Software Engineering Assignment

MODULE: 1 (SDLC)

Q:-1.What is software? What is software engineering?

ANS:- **Software**is more than just a program code. A program is an executable code, which serves some computational purpose. Software is considered to be collection of executable programming code, associated libraries and documentations. Software, when made for a specific requirement is called **software product.**

**Engineering** on the other hand, is all about developing products, using well-defined, scientific principles and methods.

**Software engineering** is an engineering branch associated with development of software product using well-defined scientific principles, methods and procedures. The outcome of software engineering is an efficient and reliable software product.

Q:2 Explain types of software.

ANS:- **Increasing complexity -**As an E-type software system evolves, its complexity tends to increase unless work is done to maintain or reduce it.

* **Conservation of familiarity -**The familiarity with the software or the knowledge about how it was developed, why was it developed in that particular manner etc. must be retained at any cost, to implement the changes in the system.
* **Continuing growth-**In order for an E-type system intended to resolve some business problem, its size of implementing the changes grows according to the lifestyle changes of the business.
* **Reducing quality -**An E-type software system declines in quality unless rigorously maintained and adapted to a changing operational environment.
* **Feedback systems-**The E-type software systems constitute multi-loop, multi-level feedback systems and must be treated as such to be successfully modified or improved.
* **Self-regulation -**E-type system evolution processes are self-regulating with the distribution of product and process measures close to normal.
* **Organizational stability -**The average effective global activity rate in an evolving E-type system is invariant over the lifetime of the product.

Q:-3.What is SDLC? Explain each phase of SDLC.

## ANS:- Why to Learn 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 a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

SDLC is a process followed for a software project, within a software organization. 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.

* SDLC is the acronym of Software Development Life Cycle.
* It is also called as Software Development Process.
* SDLC is a framework defining tasks performed at each step in the software development process.
* ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

SDLC Models

There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models. Each process model follows a Series of steps unique to its type to ensure success in the process of software development.

Following are the most important and popular SDLC models followed in the industry −

* Waterfall Model
* Iterative Model
* Spiral Model
* V-Model
* Big Bang Model

Q:4. What is DFD? Create a DFD diagram on Flipkart.

ANS:- Data flow diagrams (DFDs) or data flow charts show the flow of information throughout a system

process.

They use defined symbols, text labels, and varying levels of detail to display information. DFDs help non-technical audiences understand how data flows throughout a software system.

Gliffy is easy-to-use data flow diagram software that allows you to drag and drop shapes to make a DFD directly alongside your technical documentation in Confluence.

* [Data Flow Mapping with DFDs](https://www.gliffy.com/blog/what-is-a-data-flow-diagram#data-flow)
* [Example of a Data Flow Diagram](https://www.gliffy.com/blog/what-is-a-data-flow-diagram#example-of)
* [What's the Difference Between Data Flow Diagrams and UML?](https://www.gliffy.com/blog/what-is-a-data-flow-diagram#what-the)
* [Types of Data Flow Diagrams: Logical vs. Physical Data Models](https://www.gliffy.com/blog/what-is-a-data-flow-diagram#types-of)
* [Data Flow Diagram Levels](https://www.gliffy.com/blog/what-is-a-data-flow-diagram#diagram-levels)
* [Make a Data Flow Diagram Online](https://www.gliffy.com/blog/what-is-a-data-flow-diagram#make-a)

Q:5.What is Flow chart? Create a flowchart to make addition of two numbers.

ANS:-

A flowchart is a diagram that shows an overview of a program . Flowcharts normally use standard symbols to represent the different types of instructions . These symbols are used to construct the flowchart and show the step-by-step solution to the problem. Flowcharts are sometimes known as flow diagrams.

Q:6. What is Use case Diagram? Create a use-case on bill payment on paytm.

ANS:- So what is a use case diagram? A [**UML**](https://en.wikipedia.org/wiki/Unified_Modeling_Language) use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior (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 behavior in the user's terms by specifying all externally visible system behavior.

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.

The figure below shows the UML diagram hierarchy and the positioning of the UML Use Case Diagram. As you can see, use case diagrams belong to the family of behavioral diagrams.