# UNIT 1. Introduction to Software Engineering

----------------------------------------------------------------------

*Learning objectives*

- to acquire basic knowledge about software engineering and its types
- to understand if all software requires software engineering

*Key words and phrases. Give Russian equivalents and remember the meanings of the key words and phrases used in the text*

Concept; to be complicated; application; definition; tools; security risks; vulnerability; operational software engineering; transitional software engineering; recurrent; lifecycle; implementation; maintenance; retirement; to be congruent; stand-alone applications; interactive transaction-based applications; embedded control systems; batch processing systems

*Read the following text and do the exercises given after it*

Software engineering is a concept in and of itself, but to better understand it, you need to know what each part of the term means before you can fully understand how they operate together. It can be difficult to understand, even though it does seem straightforward. That is because the pieces are more complicated than many believe - and working with software engineering for an application is difficult and time-consuming. Software engineering has two parts: software and engineering.

**Software** is a collection of codes, documents, and triggers that does a specific job and fills a specific requirement.

**Engineering** is the development of products using best practices, principles, and methods.

What is Software Engineering?

❖ It is a branch of engineering that deals with the development of software products. It operates within a set of principles, best practices, and methods that have been carefully honed throughout the years, changing as software and technology change.

Software engineering leads to a product that is reliable, efficient, and effective at what it does. While software engineering can lead to products that do not do this, the product will almost always go back into the production stage. So, what is the complete definition of software engineering?

The IEEE fully defines software engineering as:
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.
What the software engineering meaning doesn't explain is that everything that has been software engineered needs to work on real machines in real situations, not within.

Software engineering starts when there is a demand for a specific result or output for a company, from an application. From somewhere on the IT team, typically the CIO, there is a request put into the developer to create some sort of software. The software development team breaks down the project into the requirements and steps. Sometimes, this work will be farmed out to independent contractors, vendors, and freelancers. When this is the case, software engineering tools help to ensure that all of the work done is congruent and follows best practices.

How do developers know what to put into their software? They break it down into specific needs after conducting interviews, collecting information, looking into the existing application portfolio, and talking to IT leaders. Then, they will build a roadmap of how to build the software. This is one of the most important parts because much of the "work" is completed during this stage - which also means that any problems typically occur here as well.

The true starting point is when developers begin to write code for the software. This is the longest part of the process in many cases as the code needs to be congruent with current systems and the language used in them. Unfortunately, these problems often aren't noticed until much later on in the project and then rework needs to be completed.

The code should be tested as it is written and once it has been completed – at all parts of the life cycle. With software engineering tools, you will be able to continuously test and monitor.

## Software Engineering Basics

The true work of software engineering begins before the product has even been designed – and the software engineering basics dictate that it continues long after the "work" has been completed. It all begins with a thorough and complete understanding of what your software needs to have – this includes what the software needs to do, the system in which it needs to operate, and all of the security that it entails. Security is one of the software engineering basics because it is so essential to all aspects of development. Without tools to help you better understand how your code is being built and where any security problems may fall, your team can easily become lost in the development stage.

Software engineering design basics require creating the instructions for the computer and the systems. Much of this will take place at the coding level by professionals who have comprehensive training. Still, it is important to understanding that software engineering isn't always a linear process, which means that it requires thorough vetting once it has been completed.

Not all software requires software engineering. Simplistic games or programs that are used by consumers may not need engineering, depending on the risks associated with them. Almost all companies do require software engineering because of the high-risk information that they store and security risks that they pose.

Software engineering helps to create customized, personalized software that should look into vulnerabilities and risks before they even emerge. Even when the software engineering principles of safety aren't required, it can also help to reduce costs and improve customer experience.

## Types of Software Engineering

Software engineering studies the design, development, and maintenance of software as an umbrella definition. Still, there are different types of software engineering that a company or product may need. Problems tend to emerge when software is low-quality or isn't properly vetted before deployment.

There has been a lot of demand for software engineers because of the rate of change in user requirements, statutes, and the platforms we use. Software engineering works on a few different levels:

*Operational Software Engineering:* Software engineering on the operational level focuses on how the software interacts with the system, whether or not it is on a budget, the usability, the functionality, the dependability, and the security.

*Transitional Software Engineering:* This type focuses on how software will react when it is changed from one environment to another. It typically takes some scalability or flexibility in the development. *Software Engineering Maintenance:* Recurrent software engineering focuses on how the software functions within the existing system, as all parts of it change.

Software engineering functions at all parts of the software development lifecycle, including analysis, design, development, testing, integration, implementation, maintenance, and even retirement.

It is important to understand that software engineering isn't a new practice, but it is constantly changing and can feel new on a regular basis. Software is used in everything around us, so it is important to ensure that all software is working properly. If it does not, it can result in loss of money, loss of reputation, and even in some cases, loss of life.

[https://www.castsoftware.com/glossary/what-is-software-engineering-definition-types-of-basics-introduction]

## 1. Text-based Assignments

*1.1. Make nouns from the following verbs according to the model and translate them*
*Verb+-tion (-ation)*

Inform, create, connect, integrate, explore, prepare, destine, realize, associate, implement, operate.

*1.2. Give English equivalents of the following words and word combinations:*

Казаться простым; кропотливый, занимающий много времени; конкретное требование; оттачиваться на протяжении многих лет; полное определение; проблемы безопасности; тщательная проверка; ассоциироваться с чем-л.; сократить расходы; появляться; всеохватывающее определение; потребность; взаимодействовать с системой; операционное ПО; переходная *(переход с одной платформы на другую)* разработка ПО; техническое обслуживание; количественный подход; отправная точка.

*1.3. Match the following words with their definitions:*

| | |
|---|---|
| 1. High technology | a) the range of operations that can be run on a computer or other electronic system |
| 2. ngineering | b) come into existence or greater prominence |
| 3. functionality | c) take or use another instead of |
| 4. change | d) instructions for a computer in some programming language, often machine language |
| 5. emerge | e) advanced technological development, especially in electronics |
| 6. code | f) a field of study or activity concerned with modification or development in a particular area |
| 7. application developer | g) a position or stage on a scale of quantity, extent, rank, or quality |
| 8. level | h) a person who writes computer programs to meet specific requirements |

*1.4. Answer the following questions on the text*

1) What does software engineering deal with?
2) What is the starting point for developers?
3) Does all software require software engineering?
4) How can you explain the importance of software engineering?
5) What are the types of software engineering?
6) Software engineering functions at all parts of the software development, doesn't it?

*1.5. Read the text again and decide if the following statements are true or false.*

1) Software engineering is a branch of engineering that deals with the development of software products and security problems are the most important for developers.
2) Software engineering leads to a product that is reliable, efficient, and effective at what it does.
3) All software requires software engineering.
4) Problems tend to emerge even if software is high-quality and is properly vetted before deployment.

5) Software engineering functions at four parts of the software development lifecycle, including analysis, design, development, testing.
6) Software engineering isn't a new practice, but it is constantly changing and can feel new on a regular basis.
7) Software engineering isn't always a linear process, which means that it requires thorough vetting once it has been completed.

## 2. Focus on Grammar

*2.1. Choose one of the verbs in brackets. Put them into the necessary form to complete the following sentences*

1) A lot of humans (be, have, do) dependent on technology today, which will (be, have, do) bad to them.
2) I (be, have, do) had my iPad for years now and I (be, have, do) very happy with it.
3) Because I (be, have, do) not have the chance to speak to my boss yesterday I (be, have, do) to text her in Viber.
4) The 21st century (be, have, do) the age of cutting-edge technologies.
5) They (do, be, have) doing research work on the latest applications for mobile devices.
6) All books can (be, do, have) read online.
7) We (be, have, do) not see any downsides in using personal computers at all.
8) They (be, have, do) surfing the Internet all day yesterday.
9) Every day Linda (be, do, have) a lot of exercise to keep fit.
10) Safari browser online tutorial (be, do, have) provided the user with help and support in using it.

*2.2. Identify passive structures and translate the sentences*

Все времена в страдательном залоге образуются из вспомогательного глагола to be в соответствующем лице числе и времени и смыслового глагола в форме причастия прошедшего времени /Participle II/.

1) This method has been referred to in an earlier paper.
2) I do not think this instrument can be relied upon.

3) In operational categories, the factors that decide the software performance in operations. It can be measured on: budget, usability, efficiency and correctness.
4) If the job is entered without errors it will be chosen for execution.
5) This electronic equipment has been designed for speeding up production.
6) Business variables have been and are being expressed as mathematical functions and are being statistically analyzed.
7) Even when the software engineering principles of safety aren't required, it can also help to reduce costs and improve customer experience.
8) This interview was also recorded as a video podcast. Check out the video on the Software Daily YouTube channel.
9) The code should be tested as it is written and once it has been completed – at all parts of the life cycle.

## 3. Discussion

*3.1. Possible topics for discussion*

1) What are the attributes of good software? What is your idea of it?
2) What is the difference between software engineering and computer science?
3) Do you agree? Why/why not?
4) What is interesting for you in this branch of engineering?

*3.2. Write a short summary of the text about software engineering*

*3.3. Make up and dramatize a dialogue using the top interview questions for software engineers*

Hiring a software engineer is a process that should be approached carefully and with deliberation. A good software engineer will help your company grow, but one that does not have the right skills or a good work ethic can slow down and hinder your growth.

Therefore, you should know the best questions to ask during the hiring process to successfully recruit software engineers.