**Week 1 Overview**

**Definition of Software Engineering**

The term software engineering is the product of two words, software, and engineering.

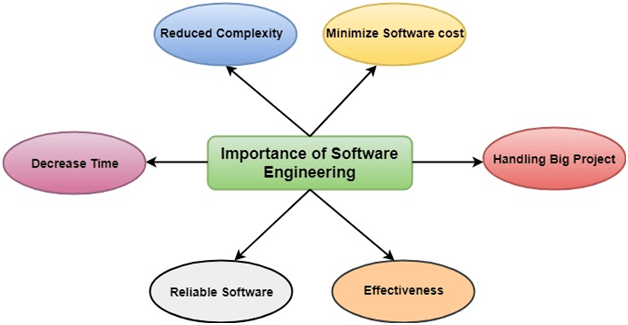
Software subsists of carefully-organized instructions and code written by developers on any of various particular computer languages.

Computer programs and related documentation such as requirements, design models and user manuals.

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

**Software Engineering** is an engineering branch related to the evolution of software product using well-defined scientific principles, techniques, and procedures. The result of software engineering is an effective and reliable software product.

**Software engineering** is an [engineering](https://en.wikipedia.org/wiki/Engineering) approach on a [software](https://en.wikipedia.org/wiki/Software) [development](https://en.wikipedia.org/wiki/Software_development) of systematic application.



**NEED OF SOFTWARE ENGINEERING**

The need of software engineering arises because of higher rate of change in user requirements and environment on which the software is working.

**Large software** - It is easier to build a wall than to a house or building, likewise, as the size of software become large engineering has to step to give it a scientific process.

**Scalability**- If the software process were not based on scientific and engineering concepts, it would be easier to re-create new software than to scale an existing one.

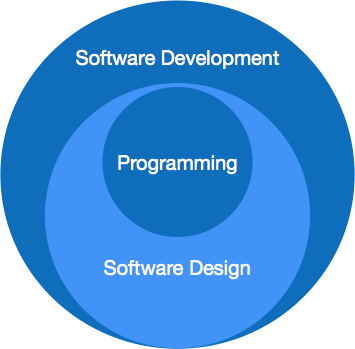
**Cost-** As hardware industry has shown its skills and huge manufacturing has lower down the price of computer and electronic hardware. But the cost of software remains high if proper process is not adapted.

**Dynamic Nature**- The always growing and adapting nature of software hugely depends upon the environment in which the user works. If the nature of software is always changing, new enhancements need to be done in the existing one. This is where software engineering plays a good role.

**Quality Management**- Better process of software development provides better and quality software product.

**Software paradigms:**

Software paradigms refer to the methods and steps, which are taken while designing the software. There are many methods proposed and are in work today, but we need to see where in the software engineering these paradigms stand. These can be combined into various categories, though each of them is contained in one another:



Programming paradigm is a subset of Software design paradigm which is further a subset of Software development paradigm.

### Software Development Paradigm

This Paradigm is known as software engineering paradigms where all the engineering concepts pertaining to the development of software are applied. It includes various researches and requirement gathering which helps the software product to build. It consists of –

* Requirement gathering
* Software design
* Programming

### Software Design Paradigm

This paradigm is a part of Software Development and includes –

* Design
* Maintenance
* Programming

### Programming Paradigm

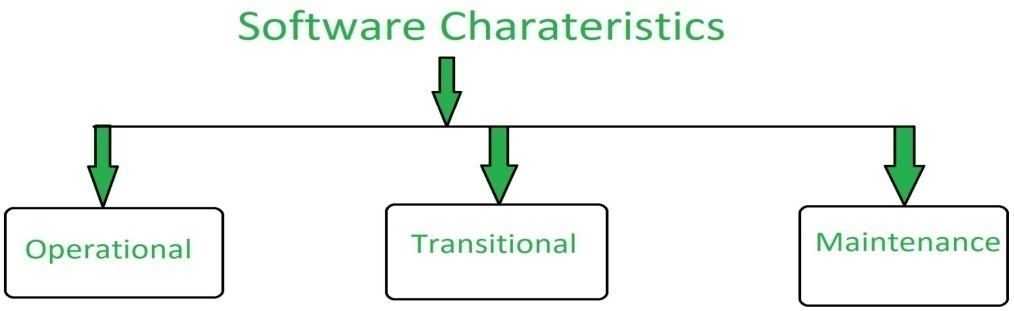
This paradigm is related closely to programming aspect of software development. This includes

* Coding
* Testing
* Integration

**Software product types: generic, customized;**

| S.No. | Generic software product development | Custom software development |
| --- | --- | --- |
| 1. | The generic software development is done for developing a general purpose software. | Customer software development is done to develop a software product as per the needs of particular customer. |
| 2. | In this development process, the software developers have to depict the end-users specifications. | In this development process, the the end-user requirements can be aggregated by communicating by them. |
| 3. | From designing and marketing perspective, this type of development is very difficult. | This development does not require marketing, because it is developed for appropriate group of users. |
| 4. | Large number of users may be using this kind of software. | This type of software is used by limited number of users. |
| 5. | Quality of the product is not a preference for generic software. | Quality is the main criterion in customer software product. Best quality of the product is focused for customer or company. |
| 6. | Development team controls the process of generic software development. | Customer determines the process of software development in this type of product. |
| 7. | Generally the software developed is economical. There may be some hidden costs such as installation and implementation cost. | Software product is of high cost as the particular product for customer is developed. |
| 8. | Example of generic software product development is Word-editing software. | Inventory control and management system are examples of customer software development. |

**Characteristics of good software**



**1.Operational:**  
In operational categories, the factors that decide the software performance in operations. It can be measured on:

* Budget
* Usability
* Efficiency
* Correctness
* Functionality
* Dependability
* Security
* Safety

**2.Transitional:**  
When the software is moved from one platform to another, the factors deciding the software quality:

* Portability
* Interoperability
* Reusability
* Adaptability

**3.Maintenance:**  
In this categories all factors are included that describes about how well a software has the capabilities to maintain itself in the ever changing environment:

* Modularity
* Maintainability
* Flexibility
* Scalability

**Challenges in software projects**

**Challenge 1: Extremely high  competition**

If your software company has a great idea, chances are another company has already thought of it. The competition is extremely high both at the local and international levels, and it affects software businesses in terms of pricing, customer reach and retention, etc. PMs have to work closely with business owners and other stakeholders to identify the right market segment and ensure the ROI of their software.

**Challenge 2: Old legacy systems**

Software companies often spend significant resources on maintaining and upgrading the old legacy systems. Having invested a lot of financial and human resources, stakeholders become resistant and don’t want to change the existing system, even when it no longer meets their needs. Instead of starting a battle and making stakeholders’ more resistant, PMs should find ways to alleviate their fears and convince them of all benefits a new solution will bring to their business.

**Challenge 3: High-level software expertise**

When it comes to software selection and implementation, the best variant for business owners is finding project managers with the relevant software expertise. And the more complex the software system is, the more experience and the more specific skill set will be required for its implementation (e.g. think of large ERP systems).

**Challenge 4: Third-party integration**

Modern companies are no longer interested in standalone solutions and look for third-party integration. In general, it looks like implementing multiple systems in one project (e.g. a PM implements a financial management system with accounting and reporting modules which interface with CRM and contract management software). This puts PMs under pressure and makes them improve their expertise and learn more about other software that integrates with the solution they are implementing.

**Challenge 5: Multiple-level users**

Most companies look for systems that allow different types of users – from basic users to strictly IT users. Project managers who are responsible for the system implementation, must be familiar with all types of users and know what user rights and permissions should be assigned to each.

**Challenge 6: Quality testing**

Successful system implementation requires numerous testing iterations to ensure that the final outcomes align with the desired results. Project managers need to make sure all bugs are discovered and all issues are fixed before the system goes live. This is essential to avoid additional rework and ensure customer satisfaction.

**Factors that influence software development;**

**1.People –**  
When we refer to people in software development, we actually mean hierarchy of roles in software development such as managers, team leaders, senior developers etc. They are ones who form development team and work in development of software. The order of priority of people among other dependency factors is **first** (highest). Software development is people-centric activity.  
Software development requires good managers, people who understand psychology of people and provide good leadership.

A good manager does not ensure success of project, but can increase probability of success. A good manager gives priority to areas like proper selection, training, compensation, career development, work culture, development team’s efficiency etc.

**2.Product –**  
A product is solution of problem that is handed to user who is struggling with problem. The priority of product among other dependency factors is **second** that is after **people**. The product or solution to problem is constructed based on requirements, which are objectives and scope work of problem. This helps managers to select best approach within constraints imposed by delivery deadlines, budgetary restrictions, personnel availability, technical interfaces etc. Without well defined requirements, it may be impossible to define reasonable estimates of cost, development time and schedule for project.

**3.Process –**  
The process is way in which we produce software. It provides framework from which comprehensive plan for software development can be established. If process is weak, end product will undoubtedly suffer. The priority of process among other dependency factors is **third**, after **people** and **product**, however, it plays very critical role for success of project. There are many life cycle models and process improvements models. A suitable process model is determined on basis of requirements specification, development team, user, type of project and associated risks of project.

**4.Project**   
A proper planning is required to monitor status of development and to control complexity. For project to be successful, it should be within budget, follow policies and requirements defined, and completed within timeline stated. Most of projects overrun cost by 100% of budget and are not delivered on time. The priority of project among other dependency factors is  **last**(lowest). In order to manage successful project, we must understand and prevent associated risks. Concrete requirements should be defined and frozen, wherever possible. Unnecessary changes should not be made to avoid software surprises. Software surprises are always risky and should be minimized. A planning mechanism should be constructed to caution from from occurrence of any surprises.

**Success software process:**



**Stage 1: Planning and Requirement Analysis**

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

**Stage 2: Defining Requirements**

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an **SRS (Software Requirement Specification)** document which consists of all the product requirements to be designed and developed during the project life cycle.

**Stage 3: Designing the Product Architecture**

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

**Stage 4: Building or Developing the Product**

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

**Stage 5: Testing the Product**

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

**Stage 6: Deployment in the Market and Maintenance**

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

**Software Development :**

In this process, designing, programming, documenting, testing, and bug fixing is done.

**Components of Software :**   
There are three components of the software: These are : Program, Documentation, and Operating Procedures.

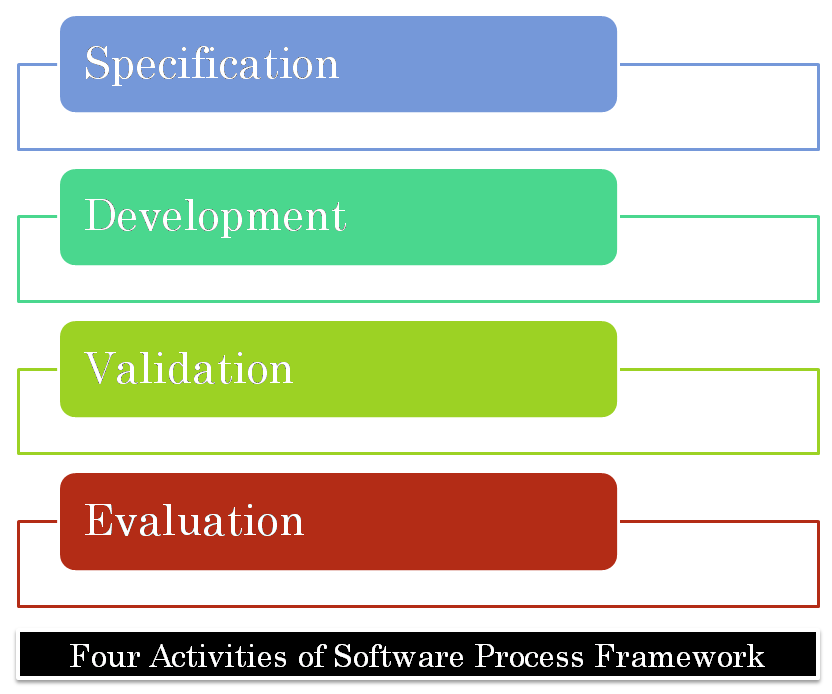
1. **Program –**  
   A computer program is a list of instructions that tell a computer what to do.
2. **Documentation –**   
   Source information about the product contained in design documents, detailed code comments, etc.
3. **Operating Procedures –**  
   Set of step-by-step instructions compiled by an organization to help workers carry out complex routine operations.

**Software Crisis :**

1. **Size and Cost –**  
   Day to day growing complexity and expectation out of software. Software are more expensive and more complex.
2. **Quality –**  
   Software products must have good quality.
3. **Delayed Delivery –**  
   Software takes longer than the estimated time to develop, which in turn leads to cost shooting up.

# Software Process

A software process (also known as software methodology) is a set of related activities that leads to the production of the software. These activities may involve the development of the software from the scratch, or, modifying an existing system.



**Any software process must include the following four activities:**

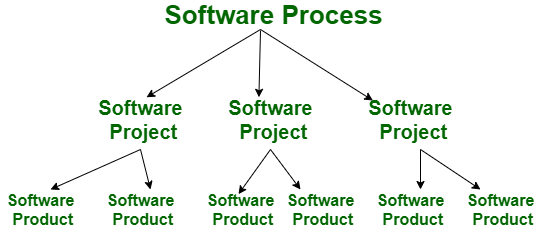
1. **Software specification** (or requirements engineering): Define the main functionalities of the software and the constraints around them.
2. S**oftware design and implementation**: The software is to be designed and programmed.
3. **Software verification and validation**: The software must conform to its specification and meet the customer's needs.
4. **Software evolution** (software maintenance): The software is being modified to meet customer and market requirements changes.

In practice, they include sub-activities such as requirements validation, architectural design, unit testing, …etc.

**Product:**   
In the context of software engineering, Product includes any software manufactured based on the customer’s request. This can be a problem solving software or computer based system. It can also be said that this is the result of a project.

**Process:**   
Process is a set of sequence steps that have to be followed to create a project. The main purpose of a process is to improve the quality of the project. The process serves as a template that can be used through the creation of its examples and is used to direct the project.

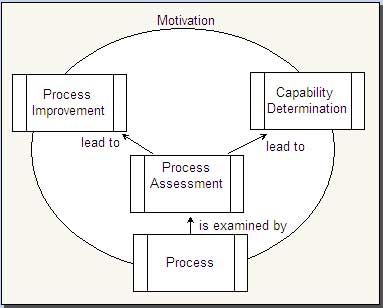
The main difference between a process and a product is that the process is a set of steps that guide the project to achieve a convenient product. while on the other hand, the product is the result of a project that is manufactured by a wide variety of people.



Let’s see the difference between Product and Process:-

| **S.NO** | **Product** | **Process** |
| --- | --- | --- |
| 1. | Product is the final production of the project. | While the process is a set of sequence steps that have to be followed to create a project. |
| 2. | A product focuses on the final result. | Whereas the process is focused on completing each step being developed. |
| 3. | In the case of products, the firm guidelines are followed. | In contrast, the process consistently follows guidelines. |
| 4. | A product tends to be short-term. | Whereas the process tends to be long-term. |
| 5. | The main goal of the product is to complete the work successfully. | While the purpose of the process is to make the quality of the project better. |

**Process Assessment and improvement:**



The process assessment leads to process capability determination and process improvement. Process capability determination is an organized assessment, which analyzes the software processes in an organization. Assessment includes five phases initiating, diagnosing, establishing, acting and learning.

SPICE (Software Process Improvement and Capability Determination) is a standard used for both process improvement and process capability determination. SPICE provides a framework for assessing the software process and is used by the organizations involved in planning, monitoring, developing, managing, and improving acquisitions. It is carried out in accordance with the **International** **Organization for Standardization (ISO)** and **International Electro-technical** **Committee (IEC),** which are used together and known as **ISO/IEC 15504.**

Software processes are assessed to ensure their ability to control the cost, time and quality of software. Assessment is done to improve the software process followed by an organization.

**Software Process Improvement (SPI) Cycle includes:**

* Process measurement
* Process analysis
* Process change

**Software engineering ethics:**

Software engineering ethics can be approached from three directions. First, it can describe the activity of software engineers making practical choices that affect other people in significant ways. Second, it can be used to describe a collection of principles, guidelines, or ethical imperatives that guide or legislative action, and third, it can be used to name a discipline that studies the relationship between the other two senses of ethics.

Software engineering ethics is clearly both an activity and a body of principles. The discipline of software engineering ethics that studies this activity and formalizes these principles, however, is in its infancy

**Software engineers shall adhere to the following Eight Principles:**

**PUBLIC** - Software engineers shall act consistently with the public interest.

**CLIENT AND EMPLOYER -** Software engineers shall act in a manner that is in the best interests of their client or employer and that is consistent with the public interest.

**PRODUCT** - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

**JUDGMENT -** Software engineers shall maintain integrity and independence in their professional judgment.

**MANAGEMENT** - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

**PROFESSION** - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

**COLLEAGUES** - Software engineers shall be fair to and supportive of their colleagues.

**SELF** - Software engineers shall participate in lifelong learning regarding the practice of their profession and promote an ethical approach to the practice of the profession.