**Error, bug, defect, SDLC, TDLC, Testing levels, Testing types, QC, QA, ODC(offline development center)**

**Commonly used Terms**

* POR – Plan of Record
* MRD – Marketing Requirements Document
* ERD – Engineering Response Document
* RAIT – Requirement Assumption Issue Test
* FS/DS – Functional Spec / Design Spec
* CR – Change Request
* RFE/ER – Request For Enhancement / Enhancement Request
* EA / CA – Engineering Advisory / Customer Advisory
* RC – Release Candidate
* GMSO – Gold Master Sign Off
* Web Release

Error: -

The difference between expected result and actual result is error.

BUG: -

If error comes at the time of development stage it is called bug.

Defect: -

If the error comes after production then we say it defect.

What is test case?

Test case is a process with set of actions and conditions to validate particular feature in a application.

What is manual testing?

If we do any testing with human interaction without using any automation tool or script that testing is called manual testing. In this testing the tester takes over the roles of an end user and tests a software to identify any unexpected behavior or bug. They are different stages in manual testing. 1) Unit testing, 2) Integration testing, 3) System testing, 4) User acceptance testing

What is automation testing, when we use?

If tester use any automation tool or scripts and any software to test software.

* Automation testing is used to rerun the test scenarios quickly and repeatedly.
* It is used for regration testing .
* Using automation we can test application on load.
* Performance and stress point of view.
* It increases the test coverage and improve accuracy.
* Save time, resource and cost compare to manual testing.

How to automate application:-

Steps:-

1. Identify areas within software for automation.
2. Select appropriate tool for test automation.
3. Through scripts
4. Development of test suites
5. Create scripts for test cases
6. Create result report
7. Identify any potential bug or performance issue.

Soft ware testing tools:-

* Win runner
* Load runner
* Selenium

15 . What is agile methodology and testing methodologies.

* Waterfall model
* V mode
* Agile model

Waterfall model is divided into following separate phases:

Waterfall module is a simple and traditional project management

This is a sequential software development model.

* Requirement Gathering & Analysis phase,
* Software Design,
* Implementation and Testing
* Maintenance

All these phases are the sequential phases & next step is totally depends on the previous step, so that second phase is begins as only if the defined set of objectives and goals are

completed first.

Advantages of Waterfall Model:

* This is linear simple model to implement & easy to maintain.
* In the SDLC initial phase spent time on reviewing requirements and design which saves the time later.
* Required resources are minimum in this model as compare to other.
* After every phase of the model a document in created which help & simpler to understand & design the system.
* Upon completion of coding, is done to check for implemented code or correctness of system.
* For each stage deadlines can be set which will help to develop the system on decided time frame.

Disadvantages of Waterfall Model:

* The biggest disadvantages of such system is it won’t allow to go back. If the problems in the design phase which creates complication in the implementation phase.
* This model is rigid model. Not flexible & make changes is not possible in the development of system.
* Unable to start the next phase before completing the previous phase so time consuming.
* In the deployment phase all requirements are not covered or all requirements are not cover so adding such requirements may create unsuitable system.
* To make it customer happy such new requirement need to be implemented in new version of system which leads to added cost to system development.

V-module:-

What is V-model- advantages, disadvantages and when to use it?

V- model means Verification and Validation model. Just like the waterfall model, the V-Shaped life cycle is a sequential path of execution of processes. Each phase must be completed before the next phase begins. Testing of the product is planned in parallel with a corresponding phase of development.

Diagram of V-model:

V-model

The various phases of the V-model are as follows:

Requirements like BRS and SRS begin the life cycle model just like the waterfall model. But, in this model before development is started, a system test plan is created. The test plan focuses on meeting the functionality specified in the requirements gathering.

The high-level design (HLD) phase focuses on system architecture and design. It provide overview of solution, platform, system, product and service/process. An integration test plan is created in this phase as well in order to test the pieces of the software systems ability to work together.

The low-level design (LLD) phase is where the actual software components are designed. It defines the actual logic for each and every component of the system. Class diagram with all the methods and relation between classes comes under LLD. Component tests are created in this phase as well.

The implementation phase is, again, where all coding takes place. Once coding is complete, the path of execution continues up the right side of the V where the test plans developed earlier are now put to use.

Coding: This is at the bottom of the V-Shape model. Module design is converted into code by developers.

Advantages of V-model:

* Simple and easy to use.
* Testing activities like planning, test designing happens well before coding. This saves a lot of time. Hence higher chance of success over the waterfall model.
* Proactive defect tracking – that is defects are found at early stage.
* Avoids the downward flow of the defects.
* Works well for small projects where requirements are easily understood.

Disadvantages of V-model:

* Very rigid and least flexible.
* Software is developed during the implementation phase, so no early prototypes of the software are produced.
* If any changes happen in midway, then the test documents along with requirement documents has to be updated.

When to use the V-model:

* The V-shaped model should be used for small to medium sized projects where requirements are clearly defined and fixed.
* The V-Shaped model should be chosen when ample technical resources are available with needed technical expertise.
* High confidence of customer is required for choosing the V-Shaped model approach. Since, no prototypes are produced, there is a very high risk involved in meeting customer expectations.

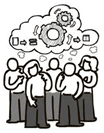
Agile :-

**What is Agile?**

Agile methodology is alternative to traditional project management.

AGILE is a methodology that promotes **continuous iteration** of development and testing throughout the software development life cycle of the project. Both development and testing activities are concurrent unlike the Waterfall model

I hope we got an idea of Agile!!! Now, we can step on to Agile Testing.

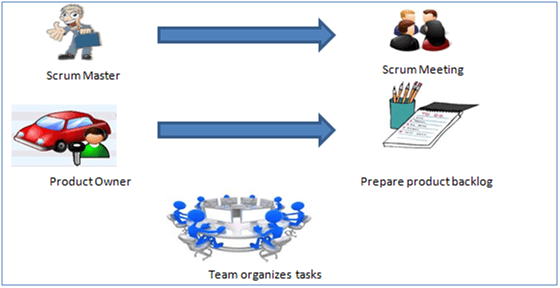
[](http://www.guru99.com/images/11-2014/agile_Processesv1_1.png)

The agile software development emphasizes on four core values.

1. Individual and team interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

**Scrum**

SCRUM is an agile development method which concentrates specifically on how to manage tasks within a team based development environment. Basically, Scrum is derived from activity that occurs during a rugby match. Scrum believes in empowering the development team and advocates working in small teams (say- 7 to 9 members). It consists of three roles, and their responsibilities are explained as follows:

[](http://www.guru99.com/images/11-2014/agile_Processesv1_3.png)

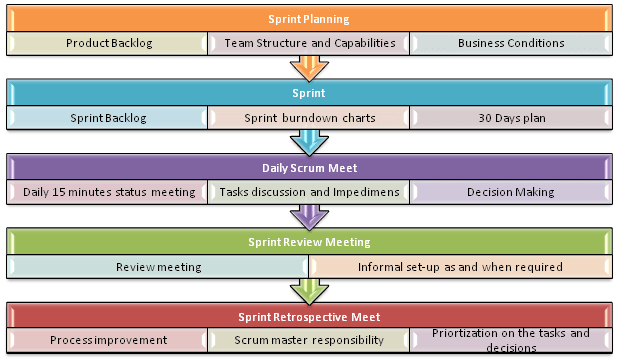
* Scrum Master
  + Master is responsible for setting up the team, sprint meeting and removes obstacles to progress
* Product owner
  + The Product Owner creates product backlog, prioritizes the backlog and is responsible for the delivery of the functionality at each iteration
* Scrum Team
  + Team manages its own work and organizes the work to complete the sprint or cycle

**Product Backlog**

This is a repository where requirements are tracked with details on the no of requirements to be completed for each release. It should be maintained and prioritized by product owner, and it should be distributed to the scrum team. Team can also request for a new requirement addition or modification or deletion

**Scrum Practices**

Practices are described in detailed:

[](http://www.guru99.com/images/11-2014/agile_Processesv1_4.png)

**Process flow of Scrum:**

Process flow of scrum testing is as follows:

* Each iteration of a scrum is known as Sprint
* Product backlog is a list where all details are entered to get end product
* During each Sprint, top items of Product backlog are selected and turned into Sprint backlog
* Team works on the defined sprint backlog
* Team checks for the daily work
* At the end of the sprint, team delivers product functionality

23 . What is retesting.

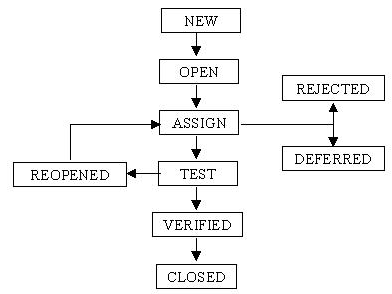
1. What is bug life cycle.

A defect is a specific concern about the quality of an Application Under Test (AUT).

Defect can be defined as an inconsistency in the behavior of the software. No software exists without a defect or bug. The elimination of bugs from the software depends upon the efficiency of testing done on the software. A defect is basically the difference between the expected result and the actual result.

The cost of finding and correcting defects represents one of the most expensive software development activities. For the foreseeable future, it will not be possible to eliminate defects. While defects may be inevitable, we can minimize their number and impact on our projects. To do this, development teams need to implement a defect management process that focuses on preventing defects, catching defects as early in the process as possible, and minimizing the impact of defects. A little investment in this process can yield significant returns.

**Defect Life Cycle**  
The bug has a life cycle in software development process. The bug should go through the life cycle to be closed. A specific life cycle ensures that the process is standardized. The bug attains different states in the life cycle. The life cycle of the bug can be shown diagrammatically as follows:



**The different states of a bug can be summarized as follows:**

1. New
2. Open
3. Assign
4. Test
5. Verified
6. Deferred
7. Reopened
8. Duplicate
9. Rejected
10. Closed

**Description of Various Stages**

1. **New :**When the bug is posted for the first time, its state will be "NEW". This means that the bug is not yet approved.
2. **Open :** After a tester has posted a bug, the lead of the tester approves that the bug is genuine and he changes the state to "OPEN".
3. **Assign :** Once the lead changes the state to "OPEN", he assigns the bug to corresponding developer or developer team. The state of the bug now is changed to "ASSIGN".
4. **Test :** Once the developer fixes the bug, he has to assign the bug to the testing team for the next round of testing. Before he releases the software with the bug fixed, he changes the state of the bug to "TEST". It specifies that the bug has been fixed and is released to the testing team.
5. **Deffered :** When the bug is changed to the deferred state, the bug is expected to be fixed in future releases. The reasons for changing the bug to this state has many factors. Some of them are the priority of the bug may be low, a lack of time for the release or the bug may not have a major effect on the software.
6. **Rejected :** If the developer feels that the bug is not genuine, he rejects the bug. Then the state of the bug is changed to "REJECTED".
7. **Duplicate :** If the bug is repeated twice or the two bugs mention the same concept of the bug, then one bug status is changed to "DUPLICATE".
8. **Verified :** Once the bug is fixed and the status is changed to "TEST", the tester tests the bug. If the bug is not present in the software, he approves that the bug is fixed and changes the status to "VERIFIED".
9. **Reopened :** If the bug still exists even after the bug is fixed by the developer, the tester changes the status to "REOPENED". The bug traverses the life cycle once again.
10. **Closed :** Once the bug is fixed, it is tested by the tester. If the tester feels that the bug no longer exists in the software, he changes the status of the bug to "CLOSED". This state means that the bug is fixed, tested and approv

26 .About SDLC ,STLC .

SDLC phases:

Requirement capturing  
Analysis  
Design  
Coding  
Testing  
Maintainance  
  
STLC phases:

Test Initiation  
Test Planning  
Test Executing  
Test monitoring  
Test Controlling  
Test Closure

|  |  |
| --- | --- |
| **SDLC (Software Development Life cycle)** | **STLC (Software Test Life Cycle)** |
| SDLC is Software Development Lifecycle, it is a systematic approach to develop a software. | The process of testing a software in a well-planned and systematic way is known as software testing life cycle (STLC). |
| Requirements gathering | Requirements Analysis is done is this phase, software requirements are reviewed by test team. |
| Design | Test Planning, Test analysis and Test design is done in this phase. Test team reviews design documents and prepares the test plan. |
| Coding or development | Test construction and verification is done in this phase, testers write test cases and finalizes test plan. |
| Testing | Test Execution and bug reporting, manual testing, automation testing is done, defects found are reported. Re-testing and regression testing is also done in this phase. |
| Deployment | Final testing and implementation is done is this phase andfinal test report is prepared. |
| Maintenance | Maintenance testing is done in this phase. |

3 . Testing concepts .

What are Software Testing Methodologies?

Software testing methodologies are the different approaches and ways of ensuring that a software application is fully tested. Software testing methodologies encompass everything from unit testing individual modules, integration testing an entire system to specialized forms of testing such as security and performance.

Importance of Testing Methodologies

As software applications get ever more complex and intertwined and with the large number of different platforms and devices that need to get tested, it is more important than ever to have a robust testing methodology for making sure that software products/systems being developed have been fully tested to make sure they meet their specified requirements and can successfully operate in all the anticipated environments with the required usability and security.

This page describes the various components of a thorough testing methodology and illustrates how SpiraTest is best suited to help you implement and manage them on your projects.

Unit Testing

The Unit testing part of a testing methodology is the testing of individual software modules or components that make up an application or system. These tests are usually written by the developers of the module and in a test-driven-development methodology (such as Agile, Scrum or XP) they are actually written before the module is created as part of the specification. Each module function is tested by a specific unit test fixture written in the same programming language as the module.

Integration Testing

The Integration testing part of a testing methodology is the testing of the different modules/components that have been successfully unit tested when integrated together to perform specific tasks and activities (also known as scenario testing). This testing is usually done with a combination of automated functional tests and manual testing depending on how easy it is to create automated tests for specific integrated components.

System Testing

The system testing part of a testing methodology involves testing the entire system for errors and bugs. This test is carried out by interfacing the hardware and software components of the entire system (that have been previously unit tested and integration tested), and then testing it as a whole. This testing is listed under the black-box testing method, where the software is checked for user-expected working conditions as well as potential exception and edge conditions.

Analogy :- During the process of manufacturing a ball pen. The cap, the body, the tail, the ink and the ballpoint are produced separately and every unit tested separately. When two or more units ready they assembled and Make integration test. When complete all units and integration testing system testing is performed after system testing they do Acceptance testing on client location.

Acceptance Testing

The acceptance testing part of a testing methodology is the final phase of functional software testing and involves making sure that all the product/project requirements have been met and that the end-users and customers have tested the system to make sure it operates as expected and meets all their defined requirements:

Non-Functional Testing

In most testing methodologies, functional testing involves testing the application against the business requirements. Functional testing is done using the functional specifications provided by the client or by using the design specifications like use cases provided by the design team.

On the other hand, non-functional testing involves testing the application against the non-functional requirements, which typically involve measuring/testing the application against defined technical qualities (also known as the ‘-ilities’ because they all end in ‘-ility), for example: vulnerability, scalability, usability. Some examples of non-functional testing are described below:

Performance, Load, Stress Testing

There are several different types of performance testing in most testing methodologies, for example: performance testing is measuring how a system behaves under an increasing load (both numbers of users and data volumes), load testing is verifying that the system can operate at the required response times when subjected to its expected load, and stress testing is finding the failure point(s) in the system when the tested load exceeds that which it can support.

We focus mainly on : Load , stress, response time, test condition, timing, stability

Security, Vulnerability Testing

Previously, security was something that was tested after-the-fact. With the rise in cyber-crime and the awareness of the risks associated with software vulnerabilities, application security is now something that needs to be designed and developed at the same time as the desired business functionality. Security testing tests the software for confidentiality, integrity, authentication, availability, and non-repudiation. Individual tests are conducted to prevent any unauthorized access to the software code.

Usability Testing

The usability testing part of a testing methodology looks at the end-user usability aspect of the software. The ease with which a user can access the product forms the main testing point. Usability testing looks at five aspects of testing, - learnability, efficiency, satisfaction, memorability, and errors.

Compatibility Testing

The compatibility part of a testing methodology tests that the product or application is compatible with all the specified operating systems, hardware platforms, web browsers, mobile devices, and other designed third-party programs (e.g. browser plugins). Compatibility tests check that the product works as expected across all the different hardware/software combinations and that all functionality is consistently supported.

4) Black box testing ?

Black box testing test the functionality of software and application, based on requirement.

It’s like : Unit testing , Integration testing, System testing, Acceptance testing

5) White box testing ?

To make white box testing tester should have knowledge about internal code and design.

It’s also known as glass box testing, structured testing clear box and open box testing.

6) Regration testing ?

In regration testing If we change any code or module we have to check entire functionality of code or module to make sure whether it is working or not properly after made change.

7) Sanity testing?

Sanity test is a narrow regration test it focus one or few areas of functionality. We make this testing when we have less time.

8. What is retesting?

Conformation testing is known as retesting. Conformation testing is to make sure the last fix or failure is working fine or not

9. Virtualization testing?

Multi os or Virtual machines on one physical machine .its reduce the cost of hardware and resource.

10. What is smoke testing?

* Smoke testing covers most of major functions of software but none of them in depth.
* The result of this test is used to decide whether to proceed with further testing if smoke testing pass we go ahead with further testing if it fails we have to stop next testing phase and ask for a new build with the required fixes.
* It is used to save time before going regression, integration and acceptance tests.
* In this testing we test very essential features of applications and hardware setups.

1. Functional testing types?
2. Unit testing
3. Smoke testing
4. Sanity testing
5. Integration testing
6. Usability testing
7. Interface testing
8. System testing
9. Regression testing
10. User acceptance testing
11. Non Functional testing types?

* Availability testing
* Compatibility testing
* Configuration testing
* Documentation testing
* Installation testing
* Load testing
* Performance testing
* Recovery testing
* Usability testing
* Stress testing
* Security testing

In non-functional testing the tester check behalf of user side, Check the application performance, scalability services, User satisfaction, system speed, response time and how much time its take to complete task.

1. Alpha testing?

* It is a first phase of testing in SDLC.
* It is done before releasing the product.
* It’s performed by testers who are internal employees of the organization.
* It’s performed at developer’s site.
* It’s involves both white box and white box testing.
* Alpha testing is to ensure the quality of the product before moving to beta testing
* Long execution cycle may be required for alpha testing.

1. Beta testing?

* It’s performed at client location
* During this test we check reliability, Security, Robustness and quality of product.
* If we observe any issue in this testing will be fix these issues in further versions of the product.
* Ensure the product is ready to real time users.

1. GUI testing?

There are two types of interfaces in a computer application.

**Command line:** - if you type text computer response to that text.

**GUI Interface: -** [graphical user interface] you can interact with the computer using images rather than text.

GUI testing involves checking the screens with the controls like menu, buttons, icons, and bars, tool bars, menu bar, dialog box, and windows etc..

16. Testing levels?

1. Unit testing
2. Integration testing
3. System testing
4. Acceptance testing