Course Content

1) Software Testing

- Introduction to s/w Testing
- Software Development Process
- Project Vs Product
- S/W quality
- S/W Testing Methods
- Roles & Responsibilities of a TE
- Environments in Project
- Quality Management

2) Software Development Life Cycle (SDLC)

- SDLC Models
- Waterfall Model
- Prototype Model
- Incremental / Iterative Model
- Spiral Model
- RAD Model
- Big-Bang Model
- Fish Model
- V-Model
- Agile Model (Scrum)

3) Testing Methodologies

- White Box Testing
- Black Box Testing
- Gray Box Testing

4) Testing Levels in SDLC

- **Review on Requirements**
- **Review on Design**

1) **Unit Testing**
   - **Statements Coverage**
   - **Loops Coverage**
   - **Conditional Statements**
   - **Path or Branch Coverage**

2) **Integration Testing**
   - **Big-bang Approach**
   - **Incremental Approach**
     - **Top Down Approach**
     - **Bottom Up Approach**
     - **Hybrid/Sandwich Approach**

3) **System Testing**
   - **Functional Testing**
     - **Object properties coverage**
     - **Error handling coverage**
     - **Input-domain coverage**
     - **Calculation coverage**
     - **Data base coverage**
     - **Links coverage**
   - **Non Functional Testing**
     - **GUI Testing**
     - **Usability Testing**
     - **Performance Testing**
     - **Recovery Testing**
     - **Security Testing**
     - **Compatibility Testing**
     - **Configuration Testing**
     - **Comparative Testing**
     - **Installation Testing**
     - **Sanitation Testing**

4) **User Acceptance Testing**
   - **Alpha Testing**
   - **Beta Testing**

5) Testing Terminology

- Smoke Testing
- Sanity Testing
- Re-Testing
- Regression Testing
- Ad-hoc Testing
- Exploratory Testing
- Jump/Monkey Testing
- L10N Testing (Localization testing)
- I18N Testing (Internationalization testing)
- Globalization testing
- Mutation Testing
- Defect seeding/be-bugging

6) Software Testing Life Cycle (STLC)

- Test Strategy
- Test Planning
- Test Case Design
  - Fundamental of TC's
  - TC Design Technique (BBT)
  - Types of TC's
  - TC Template
  - TC Reviewing
- Test Execution
- Defect Reporting & Tracking
  - Defect Reporting
  - Defect Life Cycle (BLC)
  - Severity, Priority
- Test Closure
  - Criteria for Test Closure
  - Test Summary Reports

Software Testing

- Verification & validation of software is called as testing.
- Verification means whether the software is correctly implemented or not.
- Validation means the implemented software meets the customer requirements or not.

- Software testing will help to deliver reliable application to the customer and also it will reduce maintenance cost for a project.

- The objective of Software testing is to identify defects. When those are resolved the software quality improves.

Defect:

It is a deviation between expected result and actual results in AUT (Application under Test).
Defect can also call as Issue/Incident/Fault.

Bug:

When developers are accepted defects then those are called bugs.

Failure:

When defects are reached end users then those are called failures.

Note: In general defects may present in application due to human mistakes while developing application like syntax errors and logical errors.

Software Quality:

From producer point of view when application full filled with all the user requirements and from end user point of view when application fit for use the it is consider as quality application.

In general Software Quality depends on following factors:

1) Budget/cost
2) In time Release
3) Reliability
4) Meet client requirements in terms of functionalities.
5) Meet client expectations in terms of performance, compatibility, user friendliness…etc.

Some of the activities:

Coding: {SDLC}

Writing programs using programming or scripting languages in order to develop the application. It is performed by developers or programmers.

Testing: {STLC}

In general every organization will maintain separate testing team. After successfully application developed that will be delivered to separate testing team.
Validating application based on client requirements and expectations is called testing.
Following are members in testing team.

- **Test Manager**
- **Test Lead**
- **Sr.Test Engineer**
- **Test Engineer/Trainee**

Defect Reporting: {BLC}

Notifying about the defects to the developers is called as defect reporting. It is performed by test engineer.

Debugging:

Analyze source code of the application in order to identify root cause for a defect. It is performed by developers.

Bug Fixing:

Modifying source code of application in order to solve the defects is called as bug fixing/bug resolving.

Skill set required for a functionality test engineer:

1) **Knowledge on manual testing concepts.**
2) **Knowledge on any one of functionality testing tool like Selenium, QTP.**
3) **Knowledge on defect reporting tool like QC, Bugzilla, Jira.**
4) **Some portion of knowledge on data base like Sql-Server, Oracle.**

Software Testing Methods:

In general every organization uses 2-types of testing's to test an application.

1) **Manual Testing**
2) **Automation Testing.**

Manual Testing:

1) It is the process of testing an application manually without using any automation tool.
2) In manual testing Test engineer derived & executes the test cases & generate the reports manually.
3) In manual testing there are 2-types of approaches.
    1) Static Testing
    2) Dynamic Testing

Static Testing:

1) Without executing application identify the defects is called as static testing (Verification).
2) It includes checking of requirement documents, design docs etc...
3) We can perform static testing by using peer reviews, walkthroughs, and inspections.
   Eg: Verify the spelling mistakes in buttons, labels is a static.

Dynamic Testing:

1) Perform the testing with execution of application is called as dynamic testing.
2) It includes checking of every feature of total application.
3) We can perform dynamic testing by using following methods.
    1) WBT
    2) BBT
    3) GBT
    4) Unit Testing
    5) System Testing
    6) Integration Testing
    7) UAT (User Acceptance Testing).....etc

Eg: Validate Sign in button is correctly working or not in Google.com is a dynamic testing.

Automation Testing:

1) Automating human activities with the programming language or any 3rd party tool is known as automation.
2) The process of converting manual test cases into test scripts by using some tool is also known as automation.
3) Performing testing activities by using some automation tool is known as automation testing.
4) The following are the various automation testing tools:
    1) QTP /UFT (Quick Test Professional / Unified Functional Testing Tool
    2) Selenium
    3) Lisa
    4) Tosca
    5) Win runner
    6) Test complete

5) In automation testing automation engineer is responsible to develop and exectues the test scripts.

Note:

- Test case is related manual testing and can be developed with the help of excel files.
- Test scripts is related to automation testing and can be developed by using some programming languages like Java, ruby, puthon, C# etc.

Difference between Manual Testing & Automation Testing

Manual Testing	Automation Testing
1) Manual testing can required Human Interaction to execute the Test cases.	1) Automation testing uses tools to develop and execute the Test Scripts.
2) Manual testing will required Skilled Resources long time will be high cost.	2) Automation testing save the time, cost & man power.
3) Any type of application can be tested manually. Conducting testing types ad-hoc testing, monkey testing can be done only by Manual testing.	3) Automation testing can be recommended only for stable applications and it is mostly used for regression testing.
4) Manual testing can be boring and repetitive.	4) The boring part of executing Same test cases again and again Can be handled by automation Testing.

Note:

1) 100% automation testing is not possible.
2) Every application should be tested at least manually once.
3) Manual testing can be applicable for all the applications but automation testing may not applicable.

Differences between Verification & Validation

Verification	Validation
1) It is the process of verifying whether the application is implemented correctly or not.	1) It is the process of validating whether the application meets the client requirements or not.
2) It won't requires execution of the application	2) It required execution of the application.
3) It is also known as static testing.	3) It is also known as dynamic testing.
4) It includes checking of the requirement documents like BRS, Design, FRs etc.	5) It includes testing of all thefeatures of application with execution
6) We can perform verification by using the methods. → peer review → walkthroughs → Inspections	5) We can perform validation by using the methods. → WBT → BBT → GBT
7) It finds the bugs or defects early in the development life cycle which is very essential w.r.t cost	8) It finds the bugs or defects which are not identified in verification & very costly to reslove.

Difference between Testing & Debugging

Testing	Debugging
1) It is the process of verifying Whether the application fulfills Client requirements or not whether the application contains any defects Or not.	1) It is the process of analyzing the source code of application to identify root cause for a defect.
2) Testing will be performed by the Test Engineer.	2) Debugging will be performed by the developers.
3) Testing will be performed as a part of testing phase.	3) Debugging will be performed as a part of WBT or unit testing

Roles & Responsibilities of Test Engineer:

- ➔ Review on requirements like BRS &FRS to understand application functionalities.
- ➔ Identify test scenarios for allocated modules.

Test Scenario: It describes test condition or requirement or functionality which we need to validate.

Ex: Identify test scenarios in Gmail home page.

TS01: Login validation.
TS02: Stay signed in.
TS03: Need help.

- ➔ Prepare test cases for manual testing.

Test Case: It consist set of steps or sequence of steps with user actions & subsequent response from the system.

Ex: Identify possible test cases for "Login validation".

TC01: Verify login functionality with valid data.
TC02: Verify login functionality with in valid data.
TC03: Verify login functionality without data.

Ex: Write test cases to verify login functionality with invalid data.

Test case ID/Name	Test Case Description	Step Name	Step Description	Expected Result
TC01_Gmail _Login	This TC to verify login functionality using invalid data.	Step 1	Enter invalid email valid pwd & click on "signin". Email="selenium". Pwd="selenium1".	Error message should popup. "Invalid user Id"
		Step 2	Enter valid email in valid pwd & click on "signin". Email="peers. Selenium". Pwd="selenium".	Error message should popup. "Invalid password"
		Step3	Enter invalid email in valid pwd & click on "signin". Email="selenium". Pwd="selenium".	Error message should popup. "Invalid user Id & password"

- ➔ Develop automation test script.
- ➔ Review on test cases and automation test script to check correctness and completeness.
- ➔ Execution of test cases and automation test scripts.
- ➔ Defect reporting & prioritize defects.
- ➔ Performing retesting & regression testing.

Software:

Set of statements logic and related data which gives instructions to system to perform specific task based on usage. There are 2 types of Softwares.

1) Product based Softwares
2) Project based Softwares

1) **Product based Softwares**
   Whenever Softwares developed based on standard or generic requirements and that can be selling to any customer such Softwares is called as product based Softwares.

2) **Project based Softwares**
   Whenever Softwares developed based on particular client requirements and that should be deliver that a particular client only.
   Ex: HDFC bank.com, ICICI bank.com

Based on work there are 2 types of companies in industry.

1) Product based
2) Service based

1) **Product based:** Company which will focus on their own products to develop.
2) **Service based:** Company which will develop projects for individual clients.

Company Client/Customer & End user:

Company: It is an organization where application will be developed.
Client/Customer: It is an organization or individual person who will provide requirements to develop application.
End User: Who will be use s/w at work environment with real data?

Different environments in project:

There 4 types of environments in a project.

1) Development Environment:

In this environment development team will involve to develop application based on client requirements.

2) Testing Environment:

After successful implementation of project that will be deployed/installed at test environment to validate application based on requirements.
Where separate testing team involve validating application.

3) UAT Environment:

In this environment client team involve validating application in order to confirm project is acceptable or not.

4) Production Environment:

In this environment application used by end user with real data. This is also called as live environment.

System Software & Application Software:

1) System Software:

This is also called as BIOS (Basic Input & Output System). These used to provide interface among the system components. **Ex:** Devise drives, OS…Etc.

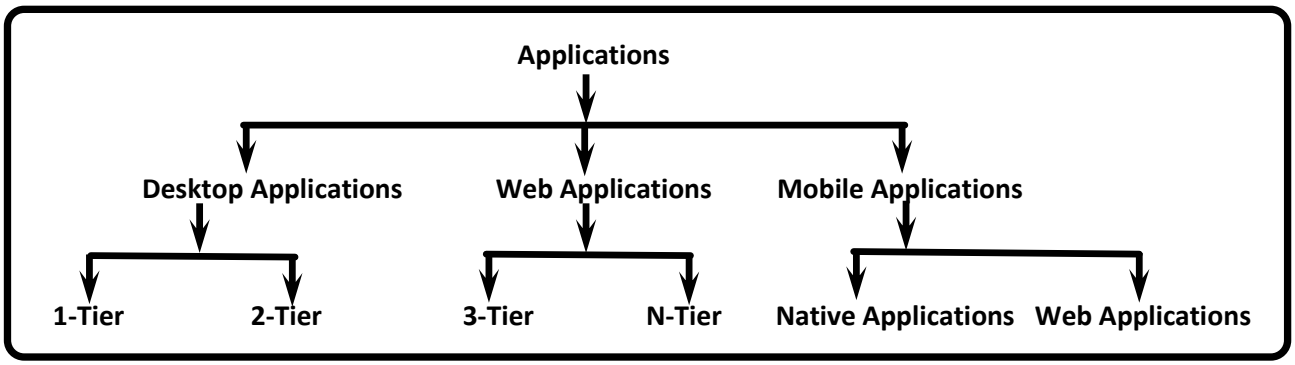
2) Application Software:

This is also called as front end application. These are developed based on user business needs whereas front end used to manipulate data into database.

Application Softwares are:

1) **Desktop Applications**
2) **Web Applications**
3) **Enterprise Applications**
4) **Distributed Applications**
5) **Mobile Applications**
6) **Embedded Applications**

All the applications are the combination of the environments.



1) Desktop application / Standalone app:

The applications which are running on a single computer called as standalone application. Two types of standalone applications are there.

1) **CUI Based**
2) **GUI Based**

1) CUI Based:

Applications which are running through console are called as CUI base app. These apps are also called as console apps.

Ex: Any Java Program

2) GUI Based:

Applications which are running with graphical user interface are called as GUI based .These are also called as desktop app.

Ex: Ms-word, calc etc.

2) We Applications:

1) The applications which can provide services over the web are called as web application. We can develop web applications by using web technologies like servlets, JSP, PHP etc.
2) Web application will be hosted on a special computer which is known as web server.
3) By using web client (Browser) end user will send request and web server will process the request & sent requested response to the client. Web client will display the response to the end user.

Web Servers: Tomcat, JETTY etc.....

Note: Every web application will be represented in a war file form. war (web archive) is compressed file which contains all the resources of the web application.

3) Enterprise applications:

These applications are more superior to web applications. We can develop enterprise applications by using technology like web technology, distributed technology etc.

Ex: servelets, JSP, EJB etc.....

Enterprise apps are banking apps, tele communication apps.....

Note: Enterprise applications can be represented in a compressed form which is nothing but ear file.

ear:(Enterprise archive) i.e nothing but build file.

Note:

1) Enterprise applications can distribute across the several machines such type of applications are called as distributed applications.
2) The main advantages of distributed applications are scalability, No Fail-over situation.

4) Mobile applications:

The applications which are running on mobile device are called as mobile applications.

Ex: BookMyShow mobile app, PayTM mobile app

The most popular technologies to develop mobile applications are android & IOS.

There are 2 types of mobile apps.

1) Native apps
2) Browser apps

3 Layers in Application:

In general Software application contains following 3 layers.

1) **Presentation Layer**
2) **Business / Application Layer**
3) **Data Layer**

1) Presentation Layer:

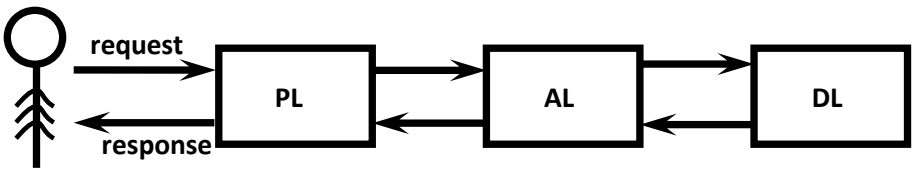
- **It is also known as client layer.**
- **It is the top most layer of the application.**
- **It is the view part of the application.**
- **This is the layer which end user can see whenever using the application.**
- **The main functionality of presentation layer is collecting the information from the end user & passing to the next level.**
  Ex: Login page of Gmail, Login page of HMS.

2) Business Layer / Application Layer:

- **It is responsible to perform total business functionality.**
- **Application layer process the information provided by the presentation layer & interact with data layer & provided information to the presentation layer.**
  Ex: whether user provides user name & password application layer is responsible to validate & provide required response with the help of data layer.

3) Data Layer:

- **The data & corresponding logic will be stored in this layer.**
- **Application layer communicates with the data layer to retrieve the data. It contains logic to interact with data base like insert, update & delete.**



Note:

Presentation Layer	**: View Part of Application**
Application Layer	**: Processing Part of Application**
Data Layer	**: Data processing Part of Application**

DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

Software Architectures:

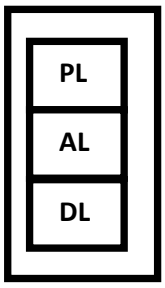
1) It is the fundamental design of the software system.
2) It describes the layers of the application.
3) It helps to every memebr of the project to understand apllication structure.
4) It is helpful for code maintainability, reusability, productivity etc.

Types of Software Architectures:

1) 1-Tier Architecture
2) 2-Tier Architecture
3) 3-Tier Architecture
4) N-Tier Architecture
5) Peer to peer Architecture

1) 1-Tier Architectures:

- If the PL, BL & DL are available in a single machine, then it is considered as 1-tier architecture.
- It is also known as standalone architecture.



Same Machine

The Main Advantage of this Architecture is:

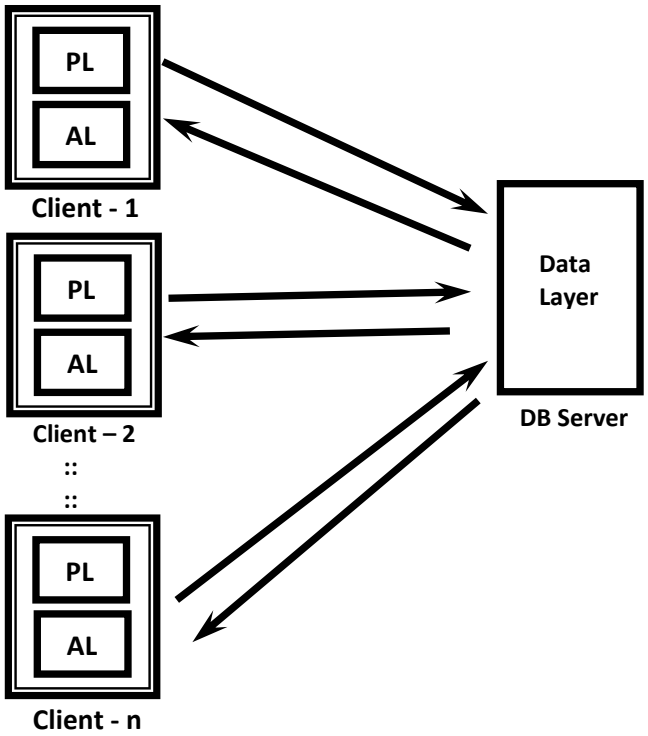
- It is very Simple.
- It is having high Performance

The main Limitations of this Architecture are:

- Not suitable for large scale projects.
- No networking, hence we can't access from remote server.

2) 2-Tier Architecture:

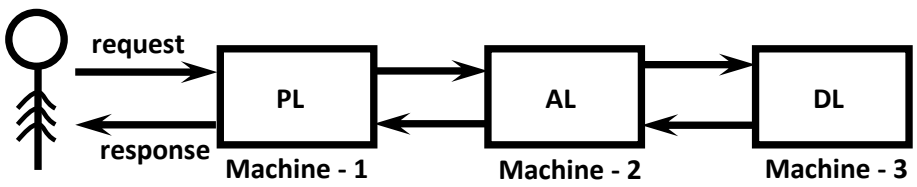
- The PL & BL are available in one machine & data layer is available in other system, then it is consider as 2-tier architecture.
- It is also known as client-server architecture. Client can communicate with server by using LAN /WAN.



- It is most rarely used architecture.
- Application communicates directly with the data base which may cause security problem.
- It is very difficult to scaleup & difficult to maintain.

3) 3-Tier Architecture:

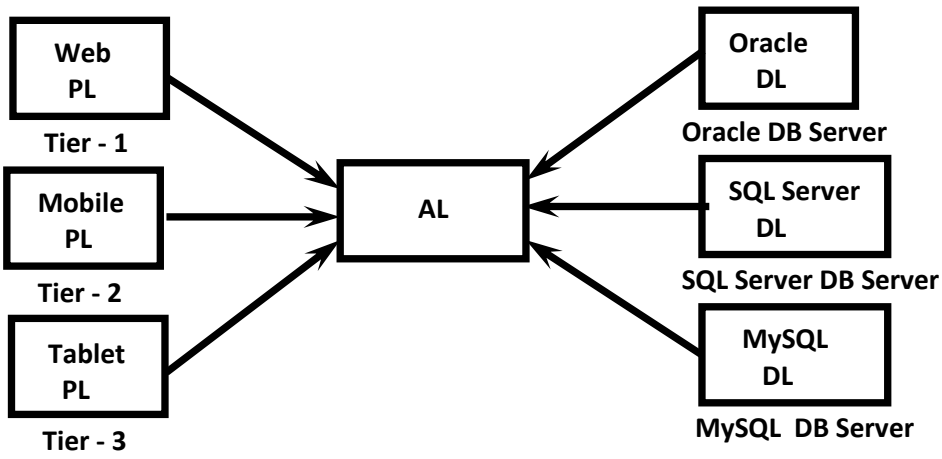
- If all the 3-layers (PL, AL, DL) are available in different systems then it is consider as 3-tier architecture.
- It is most commonly used architecture for web applications.
- It supports scalability & maintainability.
- Enhancements will become very easy.
- It provides security.



4) N-Tier Architecture:

- If PL, BL, DL are divided into multiple tiers then it is considered as N-tier architecture.
- Based on application requirement we may distribute any layer across several machines.

  Ex: PL can be divided into multiple tiers like.
  - One tier for we presentation layer
  - One tier for mobile presentation layer
  - One tier for Tablet presentation layer
- Ex: The data layer can be divided into multiple tiers like.
  - One tier for oracle data layer
  - One tier for MySQL layer etc.
- It is also known as distributed architecture, best suitable for enterprise applications.

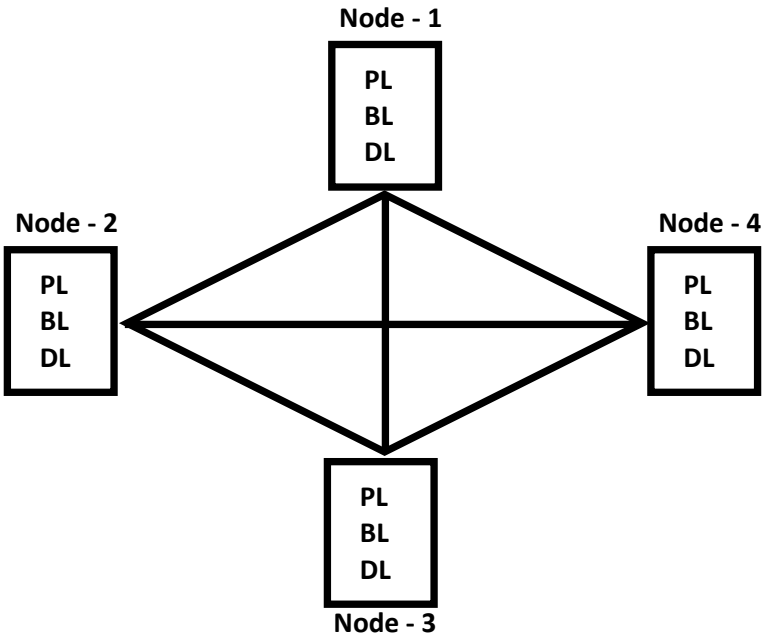


The main Advantages are:
- Scalability
- Maintainability
- Can handle Fail-over situations very easily.

- In general we can distribute business layer into multiple machines.

5) Peer-To-Peer Architecture:

- No server concept.
- All the 3-layer (PL, BL, DL) are hosted in a single machine which is nothing but a node. We can maintain multiple nodes.
- No single point of failure, even one node fails then other node can process requests.
- Best suitable for small applications with high range scalability.



Quality Management:

It is a process of preventing defects in the development of application. The aim of this process is no defects in final product.

Quality Management divided into 2-parts.

1) **Quality Assurance**
2) **Quality Control**

1) Quality Assurance:

- **QA team will define development process of application to prevent defects in application. QA team involve throughout life cycle to monitor and measure the strength of development process, if any weakness identified they will provide suggestion to improve the strength of development process.**
- **High level management team, domain experts, technical experts are the members in QA.**

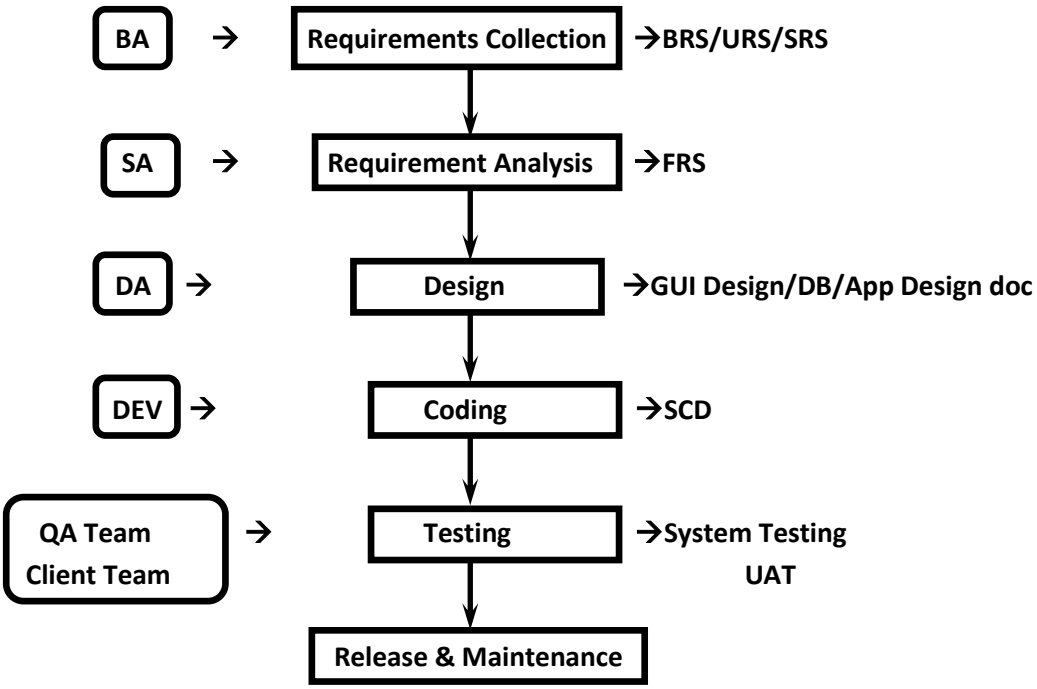
2) Quality Control:

QC team involve after product is developed. In order to deliver in the defects and make sure those should be resolved before deliver application to the customer.

SDLC

It describes developing process of software project/product to fulfill the client requirements within the specified cost & time. Following are the phases in SDLC.

- **Requirements Collection**
- **Analysis**
- **Design**
- **Coding**
- **Testing**
- **Release & Maintenance**



1) Requirement collection:

- **In this phase Business analyst (BA) will collect the requirements with an interaction of client and collected requirements will be documented as BRS/URS. BRS describes core business logic or brief description about client business needs in terms of who are the users for application and required services for those users.**
- **After preparation of BRS they will perform feasibility study to check project is acceptable or not in order to develop.**
- **Following are factors in feasibility study.**
  - **Finance feasibility**
  - **Time feasibility**
  - **Requirements are reliable or not interims of technology to develop.**

- If project is acceptable then BA will intimate to the client by releasing RFP (Request for Proposal) and SLA (Service Level Agreement).

2) Requirement Analysis:

- In this phase SA will studying user requirements from BRS based on that will prepare FRS.
- FRS describes detailed functionalities of each component in application. For successful implementation a clear FRS is an essential.

3) Design:

- In this phase DA will study user requirements using BRS/FRS based on that he will decide required architecture of the application (window/web) based on that he select requirement programming & data base technology.
- In design phase DA prepare following

⚽ **GUI Design Document:**

It contains dummy screens of application to foresee the future implementation of Application. (i.e. like Blue Print of Application).

⚽ **Data base design document:**

It describes about data base structure interims of number of tables, relation among tables & rules implemented in data base.

⚽ **Application design document:**

- **HLD Document:** It describes number of modules required in a project and relation among those modules.
- **Module:** It is some portion of application which contains related functionalities.
- **Modularization:** Splitting the project in to small units/modules for easy development of application.
- **LLD Document:** For each module there will be individual LLD documents will be prepared. It describes internal logic of each module with the help of data flow diagrams, ER diagrams.

4) Coding:

In this phase developers write the programs in order to develop the application as per design documents. Output of this phase is source code document (SCD).

5) Testing:

- After coding application is available for execution. Initially developers perform unit & integration testing using white box testing techniques.

- **Separate testing team performs system testing using black box testing techniques. Client also performs UAT.**

6) Release & Maintenance:

- **Release: After system testing & UAT when client satisfied on our work product then we deliver application for future usage is called Go-Live/Production/Release**
- **Maintenance: While using application client may identify some defects (Latent Defects) or he may need new features in application then he will send change request (CR) to company.**
- **Based on initial agreements change control board (CCB) will work on CR and then we deliver modified version to the customer.**

Q: When SDLC end for a project?

When project is no longer in use then that will be end state for SDLC of that project.

Common problems in SDLC:

- **Poor Requirements: If req are not clear or incomplete that will be a problem.**
- **Unrealistic schedule: If too much of work crammed in a too little time that will be a problem.**
- **Inadequate testing: In present scenarios it is difficult to estimate how much testing sufficient to validate application.**
- **Dynamically changes in requirements.**
- **Miss communication b/w developers, TE, DA, PM Etc.**

Q: When defects will arise while developing application?

1) Wrong requirements → Wrong project/product.
2) Mistakes in design → Design defects.
3) Mistakes in coding → Coding defects in project/product.

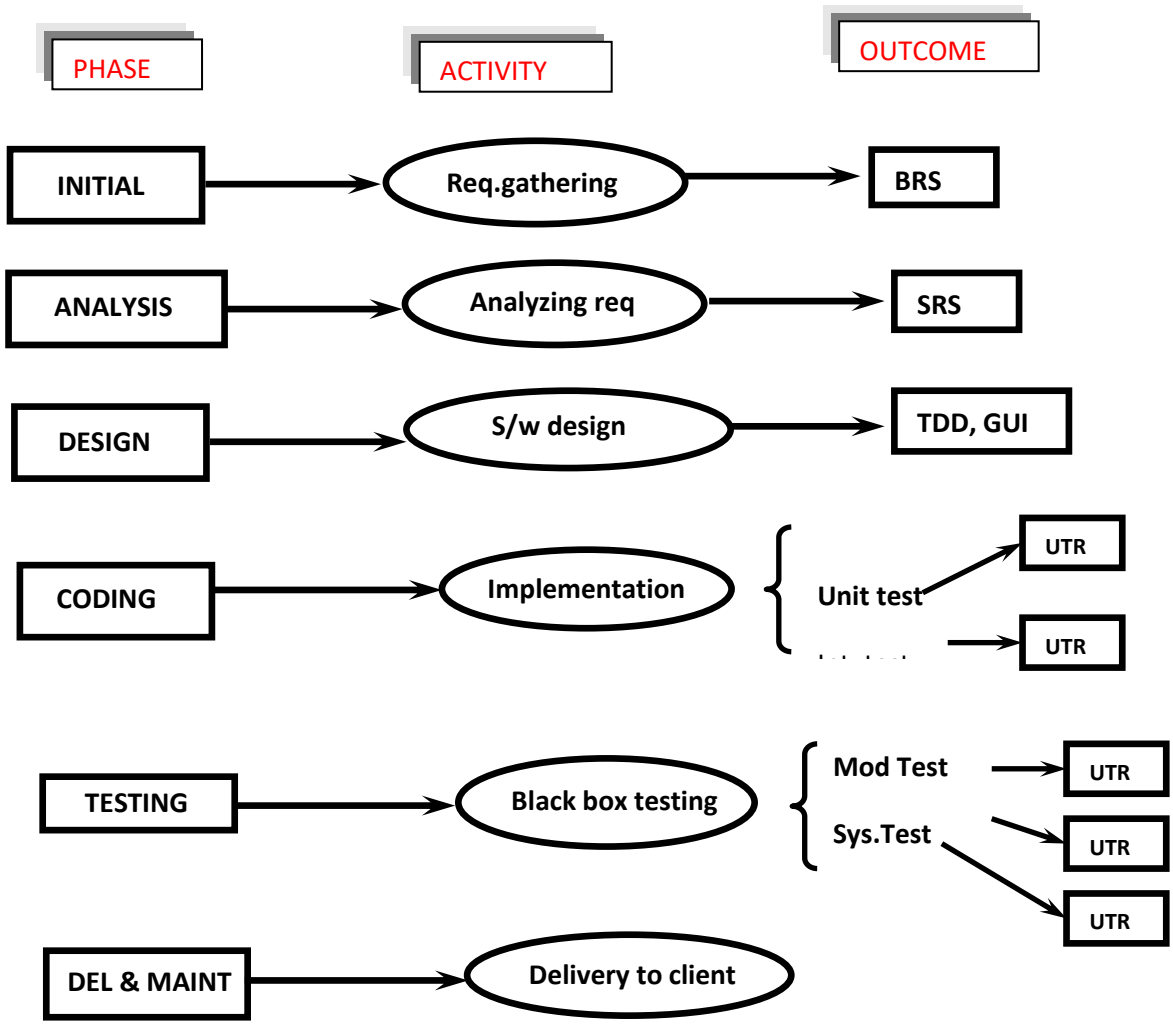
Note: Testing should start early stages. It identifies defects will take less cost to resolve those defects compare to later stages identified defects.

SDLC MODELS:

We have the below Software Development Models:

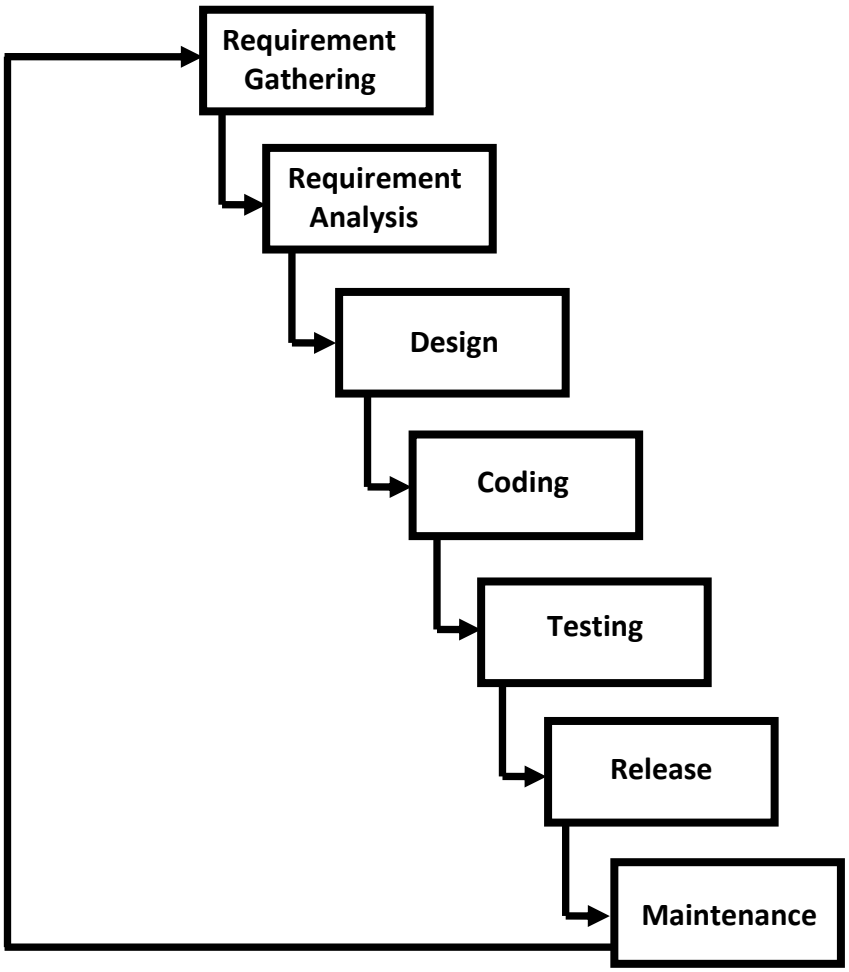
1) **Waterfall Model**
2) **Prototype Model**
3) **Incremental / Iterative Model**
4) **Spiral Model**
5) **RAD Model**
6) **Big-Bang Model**
7) **Fish Model**
8) **V Model**
9) **Agile Model (Scrum)**

1) Waterfall Model:



- It is the oldest SDLC model also known as linear sequential model.
- In this model each phase must be completed before the next phase can begin & there is no overlapping of phases, i.e all the phases will be performed one by one.
- Once the phase is completed the high level management will analyze the phase.
- How the water falls from one to other end in the same way the phases of SDLC will be implemented.

DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com



Advantages:

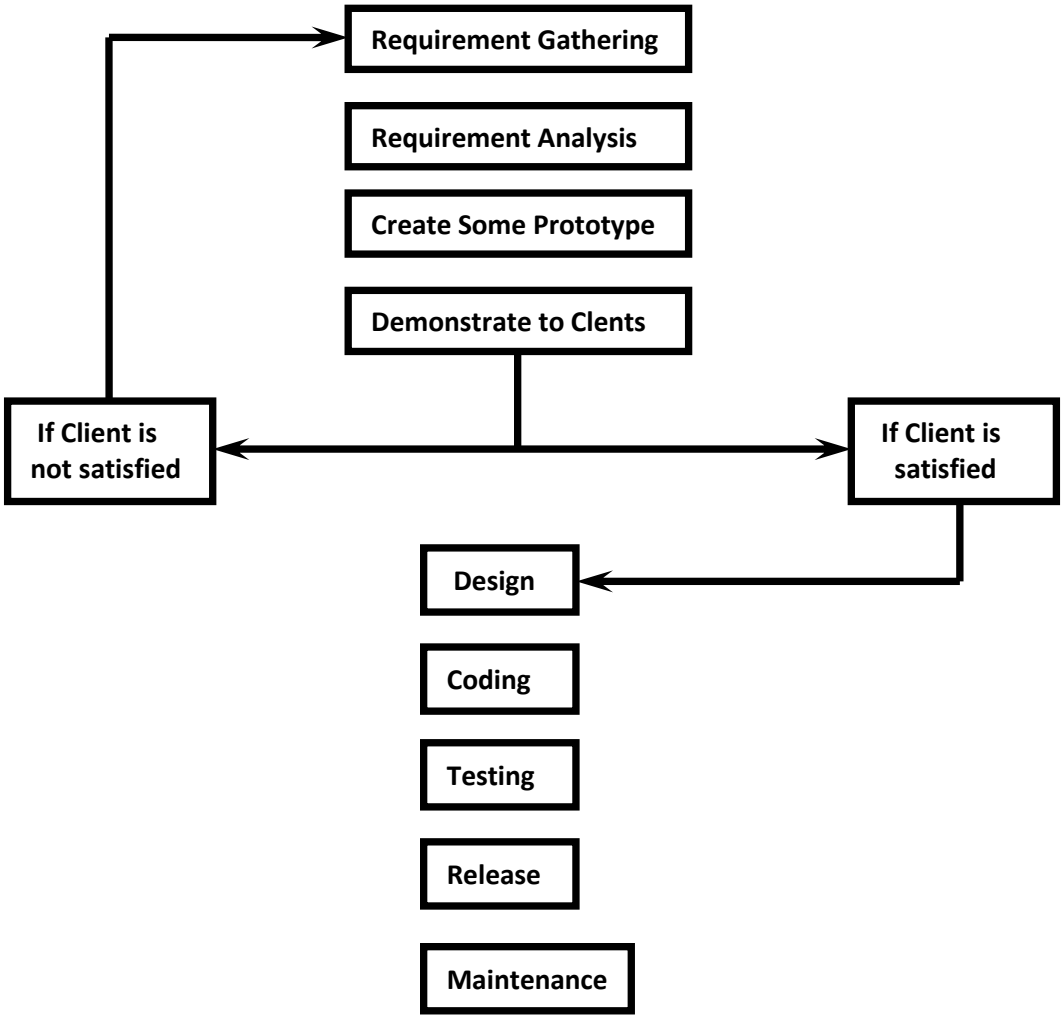
1) It is a simple and easy model.
2) Project monitoring and maintenance is easy.
3) Phases won't be overlapped hence there is no ambiguity.
4) All the phases will be executed one-by-one which gives high visibility to the project manager & client about the progress of the project.
5) Best suitable if the requirements are clear & fixed.
6) Best suitable for small projects.

Drawback:

1) It is very rigid model because it won't accept the new requirements in the middle of the project.
2) Client satisfaction is very low because most of the time client will add new requirements in the middle of the project which won't be supported.
3) Total project development time is more because testing should be done after completing development only.
4) The cost of the bug fixing is very high because we can't identify bugs in the early stages of life cycle.
5) It is not suitable for large projects.

2) Prototype Model:

- If the requirements are clear & stable then we should go for waterfall model.
- Sometimes client is not clear with the requirements and he has generic requirements then we should go for this model.
- We can listen the client requirements and based on that requirements, we can create some prototypes & we can demonstrate to the client.
- If the client is not satisfied then we can Re-create prototypes with new changed requirements & demonstrate to the client once again.
- This process will be continued again & again until client satisfied with our prototype. Once the client is satisfied with some prototype then development will be started.



Advantage:

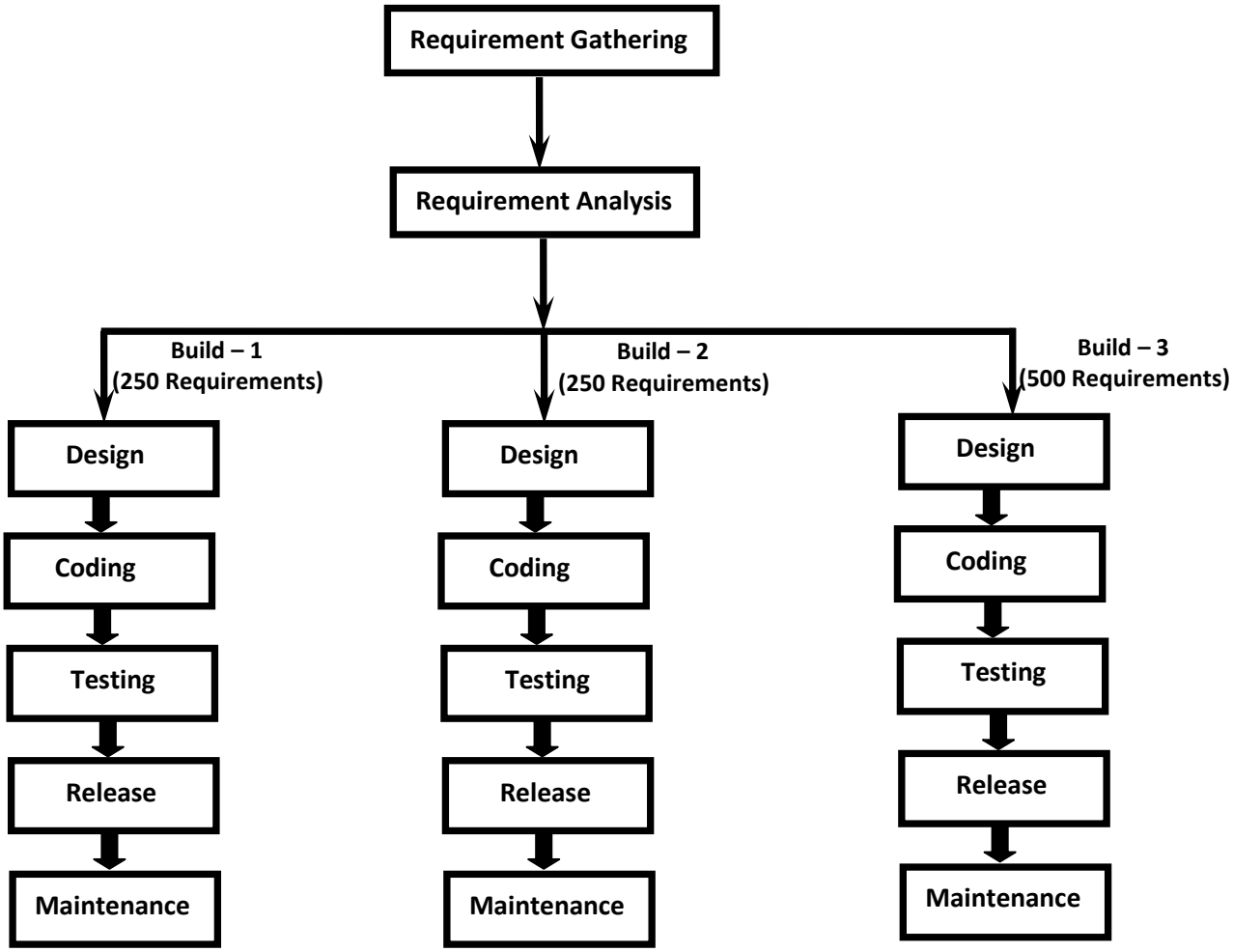
Whenever the customer is not clear idea with requirements this is best model for gathering the clear requirement.

Drawbacks:

1) It is the time consuming model because we have to create prototypes until client satisfaction.
2) It is costly to implement because of creating multiple prototypes. Prototype has to be built on companies cost.
3) Not suitable for large scale projects.

3) Incremental / Iterative Model:

- When customer requirements are clear but huge then this model is best suitable.
- In this iterative model, iterative process starts with a simple implementation of a small set of the software requirements & iteratively enhances to the versions until all the requirements completion.



- In the given diagram, software is developed as 3-Increments / 3-Iteratives.
- In the first increment we implement 250-Requirements; in the second increment we implement next 250-Requirements. In the last third increment we implement
- 500-Requirements.

- The basic idea in this model is to develop a system through repeated cycles (Iterations) and in small partitions (Increments) at a time.

Advantages:

1) Some working functionality can be developed quickly &early stages of life cycle.
2) Results are obtained early & periodically.
3) Parallel development can be planned.
4) Less cost to the change requirement.
5) Best suitable for very large projects.

Dis-Advantages:

1) Not suitable for small projects.
2) More management attention is required.
3) Overall cost of the project development may be increased.
4) Risky factor is more.

4) Spiral Model:

- If customer requirements are changing regularly then we should go for spiral model to develop software version by version.
- Every new version is: New version = Old version + Extra new features.
  Ex: windows 7==>windows 8==>windows10
- SeleniumRC==>selenium(2) webdriver==>selenium-3

- Spiral model is a combination of waterfall, incremental & prototype model. But here more importance is given to Risk Analysis.
- The spiral model has 4-phases.
  1) Planning
  2) Risk Analysis
  3) Engineering
  4) Evaluation

1) Planning:

- Requirements are gathering & analyzed during this phase.
- Feasibility study also will be performed in this phase only.

2) Risk Analysis:

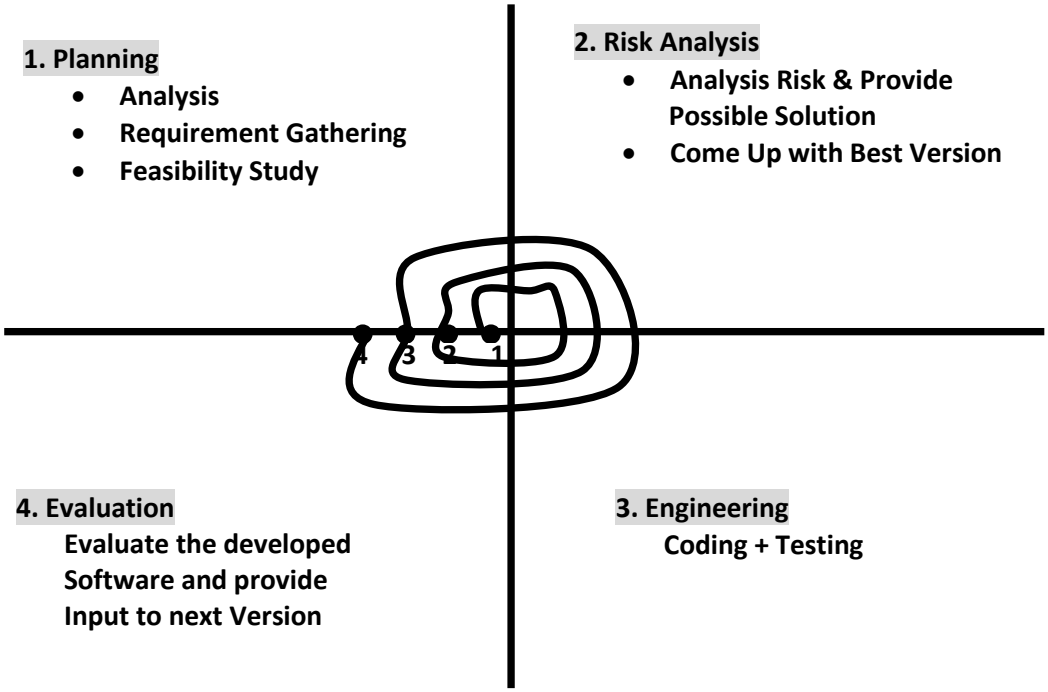
- Here the risk will be identified & analyzed.
- If any risk identified then all the possible solutions will be analyzed & will come with best solutions (Prototype).

3) Engineering:

Actual development & testing of the software takes place in this phase.

4) Evaluation:

This phase allows the customer to evaluate the implemented software, which will helpful to cintinue to the next version (Next Spiral).



Each Spiral = One Version
New Spiral = Old Version + New Features

Advantages:

1) Best choice of the requirements is keeping on changing.
2) Best choice for product based companies.
3) More and more new features will be added in systematic way.
4) Software will be delivered early.

Dis-Advantages:

1) Very costly to use.
2) Risk analysis required highly specific expertise.
3) Not suitable for small projects.
4) Spiral may go infinity.

5) RAD Model:

- RAD means Rapid Application Development.
- If client requirements are similar to old existing projects then we can follow this model.
- In this model we can develop new software with small changes in the code to meet the client requirements like logo change, address change etc....
- In this model development is very rapid, but we have to give special importance to testing.

Advantages:

1) Not required to spend much time on requirements gathering & analysis.
2) Application development will be very fast.
3) Development cost is low.
4) It increases Re-Usability of existing code / components.

Dis-Advantages:

1) We can use this model for brand new products.
2) Implementing new features / Enhancement is very costly.
3) Not suitable for large projects.

6) Big-Bang Model:

- Mainly it is used by students & new commers.
- It won't follow any specific approach to develop the project.

- We can use this approach for very small projects like academic projects & practice projects.
- We are not required to spend more time on requirement gathering, analysis & design.
- All available resources should be involved in coding.

Advantages:

1) It is a very simple model.
2) No time for planning.
3) Very few resources are required.(2 or 3).
4) It is very good learning for new comers.

Dis-advantages:

1) Very rarely used model.
2) Not suitable for real time projects.

Summary of old SDLC Models:

1) Waterfall Model: If requirements are clear & fixed.
2) Prototype Model: If client is not clear with requirements.
3) Incremental / Iterative: If requirements are clear but huge.
4) Spiral Model: If the requirements are enhancing time to time.
5) RAD Model: If requirements are similar to already existing software's.
6) Big-Bang Model: To develop very small projects like academic projects & practice projects.

Limitations of Old SDLC Models:

1) Testing should be started after coding only. It increases over all project development time. If any defects are identified then fixing those defects is very costly because we identified at last.
2) Most of the times programmers are responsible for testing. Hence there is no guarantee for the quality.

To overcome these limitations we should go for advanced SDLC models. The main advantages of advanced SDLC models are:

1) Testing will be performed in parallel to development, so that over all project development time will be reduced & bug fixing is also not costly.
2) Separate testing team is responsible for testing and hence we can give the guarantee for the quality.

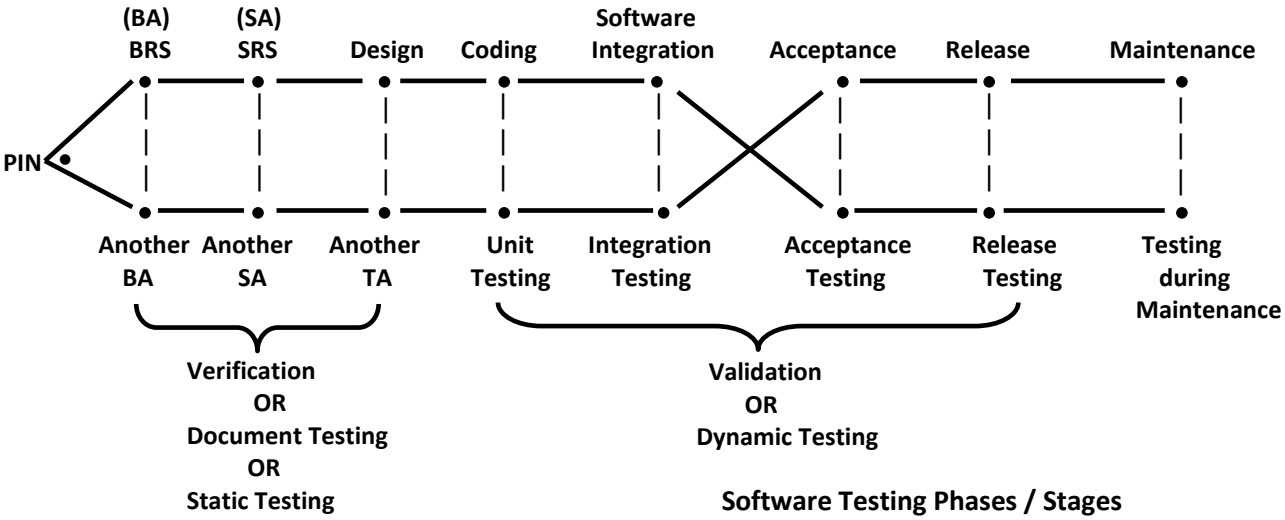
The following are the various advanced SDLC models.

- Fish Model
- V-Model
- Agile Model

7) Fish Model:

In this model one team is doing activities of SDLC & another team is responsible to perform testing parallel.

Ex: If one BA generates BRS another BA is responsible to perform review on that BRS.



Software Testing Phases / Stages

Advantages:

We can give guarantee for quality; hence this model is best suitable for projects where quality is more critical like satellites, healthcare equipment's.

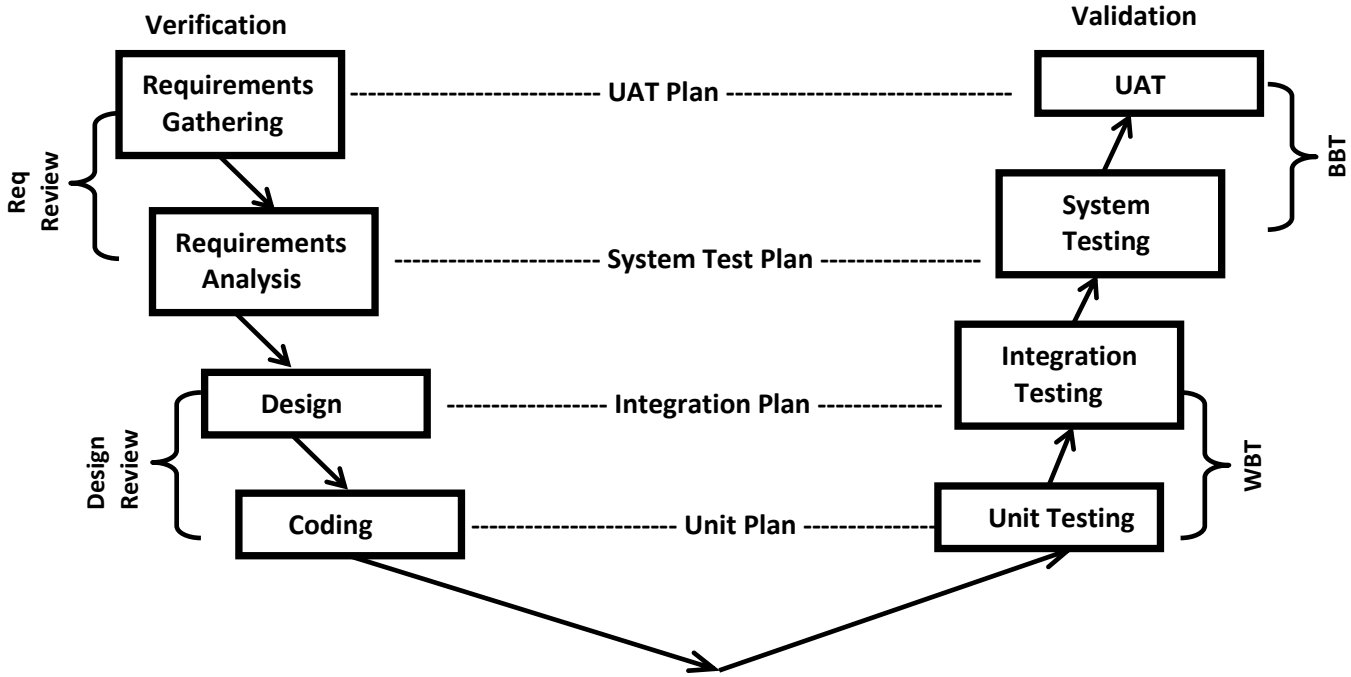
Dis-Advantages:

1) **It is more time consuming.**
2) **It is more costly to implement.**

DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

8) V-Model:

- V-Stands for Verification & Validation.
- In waterfall model, testing will starts after completing coding. But in V-Model testing related activities like preparation of test plan will be started from begining of SDLC. Hence we are reduce project development time.



- The important factor in V-Model is testing related activities will be started from the beginning only.
- After completing requirements gathering testing team will prepare UAT plan.
- After completion of requirements analysis testing team will prepare system test plan.
- After completion of design phase testing team & developers will prepare integration test plan.
- After completion of coding unit test plan will be prepared.
- Once the application ready all the test plans are ready in advance & hence we can perform testing very easily with in less time.

Advantages:

1) It is simple & easy to implement.
2) Testing activities will be performed parallel to development, hence we can reduce overall project development time.

Limitations:

1) It is very rigid model and won't accept requirement changes frequently.
2) Bug fixing is costly similar to waterfall model.
3) No working s/w released until last stage, bcoz its not incremental model.

9) Agile Model:

- This is most frequently used and hot cake model for software development.
- Agile model is divided into several sub models.
  - Rational Unify Model (RUM)
  - Adaptive Software Development (ASD)
  - Feature Driven Development (FDD)
  - Crystal Clear
  - Dynamic Software Development Method (DSDM)
  - Extreme Programming (XP)
  - Scrum Model

- After releasing agile manifest in 2001, all these models are considered as agile methodologies.
- Along all available models of agile, scrum model is the most popular & frequently used model.

Scrum Model:

- Scrum is derived from Rugby Game.
- It is a light weight process.
- It is an Incremental / Iterative model & it accepts changes very easily.
- It is people based model but not plan based model.
- Team collaboration & contentious feedback are strength of this model.

Waterfall model VS Scrum model:

- In waterfall model, before starting next phase, the previous phase should be completed. →RG→RA→Design→Coding→Testing→Release→Maintenance
- It is very rigid model & won't accept requirement changes in the middle.

Scrum:

- Scrum model is not linear sequential model.
- It is an iterative model, total s/w will be developed increment by increment & each increment is called a Sprint.
- Sprint is deliverable / shippable product in scrum model.

Points to remember in scrum model:

- Scrum is an agile model that allows us focus on delivering highest quality software in shortest time.
- In this model, s/w development follows increment by increment.
- Each increment will take one-three weeks duration.
- 7-9 members are responsible in every sprint.
- The art of doing the twice work in half time is nothing but scrum model====>Jeff Sutherland

- 3 Key Roles in scrum model:
  - Product Owner
  - Scrum Master

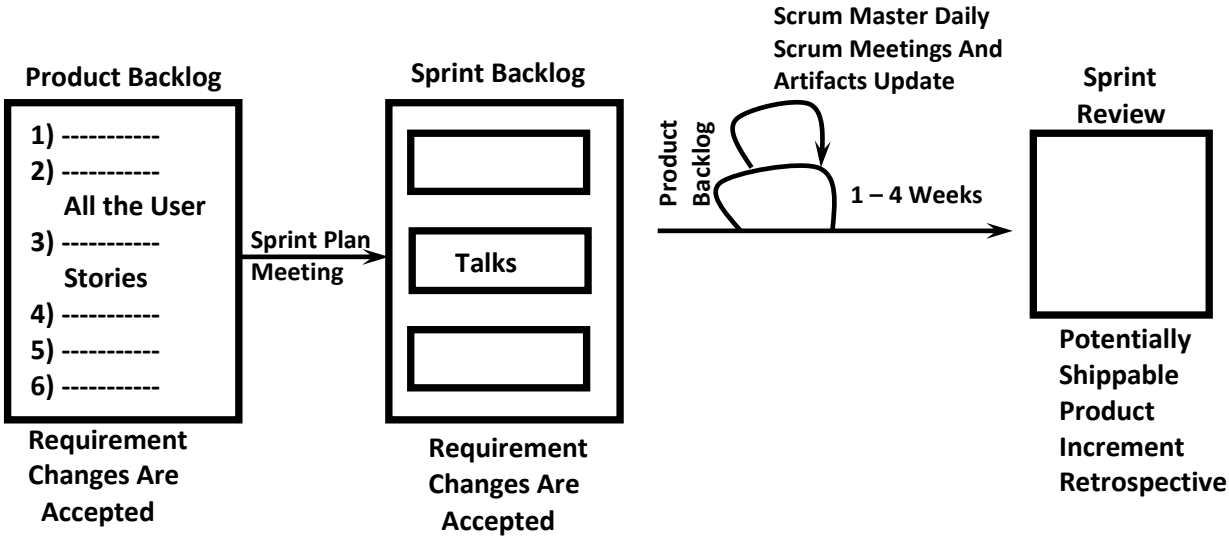
- Scrum Team

⚽ 4 important artifacts (Documents) in scrum model:
- Product Backlog
- Sprint Backlog
- Sprint Burndown Chart
- Release Burndown Chart

⚽ 4 important ceremonies (Meetings) in scrum model:
- Sprint Plan Meeting
- Daily scrum / standup meeting
- Sprint Review Meeting
- Sprint Retrospective Meeting

Characteristics of Scrum model:

⚽ Self-organizing team with 7-9 members.
⚽ In this team both developers & testers will be there.
⚽ In scrum model customer requirements are called as user stories.
⚽ All the user stories will be stored in special document which is nothing but "product backlog".
⚽ In every sprint development team will responsible to implement one or more user stories depends on priority & size of the user stories.
⚽ These selected user stories will be saved in a special document which is nothing but "Sprint Backlog".



Note:

⚽ Product backlog contains all the user stories whereas sprint backlog contains only selected user stories.
⚽ Product backlog can be converted into multiple sprint backlogs.

What is a Sprint?

- In scrum model each increment or iteration is called as a sprint.
- In every sprint "potentially shippable software" component can be developed & tested by the team of 7-9 members in the time span of 1-3 weeks.
- In every sprint a part of client required software will be released.
- All software components which are released in multiple sprints togther fullfill client requirements.

Scrum Architecture:

Roles In Scrum Model:

Scrum model contains the following 3-roles.

- Product Owner
- Scrum Master
- Scrum Team

1) Product Owner:

- Selected by company higher level management like CEO.
- While selecting PO, company may communicate with the clients & Stake Holders (Partners).
- PO is responsible to gather user stories from the customer.
- All collected user stories will be saved in a special document product backlog.
- PO is responsible to prioritize user stories based on client business needs / requirements.
- PO can attend daily scrum meetings along with scrum master & scrum teem.
- PO acts as coordinator between client, scrum master & scrum team.
- PO can guide CCB (Change Control Board) while doing changes in released sprint software.

2) Scrum Master:

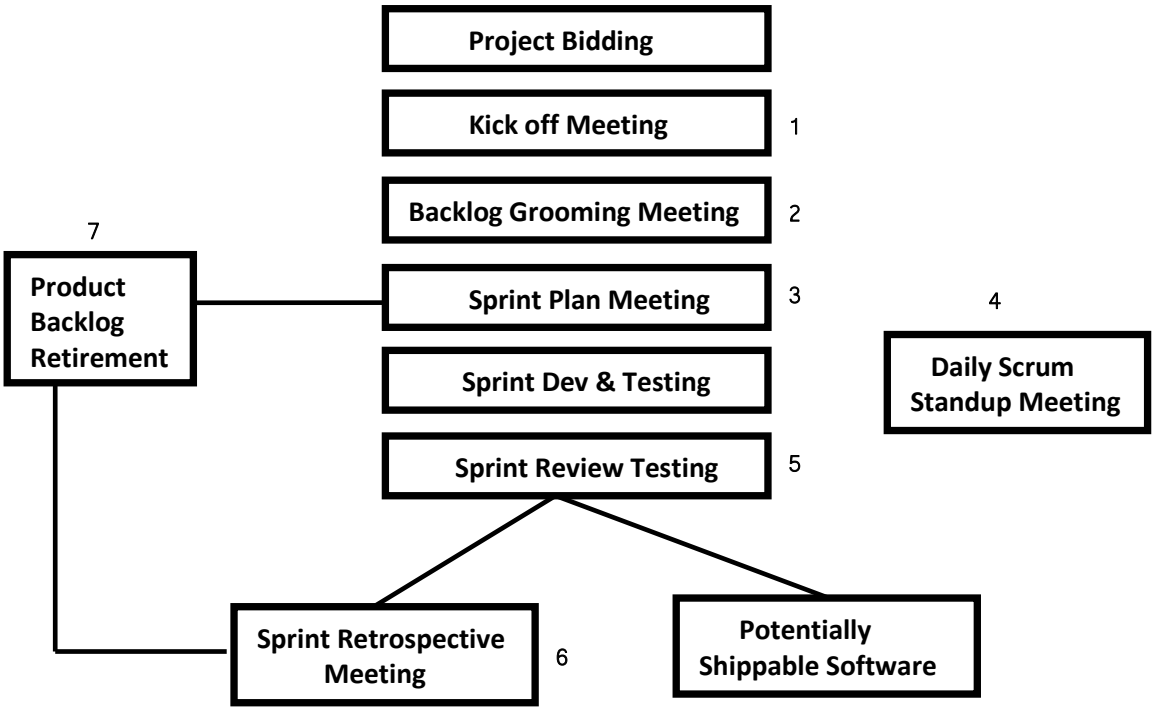
- Facilitator to the scrum team.
- He is the responsible for management of project.
- He is the responsible to implement best scrum practices.
- Responsible to ensure the team is fully functional & productive.
- Responsible for co-ordination between team members.
- Ensuring good relation between product owner and scrum team members.
- Protecting team from outside interruptions.

3) Scrum Team:

- Team size is 7-9 members.
- Cross functional because in this team QA tester, programmers, UI designers etc are the members.
- Programmers take care of design, coding, unit testing & integration testing.
- QA Testers is responsible to study user stories, writting test scenarios, Implementing test cases, defect identification, defects reporting to developers and confirm bug closing.

Important Ceremonies in Scrum Model:

In scrum model, meetings are called as ceremonies.

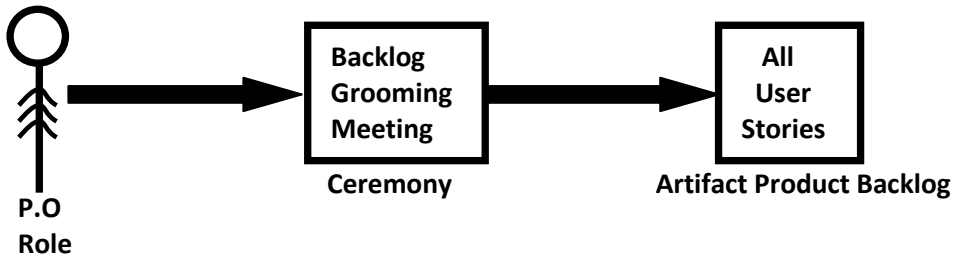


1) Kick-Off Meeting:

- Meeting before project begging.
- Selecting the Product Owner (PO) by management with the interaction of client and stake holders will be happened.

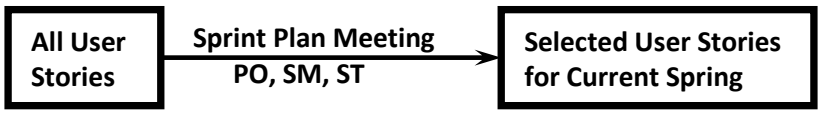
2) Backlog Grooming Meeting:

- PO can collect all the requirements (user stories) & saved into a special document is nothing but product backlog.
- Here PO can collect these requirements from the client & stake holders.
- While preparing user stories PO can take help of subject domain experts.
- If any requirement is bigger than PO will divide into smaller user stories based on client business needs for easy development & testing in right time.
- PO can prioritize user stories based on client requirements only.
- In this meeting PO, client, stakeholders, subject domain experts will be involved.



3) Sprint Planning Meeting:

- PO can sit with scrum master & scrum team to select some user stories from the product backlog based on priority and prepared sprint backlog.



- PO can explain these user stories in detailed to scrum master & scrum team.

4) Daily standup meeting / Scrum meeting:

- PO, SM & Scrum team can meet daily during sprint development & testing.
- Client & Stakeholders may involve if required.
- Duration is 15-minutes.
- In this meeting following things will be discussed.
  - →What activities completed yesterday?
  - →What activities required completing today?
  - →What obstacles are there?
  - ---
  - ---
- This meeting helps to run the sprint development & testing very smoothly.

5) Sprint review meeting:

- PO can sit with customers, stakeholders, SM & scrum team to get the feedback before release.
- Allow the changes in the current sprint or upcoming sprint.
- 2-3 hrs. Meeting.

6) Sprint Retrospective meeting:

- Scrum master sit with scrum team members to analyze the things in the current sprint & identify best practices for the next sprint.
- No involvement of PO, customer, stakeholder & it is purely scrum team meeting.
- 2-3 hrs. Meeting.
- Mandatory meeting for at least few sprints (first 5-6).

7) Backlog refinement meeting:

- PO can sit with the customers, stakeholders to allow changes for the next sprint.
- Re-prioritize product backlog w.r.t customer updated requirements.
- Ready for the next sprint planning meeting.

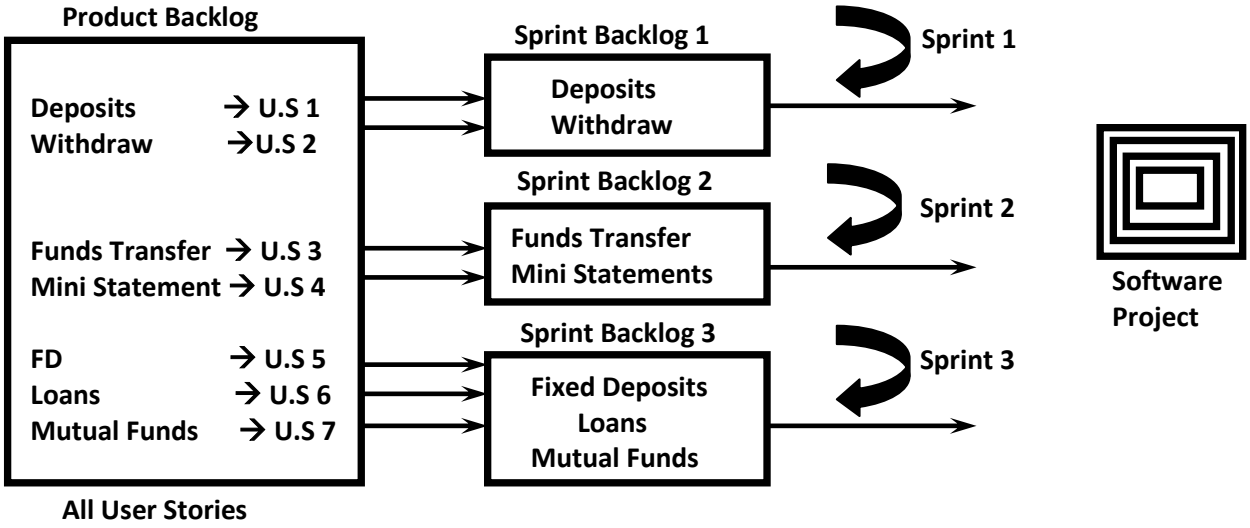
Artifacts in Scrum Model:

The documents which are prepared in scrum model are called as artifacts.

- **Product backlog**
- **Sprint backlog**
- **Sprint burn down chart**
- **Release burn down chart**

1) Product Backlog:

Product backlog is a document which consists of all the user stories, which are collected from the client & stack holders in product grooming meeting.



2) Sprint Backlog:

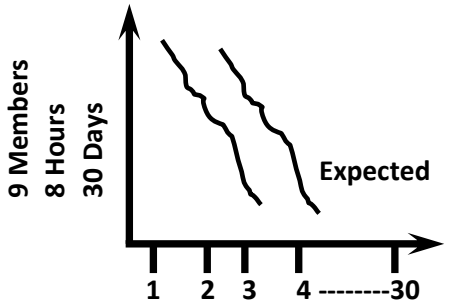
- **Agile scrum model follows incremental model. Hence the total software is developed piece by piece in every increment. Each increment cycle is called as a sprint.**
- **From the product backlog some of the user stories will be selected based on business priorities to implement current sprint. This activity will be performed in sprint planning meeting. These selected user stories will be saved in a special document which is nothing but sprint backlog.**

Note:

- **PB contains all the user stories whereas SB contains only selected user stories for the current sprint.**
- **Product Backlog can be converted into several sprint backlogs.**

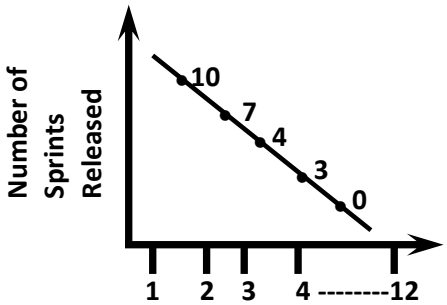
3) Sprint Burn down Chart:

- **It is a graph.**
- **Scrum master is the responsible to prepare this graph in daily scrum meeting.**
- **Scrum master can motivate the scrum team members to do work fastly by showing burn down chart.**

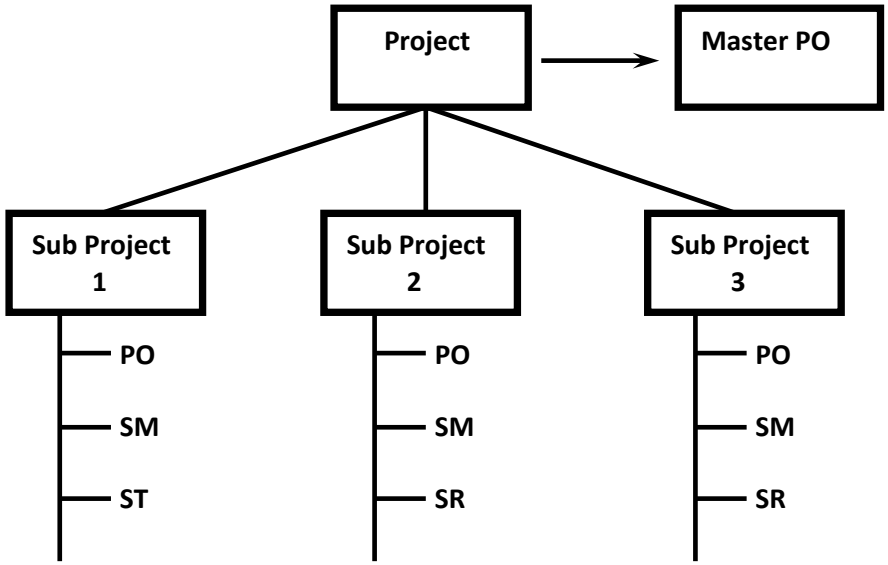


4) Release Burn down Chart:

- Release burn down chart is a graph. It can specify no.of sprints released and no.of sprints yet to release.
- PO is responsible to prepare this graph.
- During backlog refinement meeting PO can explain this chart to the client & stack holders.



Multiple Scrum Teams:



**For every sub project multiple sprints are there.

- If the project is large, then master PO devide the project into sub projects.
- For every sub project separate PO, SM, ST are available.

- Master PO will conduct kickoff meeting to select PO for every sub project.
- Subproject PO is responsible for all the activities related to the sub project.

Note:

- Master PO is the decision maker for total project. Product owners of sub projects will work under guidelines of master PO.
- Scrum masters are the facilitator for corresponding sub projects scrum teams.

Advantages:

1) There is a max. Chance for quality.
2) It ensures effective use of time and money.
3) Requirement changes will be accepted so that maximum chance for client satisfaction.
4) Project status tracking is very easy.
5) There is a possibility for the client involvement in every stage.
6) Team gets complete visibility through scrum meetings.

Dis-advantages:

1) The chance of a project failure is very high, if individuals are not committed or cooperative.
2) Adapting scrum model for large team is very big challenge.
3) Must require experienced & efficient team members.
4) If any team members leave in middle of the project, it can have huge negative impact on the project.

Testing Methodologies

To validate application source code and functionality with best possible combinations. We use following testing methodologies.

1) **White Box Testing**
2) **Black Box Testing**
3) **Grey Box Testing**

1) White Box Testing:

- **It is also called as glass box/clear box/open box/structural testing.**
- **Validating the application source code using programming knowledge is called WBT.**
- **It is performed by developers or white box testers.**

2) Black Box Testing:

- **It is also called as closed box testing. Without any programming knowledge validating application based on requirements {BRS} is called BBT.**
- **In general separate testing team will perform BBT. Following are the techniques to derive test cases.**

  →BVA
  →ECP
  →Error Guessing

3) Grey Box Testing:

It is a combination of WBT & BBT. To perform GBT we should have programming knowledge as well as complete requirements knowledge. It can be performed by TE/Developer.

DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

Testing Levels in SDLC

1) Review on Requirements
2) Review on Design
3) Unit Testing
4) Integration Testing
5) System Testing
6) User Acceptance Testing (UAT)

Review on Requirements:

When requirement documents are prepared they then perform review on those requirement documents to verify following factors.

- **All the business needs are covered or not.**
- **Requirements are correct or not.**
- **Requirements are understandable or not.**
- **Requirements are reliable or not in terms of technology to develop.**

Review on Design:

After preparation of design documents review will be conducted to verify following factors.

- **All the requirements are covered or not.**
- **Design logic is correct or not.**
- **Design logic is understandable or not.**
- **Design is possible to implement or not.**

Note: They use verification techniques on requirement review and design review like peer review, walk through and inspection.

1) Unit Testing:

- **It is also called as component testing. After coding developers will perform unit testing using WBT techniques. It is an initial validation technique.**
- **Validating individual components within the system is called Unit Testing.**
- **Following are the factors they validate in unit testing.**

- **Statements Coverage:** Verify all the statements are executing atleast once or not.
- **Loops Coverage:** Verify loop statements are terminating as per expectation or not.
- **Conditional Statements:** Verify we used right expressions or not in conditional statements.
- **Path or Branch Coverage:** Verify execution flow of the program based on condition True/False.

2) Integration Testing:

- After unit testing those individual components are connected in order to form the system.
- "Validating interfaces or connectivity between the units/components/modules is called integration testing".

- In practical all the modules construction may not possible to complete at same time in that case we use integration testing approaches.
  - Big-bang Approach
  - Incremental Approach

Big-Bang Approach:

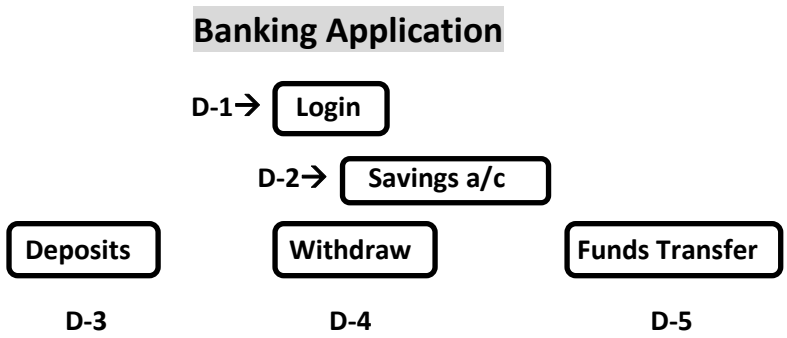
This approach says when some of the modules are under construction then we need to wait until those modules also develop to perform integration testing.

Disadvantages:

- Time consuming.
- Resource will be idle.
- In time delivery is not possible.

Incremental Approach:

This approach says when some of the modules are under construction we can start integration testing with the help of stub & driver.

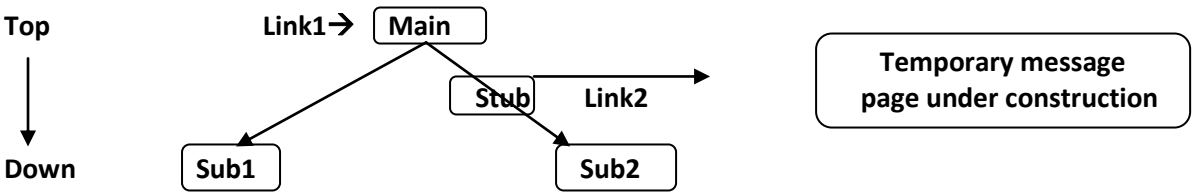


Based on availability of modules we use following incremental integration testing approaches.

- Top-Down approach
- Bottom-Up approach
- Hybrid/sand witch approach

Top-Down Approach:

In this approach we perform integration testing from main module to sub module, when some of the sub modules under construction.

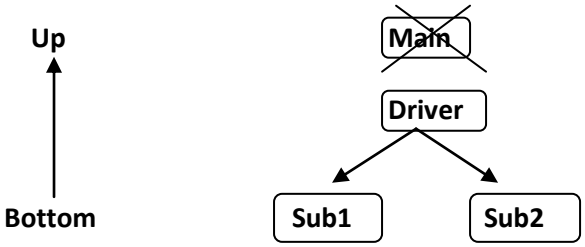


Stub:

It is a temporary program developed by developers. Stub is used when sub module is under construction. Stub will write control back to main module with some temporary result.
Stub is "Called program".

Bottom-up Approach:

When main module is under construction then we perform integration testing from sub modules with the help of driver.



Driver:

It is a temporary program developed by programmers. Driver is used when main module under construction. Driver will provide connection to the sub modules.
Driver is called "Calling program".

Hybrid/Sand witch Approach:

When main module and some of sub modules are under construction then we use stub & driver to perform integration testing among the available modules.

Integration Testing Levels:

There are 2 levels of integration testing.

1) Low-Level integration testing
2) High-Level integration testing

1) Low-Level integration testing:

This is also called as inter system testing. Validating interface or connectivity within the system components is called low-level integration testing.

Ex: Verify interface b/w Gmail application modules.

2) High-Level integration testing:

Validating interface or connectivity between different systems or application is called high-level integration testing.

Ex: Verify connectivity from E-Sava application to other applications like Airtel, Electricity bill.

Ex: Verify using HDFC bank debit card in ICICI bank ATM center.

Note:

- Integration testing can be performed by developers and TE's.
- Developers will perform integration testing after unit testing using WBT techniques.
- TE's also perform integration testing during system testing using BBT techniques.

3) System Testing:

- After unit testing and integration testing development team will release initial build to the separate testing team.
- Build means set of integrated modules an executable form of application.
- In general internal release of application from developers to TE's we call it as build.
- After receiving stable build from developers, TE's will perform system testing using BBT techniques.
- "Validating whole system based on client requirements and expectations is called system testing".
- There are 2 types of testing techniques in system testing.
    1) Functionality Testing
    2) Non-Functionality Testing

1) Functionality Testing:

- It is also called as requirements testing. Validating application functional behavior based on user business transactions is called functionality testing.
- Any type of application functionality testing is mandatory and very important. Due to that reason organization maintains separate functionality testing team.
- Functionality testing can be performed manually or using some of the functional testing tools like selenium, QTP, Silk test, rational robot…etc.
- Following are the factors we validate during functionality testing.
    - Object properties coverage
    - Error handling coverage

- Input-domain coverage
- Calculation coverage
- Data base coverage
- Links coverage

Object properties coverage:

Verify applications object property values are changing are not based on operations performed on application.

Ex: Enabled, Focused, Items Count.

Error handling coverage:

Verify how application responds whenever we performed any invalid operations.

Ex: Error messages, Warning messages, Pop-up's.

Input domain coverage:

Verify input objects allowing customer expected data or not.

Ex: Edit box/Text box

Scenario: Verify "Name" edit box should allow alphanumeric 4-16.

Test Data:

Values OR Data which we provide to application in order to validate. TE is responsible to derive test data.

We use following techniques.

- BVA (Boundary Value Analysis)
- ECP (Equivalence Class Partition)

BVA:

This technique use to validate input condition interims of range or size.

This technique says there will be 3-possibilities at each boundary level to validate.

Possible conditions are:

Min	Max
Min-1	Max-1
Min+1	Max+1

In general there will be high possibility of errors in and around the boundary conditions those we can identify with optimal test data using BVA.

Ex: Prepare test data using BVA for a Name edit box which allows from 4 to 16 characters.

BVA(Size)	
Min=4→valid	Max-1=15→valid
Max=16→valid	Min+1=5→valid
Min-1=3→In valid	Max+1=17→In valid

Ex: Prepare test data for mobile number edit box which allows 10 digits only, where as starting digit should be 7/8/9.

BVA (Range)	
Min=7000000000→valid	Max-1=9999999998→valid
Max=9999999999→valid	Min+1=7000000001→valid
Min-1=6999999999→in valid	Max+1=10000000000→in valid

Disadvantage of BVA:

- With this technique we can validate input condition with only boundary values and their adjacent values.
- In this technique it is not possible to validate data type like alphabets, special characters…Etc

ECP:

This technique use to validate input condition with data type. This technique says divide test data equivalence classes interims valid and invalid and then takes some sample data from each class to validate.

Ex: **Prepare test data using ECP to verify withdraw condition of ATM which allows 100-20000, whereas amount should be multiples of 100.**

ECP (Type)	
Valid	**Invalid**
100<=Amount<=20000 **{Amount should be multiples of 100}**	**Amount<100** **Amount>20000** **Amount which is not multiples of 100** **Ex:150,550**

Scenario: Verify "Name" edit box should allow alphanumeric 4-16.

TC01: Verify name edit box with valid data.
Test Data: {combination of valid size from BVA with valid class from ECP}
abc4→(4) **abcd12345678mnop→(16)**
mn56a→(5) **9875mnoqabcdedf→(15)**
TC02: Verify name edit box with invalid data.
Test Data: {combination of invalid size from BVA with valid class from ECP}
as3→(3) **abcd12345678mnopq→(17)**
Test Data: {combination of valid size from BVA with invalid class from ECP}
ab_4 **mn@pq** **All special characters.**

Calculation coverage/output coverage:

Verify application generating expected output values or not based on given input data.

Database coverage/Backend coverage:

- **Verify impact on data base content w.r.t operations performed on application frond end. Like insert, update, delete.**
- **In this test we also verify relation b/w expected front end component to specified data base fields.**

Links Coverage:

For web based application we need to check implemented links are working correctly or not.

2) Non-functionality Testing:

In this test we validate application characteristics based on customer expectations is called non-functionality testing.

Following are the Non-Functionality testing techniques.

→GUI Testing
→Usability Testing
→Performance Testing
→Recovery Testing
→Security Testing
→Compatibility Testing
→Configuration Testing
→Comparative Testing
→Installation Testing
→Sanitation Testing

1) GUI Testing/User Interface Testing:

In this test we verify Look-N-Feel of application. Following are the factors related to UI testing.

- **Availability of components**
- **Alignment of controls**
- **Control visibility**
- **Font size/style**
- **Control spelling correct or not**
- **Back ground color of screen**
- **Clarity of image objects like logos, graphs...etc**

2) Usability Testing:

In this test we verify user friendliness of the application to perform operations (i.e ease of use).

Following are actors related during usability test.

- **Error messages are meaningful or not.**
- **Help documents are understandable or not.**
- **Functional keys are implemented or not (F1-->Foe Help).**
- **Combination keys are implemented or not (ctrl+P-->Print).**

3) Performance Testing:

Following are the factors we validate during this testing.

❖ **Speed:**
How quickly application completes the transaction.

❖ **Scalability:**
Verify application allowing customer expected concurrent users or not (Load).

❖ **Stability:**
How application response under sudden loads.

Following are techniques:

- Load Testing/Scalability: Verify application allows customer expected concurrent users or not.
- Stress Testing: In this test we use beyond the load limit to estimate peak limit of the load on application.
- Soak/Endurance Testing: Verify how much time continuously we can run application. (I.e. Duration)
- Spike Testing: Verify application behavior under sudden loads.
- Data Volume Testing: Verify how much data we can transfer in a specified time.

4) Recovery Testing:

Verify how well application able to recover from abnormal state to normal state.

Ex: Power failure, Network problems.

5) Security Testing:

- In this test we verify privacy for end user operations in a application.
- Following are factors:

- Authentication: verify application allowing valid users and preventing invalid users or not.
- Authorization: verify application providing right services or not based on type of users.

6) Compatibility /Portability Testing :

Verify application supports customer expected OS's & browsers & compilers.

- Forward compatibility testing
- Backward compatibility testing

7) Configuration Testing:

It is also called as H/W compatibility testing. Verify application supports customer expected configuration systems or not.

Ex: RAM, Hard Disk, Processor.

8) Installation Testing:

Verify application able to install as per "Read me" file guidelines or not.

- Setup programs availability.
- During installation application providing any user friendly messages or not.
- All the related files copied or not.
- Repair/Uninstall programs availability.

9) Sanitation Testing:

- It is also called as garbage testing.
- In this test we try to identify any extra features in application which are not specified in client requirements.

10) Comparative/Parallel Testing:

Comparing our product with other existing competitive products in the market in order to identify strength and weakness of our application.

4) User Acceptance Testing:

- After system testing client team perform UAT. The objective of UAT is to confirm application is ready for release or not.
- Whereas UAT is performed based on their business needs.

Alpha Testing:

It is performed at developer side in controlled environment. If any defects identified those will be resolved and should get approval client.

Beta Testing:

After alpha testing application will be deployed at client environment. Beta testing performed at client site in uncontrolled environment.

Some other Testing techniques/Terminologies:

1) Smoke Testing:

It is performed on initial build to check the build is stable or not for further testing. In this test we validate major/core functionalities with valid data.

2) Sanity Testing:

It is also similar to smoke test. It is a subset of regression testing. It is performed on modified build to check build is stable or not.

Note: If build is unstable then we suspend test cases execution and reject build to developers. For rejected build developers will release patch files to testing team.

3) Re-Testing:

- It is performed on modified build to check bugs are resolved or not. Whereas we use fail test cases to perform execution on modified build.
- When we validate same functionality with multiple set of test data is also called Re-Testing.

4) Regression Testing:

- It is also performed on modified build to find any impact or side effects on existing passed functionalities w.r.t modifications.
- Regression testing can be performed partially or fully on application based on modification.

- In following scenarios we perform regression testing.
  - When bugs resolved in modified build.
  - When new features added in existing system based on change request (CR) from client.

5) Ad-hoc Testing:

It is an informal testing activity due to lack of requirements. Test engineer use past experience to validate the application.

6) Exploratory Testing:

Due to lack of domain knowledge while learning the application functionalities we perform testing activities.

7) Jump/Monkey Testing:

Due to lack of time we validate major functionalities in application. In this case we select test cases based on priority. This type of testing also called priority based testing.

8) L10N Testing: (Localization testing)

Verify application support domestic/local languages or not with in the country.

9) I18N Testing: (Internationalization testing)

In this test we verify application supports international languages currency, time, date format etc…

Note: Combination of L10N & I18N is called Globalization testing (G11N testing).

10) Mutation Testing:

It is performed by developers. By changing logic in program developers will perform testing.

11) Defect seeding/be-bugging:

Providing known defects in application to find strength of test engineer's is called as defect seeding/be-bugging.

Software Testing Life Cycle (STLC)

- **In SDLC testing is one of the important phases to deliver reliable application to customer.**
- **In entire SDLC separate testing team will perform 40% of activities in order to validate developed application.**
- **STLC describes systematic approach to validate application based on client requirements and expectations.**
- **It describes step-by-step process performed by separate testing team in order to validate application**

STLC Phases:

1) **Test Initiation**
2) **Test Planning**
3) **Test case design**
4) **Test Execution**
5) **Defect Reporting**
6) **Test Closure**

1) Test Initiation:

BRS/FRS
Organization Standards → **TM** → **Test Initiation** → **Test Strategy doc**

- **→Application Analysis**
- **→Risk Analysis**
  1) **Resources Risk**
  2) **Technology Risk**
  3) **Scheduled Risk**
  4) **Support Risk**
- **→Effort Estimation**

Following are Factors:

Application Analysis:

Using requirement documents like BRS/FRS application functionalities will be analyzed.

Risk Analysis:

Possible risks at test environment will be analyzed.

- **Resources risk: Availability of TE's will be analyzed.**
- **Technology Risk: Problems related to S/W & H/W will analyze.**
- **Scheduled Risk: Which factors may affect scheduled activities will analyzed?**
- **Support Risk: Availability of clarification team will be analyzed.**

Efforts Estimation:

Based on size of project required resources, duration and cost will be estimated.
After analyzing above factors TM will prepared test strategy document.

Test Strategy Document:

- **It is a high level test document, which will prepare early stages of testing.**
- **It describes testing approach, available resources and standards to be followed during testing.**
- **It is an organization specific document; information in test strategy will be uniform throughout the project.**

2) Test Planning:

In this phase TL is responsible to prepare test plan document based on current build which will received from developers.

Test Plan Document:

- **It will prepare based on build.**
- **It describes detailed testing approach, name of TE's, Features to be tested, Test Environment, Entry criteria, exit criteria, schedules, test deliverables, scope.etc…**
- **Based on build which we receive from developers information in test plan may vary.**

- **For successful software testing a clear test plan is essential.**

BRS & FRS Build Info → TL → Test Planning → Finalized Test Plan Doc

Test Planning:
- →Team Information
- →Task Allocation
- →Risk Analysis
- →Training Needs
- →Schedules
- →Prepare Test plan
- →Review on Test Plan

Finalized Test Plan Doc:
- →What to Test
- →Whom to Test
- →How to Test
- →When to Test

Components in Test plan Document:

Title: Name of the project

History of Document: It describes author of test plan, when it is prepared, reviewed by, reviewed on approved by….etc

Introduction:

- Project over View: It describes brief description about AUT.
- Purpose of Test Plan: It describes objective or core intension of test plan.
- Referred doc: Inputs to derive TP.
- Scope:
  - In scope: It describes possible testing activities under current test plan.
  - Out of Scope: It describes testing activities which are not possible under current test plan

Features to be Tested:

It describes availability modules & requirements in current build to validate.

Features not to be tested:

It describes modules which are not available under current TP to validate.

Test approach:

It describes test factors & corresponding testing techniques to validate.

Entry & Exit Criteria:

⚽ Entry Criteria: It describes when to perform dynamic testing/ test execution.

- All test cases should prepare & reviewed.
- Build should deploy in test environment.
- Build should passed BVT (i.e. Build is stable)
- Test environment set-up should be completed.

⚽ **Exit Criteria: It describes when to stop testing for a project.**

In general following factors will be considered in order to stop testing activities for a project.

- **All test cases should be executed at least once.**
- **Majority of TC should be passed like 90-95%.**
- **Defects which are identified that status either "Differed" or "Closed"**
- **When UAT completed and satisfied.**
- **When budget and time constrains overflow.**

Suspension & Resumption Criteria:

Suspension: It describes when we can temporarily stop the test execution.

Ex: **1) Build is failed in BVT.**
2) Delayed in deployment of build.
3) When there is a change request from client.
4) When we identified show stopper/critical/fatal defects.

Resumption: It describes when to continue test execution.

1) When patch file received for rejected build.
2) When input documents are published.
3) When specifications are refined and based on CR.
4) When show stopper defects are resolved.

1) Roles & Responsibilities:
It describes name of TE's allocated task.

2) Test Deliverables:
It describes deliverables documents for a client at end of test. Test cases review report, defect report, test execution report, test summary report.

3) Test Environment:
It describes required s/w & h/w configuration in the system to perform test execution.

4) Schedule:
It describes time lines to perform each testing activity.

5) Training Needs:

- **It describes required training sessions for TE's.**
- **After preparation of test plan review will be conducted along with Sr. Test Engineer and then finalized test plan given to TE's to perform their activity.**

Example: below is the sample Test Plan document for Spice Jet.

REVISION HISTORY

Reference	**4.7_UseCases_TestCases_Features_Interim Build 9-24**	**Version No.**	**4.7**
Release Date		**Total No. of pages**	
	Name	**Designation**	**Date**
Prepared by	**Mahesh.D**	**QA Lead**	**19-Oct-2010**
Reviewed By			
Approved by			

Table of Contents

Introduction

The Test Plan has been created to communicate the test APPROACH to client and team members. It includes the objectives, scope, schedule, risks and approach. This document will clearly identify what the test deliverables will be and what is deemed in and out of scope.

1. Objective

The primary objective of this document is to establish a Test Plan for the activities that will verify SPICEJET as a high quality product that meets the needs of the SPICEJET business community. These activities will focus upon identifying the following:

- **⇨ Items to be tested**
- **⇨ Testing approach / Strategy adopted**
- **⇨ Resource Requirements**
- **⇨ Roles and Responsibilities**
- **⇨ Milestones**
- **⇨ Risks and contingencies**
- **⇨ Test deliverables**

1.1. Scope

This document is a very high level vision of how SPICEJET applications will be tested and will not aim at providing any details about each testing engaged at various levels of testing.

Scope of Testing

1. Test cases identification and documentation for the new features/use cases and NFR
2. Creation of new test cases and updating existing test cases for all modules
3. Functional Testing for the new functionalities(New Features)
4. Non Functional Requirement(NFR) testing(Performance testing)
5. Regression testing for the SPICEJET functionalities
6. Complete SPICEJET Application testing
7. Recording of bugs and verification of resolved bugs for each build

2. Reference Documents

The table below identifies the documents and their availability, used for developing the test plan

Document (Version / Date)	Created / Available	Received / Reviewed	Author / Resource	Remarks
Homepage SRS doc			BA	
Book a flight SRS doc			BA	

3.Test Items

3.1 . Features to be tested

All change requests / enhancements / bug fixes on above listed applications will be tested on need- basis:

Test cases will be prepared based on the following documents:

- Understanding Use Case documents
- Software design documents
- Data Validation documents

Business Requirements	Ref. No.	Feature	Functional Specification

3.2 . Features not to be tested

Security Testing and mobile testings are not part of the engagement

4. Test Strategy

The SPICEJET modules and sub modules testing will be performed with below testing types

4.1. Testing Types

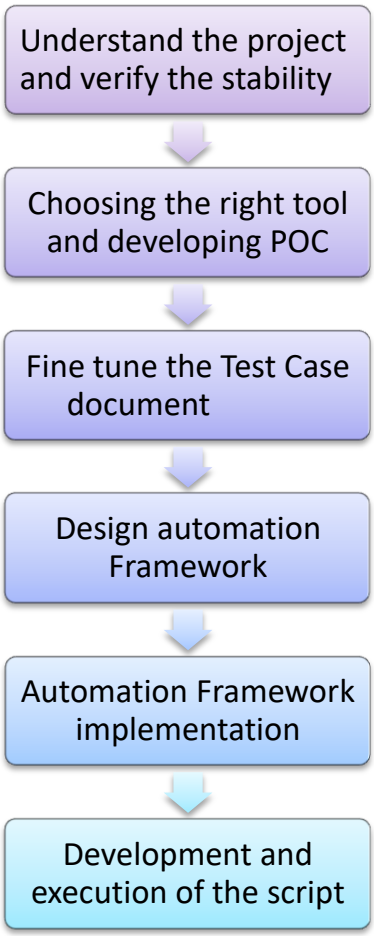
4.1.1 Functional Testing

The primary functional areas in various SPICEJET modules will be thoroughly tested according to the functional specification document or understanding document or software requirement specification document. During the testing phase, complete functionality will be tested at least two to three times based on the complication involved in the application.

Test Objective	**Ensure that each function specified in the functional document and SRS works correctly while passing/retrieving parameters without data corruption.**
Technique	**Test the each function by providing the valid and invalid input values and inspect the input data has been operated by the function, also inspect the output data as intended, and ensure that all implemented functions are processing the data properly or review the output data to ensure that the correct data was retrieved.**
Completion Criteria	**Both interfaces (back-end application and end user site) should process data correctly without any mismatch. Error handling cases should also be checked.**
Special Considerations	**Testing may require the huge test inputs to test the functionality of sending test SMS or test e-mails.**

5. Automation Testing

This section describes the strategy and the activities performed as part of the planned strategy for implementing the automation of test scenarios. The following steps should be adopted as a part of strategy of implementing the test automation



Understanding the product and verify the stability of the product:

First analyze the product and assess the feasibility of automating the test scenarios identified for application. It i also ascertain that the future releases of the product would not undergo major changes.

Design Automation Framework:

Based on the product , evaluation of various test automation tools with an objective to suggest a suitable automation tool in all respects should under taken.

Developing proof of concept:

A proof of concept developed to show the capabilities and computability of the tool with the application. In the POC we have taken AOA and BBM scenarios by covering the various verification

points and actions.

Design the Automation Frame work:

To design the Frame Work we use the Data Driven Approach as the applications are stable and no GUI changes are identified and which would also cover AUT(Application Under Test).This also identifies the various reusable action and common steps.

Automation Frame work implementation:

In this all the identified reusable function are implemented followed by coding convention.

Development and Execution of the Script:

The test scripts are developed based on design. Finally, a clear log report is produced covering all the verification points. Adherence to the define coding conventions and elimination of common coding.

Automation – Phases	ACTIVITIES	DELIVERABLES
Automation Assessment	• Walkthrough of requirements. • Walkthrough of systems. • Analyze requirements from automation perspective. • Focused discussions - Preparing test cases - Test data requirement - Test data conditioning - Technical aspects - Priorities - Workarounds	• Plan for Framework phases.
Framework Design	• Identification of test scenarios for automation. • Identification of Reusable components. • Preparing Design document.	• Detailed design document • Test Scenarios with coverage's to be automated.

Framework Development	• Prepare framework code • Code reviews • Testing framework	• Baseline Design document • Framework code
Test scenarios Automation	• Develop automated script based on the framework developed • Test data conditioning input file preparation • Testing of the automated script. • Peer review of the scripts for adherence to standards. • Prepare User manual guide. • Package the automated scripts for release	• Automated Test scripts • Automate Manual User Guide.

Testing Tool: Selenium

6. Test Environment

The test environment preparation step will ensure that hardware, software, and tools required for testing will be available to the testing team when they are needed. This would involve co-ordination with IT infrastructure team and other providers in regard to Equipment, Operating systems, networks, etc.

Machine type : Windows server Enterprise

OS : Windows

Processor : Intel® xeon® CPU

Memory : 4 GB / 2.13 GHZ

Hard disk : 150 GB

Database : Microsoft SQL Server 2008 Standard Edition
Web server : IIS 7.0

Client : Microsoft Internet Explorer, Firefox, GoogleChrome

7. Test Pass / Fail Criteria

Defects will be classified as follows according to severity of the impact on the system:

*Severity Level	Description
Sev 1 - Blocker	Calico Software is not operational in production and a work-around is not available. Critical Errors include the following: • Calico Software may cause corruption or destruction of data • The System fails catastrophically (50% or greater reduction of service) • Two or more reboots of the System per day
Sev 2 - Major	A major function in the Calico Software is not operational and no acceptable work-around is available, but Customer is able to do some production work. High Errors include the following: • System is usable but incomplete (one or more documented commands/functions are inoperable/missing) • System fails catastrophically (10-50% reduction of service) • One reboot per day of the System
Sev 3 - Minor	There is a loss of a function or resource in Calico Software that does not seriously affect the Customer's operation or schedules. Medium Errors include the following: • Issues associated with the installation of Calico Software • Any "Critical" or "High" Error that has been temporarily solved with a work-around
Sev 4 - Suggestion	All other issues with Calico Software. Low Errors include the following: • Errors in Documentation • Calico Software does not operate strictly according to specifications

8. Defect Analysis and closure

Following are the defect tracking activities till its closure. It includes:

- Logging of defects
- Analysis of defects
- Fixing of defects
- Re-testing of fixes
- Regression testing to ensure that fixes have not impacted the original functionality.
- Defect tracking till closure

9. The Test Deliverables:

Phase No	Modules	DeadLines (Date of Delivery)
1	1. BookaFlight 2. ManageMy Booking 3. PNR Status	30th Sept 2014
2	4. Flight Schedules	31st Dec 2014
3	5. Corporate Benefit	30th Mar 2015

10. Risks and Contingencies

Risks	Contingencies
Person shortfall	Maintain buffer resources
Continuous Requirement Changes	Analyze the requirements
Lack of peer reviews	Monitor Peer reviews

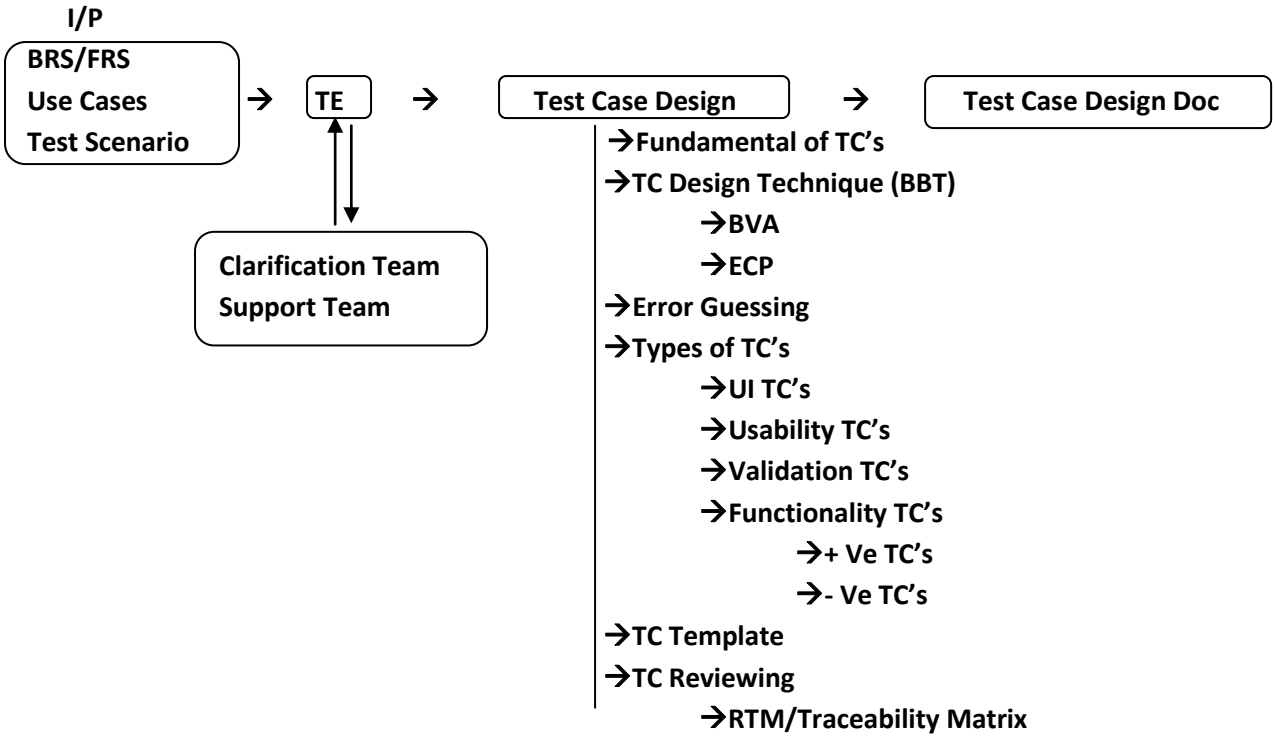
11 HARD WARE AND SOFTWARE REQUIREMENTS

12 RESOURCE PLAN

Test Engg – 10 Automation TE-5 DB Testers-2

3) Test Case Design:

After training session Te's are responsible to derive test cases for allocated modules.



Test Case:

It consist set of steps or sequence of steps with the user actions and subsequent response from system to validate specific functionality.

*Pre-Condition:

It describes things to ensure in application to perform test case execution.

Ex:

TC01: Verify delete mail functionality in inbox.

Pre-Condition: In inbox some mails should exist.

Fundamentals of TC's:

Following are basic factors to consider while writing TC's.

- **TC's should be simple & easy to understand.**
- **Naming convention should be followed.**
- **No duplicate/redundant TC's.**

Inputs to derive TC's:

- Requirement document documents like BRS & FRS.
- Use case document: It is optional prepared by business users. It describes functional behavior from user perspective interims of actor's action and system response.
- Test Scenarios: It describes a test condition or requirement or functionality which we need to validate.
- In general test scenarios identified by TL/TE's.
- For a scenario there can be one or more number of TC's.

Advantages of Test Scenarios:

- Based on TS we can identify possible number of TC's need to be drafted.
- With the help of TS we can provide complete test coverage.

Note: TS are not mandatory for some of the projects.

Difference among use case, test scenario, TC:

UC: It describes functional behavior from user perspective
{How system need to work}

TS: It describes test condition or requirement or functionality which we need to validate.
{What to Test}

TC: It describes validation procedure of each requirement or functionality.
{How to Test}

Test Case Design Techniques: Also called as BBT techniques.

→BVA
→ECP
→Error Guessing: It is an experienced based technique there is no specific format.
Based on functionality knowledge we can identify possible test cases related to error guessing.

Types of Test Cases:

There are 4 types of Test Cases.

1) **User Interface TC:** These are derived to verify Look-N-Feel of the application.
Ex: Availability of components, visibility, alignments…etc

2) **Usability TC:** These are derived to verify user friendliness of the application to perform operation.
Ex: Error messages, user manuals, Function Keys implementation, Combination keys etc...

3) **Validation TC:** These are derived to validate input components.
   Ex: Edit box, Text box, and List box.

4) **Functional TC:** These are derived to functional behavior of application based on user operations.
   Ex: Click on push buttons/Images.

Note:

- Usability, users interface & validate test cases will be part in functionality test cases.
- Based on validation & functionality there can be 2 types of test cases.

1) Positive Test Cases: TC's with valid inputs.
2) Negative Test Cases: TC's with invalid inputs

Test Case Template:

In general organization maintains their own template to derive test cases.

Test case ID	Test Scenario	Precondition	Test steps	Expected Result

Components in TC Template:

1) Project: Name of the project

2) Module: Name of the module

3) Created By: Name of Test Engineer

4) Created On: On which date TC's are prepared

5) Reviewed By: Who conducted review on TC's (i.e Test Lead?)

6) Reviewed On: On which date review conducted on TC's.

7) Referred Documents: Inputs to derive TC's.

8) Test Case ID: Name of the test case which should be unique for entire project.

9) Test Case Description: It describes purpose or objective of test case.

10) Priority:

- **It defines importance of TC during execution. It is derived based on importance of the functionality w.r.t client business needs.**
- **There can be 3 levels of priorities.**
  - **High**
  - **Medium**
  - **Low**

Note: Due to lack of time first we execute high priority test cases and we perform next level priority test cases execution.

11) Step Name: It describes number of steps in a test case. A test case may consist one or more steps.

12) Test Data: Data or values which we use to validate application during execution.

13) Step Description: It describes user operation to be performed on AUT during TC execution.

14) Expected Result:

It describes expected behavior of the application based on user operation.

****Expected results will be drafted from requirements.**

15) Actual Results: It describes actual behavior of AUT during test case execution.

16) Status: It describes status of steps interims of PASS/FAIL.

17) QC path/Subject: To export test case in to QC we provide folder name under "QC Path" column.

Sample Test Cases

Testcase ID	Test Scenario	Precondition	Teststeps	Expected Result
1	Verify the GUI of the Book a flight page	Open the URL http://spicejet.com	1. Check the spell, font, allignment and color 2. Check the look and feel of the page	Application should maintain the consistency
2	Verify the availability of the fields in Book a Flight page	Open the URL http://spicejet.com	Check the below fields 1.Field Name Field Type 2.Round trip Radio 3.Oneway Radio 4.Leaving From Dropdown 5.Going To Dropdown 6.Date Picker1 Date Picker 7.Date Picker2 Date Picker 8.Adult Dropdown 0.Children Dropdown 10.Infants Dropdown 11.Indian armed forces 12.personnel Checkbox 13.Student Check box 14.Find flights Button	Application should display all the fields in Book a flight page
3	Verify the functionality of one way radio button	Open the URL http://spicejet.com	Click on one way radio button	Application should display one way date picker
4	Verify the functionality of Roundtrip radio button	Open the URL http://spicejet.com	Click on Roundtrip radio button	Application should display both date pickers
5	Verify the availability of origins in Leaving from field	Open the URL http://spicejet.com	Click on Leaving from field	Application should display all the origins. Please refer SRS doc
6	Verify the functionality of date picker field	Open the URL http://spicejet.com	Click on date picker	1. Application should display all the dates based on system date 2. Previous dates should be disabled
7	Verify the availability of the dropdown fields with values	Open the URL http://spicejet.com	Check the Adult, child and infant dropdown	Adult should contain 1 to 9 adults Child should

				contain 0 to 4 infant should contain 0 to 4
8	Verify the one way search functionality with 1 adult	Open the URL http://spicejet.com	1. Click on one way radio button 2. Select from and TO 3. Select date 4. Select 1 adult 5. Click on find flight button	Application has to navigate to select page
9	Verify the one way search functionality with 2 adults and 1 child	Open the URL http://spicejet.com	1. Click on one way radio button 2. Select from and TO 3. Select date 4. Select 2 adult 5. Select 1 child 6. Click on find flight button	Application has to navigate to select page
10	Verify the one way search functionality with 3 adults, 2 children and 1 infant	Open the URL http://spicejet.com	1. Click on one way radio button 2. Select from and TO 3. Select date 4. Select 3 adults 5. Select 2 children 6. Select 1 infant 7. Click on find flight button	Application has to navigate to select page

Sample Test Cases for Gmail:

Req ID	Test Type	Priority	TC ID	Test Scenario	Pre condition	Teststeps	Actual Result	Expected Result	Result
1	GUI	P3	1	Verify the GUI of login functionality	Open the url http://gmail.com	1. Check the spell of all the fields 2. Check the font, allignment, color of all the fields 3. Check the look and feel of the page		Application should maintain the consistency	
1	Validation	P1	2	Verify the availability of the fields in login page	Open the url http://gmail.com	Check the below fields Email - Textbox Password - Textbox Sign in - Button		Application should display all the fields	
1	+ve	P1	3	Verify the login functionality with valid credentials	Open the url http://gmail.com	1.Enter the username jan30selenium 2. Enter the password Selenium5 3. Click on Sign in		Application should navigate to home page	
1	-ve	P2	4	Verify the login functionality with invalid credentials	Open the url http://gmail.com	Test the login with below credentials Username / Password jan30selenium/Selen jan30 / Selenium5 jan30/ selen12 / selenium jan30selenium /		Application should display the error msg "The email or password you entered is incorrect."	
1	DB	P1	5	Verify the credentials in DB	Login into the DB	Write and execute the valid query		DB should display the credentials	
2	GUI	P3	6	Verify the GUI of home page	Login into gmail.com	1. Check the spell of all the fields 2. Check the font, allignment, color of all the fields 3. Check the look and feel of the page		Application should maintain the consistency	

2	Validation	P1	7	Verify the availability of the compose field	Login into gmail.com	Check the compose field		Application should display the field	
2	+ve	P1	8	Verify the functionality the compose field	Login into gmail.com	Click on compose field		Application should navigate to compose page	
2	GUI	P3	9	Verify the GUI of the compose page	Click on compose field	1. Check the spell of all the fields 2. Check the font, allignment, color of all the fields 3. Check the look and feel of the page		Application should maintain the consistancy	
2	Validation	P1	10	Verify the availability of the fileds	Click on compose field	Check the below fields To - Textbox Subject - Textbox Body - Text area Send - Button		Application should display all the fields	

Sample Test Cases for Mobile:

Req ID	Test Type	Priority	TC ID	Test Scenario	Pre condition	Teststeps	Actual Result	Expected Result	Result
1	GUI	P3	1	Verify the GUI of mobile	Mobile should be available	1. Check the look and feel of a mobile. 2.Check whether the dimensions of mobile are according to the requirement		Mobile should maintain the consistency	
1	Validation	P1	2	Verify the availability of all fields / parts	Mobile should be available	1.Check battery 2.Check the charger, headset, panels 3. Check the keys		Mobile should contain the all required fields	

1	+ve	P1	3	Verify functionality of power button	Mobile should be available	Test the power button by long pressing it.		Mobile should be turn on	
1	+ve	P2	4	Verify functionality of calling	Switch on the mobile	1.Dail the mobile number 2.Press the call button		Mobile should make a call	
1	+ve	P2	5	Verify the functionality of message	Switch on the mobile	1.Press the message button . 2.Type the message 3.Enter the recipient number 4. Press send		Message should be delivered	
1	+ve	P1	6	Verify the functionality of menu button	Switch on the mobile	Press the menu button		Mobile should display the menu button	
1	+ve	P2	7	Verify the functionality of charging socket	Switch on the mobile	1. Insert the charger pin to device 2. Switch on the charger		Device should be switch into charging mode	
1	-ve	P2	8	Verify the Call functionality without Sim	Switch on the mobile	1.Switch of the device. 2. Remove the sim card. 3. Switch on the Device 4. Make a call		Device should display a message "sim card is not inserted"	
1	-ve	P1	9	Verify the power button functionality without battery	Switch on the mobile	long press the power button		Device should not turn on	
1	-ve	P2	10	Verify the incoming call without sim card	Switch on the mobile	1.Switch off the device. 2. Remove the sim card. 3. Switch on the Device 4. Make a call		Device should not respond	

						from other device.			

Sample Test Cases for Bike:

Req ID	Test Type	Priority	TC ID	Test Scenario	Pre condition	Teststeps	Actual Result	Expected Result	Result
1	GUI	P3	1	Verify the GUI of a bike	Bike should be available	1.Check the colour of a bike. 2.Check the Parts of a bike. 3.Check the look and feel.		Bike should maintain proper look and feel.	
1	Validation	P2	2	Verify the parts of a bike	Bike should be available	1.Check the Head light. 2. Check the acclerator, clutch, gear etc		Bike should be contain the all parts	
1	+ve	P1	3	Verify the functionality of the lock	Select the bike	1.Insert the bike key into the lock.		Lock should work properly	
1	+ve	P1	4	Verify the functionality of the start	Switch on the bike lock	Press the start button		Bike should start.	
1	+ve	P2	5	Verify the functionality of Handle lock	Select the bike	1.Insert the key in to Handle lock 2. Lock the bike handle		Bike handle should be locked	
1	+ve	P2	6	Verify the functionality of head light.	Start the bike	Switch on head light		Head light should ON.	
1	+ve	P1	7	Verify the functionality of Gear	Start the bike	1.Hold the clutch. 2. Change the gear		Gear should work properly	
1	-ve	P1	8	Verify the functionality of a lock with invalid key	Bike should be available	Insert the bike key into lock.		Lock should not insert in to lock	
1	-ve	P2	9	Verify the functional of tires wit out	Bike should be available	Remove the air in the tire and move		Bike should not move in a	

				air				proper manner	
1	-ve	P1	10	Verify the start functionality without key	Bike should be available	Press the start button without key.		Bike should not start	

Test Cases Review:

After preparation of test cases we perform review in order to check correctness and completeness of TC's w.r.t to requirements.
There are 2 levels of reviews on TC's.

1) Peer Review.
2) Test Lead Review.

1) **Peer Review:**
   By interchanging within the TE's will conduct review on their written test cases is called as peer review.

2) **Test Lead Review:**
   After peer review on TC's those will be send to test lead for review.

Requirements Traceability Matrix (RTM/TM):

In general it is difficult to understand all the requirements are covered or not with written test cases using TC document alone.
**Whereas to analyze gap between requirements to test cases we use RTM/TM.
RTM we can prepare by mapping each test case to the corresponding requirement/Scenario.
RTM can prepared by TL/TE.

RTM can prepared in excel sheet as shown below

Req ID	Test case id	Commnents
1	1	
2	2	
3	3 to 5	
4	6 to 8	
5	9 t0 12	
6	13 to 16	
7	Not prepared	Req is not clear, waiting for comments from BA
8	17 to 18	

4) Test Execution:

After TC's review TL will give the permission for TE's to execute test cases.

Test Execution Flow:

Test Execution levels:

⚽ Level-0 (Smoke test):

When build is deployed at test environment then initially TE's will perform smoke test in order to check build is stable or not for thorough testing. In general to perform smoke test we use checklist.

Ex: Prepare check list for Gmail application to perform smoke test.

1) Launch Gmail application.
2) Login with valid data.
3) All options are properly working or not like compose, inbox, drafts…etc
4) Navigation flow of application.
5) Logout is properly working or not.

⚽ Level-1 (Real testing/Comprehensive testing):

- **When build is stable then we perform thorough testing using test cases.**
- **Whereas test cases are executed inform of Test sets/Test suites/Test batches.**
- **Test set is a collection of test cases based on functionality flow in application (i.e. parent to child functionalities).**

During test cases execution TE will perform following activities (i.e. manual testing approach)

- **Step1: Perform operation on AUT as per "Step Description" given in TC.**
- **Step2: Record actual behavior of the AUT under "Actual Result" column.**
- **Step3: Compare expected result and actual results based on that provide status for a step in terms of PASS/FAIL.**

During test execution we provide following status for a Test case:

- **Pass: When all the expected values are matching with actual values in a test case.**
- **Fail: When expected values are not matching with actual values in a test case.**
- **Blocked: Due to parent functionality failure which test case not possible to execute.**

⚽ Level-2 (Re-Testing & Regression Testing):

- **Re-Testing: It is performed on modified build to check bugs are resolved or not.**
- **Regression Testing:**
  - **It is performed on modified build to find any impact on existing passed functionalities w.r.t to modifications.**
  - **Re-Testing & Regression testing both are performed at same level on modified build.**

⚽ **Level-3 (Final Regression test/End-to-End Scenario testing);**

Before deliver application for UAT we perform end-to-end scenario testing.
In this test we validate some of the functionalities from login session to logout session based on defect density during comprehensive testing (i.e. Level1).
***Number of defects identified in a module is called Defect Density**

5) Defect Reporting & Defect Tracking:

During test case execution whenever we identify deviation b/w expected to actual result is called as Defect/Issue/Incident/Fault.
Notifying about the defects to the developers is called defect reporting. Defects can be reported manually (i.e. using common repository) or using any one of the defect tracking tool like.

- **Quality Center (ALM) →HP**
- **Clear Quest→IBM**
- **Bugzilla**
- **D-Tracker**
- **Issue Tracker**
- **P-Tracker**

Defect Submission process:

Whenever defect identified then we prepare defect profile and that will be send to test lead to get an approval, if defect is valid then TL will intimate to the reporting manager in development team.

Components in Defect Template:

Summary: Brief description about defect.

Defect Id: Unique identification.

Reported by: Name of TE.

Reported on: On which date defect reporting.

Note: In general defect should be reported on the same day when it identified.

Reported to: To whom we reporting defects (i.e. Test Lead).

Reproducible Defect: [YES/NO]

Whenever defect identified in functionality then we need to recheck the same functionality to confirm defect will rise or not. If defect is raised then it is consider as Reproducible defect.

Priority:
It describes importance of the functionality in which we identified defect.
Priority also describes in what time that defect should be resolved.
There are different levels of priorities.

- **High**
- **Medium**
- **Low**

Severity: It describes seriousness of the defect or impact of the defect to perform test execution.
There are different levels of severities:
Critical (High functional defect and there is no workaround to continue execution)
Major (High functional defect but there is a workaround to continue execution)
Medium (Medium and low level functional defects)
Minor (Defect related to UI and Usability)

Project: Project Name.

Build Version Id: In which version of build we identified defects.

Test Set: Name of the test set in which TC failed.
TC Id/Name: Name of the TC.

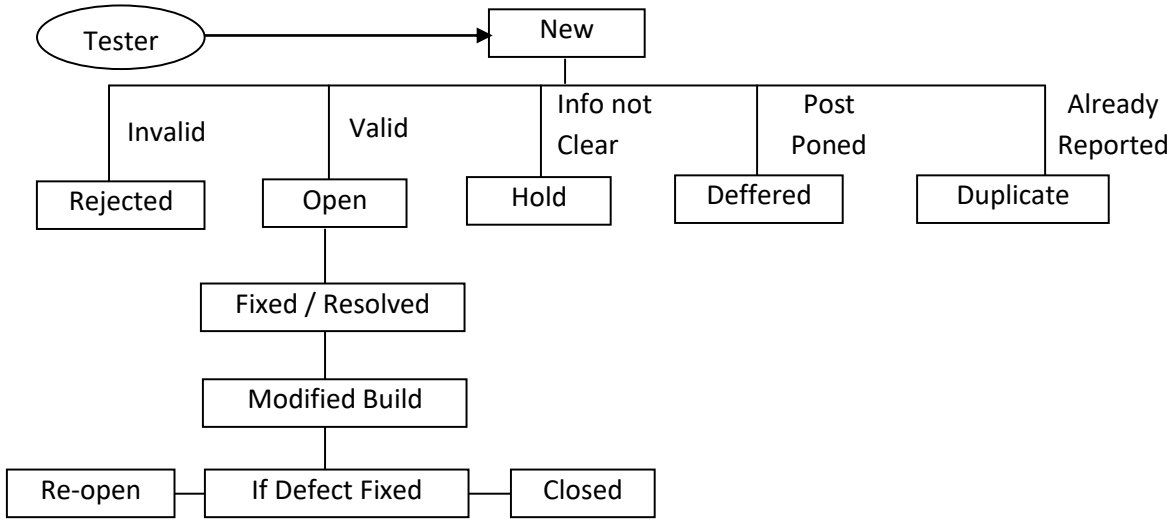
Snapshot: Defect Snapshot.

Status of Defect: It describes state of the defect

Ex: NEW
Note: When we reporting first time about the defect should have the status as "NEW".

Bug Life Cycle: (BLC)

It describes varies status of defect from it identification to closed.



TE will provide following status for a defect.

1) New: First time reporting about the defect.
2) Re-Open: When TE not satisfied about defect resolve work during Re-testing.
3) Closed: When TE satisfied about defect resolve work during Re-testing.

Developers will provide following status for a defect.
4) Duplicate: When defect is similar to earlier defect.
5) Hold: When developers required more information about the defect.
6) Rejected: When developers are not accepting that as a defect.
7) Deferred: Defect is accepted but that will be considered in next release of the application in order to solve.
8) Open: When developers are accepted and ready to resolve.
9) Assigned: When task assigned for developer to solve the defect.
10) Fixed: When defect/bug is resolved.
Defect Tracking: Maintenance about the defects for a project is called effect tracking.
Defect Age: The time gap between defect identification to closed date.

6) Test Closure:

It is performed by Test Lead. Whereas TL will analyze "Exit criteria" factors, if majority of the factors are satisfied then he will stop testing activities for a project.

Test Report Template:

Build Status Report / Test Report

Test Engg Name:	
Build No	**1**
Login Credentials	
Browser	**FF, IE GoogleChrome**
Test Matrics	
Total no of test cases	**200**
No of Test cases executed	**150**
No of Test cases pending	**50**
No of Test cases Pass	**100**
No of Test cases Fail	**50**
No of Test cases Skipped	**10**
No of Bugs Reported	**20**

Daily Status Report: **The daily status report template looks as follows**

Daily Status Report	
Tasks done for Today	
Task	**Comments**
Executed 50 Testcases	**40 - Pass** **10- Fail**
Prepared 20 new Test cases	
Tasks planned for Tomorrow	
Task	**Comments**
Will execute 50 more test cases	
Will prepare 20 more test cases	

Manual FAQ's

Q. What is Testing?

- Verification & validation of software is called as testing.
- Verification means whether the s/w is correctly implemented or not.
- Validation means the implemented s/w meets the customer requirements or not.

Q. What phases are available in SDLC?

It describes developing process of software project/product to fulfill the client requirements within the specified cost & time. Following are the phases in SDLC.

- Requirements Collection
- Analysis
- Design
- Coding
- Testing
- Release & Maintenance

Q. What is the difference between Bug, Defect and Failure?

Defect:

It is a deviation between expected result and actual results in AUT (Application under Test). Defect can also call as Issue/Incident/Fault.

Bug:

When developers are accepted defects then those are called bugs.

Failure:

When defects are reached end users then those are called failures.

Q. Difference between Waterfall, V-model?

Waterfall Model: It is a sequential execution. Once the phase is completed the high level management will analyze the phase. Its costly process, the requirements can't be changed in between the phase and the risk can't be identified and prevented at the early stage of the life cycle

DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

V-Model: **Testing activities will be started at the early stage of the life cycle. The validation team will participate from requirement phase on-wards. Will prepare the test plan once the SRS doc is base lined and also prepares the system test plan, Design plan once the Analysis phase is completed.**

Advantages:

1. We can ensure for quality because the testing activities are starting at early stage of SDLC.

2. Risk can be identified at early stage and it can be prevented.

3. Requirement change can be accepted at early stage of SDLC

Q. What is agile process?

Agile software development is a group of software development methods in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, continuous improvement and encourages rapid and flexible response to change.

Customer satisfaction by rapid delivery of useful software

Welcome changing requirements, even late in development

Working software is delivered frequently for each sprint

Close, daily cooperation between business people and developers

Projects are built around motivated individuals, who should be trusted

Face-to-face conversation is the best form of communication

Working software is the principal measure of progress

Sustainable development, able to maintain a constant pace

Continuous attention to technical excellence and good design

Simplicity—the art of maximizing the amount of work not done—is essential

Self-organizing teams

Regular adaptation to changing circumstances

Q. What is Re-testing testing and Regression testing?

Re-Testing:

- **It is performed on modified build to check bugs are resolved or not. Whereas we use fail test cases to perform execution on modified build.**
- **When we validate same functionality with multiple set of test data is also called Re-Testing.**

Regression Testing:

- **It is also performed on modified build to find any impact or side effects on existing passed functionalities w.r.t modifications.**
- **Regression testing can be performed partially or fully on application based on modification.**

Q. What is the difference between Sanity and smoke Testing?

Smoke Testing:

It is performed on initial build to check the build is stable or not for further testing. In this test we validate major/core functionalities with valid data.

Sanity Testing:

It is also similar to smoke test. It is a subset of regression testing. It is performed on modified build to check build is stable or not.

Note:

If build is unstable then we suspend test cases execution and reject build to developers. For rejected build developers will release patch files to testing team.

Q. What is Load and Stress Testing?

Load Testing/Scalability: Verify application allows customer expected concurrent users or not.
Stress Testing: In this test we use beyond the load limit to estimate peak limit of the load on application.

Q. What is Usability testing?

In this test we verify user friendliness of the application to perform operations (i.e ease of use). Following are actors related during usability test.

- **Error messages are meaningful or not.**
- **Help documents are understandable or not.**
- **Functional keys are implemented or not (F1-->Foe Help).**
- **Combination keys are implemented or not (ctrl+P-->Print).**

Q. What is UAT?

After system testing client team perform UAT. The objective of UAT is to confirm application is ready for release or not.
Whereas UAT is performed based on their business needs.

Alpha Testing:
It is performed at developer side in controlled environment. If any defects identified those will be resolved and should get approval client.

Beta Testing:
After alpha testing application will be deployed at client environment. Beta testing performed at client site in uncontrolled environment.

Q. What is BVA and ECP?

BVA:

- **This technique use to validate input condition interims of range or size.**
- **This technique says there will be 3-possibilities at each boundary level to validate.**
- **Possible conditions are:**

Min	**Max**
Min-1	**Max-1**
Min+1	**Max+1**

- **In general there will be high possibility of errors in and around the boundary conditions those we can identify with optimal test data using BVA.**
  Ex: Prepare test data using BVA for a Name edit box which allows from 4 to 16 characters.

BVA(Size)	
Min=4→valid	**Max-1=15→valid**
Max=16→valid	**Min+1=5→valid**
Min-1=3→In valid	**Max+1=17→In valid**

ECP:
This technique use to validate input condition with data type. This technique says divide test data equivalence classes interims valid and invalid and then takes some sample data from each class to validate.

Ex: Prepare test data using ECP to verify withdraw condition of ATM which allows 100-20000, whereas amount should be multiples of 100.

ECP (Type)	
Valid	**Invalid**
100<=Amount<=20000 **{Amount should be multiples of 100}**	**Amount<100** **Amount>20000** **Amount which is not multiples of 100** **Ex:150,550**

Q. What is system testing? Explain in detail?

- **After unit testing and integration testing development team will release initial build to the separate testing team.**
- **Build means set of integrated modules an executable form of application.**
- **In general internal release of application from developers to TE's we call it as build.**
- **After receiving stable build from developers, TE's will perform system testing using BBT techniques.**
- **"Validating whole system based on client requirements and expectations is called system testing".**
- **There are two types of testing techniques in system testing.**
    - **Functionality Testing**
    - **Non-Functionality Testing**

Q. What is the differences between web application testing and client server application testing?

In Client Server testing, there are two different components to test.

Application is loaded on server machine and the application exe on every client machine.

Testing is done broadly in categories like GUI on both sides, functionality, Load, Client Server interaction and backend testing. Most of the client server applications are Intranet networks. Web Application testing is complex to test as tester doesn't have control over the application. Application is loaded on the server whose location may not be known and no exe is installed on the client machine. Hence, Web Applications are supposed to test on different browsers and OS platforms. It is tested mainly for browser and OS compatibility, error handling, static pages, backend testing and Load Testing.

Q. Explain in detail what is integration testing?

After unit testing those individual components are connected in order to form the system. "Validating interfaces or connectivity between the units / components / modules is called integration testing".

In practical all the modules construction may not possible to complete at same time in that case we use integration testing approaches.

→Big-bang Approach

→Incremental Approach

Q. Difference among Use case, Test Scenario & Test Case?

UC: It describes functional behaviour from user perspective

{How system need to work}

TS: It describes test condition or requirement or functionality which we need to validate.

{What to Test}

TC: It describes validation procedure of each requirement or functionality.

{How to Test}

Q. Differences between QA and QC?

Quality Assurance:

QA team will define development process of application to prevent defects in application. QA team involve throughout life cycle to monitor and measure the strength of development process, if any weakness identified they will provide suggestion to improve the strength of development process.

High level management team, domain experts, technical experts are the members in QA.

Quality Control:

QC team involve after product is developed. In order to deliver in the defects and make sure those should be resolved before deliver application to the customer

Q. In defects in the software, how are priority and severity defined?

Severity describes how seriously the bug is impacting the application. Each and every bug will have the severity. Severity has below types,

1. Blocker 2. Very High 3. High 4.Medium 5. LowPriority describes the order to fix the bugs. It is of type P1, P2, P3, P4 and P5.

P1 bugs are fixed initially, then P2, P3, P4, P5

We will maintain the priority for each bug based on severity.

Q. What is the difference between priority in test cases and priority in bugs?

Priority in TestCases: Every test case/scenario has a priority. It describes the importance of the test cases. There are 4 types of priorities.

Priority1 (P1) – The Test Case describes about the main functionality.

Priority2 (P2) – The Test Case describes about the field level.

Priority3 (P3) – All the GUI's.

Priority4 (P4) - If the test engineer is planning to provide any suggestion to the application, then he can write test cases where the priority is P4

Priority in Bugs: Based on the severity the priority will be assigned. Priority describes in which order the has to be fixed by the developer

Blocker – P1

Very High – P2

High – P3

Medium – P4

Low – P5

Q. What do you write in a test plan?

It will prepare based on build. It describes detailed testing approach, name of TE's, Features to be tested, Test Environment, Entry criteria, exit criteria, schedules, test deliverables, scope.etc… Based on build which we receive from developers information in test plan may vary.

For successful s/w testing a clear test plan is essential.

BRS & FRS Build Info → TL → Test Planning → Finalized Test Plan Doc

Test Planning:
- →Team Information
- →Task Allocation
- →Risk Analysis
- →Training Needs
- →Schedules
- →Prepare Test plan
- →Review on Test Plan

Finalized Test Plan Doc:
- →What to Test
- →Whom to Test
- →How to Test
- →When to Test

Q. What is AUT?

AUT stands for Application under Test. The project which we are testing is known as AUT.

Q. What is Ad-hoc Testing?

It is an informal testing activity due to lack of requirements. Test engineer use past experience to validate the application.

Q. What is Alpha Testing and Beta Testing?

Alpha Testing:

It is performed at developer side in controlled environment. If any defects identified those will be resolved and should get approval client.

Beta Testing:

After alpha testing application will be deployed at client environment. Beta testing performed at client site in uncontrolled environment.

Q. What is Compatibility Testing/Portability Testing?

Verify application supports customer expected OS's & browsers & compilers.

- **Forward compatibility testing**
- **Backward compatibility testing**

Q. What is End-to-End testing?

Before deliver application for UAT we perform end-to-end scenario testing.In this test we validate some of the functionalities from login session to logout session based on defect density during comprehensive testing (i.e. Level1).
***Number of defects identified in a module is called Defect Density**

Q. What is Localization Testing (L10N Testing)?

Verify application support domestic/local languages or not with in the country.

Q. What is Internationalization Testing (I18N Testing)?

In this test we verify application supports international languages currency, time, date format etc…

Q. What is Monkey / Jump Testing?

Due to lack of time we validate major functionalities in application. In this case we select test cases based on priority. This type of testing also called priority based testing.

Q. Explain the bug which is High Severity and Low priority ?

High severe bugs will be having high priority. But if the bug is not related to the current release then the priority of the bug will became low priority. The development lead will have the permission to change the priority.

Q. Explain the bug which is Low Severity and High priority ?

Low severe bugs will be having low priority. But if the bug is related to the current release then the priority of the bug will became high priority. The development lead will have the permission to change the priority.

Q. What is Endurance Testing (or Soak Testing)?

Soak/Endurance Testing: Verify how much time continuously we can run application. (I.e. Duration).

Q. What is Data Volume Testing?

Data Volume Testing: Verify how much data we can transfer in a specified time.

Q. Skill set required for a functionality test engineer:

- **Knowledge on manual testing concepts.**
- **Knowledge on any one of functionality testing tool like Selenium, QTP.Knowledge on defect reporting tool like QC, Bugzilla, Jira.**
- **Some portion of knowledge on data base like Sql-Server, Oracle.**

Q. What is automation testing?

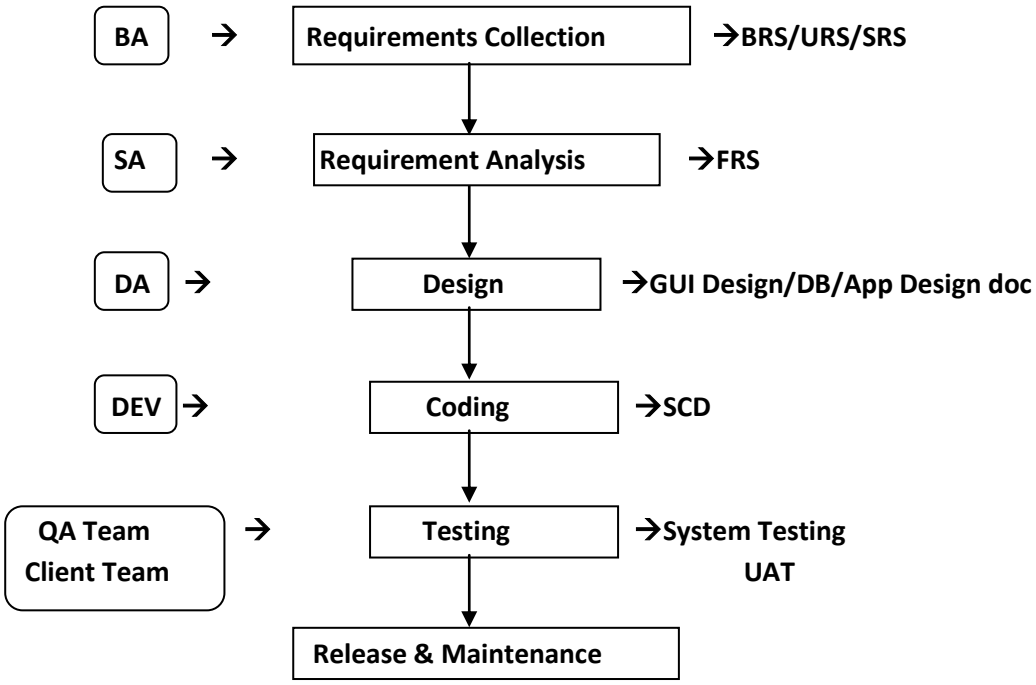
- **Any task or activity performed with the help of any programming or scripting language or tool is called as automation.**
- **Automating human activities in order to validate application is called automation testing.**
- **Automation testing can be performed using programming language or scripting language or any one of the 3rd party tools selenium, QTP, silk, rational robot…etc.**
- **In automation testing we develop the automation test script.**

Q. Ex: Write test cases to verify login functionality in gmail with invalid data.

Test case ID/Name	Test Case Description	Step Name	Step Description	Expected Result
		Step 1	Enter invalid email & valid pwd & click on “signin”.	Error message should popup. “Invalid user Id”

TC01_Gmail_Login	This TC to verify login functionality using invalid data.	Step 2	Enter valid email in valid pwd & click on "signin".	Error message should popup. "Invalid password"
		Step3	Enter invalid email in valid pwd & click on "signin".	Error message should popup. "Invalid user Id & password"

SDLC Phases:



Q. When SDLC end for a project?

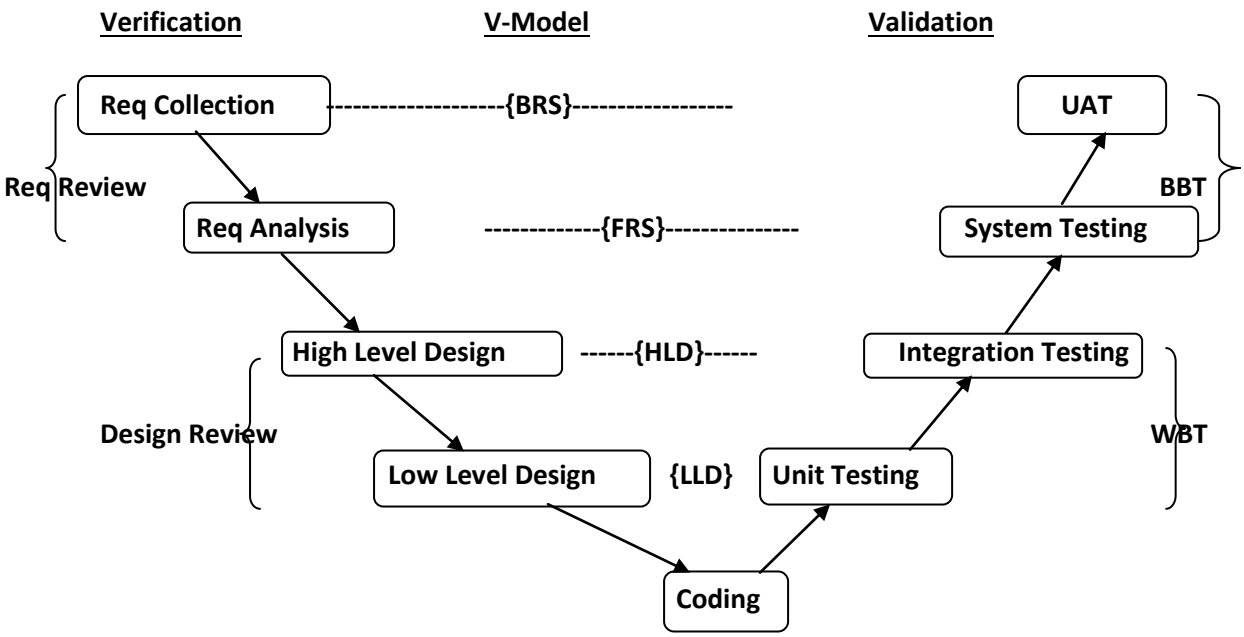
When project is no longer in use then that will be end state for SDLC of that project.

Q. What is Unit Testing?

It is also called as component testing. After coding developers will perform unit testing using WBT techniques. It is an initial validation technique.

Validating individual components within the system is called Unit Testing.
Following are the factors they validate in unit testing.

V-Model:



Q. Describe about Functionality Testing?

- It is also called as requirements testing. Validating application functional behavior based on user business transactions is called functionality testing.
- Any type of application functionality testing is mandatory and very important. Due to that reason organization maintains separate functionality testing team.
- Functionality testing can be performed manually or using some of the functional testing tools like selenium, QTP, Silk test, rational robot…etc.
- Following are the factors we validate during functionality testing.
  - →Object properties coverage
  - →Error handling coverage
  - →Input-domain coverage
  - →Calculation coverage
  - →Data base coverage
  - →Links coverage

Q. Describe about Non-Functionality Testing?

In this test we validate application characteristics based on customer expectations is called non-functionality testing.

Following are the Non-Functionality testing techniques.

- →GUI Testing
- →Usability Testing
- →Performance Testing
- →Recovery Testing
- →Compatibility Testing
- →Configuration Testing
- →Comparative Testing
- →Installation Testing

→Security Testing →Sanitation Testing

Q. STLC Phases:

→Test Initiation
→Test Planning
→Test case design
→Test Execution
→Defect Reporting
→Test Closure

Test Initiation:

BRS/FRS
Organization Standards doc → TM → Test Initiation → Test Strategy

→Application Analysis
→Risk Analysis
5) Resources Risk
6) Technology Risk
7) Scheduled Risk
8) Support Risk
→Effort Estimation

Test Planning:

BRS & FRS
Build Info → TL → Test Planning → Finalized Test Plan

→Team Information
→Task Allocation
→Risk Analysis
→Training Needs
→Schedules
→Prepare Test plan
→Review on Test Plan

→What to Test
→Whom to Test
→How to Test
→When to Test

Test Case Design:

I/P

BRS/FRS
Use Cases
Test Scenario

→ TE →

Clarification Team
Support Team

Test Case Design → Test Case Design Doc

- →Fundamental of TC's
- →TC Design Technique (BBT)
  - →BVA
  - →ECP
- →Error Guessing
- →Types of TC's
  - →UI TC's
  - →Usability TC's
  - →Validation TC's
  - →Functionality TC's
    - →+ Ve TC's
    - →- Ve TC's
- →TC Template
- →TC Reviewing
  - →RTM/Traceability Matrix