SOFTWARE ENGINEERING

- The term software engineering is composed 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 software product.
- Engineering on the other hand, is all about developing products, using well-defined, scientific principles and methods.

Generic software development:

- Generic software development is being undertaken by the company that owns the resulting product. This product has a reliable need in the market over a period of time and fit for most of the business needs.
- Software development companies develop generic software at own cost and sell it as a product for customers. Then generic software is ready, set to use and available in the market.

Custom software development :

Custom software development is being done as a work-for-hire project for a particular client (the group of people or another company). The list of requirements, product's idea, need and money on development come from the client/customer. Usually, it takes a lot of time to develop a custom product.

Types of softwares:

System Software:

- System software is necessary to manage the compiler resources and support the exectution of application programs.
- Ex:Compilers

Application software:

- It is designed to solve user problems as per the user's requirements. Application software can be generic or customized. Ex: MS Office

Scientific software:

- It deals with processing requirements in a specific application.
- These software are used in the field of mechanical, electrical, drafting, engineering and Analysis. They run on mainframes, general purpose workstation, and PCs (Personal computers).

Embedded software:

- This software is embedded into hardware as a part of larger systems to control its various functions.
- Ex:Keypad control software embedded in a microwave oven or washing machine

Product_line software:

- A software product line (SPL) is a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.

Web apps Software:

- It is an application that is accessed via web browsers over a network such as an internet or an intranet.
- Ex: Facebook

AI software:

- This software makes the use of non-numerical algorithms that use the data generated in the system to solve complex problems
- Ex: Robotics

• Software engineering principles use two important techniques to reduce problem

complexity:

- abstraction

- decomposition.

ABSTRACTION:the main purpose of abstraction is to consider only those aspects of the problem that are relevant for certain purpose and suppress other aspects that are not relevant for the given purpose.

Abstraction is a powerful way of reducing the complexity of the problem.

DECOMPOSITION: In decomposition, a complex problem is divided into several smaller problems and then the smaller problems are solved one by one.The problem has to be decomposed such that each component of thedecomposed problem can be solved independently and then the solution of the different components can be combined to get the full solution.

APPLICATIONS AND NEED OF SOFTWARE ENGINEERING

Software Development for various domains

• To perform various operations on the software like testing

• Maintenance of various software products

• To apply the knowledge, practices, and technologies to build high-quality

• software products that enhance productivity in every industry

CHARACTERISTICS OF GOOD SOFTWARE

Every software must satisfy the following attributes:

• Operational

• Transitional

• Maintenance

Operational

This tells us how well software works in operations. It can be measured on:

• Budget

• Usability

• Efficiency

• Correctness

• Functionality

• Dependability

• Security

• Safety

Transitional

This aspect is important when the software is moved from one platform to another:

• Portability

• Interoperability

• Reusability

• Adaptability

Maintenance

This aspect briefs about how well a software has the capabilities to maintain itself in the ever-changing environment:

• Modularity

• Maintainability

• Flexibility

• Scalability

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 PHASES

• 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:

1)Requirement collection and analysis:

- It is conducted by the senior team members with inputs from all the sales department and domain experts in the industry.
- Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage.
- This stage gives a clearer picture of the scope of the entire project.This helps companies to finalize the necessary timeline to finish the work of that system.

Feasibilty study

• Economic: Can we complete the project within the budget or not?

• Legal: Can we handle this project as cyber law and other regulatory compliances.

• Operation feasibility: Can we create operations which is expected by the client?

• Technical: Need to check whether the current computer system can support the software

• Schedule: Decide that the project can be completed within the given schedule or not.

Design

- the system and software design documents are prepared as per the requiremens specification document.
- **High-Level Design (HLD)**

  • Brief description and name of each module

  • An outline about the functionality of every module

  • Interface relationship and dependencies between modules

  • Database tables identified along with their key elements

  • Complete architecture diagrams along with technology details
- **Low-Level Design(LLD)**

  • Functional logic of the modules

  • Database tables, which include type and size

  • Complete detail of the interface

  • Addresses all types of dependency issues

  • Listing of error messages

  • Complete input and outputs for every module.

Coding

- In this phase, developers start
- build the entire system by writing code using the chosen programming language.In the coding phase, tasks are divided into units or modules and assigned to the various developers.
- It is the longest phase of the Software Development Life Cycle process.In this phase, Developer needs to follow certain predefined coding guidelins.

Testing

- Once the software is complete, and it is deployed in the testing environment. The testing team starts testing the functionality of the entire system. This is done to verify that the entire application works according to the customer requirement.
- During this phase, QA and testing team may find some bugs/defects which they communicate to developers.
- The development team fixes the bug and send back to QA for a re-test. This process continues until the software is bug-free, stable, and working according to the business needs of that system..

INSTALLATION/DEPLOYMENT

• Once the software testing phase is over and no bugs or errors left in the system then the

final deployment process starts.

• Based on the feedback given by the project manager, the final software is released and

checked for deployment issues if any.

MAINTENANCE

Once the system is deployed, and customers start using the developed system, following 3 activities

occur

• Bug fixing - bugs are reported because of some scenarios which are not tested at all

• Upgrade - Upgrading the application to the newer versions of the Software

• Enhancement - Adding some new features into the existing software

The main focus of this SDLC phase is to ensure that needs continue to be met and that the system

continues to perform as per the specification mentioned in the first phase.

Life cycle models

1)WATERFALL MODEL

i] Requirement analysis and specification phase :-

- This phase aims to understand and properly document the exact requirements of the customer. This task is usually carried out in collaboration with the customer, as the aim is to record all specifications for the application.

- The SRS document may act as a contract between the developer and the customer. If the developer fails to implement a full set of requirements, it may fail to implement the contracted set of requirements.

ii]Design phase:-

• The purpose of this step is to convert the requirement specification into a structure that is suitable for implementation in some programming languages.overall software architecture is defined, and the high level and detailed design work are performed.

• This work is documented and known as a software design description (SDD) document. The information included.

Implementation and unit testing phase:-

• During this phase, the design is implemented. If the SDD is complete, the

implementation or coding phase proceeds smoothly, because of all the information

needed by the software developers contained in the SDD.

• During testing, the primary activities are centered around the examination and

modification of the code. Small modules were initially tested in isolation from the rest

of the software product.

• There are problems associated with the evaluation of unit tests in isolation. How do

we run a module without anything to label it, to call it, or, possibly, to intermediate

output values obtained during execution?

• These problems are resolved at this point, and the modules are checked after writing

some overhead code.Integration and system testing:-

iii]Integration and system testing:-

• This is a very important phase. Good testing can help to deliver higher quality software products, more happy customers, lower maintenance costs, and more accurate and reliable performance.

• It is a very expensive activity and consumes one-third to at least one half the value of a typical development project.

• As we know, the purpose of unit testing is to determine whether each independent module is implemented correctly.

• It gives very little chance to determine if the interface between modules is also correct and, therefore, an integration testing is performed. System testing involves the testing of the entire system, whereas software is a part of the system.

iv]Operation and maintenance phase:-

• Software maintenance is a challenge that every team has to face when the software is deployed and installed to the customer’s site.

• Software maintenance is a very broad activity that includes error correction, enhancement of

capabilities, and optimizations.

• The purpose of this phase is to maintain the value of the software over time. This stage can last from 5 to 50 years, while development can last from 1 to 3 years.