# Software Engineering Assignment

## Module : 1 (SDLC)

### ❖ What is Software? What is software engineering?

Software is a set of instructions, data or programs used to operate computers and execute specific tasks. It is the opposite of hardware, which describes the physical aspects of a computer. Software is a generic term used to refer to applications, scripts and programs that run on a device.

Without software, most computers would be useless. For example, a web browser is a software application that allows users to access the internet. Without the web browser software, reading this page on Webopedia wouldn't be possible. An operating system (OS) is a software program that serves as the interface between other applications and the hardware on a computer or mobile device. TCP/IP is built into all major operating systems to allow computers to communicate over long distance networks. Without the OS or the protocols built into it, it wouldn't be possible to access a web browser.

Let us first understand what software engineering stands for. The term is made of two words, software and engineering. **Software** is more than just a program code. A program is an executable code, which serves some computational purpose. Software is considered to be a collection of executable programming code, associated libraries and documentations. Software, when made for a specific requirement is called a software product. **Engineering** on the other hand, is all about developing products, using well-defined, scientific principles and methods.

➔ **IEEE defines software engineering as:**

The application of a systematic,disciplined,quantifiable approach to the development,operation and maintenance of software; that is, the application of engineering to software.

# ❖ Explain types of software

The two main categories of software are application software and system software. An application is software that fulfill a specific need or performs tasks. System software is designed to run a computer's hardware and provides a platform for applications to run on top, and some other related types of software are as follows.

## 1. <u>Application software.</u>

The most common type of software, application software, is a computer software package that performs a specific function for a user, or in some cases, for another application. An application can be self-contained, or it can be a group of programs that run the application for the user. Examples of modern applications include office suites, graphics software, databases and database management programs, web browsers, word processors, software development tools, image editors and communication platforms.

## 2. <u>System Software</u>

These software programs are designed to run a computer's application programs and hardware. System software coordinates the activities and functions of the hardware and software. In addition, it controls the operations of the computer hardware and provides an environment or platform for all the other types of software to work in. The OS is the best example of system software; it manages all the other computer programs. Other examples of system software include the firmware, computer language translators and system utilities.

## 3. <u>Driver Software</u>

Also known as device drivers, this software is often considered a type of system software. Device drivers control the devices and peripherals connected to a computer, enabling them to perform their specific tasks. Every device that is connected to a computer needs at least one device driver to function. Examples include software that comes with any nonstandard hardware, including special game controllers, as well as the software that enables standard hardware, such as USB storage devices, keyboards, headphones and printers.

## 4. <u>Middleware</u>

The term middleware describes software that mediates between application and system software or between two different kinds of application software. For example, middleware enables Microsoft Windows to talk to Excel and Word. It is also used to send a remote work request from an application in a computer that has one kind of OS, to an application in a computer with a different OS. It also enables newer applications to work with legacy ones.

**5. Programming Software**

Computer programmers use programming software to write code. Programming software and programming tools enable developers to develop, write, test and debug other software programs. Examples of programming software include assemblers, compilers, debuggers and interpreters.

# ❖ What is SDLC? Explain each phase of SDLC

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares. The SDLC aims to produce high-quality software that meets or exceeds customer expectations, reaches completion within time and cost estimates.

SDLC is a process followed for a software project, within a software organisation. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.



- Phase 1: Requirement collection and analysis
- Phase 2: Feasibility study
- Phase 3: Design
- Phase 4: Coding
- Phase 5: Testing
- Phase 6: Installation/Deployment
- Phase 7: Maintenance

(note : Basically 6 phases, Installation/Deployment are included in Maintenance.)

## ➔ 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.

## ➔ 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.

## ➔ 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.

## ➔ Coding and Build 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 organised manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organisation 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.

## ➔ 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 The QA (Quality Assurance) and testing teams may discover faults or defects, which they report to developers. The development team fixes the bug and sends it back to QA for another round of testing. This procedure is repeated retesting and regression testing) until the program is bug-free, stable, and meets the system's business requirements. Testers refer to the SRS document to ensure that the software meets the customer's standard.
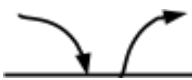
➔ **Deployment in the Market and Maintenance**

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 organisation. 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.

## ❖ What is DFD? Create a DFD diagram on Flipkart.

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

It shows how data enters and leaves the system, what changes the information, and where data is stored. The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called a data flow graph or bubble chart.

| Symbols | Name | Function |
|---|---|---|
| ⟶ | Data Flow | Line shows the flow of data into or out of a process or data store. |
| ⬭ | Process | Circle shows a process that transforms data inputs into data outputs. |
| ▭ | External Entity | Source of System inputs or sink of System outputs. |
| ⤵⤴ | Data Store | The arrow heads indicate net inputs and net outputs to store. |

**Symbols of Data Flow Diagrams**
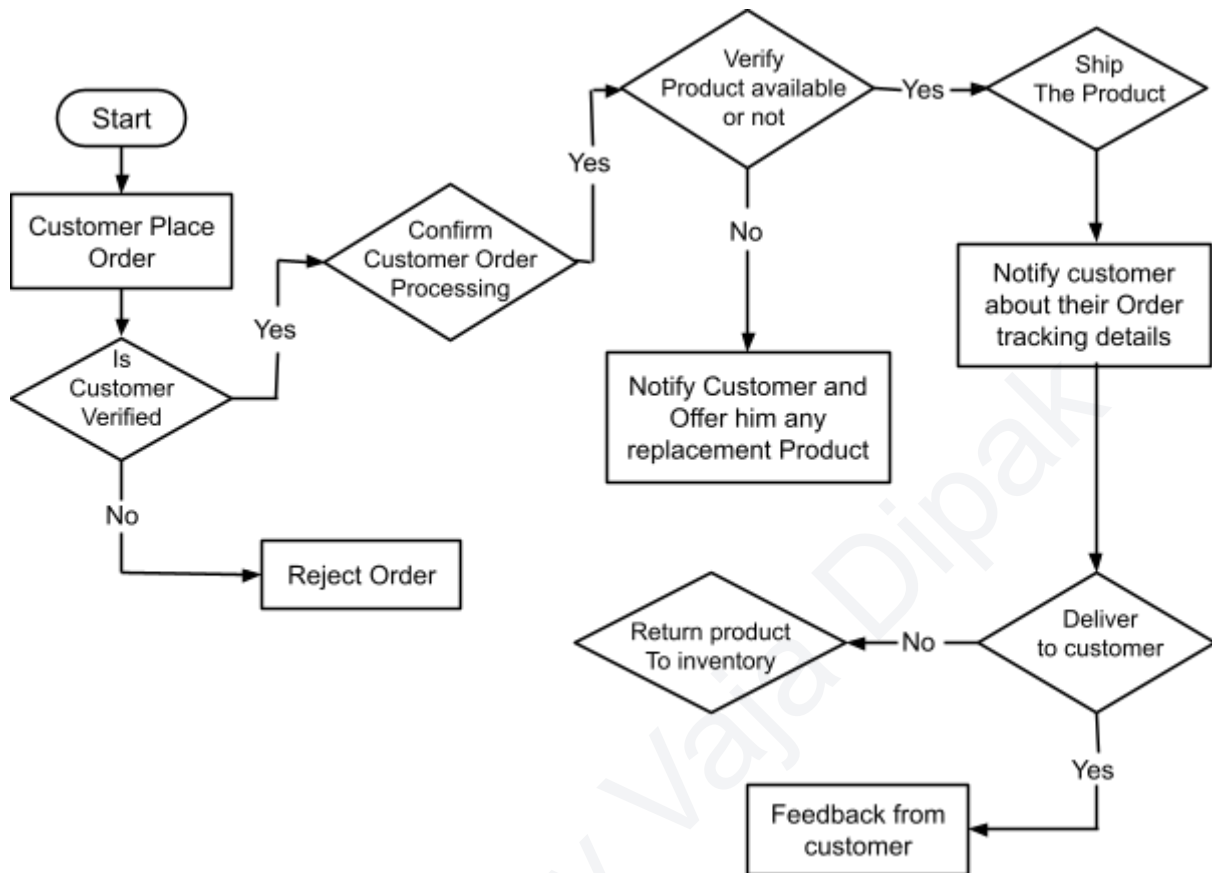
# ❖ What is Flow chart? Create a flowchart to make addition of two numbers.

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task.

➢ Flow Chart is a diagrammatic representation to get solution of any problem.
➢ Flow Charts are drawn in an earlier time to get solutions.
➢ In terms of programming language, the Flow charts are used to represent any problems graphically.
➢ In short, the logic techniques of solving problems are known as "Flow Chart".
➢ Flow Chart facilitates communication between programmers and business people.

➢ Once the flow chart is drawn, it becomes easy to write the program in High Level Language.

➢ There are several symbols, which are used to draw flowcharts:

### 1. Terminator/Oval:

This symbol is used to represent the starting and ending point of "Flow Chart". This symbol may contain the words like Start, Begin, Stop, and End Etc.
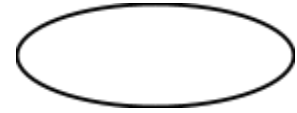
### 2. Rectangle:

This symbol is used to represent Process associated with the problem. This symbol is also used to declare the variables.

### 3. Input/Output:

This symbol is used to represent All Inputs and Outputs, which are associated with the flowcharts.

### 4. Diamond:

This symbol is used to represent the conditions associated with the flowcharts.This symbol always contains the conditions or expressions.
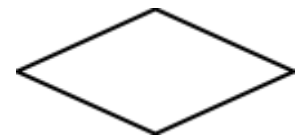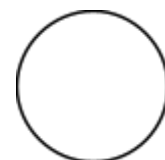
### 5. Flow Lines:

The flow lines are used to indicate the control flow of the flow chart. Generally the flow of control may be top to bottom or left to right.
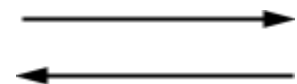
### 6. Connector:

This symbol is used to connect the multiway flow of the control, and direct the control to the particular point.
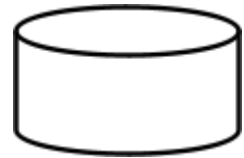
### 7. Magnetic Tape (Sequential Access Storage):

Although it looks like a 'Q', the symbol is supposed to look like a reel of tape.This symbol implies that the stored data can directly be accessed.
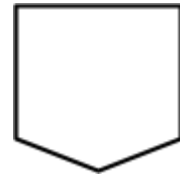
## 8. Magnetic Disk (Database)

The most universally recognizable symbol for a data storage location, this flowchart shape depicts a database.

## 9. Off-page Connector

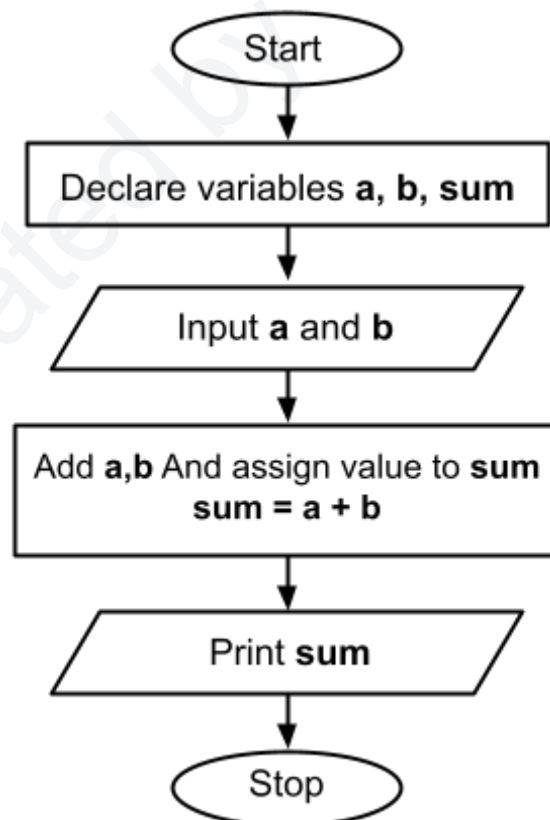It is used to signify a connection to a process held on another sheet screen.

## 10. Annotation

It is used to keep data secret from the outside world.

## 11. Display

Indicates a process step where information is displayed to a person (e.g., PC user, machine operator)



**Flow chart of Input two numbers and perform addition.**

## ❖ What is Use case Diagram? Create a use-case on Online shopping site.

A use case diagram is a way to summarise details of a system and the users within that system. It is generally shown as a graphic depiction of interactions among different elements in a system. Use case diagrams will specify the events in a system and how those events flow, however, use case diagrams do not describe how those events are implemented.

### ➜ Use case diagram uses

The reasons why an organisation would want to use case diagrams include:

- Represent the goals of systems and users.
- Specify the context a system should be viewed in.
- Specify system requirements.
- Provide a model for the flow of events when it comes to user interactions.
- Provide an outside view of a system.
- Show's external and internal influences on a system.

### ➜ Use case diagram contains four components.

1. The boundary, which defines the system of interest in relation to the world around it.

2. The actors, usually individuals involved with the system defined according to their roles.

3. The use cases, which are the specific roles played by the actors within and around the system.

4. The relationships between and among the actors and the use cases.

### ➜ Following are the purposes of a use case diagram given below:

1. It gathers the system's needs.
2. It depicts the external view of the system.
3. It recognizes the internal as well as external factors that influence the system.
4. It represents the interaction between the actors.

# ➔ Use case diagram of Online shopping