Software Development Life Cycle – SDLC

It is a step-by-step process to develop a new software.

Stages in SDLC

1) Requirement Collection

2) Feasibility Study

3) Design

4) Coding

5) Testing

6) Installation

7) Maintenance

1) Requirement **collection: B**usiness analyst from the company collects the requirement in business Language and converts that language into software language and explains it to the entire team (testing engineer, architect, developers, PM etc.…). this process is called Requirement collection.

Requirement collection is done by BA (business analyst) or PA (product analyst). BA acts as a bridge Between company and customer.

Who can become BA

a) Domain expert

b) Senior developer / senior test engineer.

2) Feasibility Study stage:

It is done by a team which consists of PM, architect, BA, finance team and HR team. Project manager will interact or communicate with BA, HR team, or architect and gather all The Info and project manager will decide whether to take up the project or not.

3) Design: Once feasibility study is completed, we go to the design stage.Design is done by an architect / technical architect /senior architect.

There are two types of design:

1) High level design is also called external design.

2) Low level design is also called internal design.

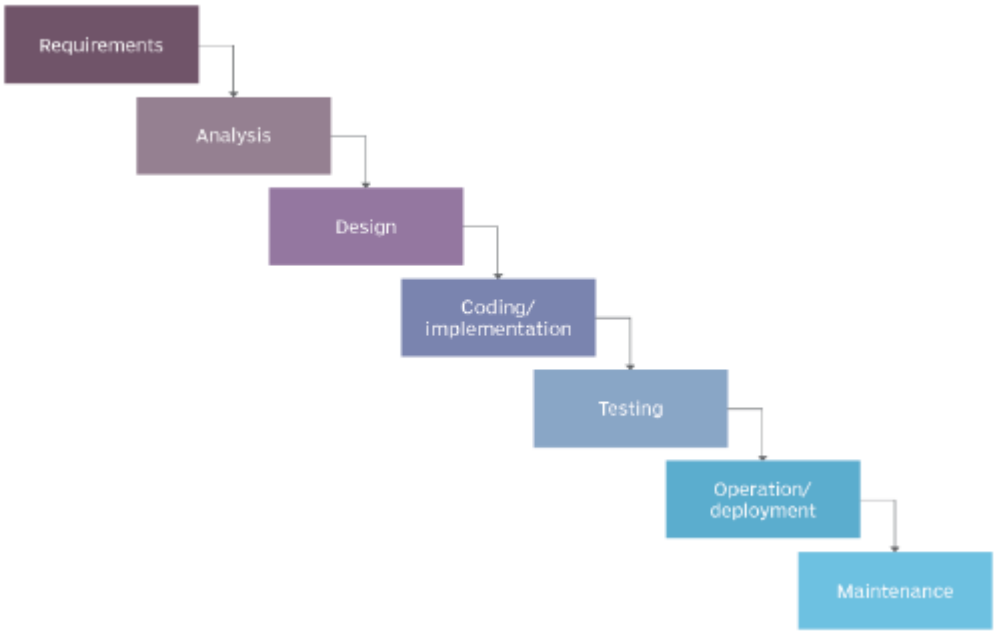
4) **Coding:** Once design is completed, then we go for the coding phase or stage.It is done by looking into customer requirements and LLD. It is done by Senior developer, Junior developer and freshers by looking into the LLD and the requirements.

5) **Testing**: Testers test the software which is given by developers and test software to find bugs according to customer requirements.

6) **Installation**: Once Testing is done, installation will be done by IT Engineers. IT Engineers from the company go to the customer's place, install the software and provide a user guide to customers.

7) **Maintenance period**: This is nothing but an agreement period between company and the customer where they sign in for a particular period of time, in between this time, if there is any issue, we are responsible to fix the issues.

WATERFALL MODEL



It is also called a Traditional model or sequential model.

It follows the same stages of SDLC, where it starts with Requirement collection done by BA. BA collects complete requirements in Business language and explains to the team in software language. Later on we go with a feasibility Study where the Project Manager takes opinions from HR, Finance team, Architect, BA and other team members and decides whether we can take up this project or not.

Once Project is started, initially Architects need to design software in terms of HLD(High level design) and LLD(Low level design).

Developers by looking into LLD and requirements starts writing the code. Once done it will be moved to the Testing team, where Testers will find bugs according to customer requirements.

It is important to note that Requirements and Design is not tested in this model. Because of which if there is any defect in the requirements such as missing requirement, wrong requirement, and conflict requirements, it will flow to the next stages.

In Waterfall model, once Requirement collection and Feasibility study, we cannot go back and make any changes i,e,. backtracking is not possible because of which any defects found will flow till testing phase. To fix this issue, it consumes a lot of time and reworking. Hence this model is not preferred.

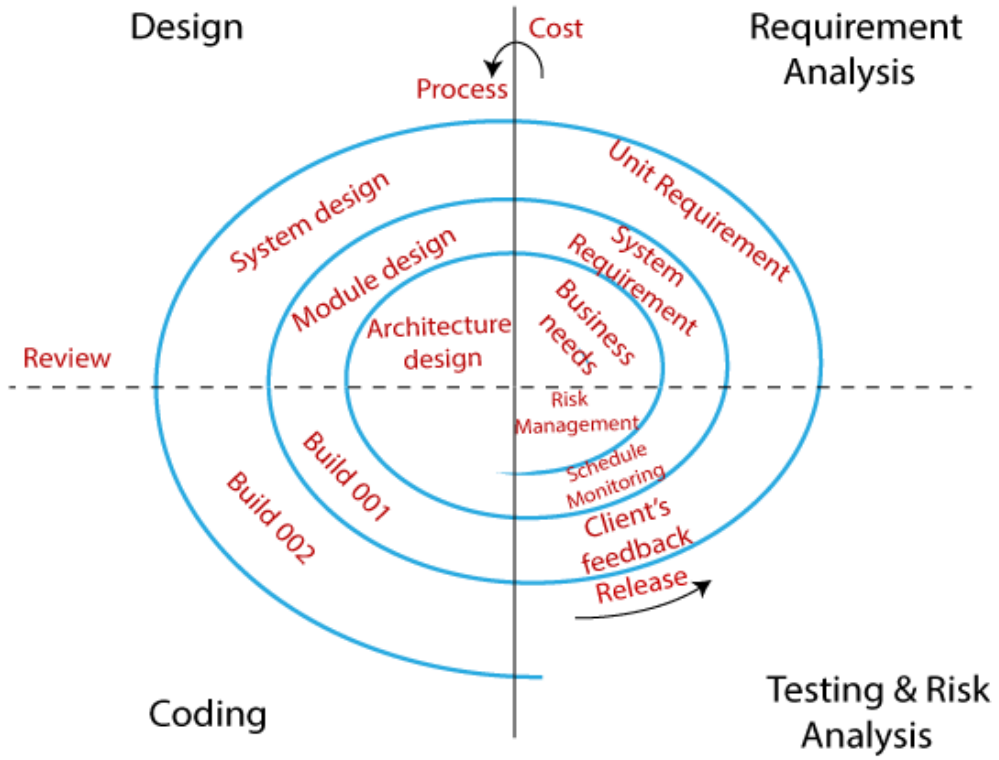
Advantages:

1) Product will be robust
2) Simple model to follow

Disadvantages:

1) Requirement collection and design is not tested
2) Backtracking is not possible

SPIRAL MODEL



It is also called the Iterative model or Incremental model.

This model is called Incremental model because the customer keeps on adding the module after every cycle. It is called an Iterative model because the steps keep on repeating.

This model is divided into 4 stages:

1) Requirement and Feasibility Study
2) Design
3) Coding
4) Testing

Module A - Let's say customer gives module A, in stage 1, we do Requirement collection and feasibility study, later on it is moved to Design stage, here Architect will do HLD and LLD and will be moved to coding stage, where Developer will look into the Requirement and LLD and starts writing the code. Later once the module is moved Testing, testers will test if the module is stable or not. If the module is stable, it will be released to the customer.

Module B - Let's say customer gives module A, in stage 1, we do Requirement collection and feasibility study, later on it is moved to Design stage, here Architect will do HLD and LLD and will be moved to coding stage, where Developer will look into the Requirement and LLD and starts writing the code. Later the module is moved to Testing, here

1) Testers need to test for module B
2) Testers need to test for module A
3) Later, data flow between module A and module B needs to be tested

Once the modules are stable, we can release it to the customer.

Module C -- Let's say customer gives module C, in stage 1, we do Requirement collection and feasibility study, later on it is moved to Design stage, here Architect will do HLD and LLD and will be moved to coding stage, where Developer will look into the Requirement and LLD and starts writing the code. Later once the module is moved Testing, here

1) Testers need to test for module C
2) Testers need to test for module B
3) Testers need to test for module A
4) Later test the dataflow between module A and B

5) Test the dataflow between A and C

Finally, once the modules are stable, it will be released to the customer.

Advantages:

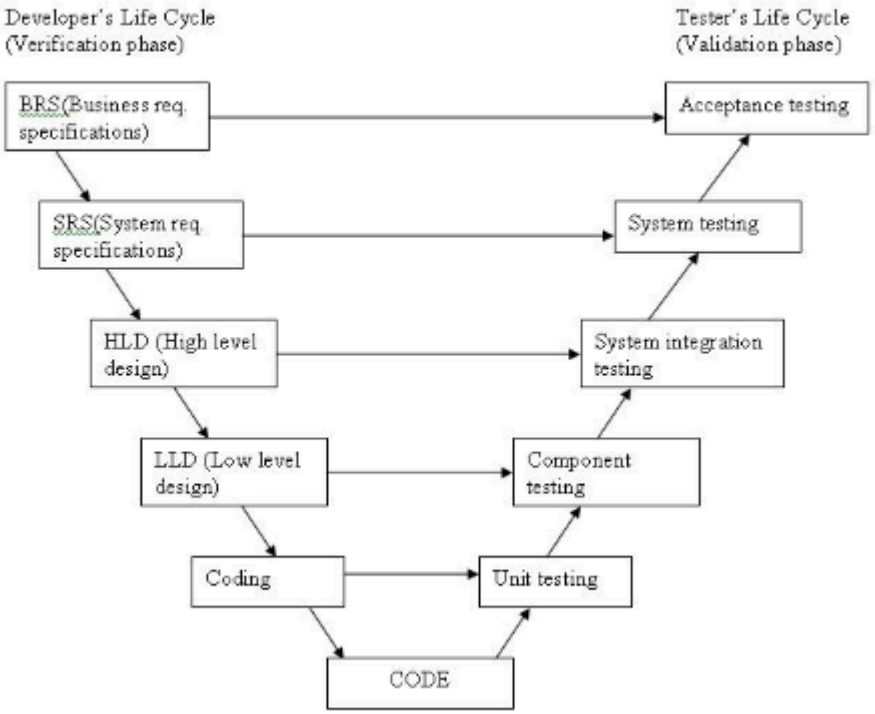
1) Initial investment is less
2) Integration between the modules is tested
3) Requirements changes are allowed after every cycle

Disadvantages:

1) Requirements changes are allowed only after the end of the cycle, not in between the cycle.
2) Every stage looks like a waterfall model.

V & V model(Verification and Validation model)

Diagram of V-model:



In this model, Developers and Test Engineers work parallely, hence it is called V & V model.

Explanation for the model:

Customer will give the requirement in the form of 'CRS'(it is also called BRS as in the above diagram) that will be in Business language. BA will go to the customer place and collect the CRS and Explain to 'PM' in the business language.

PM will take up the 'CRS' and send it to both developer and test engineers. Test engineer review CRS and start writing Acceptance test plan and test cases, parallelly developer will be converting 'CRS' to SRS, while reviewing CRS, if test engineer find any bug(conflicting requirement, missing requirement, wrong requirement) they will communicate to developer and developer will communicate bug to customer and customer will update 'CRS' and send updated 'CRS' to both test engineer and developers. Tester will review updated CRS and also update corresponding Acceptance test plan and Acceptance test cases, parallelly developer will update defect in 'SRS' and continue converting 'CRS' to 'SRS', once both developers and test engineers complete the work, 'SRS' will be ready.

'PM' will take up 'SRS' and send it both to developers and test engineers. Test engineers will review 'SRS' against 'CRS' write System test plan and test cases. Parallelly developers will be converting 'SRS' to 'HLD' once both developers and Tester complete the work, 'HLD' will be ready. Like this way flow continues till the coding once after coding is completed, Developers will do WBT (testing each and every time of the code). While doing WBT, if developers find the defect in the code, developers will fix the defect and do WBT. Once WBT is completed, developer will give software to tester and Tester will start with validation

activities, where in first tester will first do Functional testing by executing Functional test cases, while doing functional testing, if test engineer finds any bug in the software, then test engineer will inform to developer. Developer will fix the defect and do WBT and give new software to the tester and Tester will uninstall the old software, install new software & test the defect first and continue doing functional testing, after functional testing is completed. Testers will do Integration and System Testing respectively. Later on, customer/end-users will do Acceptance testing by executing test cases once Acceptance testing is completed, software quality is good, then software will be released to the customer, where the customer will use software and run the business.

Advantages:

1) Total time taken is less since testing starts from the early stage itself.

2) All the stages are tested (CRS, SRS, HLD, LLD, Code).

3) Since testing is done in every stage, the downward flow of defects is less because of this less Reworking and less time consuming.

4) Requirement changes can be done in any stage.

Disadvantages:

1) Initial investment is high.

2) Documentation is more (in every stage, we review documents and write test plan and test cases).

AGILE

It is an Iterative and Incremental approach where requirements keep on changing, as a company we should be flexible to take up all the requirements, develop the changes, test the changes and give quality software to customers within a short span of time.

Through Agile, we can provide customer satisfaction through “quick delivery of working piece of the software”.

Advantages:

1) Customers can change the requirement at any stage of development.

2) Release will be short.

3) Developers, Test Engineers and BA will be having meeting regularly. It will be helpful to improve the process.

4) Teams will be self-organized.

5) This methodology is fast and flexible.

Disadvantages:

1) We are not sure when the customer will stop changing the requirements.

2) We cannot do effort estimation at the beginning of the project.

3) More pressure in the process.

Few Types/Flavours of Agile:

1) Scrum model

2) Extreme Programming

3) Feature Driven Development

4) Crystal clear

5) Lean and Kanban

Scrum model – Scrum model is a standard procedure/framework which helps team to work together and develop new software.

Agile Terminologies:

1) **Epic** – Collection of user stories.

2) **User stories** – These are nothing but functionalities or features

3) **Sprint** – Actual time taken to develop and test one or more user stories.

4) **Story point** – It is a rough estimation given to develop and test individual user stories. It is calculated in terms of the Fibonacci series.

5) **Burndown chart** – This chart is used to calculate the total amount of work remaining at the beginning of every sprint.

6) **Burnup chart** – This chart is used to calculate the total amount of work completed at the end of every sprint.

7) **Capacity** – Total amount of available hours of every individual engineer for the given sprint is called Capacity.

8) **Velocity** – Total amount of work completed by every individual in the given sprint is called Velocity.

Q) Who is a Scrum master?

A) Scrum master is a person who is responsible for delivery of the software in the planned period of time. He will track each and every activity of team members working on the project. Scrum master mainly works on Project management tools like Jira where he creates Epic for requirements, create user stories for the epic and create tasks for the user stories and assign tasks to every individual engineer.

Scrum ceremonies/meetings in Scrum model

1) **Product Backlog** – This contains a list of all user stories or tasks to develop a software. Product Owner is responsible for backlog of product, based on Product owner feedback, the user stories will be divided to work efficiently in further stages.

2) **Sprint Planning** – It is a meeting conducted by Scrum master on the first day of the Sprint(some do conduct on the last day of the Sprint). In this ceremony, BA will explain how each and every user story should be worked and explains business workflow. In this meeting, user stories will be prioritized according to customer requirements and work will be assigned to every team member respectively.

3) **Sprint Review** – This is a demo/report given to Product owner/Stakeholders, how the user stories have been worked so far. The intention of conducting this review to get feedback on the product worked so far. This review is helpful in analyzing the progress of the product.

4) **Sprint Retrospect** – This meeting is usually conducted at the end of the Sprint, where Scrum team members discuss what went well, what did not go well and what are the action plans. The main purpose of conducting this meeting is to get feedback on the

process we are working on and implement the ideas to make it better. This meet can also be called a Sprint closure meeting.

5) **Scrum meeting** – This is also called a Daily stand-up meeting, conducted everyday for a short duration, ranging from 15-20 mins.

6) **Bug Triage Meeting** – This meet is conducted to know the status of pending and open defects and we will prioritize bugs according to customer business workflow and fix it accordingly. Sr Testers are responsible for this meeting as they have better ideas on bug priorities. As Testers, we are supposed to give a Live report once a bug triage meeting is done before production.

7) **Project closure meeting** – In this meeting we discuss the workflow done so far, how efficient we were in delivering quality products and what are the areas of improvement. This is usually conducted at the end of the Project.

Q) What is a Sprint Backlog?

A) In Jira, during Sprint planning, we decide what are the stories we are prioritizing during the start of the Sprint and those user stories will be stored in the Sprint backlog.

Q) How do you rate story points to your user stories?

A) We need to first breakdown user story into task and subtasks and convert those into scenarios, Test cases, identify time taken to execute Test cases, roughly estimate time period and rate story point using Fibonacci series.(Fibonacci series – 0,1,1,2,3,5,8.13,21…..)

Q) What is the Test Cycle duration?

A) Time period taken to complete test 1 build is called Test cycle duration.

DEFECTS

Definition - Any deviation from customer requirements specification is called a defect.

It can be categorised into Severity and Priority.

Severity and Priority

1) **Severity** – Impact of Bug on customer business workflow is called Bug.

If the impact is high, it can be considered as high severity.

If the impact is low, it can be considered as low severity.

2) **Priority** – Importance given to fix the bug(how soon the bug needs to be fixed) is called Priority.

Types of Severity:

1) Blocker
2) Critical
3) Major
4) Minor

1) **Blocker** – Lets say, when the Tester is testing the application it is not allowing the user to proceed further, it may say internal server error or might take the user to a blank page, the functionality itself is blocked. This is considered a Blocker bug.

2) **Critical** – If the main functionality of the application is not working or not according to customer requirement specification, it can be considered a Critical bug. In other words, if Tester is sure that this bug affects customer business flow, then it can be termed as Critical bug. Note that it is not blocking the user, but customer business work flow is affected largely because of this bug.

3) **Major** – If the Tester is not sure that this bug which is found is affecting the customer business workflow or not, it can be considered as a Major bug. In other words, if minor functionalities of an application is affected, it can be considered a Major bug.

4) **Minor** – If the Tester is sure that this bug is definitely not affecting the customer business workflow, it can be considered a minor bug. These are usually with respect to the look and feel of the application, alignment issues, spell issues etc.

Types of Priority

1) P0 – High Priority, the developer has to fix the bug immediately.

2) P1 – Medium Priority, developers can fix the bug in upcoming builds.

3) P2 – Low Priority, developers can fix the bug in the coming releases.

Blocker and Critical bugs can be considered as High severity.

Examples:

High Severity High Priority – While logging into the login page, a blank page is displayed. It is a blocker defect and has a large impact on customer business workflow, so it is a High severity and needs to be fixed immediately.

High Severity Low Priority – When a user clicks on the Terms and conditions page, a blank page is displayed. It is High Severity low priority because though it is a blocker defect, this does not affect customer business workflow and can be fixed in later stage.

Low Severity High Priority – When logging in to the login page, let's say login spelling is wrong (logone), now this is low severity because it is a minor defect, but it will impact the user as it is displayed in the first page itself, hence we fix the defect on High priority.

Low Severity Low Priority – Lets say there is a spelling mistake in the inside contact information page with respect to email Id, phone etc or any color formation issue or alignment issues. This can be considered as a Low severity as it is a spelling mistake issue and does not have any impact on customer business workflow and low priority as it can be fixed in the later stage.