Software engineering is a technique through which we can develop or create software for computer systems or any other electronic devices. It is a systematic, scientific and disciplined approach to the development, functioning, and maintenance of software.

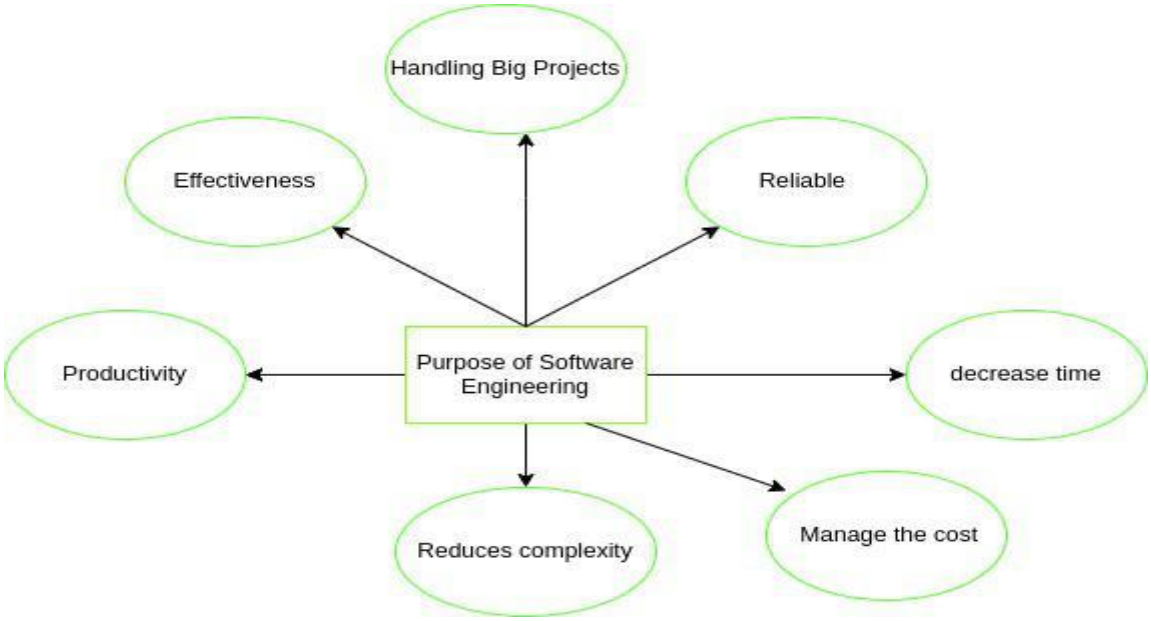
Basically, Software engineering was introduced to address the issues of low-quality software projects. Here, the development of the software uses the well-defined scientific principal method and procedure.

In other words, software engineering is a process in which the need of users are analysed and then the software is designed as per the requirement of the user. Software engineering builds this software and application by using designing and programming language.

In order to create complex software, we need to use software engineering techniques as well as reduce the complexity we should use abstraction and decomposition, where abstraction describes only the important part of the software and remove the irrelevant things for the later stage of development so the requirement of the software becomes simple. Decomposition breakdown of the software in a number of modules where each module procedure as well defines the independent task

Need of Software Engineering:

- **Handling Big Projects:** A corporation must use a software engineering methodology in order to handle large projects without any issues.
- **To manage the cost:** Software engineering programmers plan everything and reduce all those things that are not required.
- **To decrease time:** It will save a lot of time if you are developing software using a software engineering technique.
- **Reliable software:** It is the company's responsibility to deliver software products on schedule and to address any defects that may exist.
- **Effectiveness:** Effectiveness results from things being created in accordance with the standards.
- **Reduces complexity:** Large challenges are broken down into smaller ones and solved one at a time in software engineering. Individual solutions are found for each of these issues.
- **Productivity:** Because it contains testing systems at every level, proper care is done to maintain software productivity.



Difference between Custom and Generic Software:

Generic software is an off-the-shelf product designed for many consumers and can meet many clients' general requirements. An example of generic software is a word or spread sheet document.

Custom software is a bespoke design developed to meet one client's specific needs, based on the budget and requirements predefined by them. It's meant to be operated by one user or a group of users and meets the needs not fulfilled by off-the-shelf software.

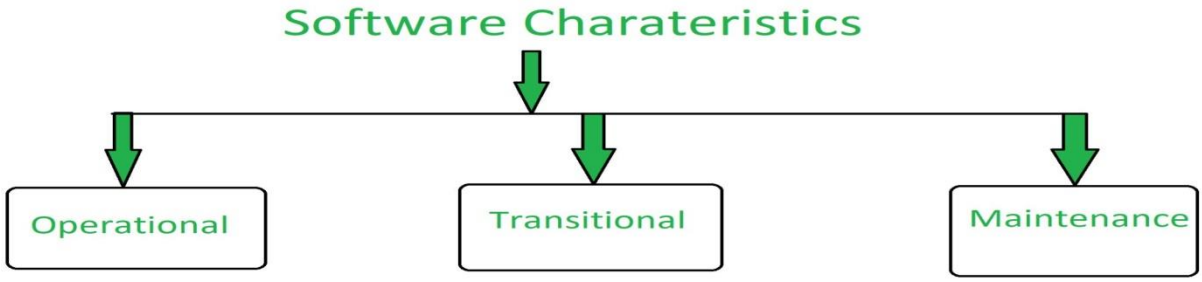
An example of custom software is B2B accounting software or a student portal for a university.

Which is better?

There's no better or worse option. It really depends on your requirements and budget.

SL NO.	Generic Software	Custom software
1	The generic software development is done for developing 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 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.

Software is treated as a good software by the means of different factors. A software product is concluded as a good software by what it offers and how well it can be used. The factors that decide the software properties are divided into three categories: Operational, Transitional, and Maintenance. These are explained as following below.



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

Most Common Software Development Challenges & How to Solve Them:

Lack of Guidance from Management

One of the most common software development challenges is the lack of guidance from the project managers. Developers need clear instructions to be able to do their job effectively, and if they're left in the dark, it can lead to chaos and confusion. This can cause delays in the project and affect its overall software quality.

The best method to overcome this challenge is to establish a clear and concise project plan. This document should outline all of the critical tasks that need to be completed and who is responsible for each one. It should also include a timeline so developers can track their progress and ensure they're on schedule. If the project manager changes their mind about something, it's important to update the project plan, so everyone is on the same page.

Difficulty Estimating Time and Resources

The difficulty of estimating time and resources for software development projects is another common problem. It's important to have a realistic timeline, given the constraints of budgets and timelines. This way, developers know how much time they can spend on enhancing a certain feature or working on software testing and bug fixes, for instance. If a developer has too little time to work on the part of the project, it may not function properly or have features that are lacking. If we underestimate the amount of time it takes to complete a task, there will be delays in the project, and other members of the team may run out of work. On the other hand, if it takes longer than expected, then the project will stretch into an uncomfortable timeframe.

One way to avoid this common challenge is to break larger tasks into smaller chunks so they're more manageable. Developers should also leverage project management tools or time-tracking tools like Toggl or Harvest to ensure they're keeping track of how much effort they're investing in each task. In addition, setting deadlines for work will help developers prioritize their tasks and know which ones need to be done by a specific date. They should also factor in any potential software development problems that may arise and have a backup plan ready to go.

Lack of Resources for the Software Development Process

Lack of budget and resources is another common software development challenge. Part of it is due to the increased complexity of software projects, which takes a lot more time and money to complete. In addition to these constraints, developers may also be faced with a lack of access to computers for testing, a shortage of software engineers, or a lack of appropriate technology for their projects.

The best way to work through these software development challenges is by leveraging free assets from the web. Since many different types of assets are available on the internet – from fonts and graphics to code libraries and scripts – it's possible to find something that addresses your specific needs without having to spend anything at all. Additionally, development teams need to be vocal about the resources they need in order to do their job properly. If they don't have the right tools, it will be impossible for them to meet deadlines or produce a high-quality product. One more solution for this problem is not to request too many features or enhancements at once. The project teams should also seek out areas where they can cut costs and save time and use those funds and hours on other aspects of the project.

Defining the Requirements of the Software Development Projects

One of the most time-consuming challenges for software developers is to define the requirements. Simply, this means figuring out what the product should do and how it should work. The requirements need to be clear, concise, and complete so that developers have a good understanding of what they need to do. Unfortunately, it's often difficult to get a clear picture of what's required and even more difficult to translate that into specific instructions. If these requirements are not well-defined, it can lead to confusion and frustration on the part of the developers, which will ultimately delay the project.

It's essential to have clear software development initiatives and an exemplary process for defining the final agreed-upon requirements. This way, the development teams will know at a

glance what they need to do, and their work won't get unnecessarily stalled. One useful way to address this problem is by holding discussions with customers or clients when there are any questions or unclear requirements during the project cycle. Having these discussions will allow for greater clarity on what needs to be done and when things need to be completed. Moreover, the project team should create prototypes and test them with the customers for feedback.

Miscommunication with Customers/Stakeholders

Another major challenge for software developers is miscommunication with customers and stakeholders. This can happen for many reasons, such as a lack of communication channels, misunderstandings about the requirements, or failures to document the project properly. Also, it can result from a lack of communication between members of the software development teams, which means it's crucial to establish an effective communication process. Whatever the cause may be, miscommunication can lead to delays in the project and a poor final product.

To prevent these problems from occurring, we should have an open line of communication with customers and stakeholders. This means establishing regular communication channels (such as weekly meetings or daily updates), being clear about the requirements, and documenting everything that happens during the project. In addition, developers should ask lots of questions and seek clarification whenever there is any doubt. By taking these measures, miscommunication can be reduced, and the development can proceed smoothly.

Strict Time Constrains

The other challenge for software developers is dealing with strict time constraints. In many cases, deadlines are agreed upon before they even know what work needs to be done, leading to a lot of pressure and frustration. Even if this issue doesn't happen initially, it will most likely arise later in the project when the software development team encounters problems that might take longer than expected to address.

To deal with this challenge, companies better put an emphasis on good time management. Developers and project managers should set realistic deadlines and clear expectations from the start. Moreover, it's advisable for software development teams to take the necessary time off for rest and recovery and build in time for contingencies. This way, if unexpected events arise or the team finds themselves running behind, they can make up for lost time quickly without putting the project at risk.

The Complexity of Software Projects

The complexity of projects is another common challenge that developers commonly face. This is often due to a large number of dependencies and interactions between different parts of the software system. In addition, the number of potential problems that can occur during development can be daunting. As a result, it's often difficult for developers to know where to start and how to proceed.

To address this challenge, developers need to understand the system they are working on clearly. They should also break down the project into manageable tasks and establish a plan for dealing with potential problems. Moreover, they should always stay in close

communication with their team members so that everyone is aware of what is happening and can help out when needed. Finally, they should take the time to devise strategies for dealing with false starts. This will help them stay organized and stay on track with the development process.

Finding Qualified Talents

Finding qualified talents is one of the most significant challenges in software development for enterprises. This can be difficult for a number of reasons, such as a shortage of qualified workers, the high cost of hiring, or the difficulty of finding the right person for the job. In addition, many software development companies are reluctant to hire new graduates because they often have little experience and lack the necessary skills.

To overcome this challenge, companies need to be more proactive in their search for talent. They should identify the skills they need and target candidates who have those skills. They should also create an attractive job offer that includes benefits and a competitive salary. In addition, they should actively promote their company and its culture. By doing these things, companies can attract more talent and continue to grow their businesses.

Moreover, companies can also consider using recruitment agencies or online platforms that provide ready-made resumes for professionals with different qualifications and skillsets. Or they should be more flexible in talent acquisition by giving more opportunities to interns. Currently, more and more many software companies are outsourcing software development work to third-party companies, which can be a more cost-effective way to find the right talent.

Testing and Debugging

Last but not least, testing and debugging is one of the major challenges in software development that software testing teams often encounter. This usually involves identifying and fixing the errors in the code. In addition, it can be challenging to determine the source of the error and how to fix it.

To address this challenge, developers need to be meticulous in their work. They ought to understand the system they are working on and the code they are writing. In addition, they should use automated debugging tools to help them find and fix errors. They should also establish a testing plan and test their code thoroughly before releasing it to production. By doing these things, you can ensure that your code is error-free and ready for use while also saving time and effort.

Maintaining the Competitive Edge

One of the increasing software development challenges in the software industry is maintaining the competitive edge. This is often due to the fast-paced and ever-changing technology landscape. In addition, there are many companies that offer similar products and services.

Software is the set of instructions in the form of programs to govern the computer system and to process the hardware components. To produce a software product the set of activities is used. This set is called a software process.

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.

There are four basic key process activities:

- **Software Specifications –**
  In this process, detailed description of a software system to be developed with its functional and non-functional requirements.

- **Software Development –**
  In this process, designing, programming, documenting, testing, and bug fixing is done.

- **Software Validation –**
  In this process, evaluation software product is done to ensure that the software meets the business requirements as well as the end users needs.

- **Software Evolution –**
  It is a process of developing software initially, then timely updating it for various reasons.

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.

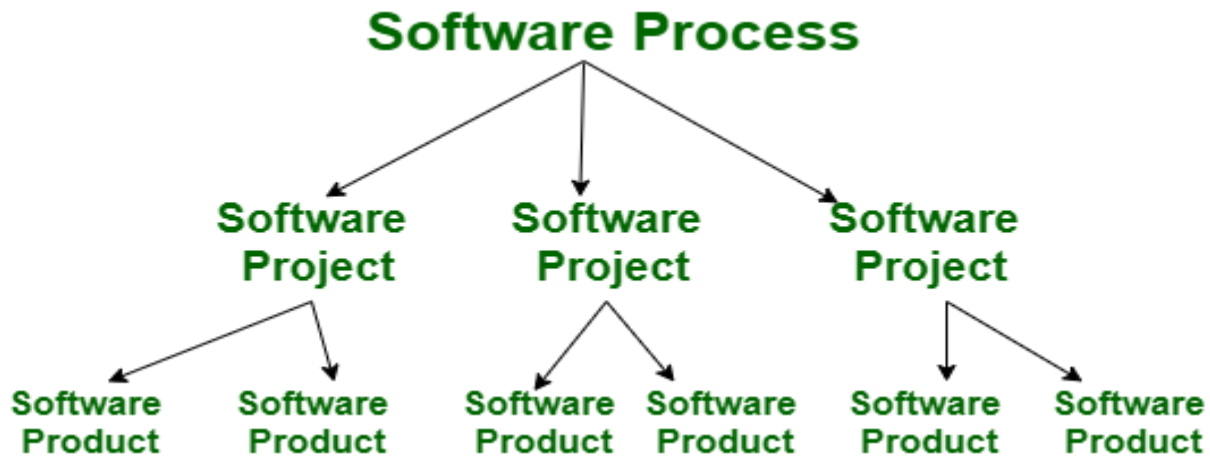
Product:

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.



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	A product tends to be short-term.	Whereas the process tends to be long-term.
4	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.
5	Product is created based on the needs and expectations of the customers.	A process serves as a model for producing various goods in a similar way.

Software Process Assessment is a disciplined and organized examination of the software process which is being used by any organization bases the on the process model. The Software Process Assessment includes many fields and parts like identification and characterization of current practices, the ability of current practices to control or avoid significant causes of poor (software) quality, cost, schedule and identifying areas of strengths and weaknesses of the software.

Types of Software Assessment:

- **Self-Assessment :** This is conducted internally by the people of their own organisation.

- **Second Party assessment:** This is conducted by an external team or people of the own organisation are supervised by an external team.

- **Third Party assessment:**

  In an ideal case Software Process Assessment should be performed in a transparent, open and collaborative environment. This is very important for the improvement of the software and the development of the product. The results of the Software Process Assessment are confidential and are only accessible to the company. The assessment team must contain at least one person from the organization that is being assessed.

Software Engineering Ethics:

Like other engineering disciplines, software engineering is carried out within a social and legal framework that limits the freedom of people working in that area. As a software engineer, you must accept that your job involves wider responsibilities than simply the application of technical skills.. Some of these are:

1. **Confidentiality** You should normally respect the confidentiality of your employers or clients regardless of whether or not a formal confidentiality agreement has been signed.
2. **Competence** You should not misrepresent your level of competence. You should not knowingly accept work that is outside your competence.
3. **Intellectual property rights** You should be aware of local laws governing the use of intellectual property such as patents and copyright. You should be careful to ensure that the intellectual property of employers and clients is protected.
4. **Computer misuse** You should not use your technical skills to misuse other people's computers. Computer misuse ranges from relatively trivial (game playing on an employer's machine) to extremely serious (dissemination of viruses or other malware).

Professional societies and institutions have an important role to play in setting ethical standards. Organizations such as the ACM, the IEEE (Institute of Electrical and Electronic Engineers), and the British Computer Society publish a code of professional conduct or code of ethics. Members of these organizations undertake to follow that code when they sign up for membership. These codes of conduct are generally concerned with fundamental ethical behaviour.