SDLC - Quick Guide

Previous Page Next Page

SDLC Overview

SDLC, Software Development Life Cycle is a process used by 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 the acronym of Software Development Life Cycle.

It is also called as Software development process.

The software development life cycle (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.

A typical Software Development life cycle consists of the following stages:

Stage 1: Planning and Requirement Analysis

Stage 2: Defining Requirements

Stage 3: Designing the product architecture

Stage 4: Building or Developing the Product

Stage 5: Testing the Product

Stage 6: Deployment in the Market and Maintenance

SDLC Models

There are various software development life cycle models defined and designed which are followed during 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, in order to ensure success in process of software development.

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

Waterfall Model

Iterative Model

Spiral Model

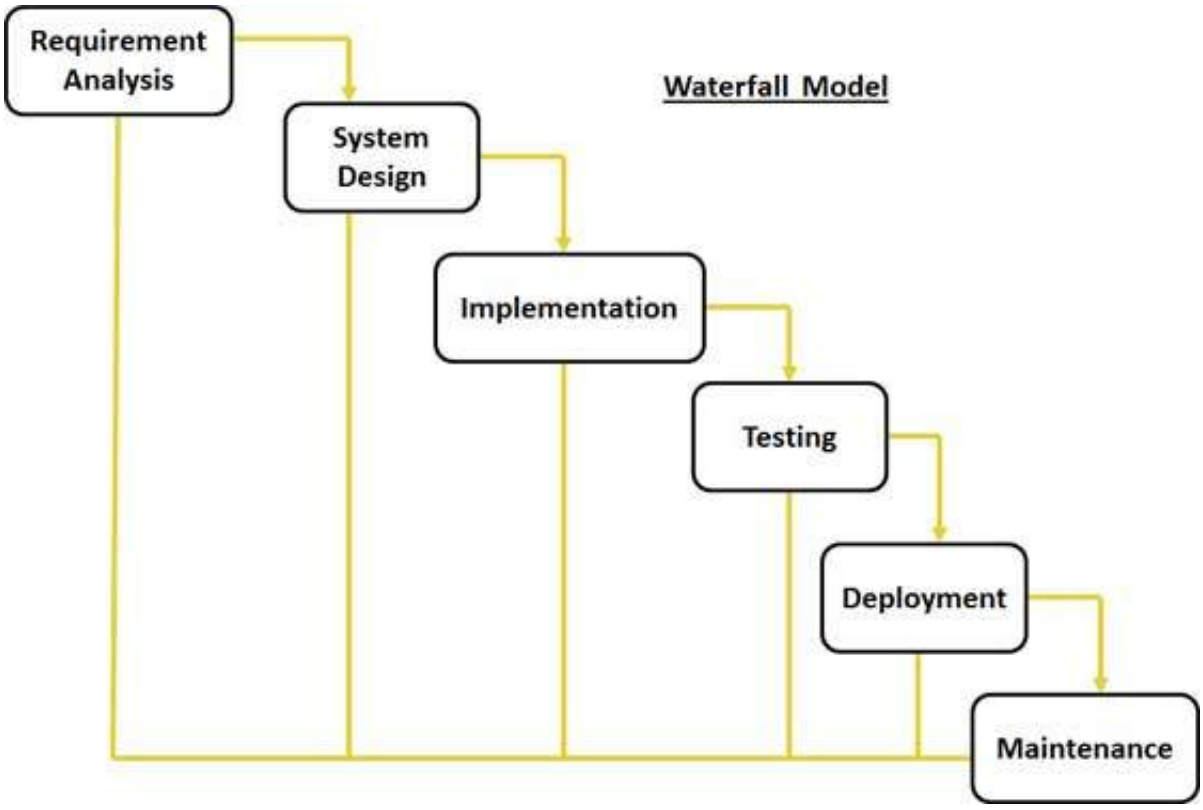
V-Model

Big Bang Model

The other related methodologies are Agile Model, RAD Model, Rapid Application Development and Prototyping Models.

SDLC Waterfall Model

Following is a diagrammatic representation of different phases of waterfall model.



Waterfall Model

The sequential phases in Waterfall model are:

Requirement Gathering and analysis: All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

System Design: The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

Implementation: With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

Integration and Testing: All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

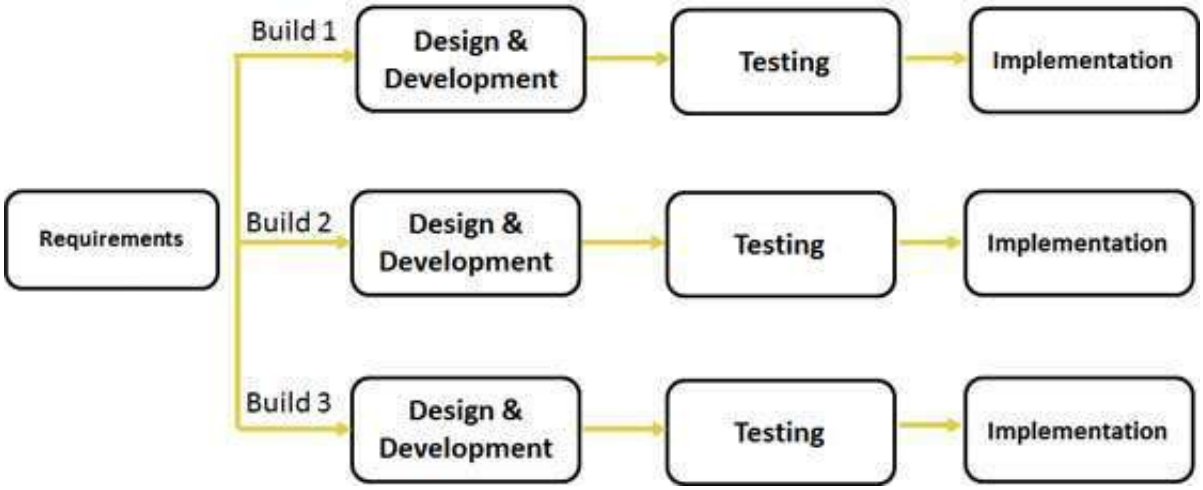
Deployment of system: Once the functional and non functional testing is done, the product is deployed in the customer environment or released into the market.

Maintenance: There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model phases do not overlap.

SDLC Iterative Model

Following is the pictorial representation of Iterative and Incremental model:



This model is most often used in the following scenarios:

Requirements of the complete system are clearly defined and understood.

Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.

There is a time to the market constraint.

A new technology is being used and is being learnt by the development team while working on the project.

Resources with needed skill set are not available and are planned to be used on contract basis for specific iterations.

There are some high risk features and goals which may change in the future.

SDLC Spiral Model

The spiral model has four phases. A software project repeatedly passes through these phases in iterations called Spirals.

Identification: This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.

This also includes understanding the system requirements by continuous communication between the customer and the system analyst. At the end of the spiral the product is deployed in the identified market.

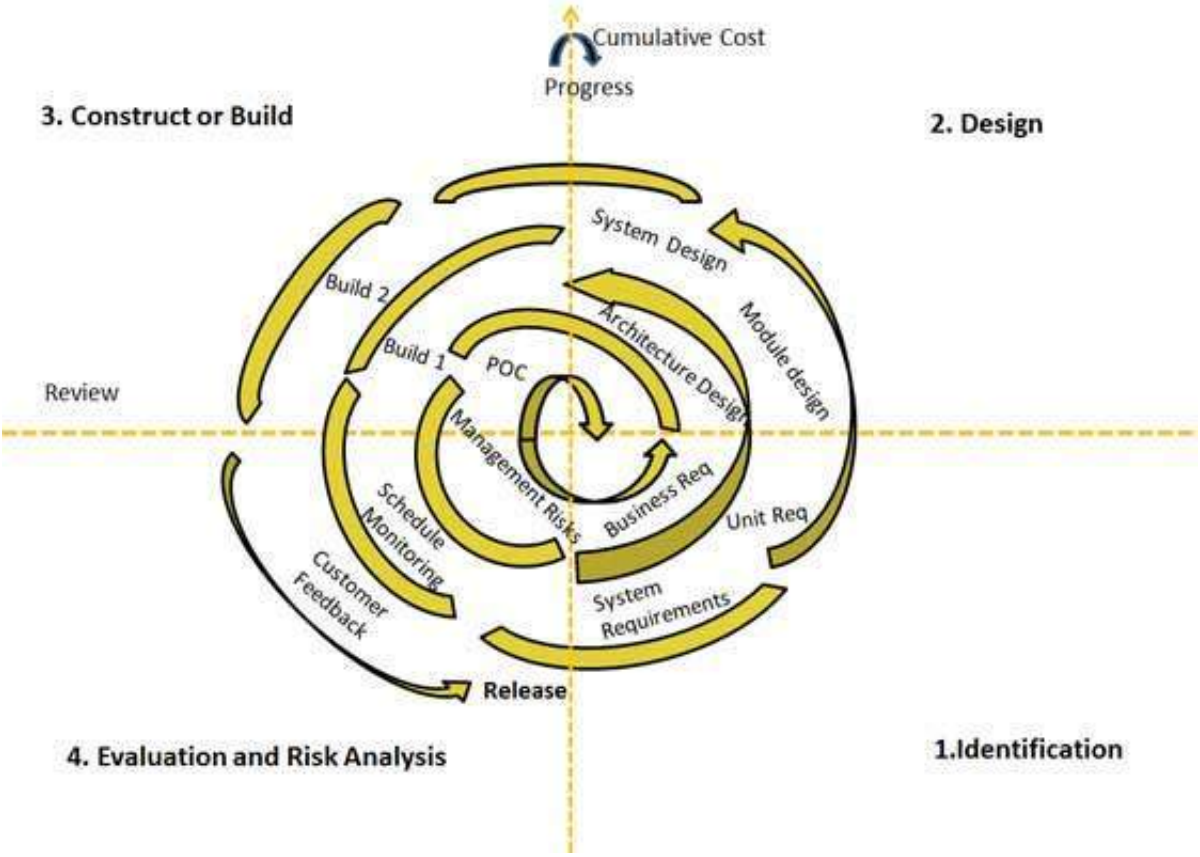
Design: Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and final design in the subsequent spirals.

Construct or Build: Construct phase refers to production of the actual software product at every spiral. In the baseline spiral when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.

Then in the subsequent spirals with higher clarity on requirements and design details a working model of the software called build is produced with a version number. These builds are sent to customer for feedback.

Evaluation and Risk Analysis: Risk Analysis includes identifying, estimating, and monitoring technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

Following is a diagrammatic representation of spiral model listing the activities in each phase:

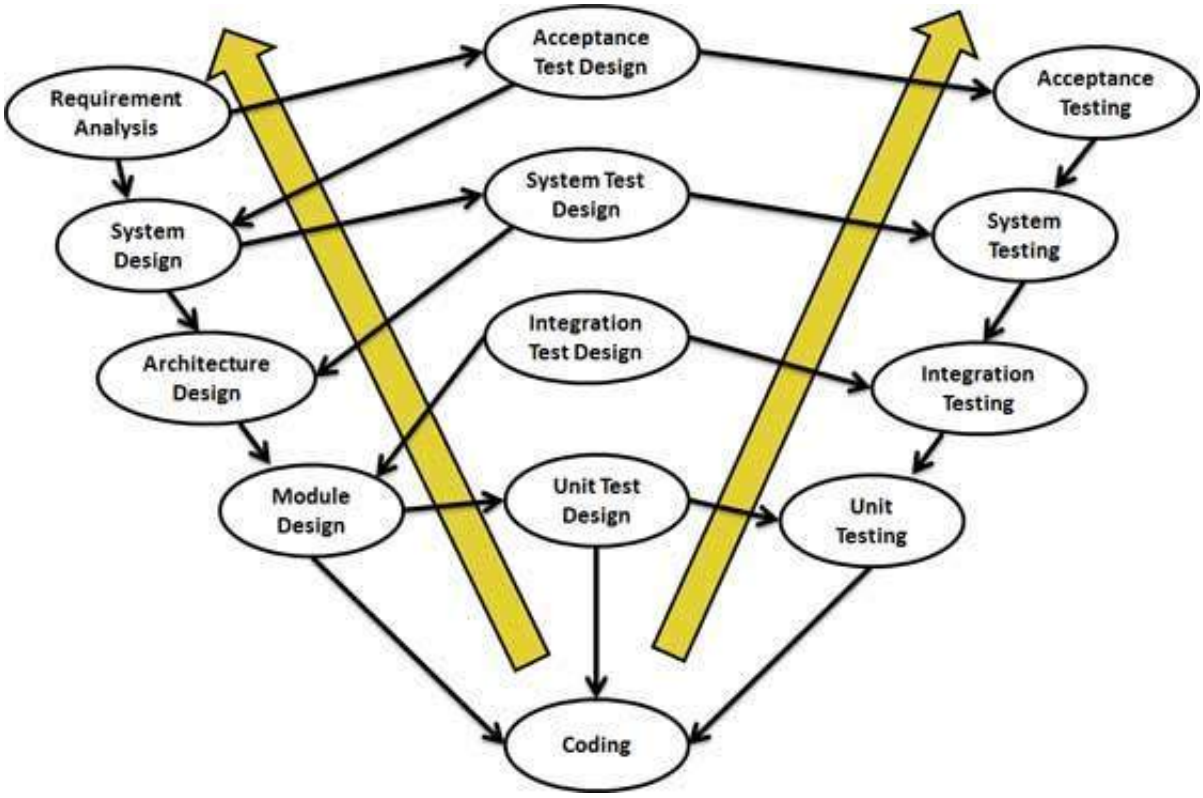


V Model

The V - model is SDLC model where execution of processes happens in a sequential manner in V-shape. It is also known as Verification and Validation model.

V - Model is an extension of the waterfall model and is based on association of a testing phase for each corresponding development stage. This means that for every single phase in the development cycle there is a directly associated testing phase. This is a highly disciplined model and next phase starts only after completion of the previous phase.

The below figure illustrates the different phases in V-Model of SDLC.



SDLC Big Bang Model

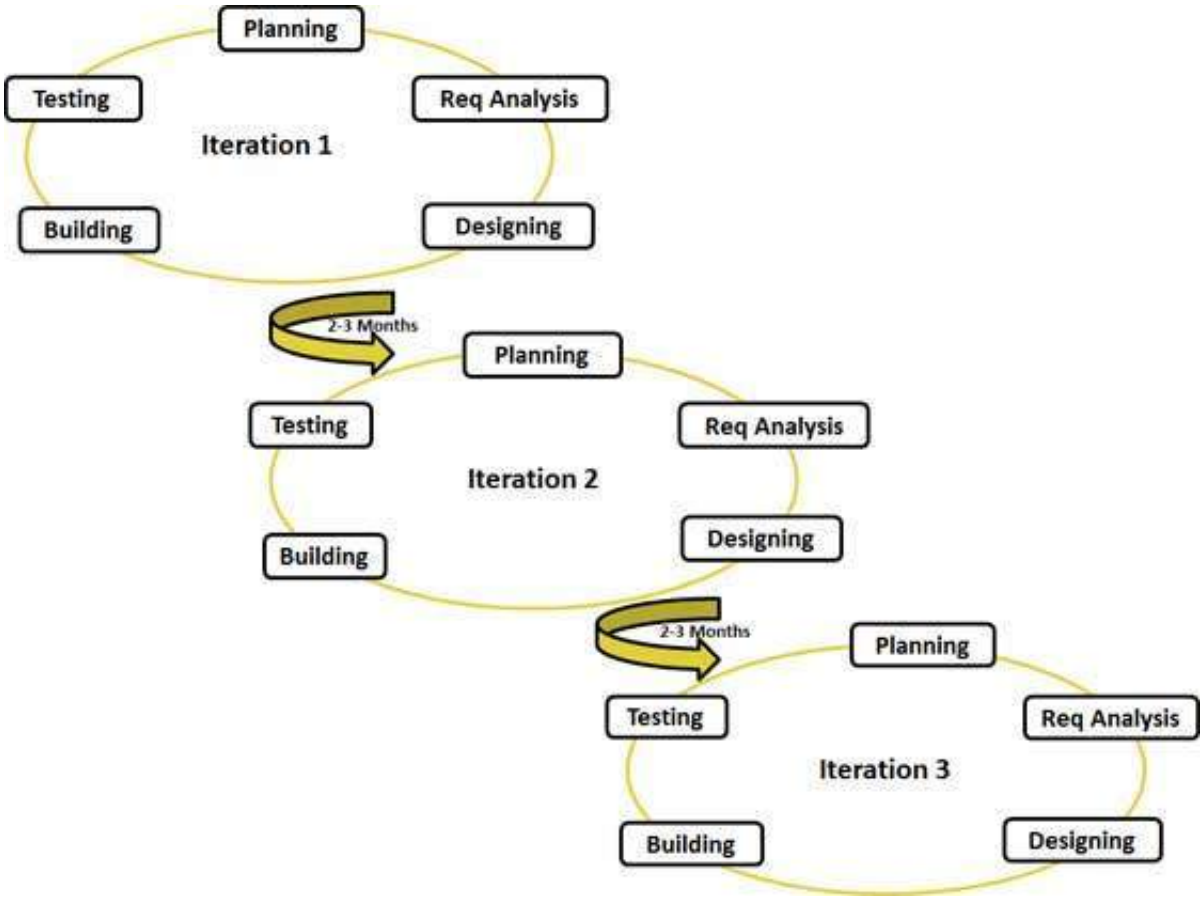
The Big Bang model is SDLC model where there is no specific process followed. The development just starts with the required money and efforts as the input, and the output is the software developed which may or may not be as per customer requirement.

B ig Bang Model is SDLC model where there is no formal development followed and very little planning is required. Even the customer is not sure about what exactly he wants and the requirements are implemented on the fly without much analysis.

Usually this model is followed for small projects where the development teams are very small.

Agile Model

Here is a graphical illustration of the Agile Model:



Agile thought process had started early in the software development and started becoming popular with time due to its flexibility and adaptability.

The most popular agile methods include Rational Unified Process (1994), Scrum (1995), Crystal Clear, Extreme Programming (1996), Adaptive Software Development, Feature Driven Development, and Dynamic Systems Development Method (DSDM) (1995). These are now collectively referred to as agile methodologies, after the Agile Manifesto was published in 2001.

Following are the Agile Manifesto principles

Individuals and interactions . in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.

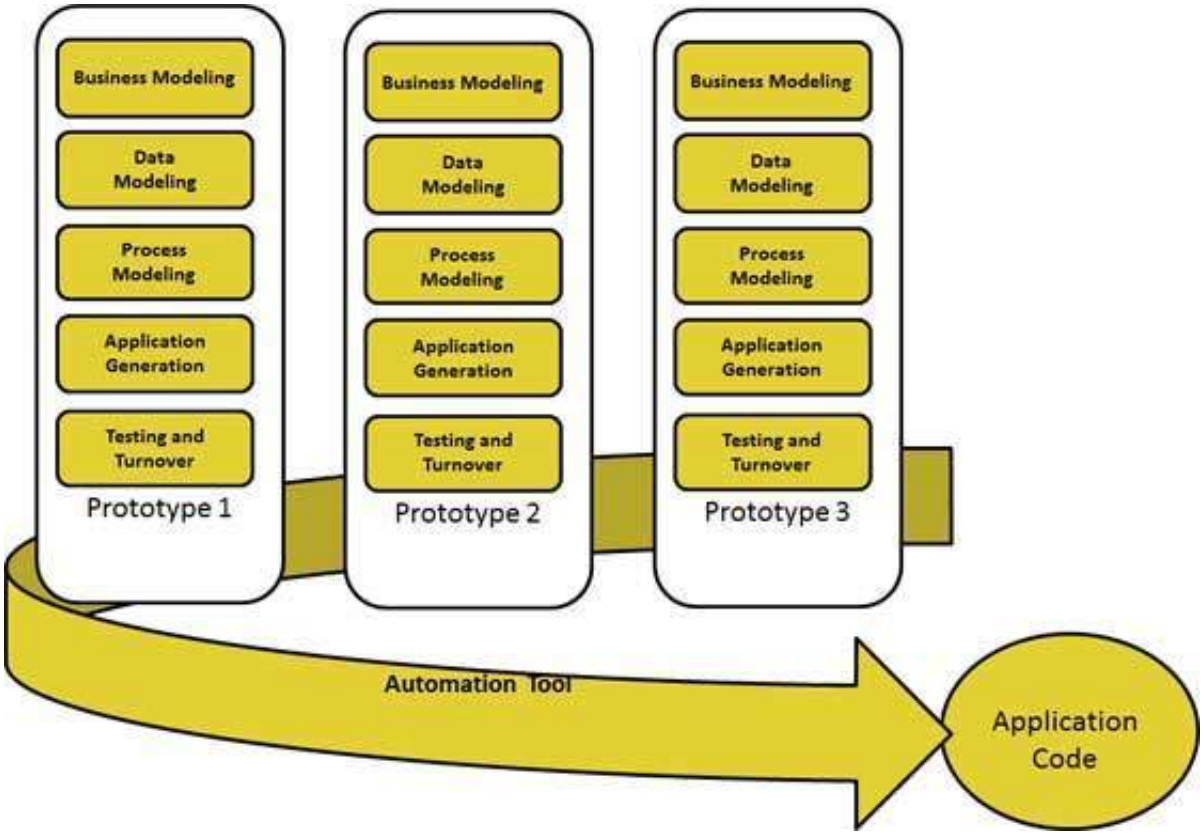
Working software . Demo working software is considered the best means of communication with the customer to understand their requirement, instead of just depending on documentation.

Customer collaboration . As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.

Responding to change . agile development is focused on quick responses to change and continuous development.

RAD Model

Following image illustrates the RAD Model:



Following are the typical scenarios where RAD can be used:

RAD should be used only when a system can be modularized to be delivered in incremental manner.

It should be used if there.s high availability of designers for modeling.

It should be used only if the budget permits use of automated code generating tools.

RAD SDLC model should be chosen only if domain experts are available with relevant business knowledge.

Should be used where the requirements change during the course of the project and working prototypes are to be presented to customer in small iterations of 2-3 months.

Software Prototyping

The Software Prototyping refers to building software application prototypes which display the functionality of the product under development but may not actually hold the exact logic of the original software.

Software prototyping is becoming very popular as a software development model, as it enables to understand customer requirements at an early stage of development. It helps get valuable feedback from the customer and helps software designers and developers understand about what exactly is expected from the product under development.

Following is the stepwise approach to design a software prototype:

Basic Requirement Identification: This step involves understanding the very basics product requirements especially in terms of user interface. The more intricate details of the internal design and external aspects like performance and security can be ignored at this stage.

Developing the initial Prototype: The initial Prototype is developed in this stage, where the very basic requirements are showcased and user interfaces are provided. These features may not exactly work in the same manner internally in the actual software developed and the workarounds are used to give the same look and feel to the customer in the prototype developed.

Review of the Prototype: The prototype developed is then presented to the customer and the other important stakeholders in the project. The feedback is collected in an organized manner and used for further enhancements in the product under development.

Revise and enhance the Prototype: The feedback and the review comments are discussed during this stage and some negotiations happen with the customer based on factors like , time and budget constraints and technical feasibility of actual implementation. The changes accepted are again incorporated in the new Prototype developed and the cycle repeats until customer expectations are met.

Summary

This was about the various SDLC models available and the scenarios in which these

SDLC models are used. The information in this tutorial will help the project managers decide what SDLC model would be suitable for their project and it would also help the developers and testers understand basics of the development model being used for their project.

We have discussed all the popular SDLC models in the industry, both traditional and Modern. This tutorial also gives you an insight into the pros and cons and the practical applications of the SDLC models discussed.

Waterfall and V model are traditional SDLC models and are of sequential type. Sequential means that the next phase can start only after the completion of first phase. Such models are suitable for projects with very clear product requirements and where the requirements will not change dynamically during the course of project completion.

Iterative and Spiral models are more accommodative in terms of change and are suitable for projects where the requirements are not so well defined, or the market requirements change quite frequently.

Big Bang model is a random approach to Software development and is suitable for small or academic projects.

Agile is the most popular model used in the industry. Agile introduces the concept of fast delivery to customers using prototype approach. Agile divides the project into small iterations with specific deliverable features. Customer interaction is the backbone of Agile methodology, and open communication with minimum documentation are the typical features of Agile development environment.

RAD (Rapid Application Development) and Software Prototype are modern techniques to understand the requirements in a better way early in the project cycle. These techniques work on the concept of providing a working model to the customer and stockholders to give the look and feel and collect the feedback. This feedback is used in an organized manner to improve the product.

The Useful Resources section lists some suggested books and online resources to gain further understanding of the SDLC concepts.

Keep visiting to us, Happy Learning!

Previous Page Next Page

Enter email for newsletter go