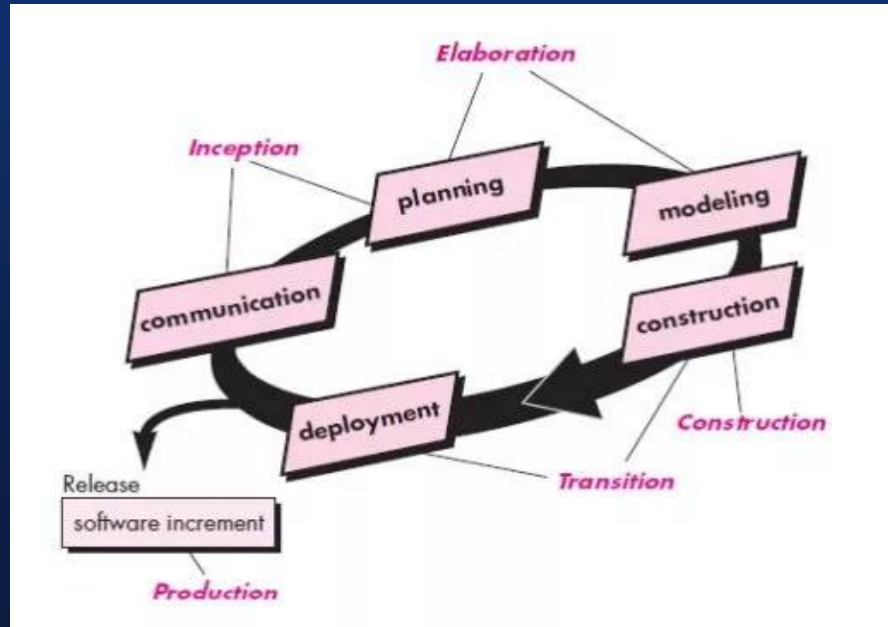
1. This model is applicable to all types of software development processes.
2. It is easy for understanding and use.
3. It gives immediate feedback from testing.
4. It provides an accurate picture of the current state of a project.

Disadvantages of the concurrent development model

1. It needs better communication between the team members. This may not be achieved all the time.
2. It requires to remember the status of the different activities.

Unified Process

Unified process (UP) is an architecture centric, use case driven, iterative and incremental development process. UP is also referred to as the unified software development process.



Unified Process

- Unified process (UP) is an architecture centric, use case driven, iterative and incremental development process. UP is also referred to as the unified software development process.
- The Unified Process recognizes the importance of customer communication and streamlined methods for describing the customer's view of a system.
- It suggests a process flow that is iterative and incremental, providing the evolutionary feel that is essential in modern software development.

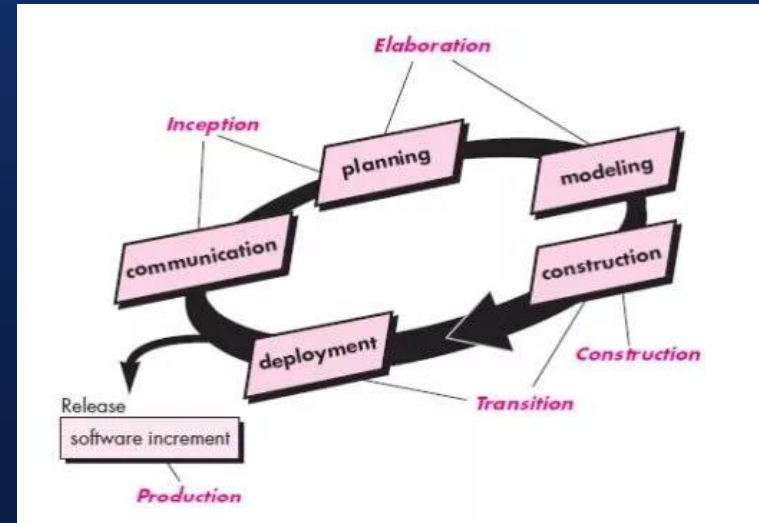
Phases of the Unified Process

This process divides the development process into five phases:

1. Inception
2. Elaboration
3. Conception
4. Transition
5. Production

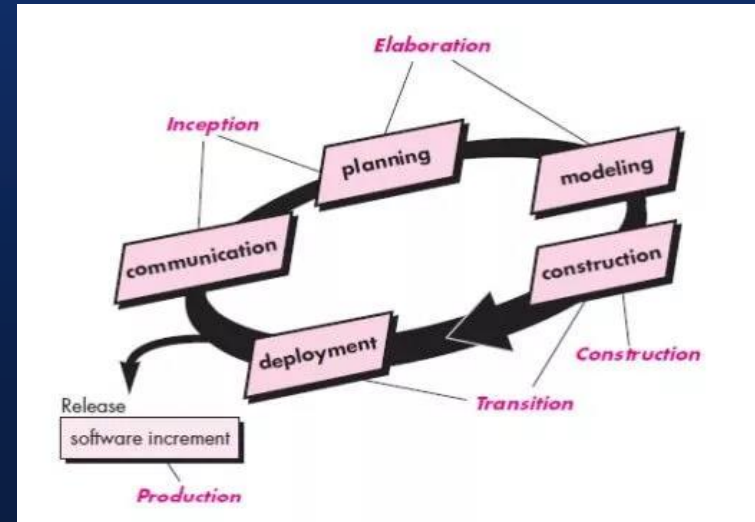
1. Inception –

- > Communication and planning are main.
- > Identifies Scope of the project using use-case model allowing managers to estimate costs and time required.
- > Customers requirements are identified and then it becomes easy to make a plan of the project.
- > Project plan, Project goal, risks, use-case model, Project description, are made.
- > Project is checked against the milestone criteria and if it couldn't pass these criteria then project can be either cancelled or redesigned.



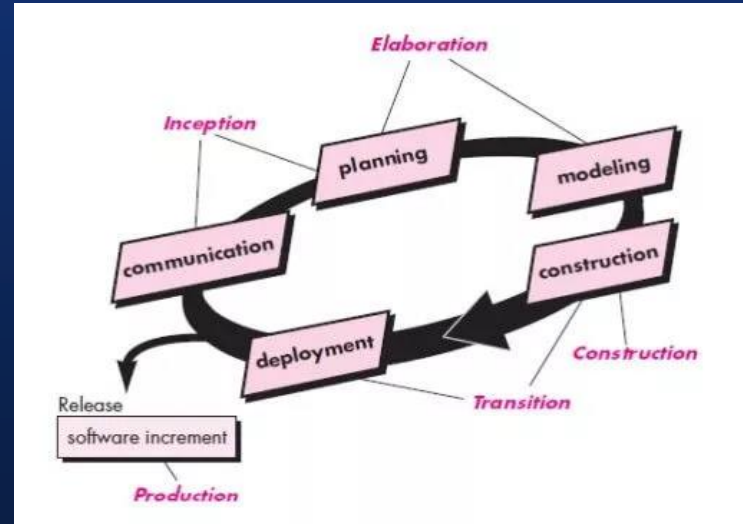
2. Elaboration –

- > Planning and modeling are main.
- > Detailed evaluation, development plan is carried out and diminish the risks.
- > Revise or redefine use-case model (approx. 80%), business case, risks.
- > Again, checked against milestone criteria and if it couldn't pass these criteria then again project can be cancelled or redesigned.
- > Executable architecture baseline.



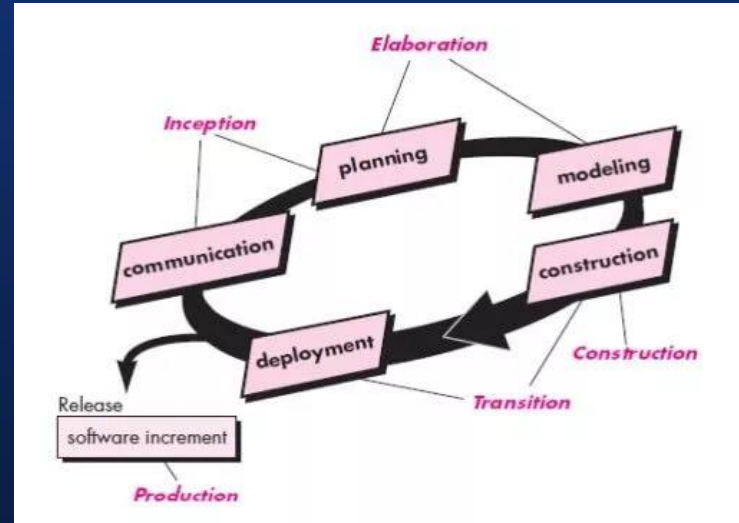
3. Construction –

- > Project is developed and completed.
- > System or source code is created and then testing is done.
- > Coding takes place.



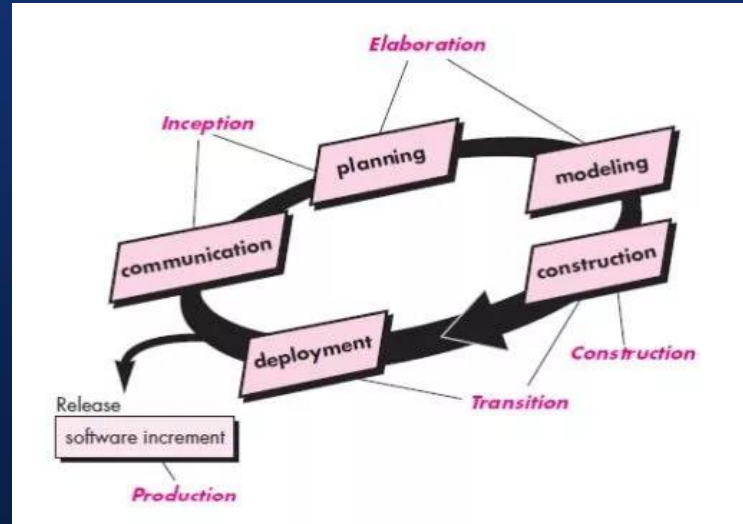
4. Transition –


- > Final project is released to public.
- > Transit the project from development into production.
- > Update project documentation.
- > Beta testing is conducted.
- > Defects are removed from project based on feedback from public.



5. Production –

- > Final phase of the model.
- > Project is maintained and updated accordingly



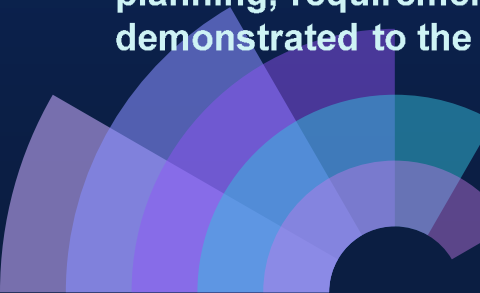


Agile Model



Agile Model



- The meaning of Agile is swift or versatile.
 - "Agile process model" refers to a software development approach based on iterative development.
 - Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning.
 - The project scope and requirements are laid down at the beginning of the development process.
 - Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.
 - Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks.
 - The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements.
 - Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.
- 

Phases of Agile Model

Following are the phases in the Agile model are as follows:

1. Requirements gathering
2. Design the requirements
3. Construction/ iteration
4. Testing/ Quality assurance
5. Deployment
6. Feedback




Phases of Agile Model



1. Requirements gathering: In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.

2. Design the requirements: When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.



Phases of Agile Model




3. Construction/ iteration: When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.

4. Testing: In this phase, the Quality Assurance team examines the product's performance and looks for the bug.

5. Deployment: In this phase, the team issues a product for the user's work environment.

6. Feedback: After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.



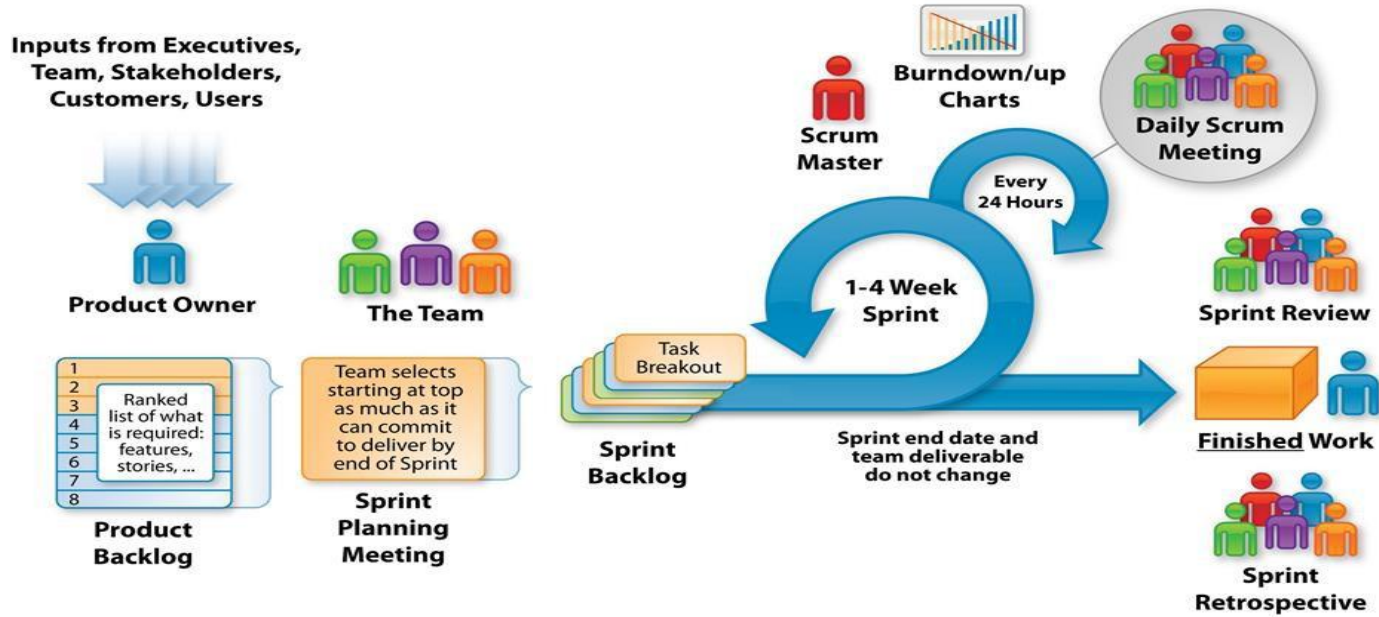
Agile Testing Methods:

1. Scrum
2. Crystal
3. Dynamic Software Development Method(DSDM)
4. Feature Driven Development(FDD)
5. Lean Software Development
6. eXtreme Programming(XP)



1. Scrum

The Agile - Scrum Framework



Scrum

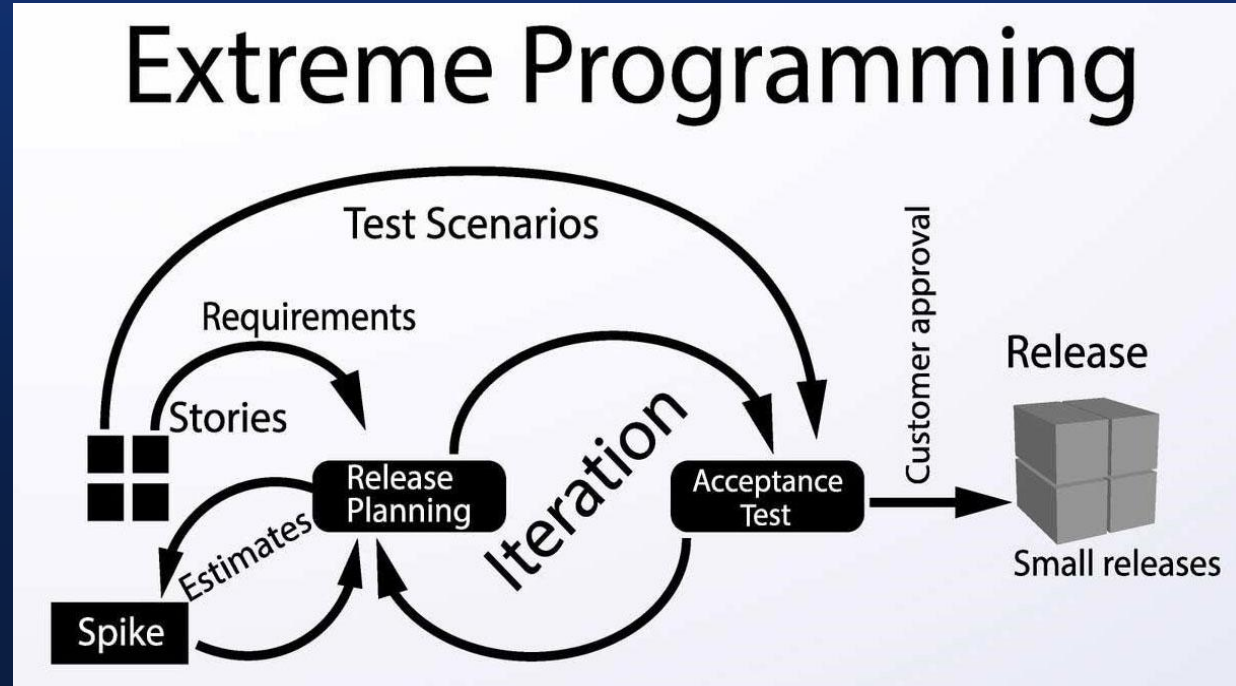
SCRUM is an agile development process focused primarily on ways to manage tasks in team-based development conditions.

There are three roles in it, and their responsibilities are:

1. Scrum Master: The scrum can set up the master team, arrange the meeting and remove obstacles for the process
2. Product owner: The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.
3. Scrum Team: The team manages its work and organizes the work to complete the sprint or cycle.

2. eXtreme Programming(XP)

This type of methodology is used when customers are constantly changing demands or requirements, or when they are not sure about the system's performance.



3. Crystal

There are three concepts of this method-

1. Chartering: Multi activities are involved in this phase such as making a development team, performing feasibility analysis, developing plans, etc.
2. Cyclic delivery: under this, two more cycles consist, these are:
 - A. Team updates the release plan.
 - B. Integrated product delivers to the users.
3. Wrap up: According to the user environment, this phase performs deployment, post-deployment.



4. Dynamic Software Development Method(DSDM):

DSDM is a rapid application development strategy for software development and gives an agile project distribution structure. The essential features of DSDM are that users must be actively connected, and teams have been given the right to make decisions. The techniques used in DSDM are:

1. Time Boxing
2. MoSCoW Rules
3. Prototyping

Dynamic Software Development Method(DSDM):

The DSDM project contains seven stages:


1. Pre-project
2. Feasibility Study
3. Business Study
4. Functional Model Iteration
5. Design and build Iteration
6. Implementation
7. Post-project





5. Feature Driven Development(FDD):

This method focuses on "Designing and Building" features. In contrast to other smart methods, FDD describes the small steps of the work that should be obtained separately per function.



6. Lean Software Development:

Lean software development methodology follows the principle "just in time production." The lean method indicates the increasing speed of software development and reducing costs. Lean development can be summarized in seven phases.

1. Eliminating Waste
2. Amplifying learning
3. Defer commitment (deciding as late as possible)
4. Early delivery
5. Empowering the team
6. Building Integrity
7. Optimize the whole

When to use the Agile Model?

1. When frequent changes are required.
2. When a highly qualified and experienced team is available.
3. When a customer is ready to have a meeting with a software team all the time.
4. When project size is small.

Advantage(Pros) of Agile Method:

1. Frequent Delivery
2. Face-to-Face Communication with clients.
3. Efficient design and fulfils the business requirement.
4. Anytime changes are acceptable.
5. It reduces total development time.





Disadvantages(Cons) of Agile Model:

- Due to the shortage of formal documents, it creates confusion and crucial decisions taken throughout various phases can be misinterpreted at any time by different team members.
 - Due to the lack of proper documentation, once the project completes and the developers allotted to another project, maintenance of the finished project can become a difficulty.
- 