**MODULE: 1 (SDLC)**

**(Q.1)** What is software? What is software engineering?

* Software is a set of programs (sequence of instructions) that allows the users to perform a well-defined function or some specified task. Software is responsible for directing all computer-related devices and instructing them regarding what and how the task is to be performed. However, the software is made up of binary language (composed of ones and zeros), and for a programmer writing the binary code would be a slow and tedious task. Therefore, software programmers write the software program in various human-readable languages such as Java, Python, C#, etc. and later use the source code.
* Software engineering is defined as a process of analyzing user requirements and then designing, building, and testing software application which will satisfy those requirements. IEEE, in its standard 610.12-1990, defines software engineering as the application of a systematic, disciplined, which is a computable approach for the development, operation, and maintenance of software.
* Fritz Bauer defined it as ‘the establishment and used standard engineering principles. It helps you to obtain, economically, software which is reliable and works efficiently on the real machines’.
* Boehm defines software engineering, which involves, ‘the practical application of scientific knowledge to the creative design and building of computer programs. It also includes associated documentation needed for developing, operating, and maintaining them.’

**(Q.2)** Explain types of software?

There are several types of software based on their functionalities, which are as follows:-

System Software

* System software allows the user to run computer software or hardware and is responsible for managing their interaction with each other. It perpetually runs in the background to maintain the basic functionalities and the hardware of the computer. It essentially acts as a mediator between the computer and the user to facilitate operations. System software is of four types, which are as follows-
* Operating System – The operating system is a collection of software that helps execute programs and offers a computer application’s general services. There are various types of operating systems, such as iOS, macOS, Windows, Unix, [Linux](https://www.simplilearn.com/linux-programming-for-beginners-article), and Ubuntu, among others.
* Device Drivers – Device drivers handle the operations of the hardware devices connected to a computer. They act as a software interface for the hardware devices so that applications and the operating system of a computer can run the hardware functions without having the knowledge of the hardware’s exact specifications.
* Firmware – Firmware is embedded in the read-only memory of a system and is a type of permanent software that offers low-level control for certain device hardware.
* Utility – Utility software functions to configure, maintain, and offer support in the analysis and optimization of the computer.

Application Software

* [Application software](https://www.simplilearn.com/tutorials/programming-tutorial/what-is-application-software) or application programs are end-user programs that serve specific functionality to help users accomplish certain tasks. This includes graphic designing, researching online, drafting documents, watching movies or playing games, and managing finance among others. Developers keep creating software applications based on the evolving needs of users. There are various types of application software, which are as follows-
* Word Processors – Word processor software, as the name suggests, is designed for making notes, typing data, and documentation. Users can also format, store, and print their data and documents respectively using word processes.
* Database Software – Also known as [Database Management System](https://www.simplilearn.com/dbms-interview-questions-and-answers-article) (DBMS), database software allows users to perform various operations on quickly retrieved data, such as creation, management, organization, and modification. Some popular examples of DBMS are MS Access, dBase, [MySQL](https://www.simplilearn.com/tutorials/mysql-tutorial), Oracle, [Microsoft SQL Server,](https://www.simplilearn.com/what-is-microsoft-sql-server-architecture-article) and FileMaker.
* Multimedia Software – Users can perform certain actions on their computer, such as playing and recording music and video files as well as creating images with the help of multimedia software. This type of software has a massive application in the field of graphic designing where users create gifs, images, animations, and edit videos. Adobe Photoshop and Illustrator, Windows Movie Maker and Media Player, Corel Draw, and Picasa are some common types of multimedia processors.
* Web Browsers – Web Browsers are used for browsing the internet and their primary function is to allow users to fetch data across the web and also position the data. In simpler terms, you would access the internet with the help of web browsers so that you find the information that you need. Chrome, Opera, Mozilla Firefox, Microsoft Edge, Apple Safari, and UC Browser are very commonly used web browsers.
* Freeware – Freeware software is made available free of cost for a lifetime and it can be downloaded from the internet. This type of software is generally created by companies to improve their reach and gain more popularity. Some such software that is available free of cost for different purposes are Skype, Audacity, Zoom, Adobe Reader, WhatsApp, etc.
* Shareware – Shareware, much like Freeware, can be downloaded from the internet. However, the main difference between both is that, while freeware can be used for an unlimited amount of time without having to make any payment, shareware can be used on a trial basis. Adobe Acrobat and Photoshop, WinZip, and PHP Debugger are some popular types of shareware software.
* Open-Source – Open-source software is also available on the internet free of cost. However, what differentiates them from freeware is that they are available with their source code. This means users who download open-source software can make changes and transformations to it and even add features to it.

**(Q.3)** What is SDLC? Explain each phase of SDLC?

* SDLC or the Software Development Life Cycle is a process that produces software with the highest quality and lowest cost in the shortest time possible. SDLC provides a well-structured flow of phases that help an organization to quickly produce high-quality software which is well-tested and ready for production use. SDLC works by lowering the cost of software development while simultaneously improving quality and shortening production time. SDLC achieves these apparently divergent goals by following a plan that removes the typical pitfalls of software development projects. That plan starts by evaluating existing systems for deficiencies. Next, it defines the requirements of the new system. It then creates the software through the stages of analysis, planning, design, development, testing, and deployment. By anticipating costly mistakes like failing to ask the end-user or client for feedback, SLDC can eliminate redundant rework and after-the-fact fixes. It’s also important to know that there is a strong focus on the testing phase. As the SDLC is a repetitive methodology, you have to ensure code quality at every cycle. Many organizations tend to spend few efforts on testing while a stronger focus on testing can save them a lot of rework, time, and money. Be smart and write the right types of tests.
* Next, let’s explore the different stages of the Software Development Life Cycle.

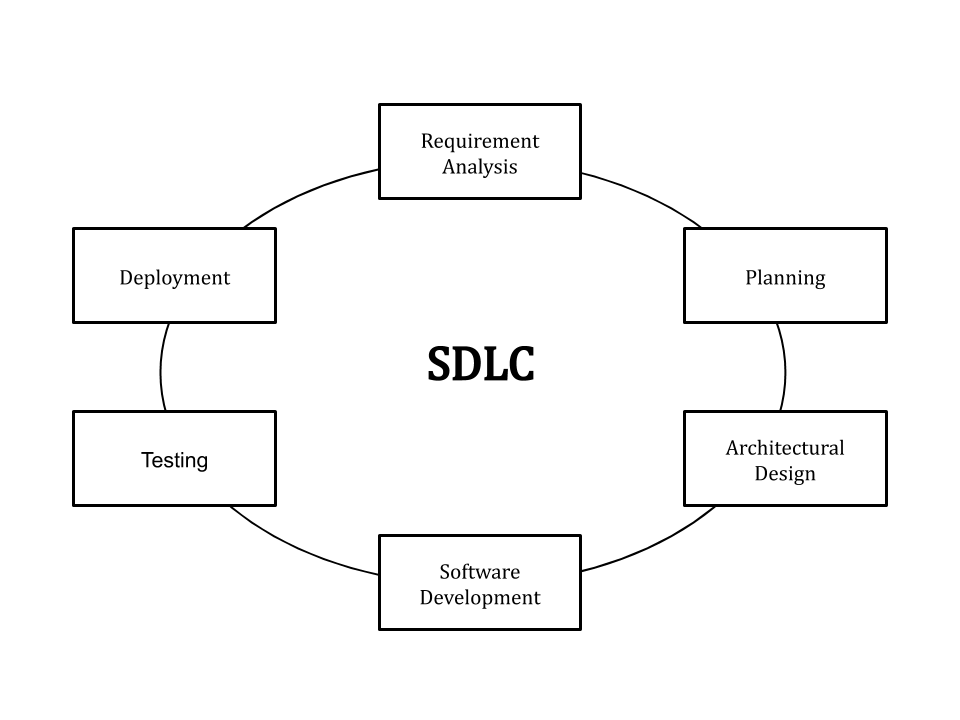


Fig.1 Stages of SDLC

1. Identify the Current Problems :-

This stage of the SDLC means getting input from all stakeholders, including customers, salespeople, industry experts, and programmers. Learn the strengths and weaknesses of the current system with improvement as the goal.

1. Plan :-

In this stage of the SDLC, the team determines the cost and resources required for implementing the analyzed requirements. It also details the risks involved and provides sub-plans for softening those risks. In other words, the team should determine the feasibility of the project and how they can implement the project successfully with the lowest risk in mind.

1. Design :-

This phase of the SDLC starts by turning the software specifications into a design plan called the Design Specification. All stakeholders then review this plan and offer feedback and suggestions. It’s crucial to have a plan for collecting and incorporating stakeholder input into this document. Failure at this stage will almost certainly result in cost overruns at best and the total collapse of the project at worst.

1. Build :-

At this stage, the actual development starts. It’s important that every developer sticks to the agreed blueprint. Also, make sure you have proper guidelines in place about the code style and practices. For example, define a nomenclature for files or define a variable naming style such as camelCase. This will help your team to produce organized and consistent code that is easier to understand but also to test during the next phase.

1. Code Test :-

In this stage, we test for defects and deficiencies. We fix those issues until the product meets the original specifications. In short, we want to verify if the code meets the defined requirements. Try Stackify’s free code profiler, Prefix, to write better code on your workstation. Prefix works with .NET, Java, PHP, Node.js, Ruby, and Python.

1. Software Deployment

At this stage, the goal is to deploy the software to the production environment so users can start using the product. However, many organizations choose to move the product through different deployment environments such as a testing or staging environment. This allows any stakeholders to safely play with the product before releasing it to the market. Besides, this allows any final mistakes to be caught before releasing the product.

**(Q.4)** What is DFD? Create a DFD diagram on Flipkart?

* A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually “say” things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That’s why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.

**DFD Level 0**

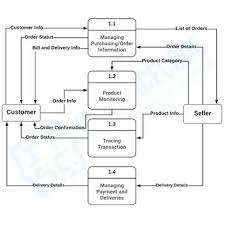
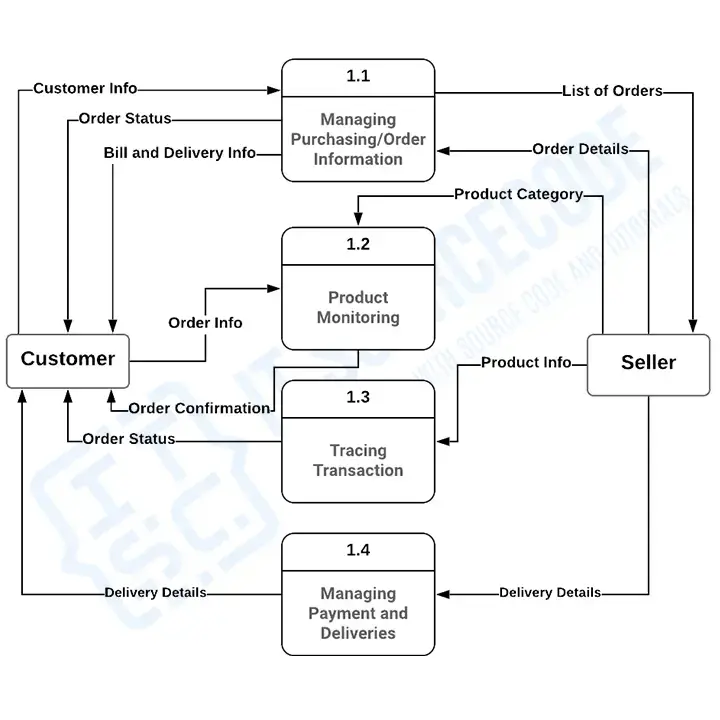


Fig.2 DFD level 0

**DFD Level 1**



**Fig.3 DFD level 3**

**DFD Level 2**

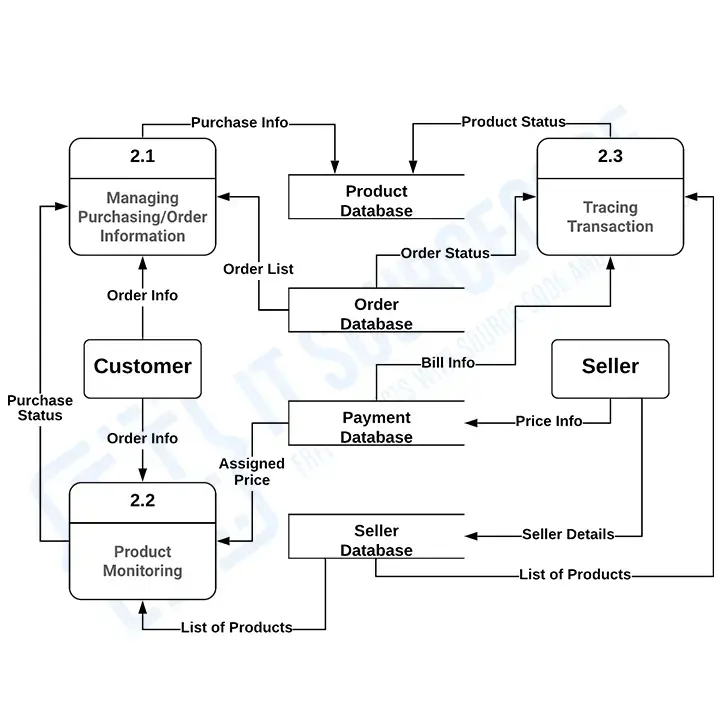


Fig.4 DFD level 2

**(Q.4)** What is Flow chart? Create a flowchart to make addition of two number?

* A flowchart is a diagram that depicts a process, system or computer algorithm. They are widely used in multiple fields to document, study, plan, improve and communicate often complex processes in clear, easy-to-understand diagrams. Flowcharts, sometimes spelled as flow charts, use rectangles, ovals, diamonds and potentially numerous other shapes to define the type of step, along with connecting arrows to define flow and sequence. They can range from simple, hand-drawn charts to comprehensive computer-drawn diagrams depicting multiple steps and routes. If we consider all the various forms of flowcharts, they are one of the most common diagrams on the planet, used by both technical and non-technical people in numerous fields. Flowcharts are sometimes called by more specialized names such as Process Flowchart, Process Map, Functional Flowchart, Business Process Mapping, Business Process Modeling and Notation (BPMN), or Process Flow Diagram (PFD). They are related to other popular diagrams, such as Data Flow Diagrams (DFDs) and Unified Modeling Language (UML) Activity Diagrams.
* Flow chart to make addition of two numbers :-

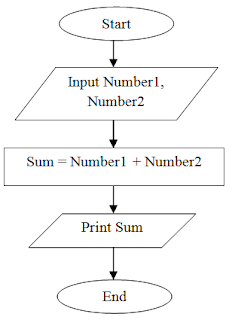


Fig.5 flow chart to add two numbers

**(Q.5)** What is Use case Diagram? Create a use-case on bill payment on paytm?

* A use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behaviour (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behaviour in the user's terms by specifying all externally visible system behaviour.
* A use case diagram is usually simple. It does not show the detail of the use cases: It only summarizes some of the relationships between use cases, actors, and systems. It does not show the order in which steps are performed to achieve the goals of each use case. As said, a use case diagram should be simple and contains only a few shapes. If yours contain more than 20 use cases, you are probably misusing use case diagram.
* Use-case diagram on bill payment on paytm :-

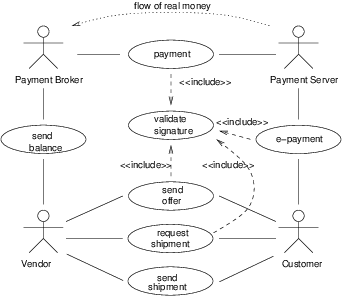


Fig. 6 use-case diagram on bill payment