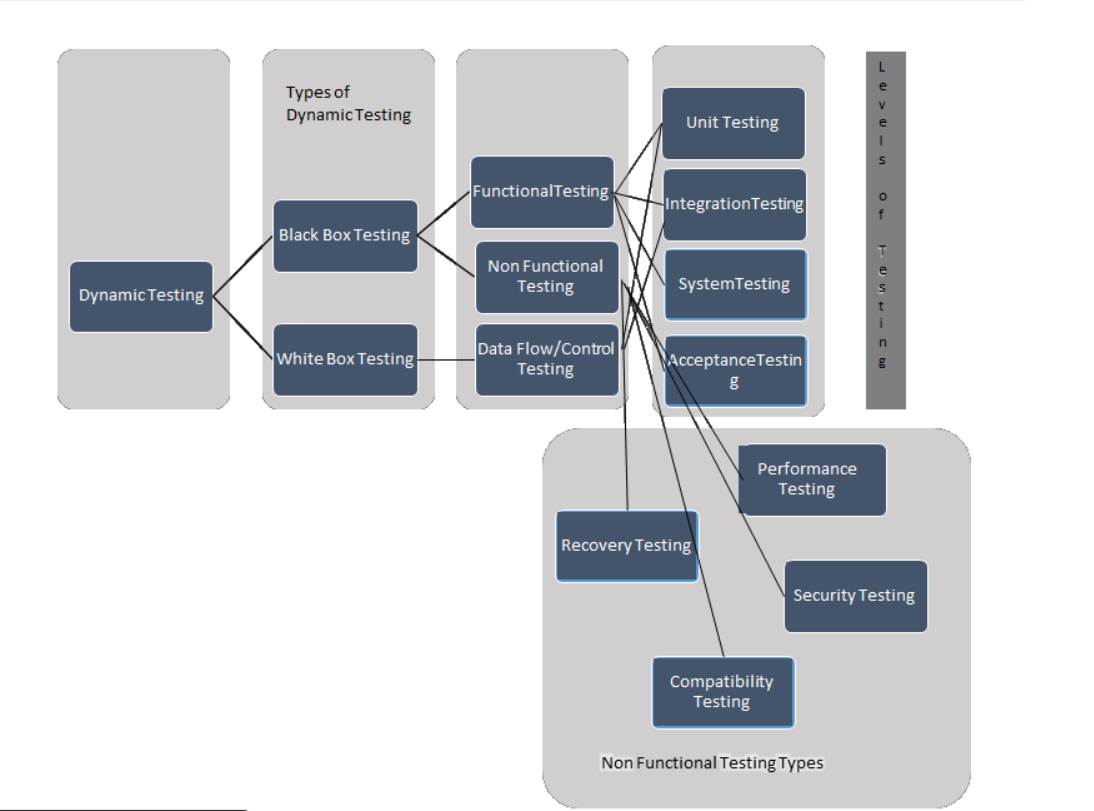
We will discuss only those testing which we are gonna use here



301 testing

Black Box

White box Testing

Unit testing tick👍

Database testing cross👎

The goal of whitbox testing is to verify all decision branches ,loops,statements in code

Security testing

Finding and estimating the dead code

**Security testing penetration testing proxy testing**

**URL encryption**

**Burp Suite**

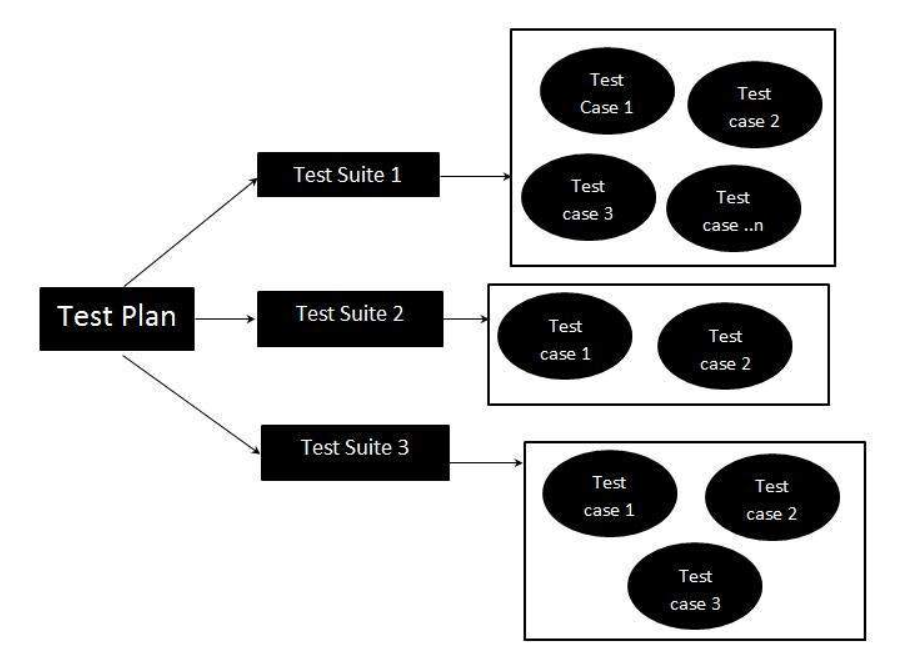
**AUT is the built or version**

**0.1** changes are there but not apt for delivery(development stage) **and 1.0** (1.x initial) after release

Regex regular expression

Typing doesn’t do this but copying is

**Test Cases**



**Test Suite**

Test suite is a container that has a set of tests which helps testers in executing and reporting the test execution status. It can take any of the three states namely Active, Inprogress and completed.

Test suites are created based on the cycle or based on the scope. It can contain any type of tests, viz - functional or Non-Functional.

**Bugs**

When something doesn't match with req

Go acc to SRS

Take priority for functional bugs(depends from client to client)

**priority**

-critical,blocker,high

**Severity**

Example - if company logo is wrong

High Severity(Its )

Low priority(not impacting the application)

Make sure whether its a api bug or application bug,to identify which bug it is ,test it independently

Advice-go for priority over severity and if time allows go for severity

Don’t get emotionally attached to a bug

Cross question scrum master for non addressal of bugs

Create issue---> project---->Summary(crisp and detailed)

Issue type Description(steps to produce the bug and actual vs expected data)

Components-module

Epic

Test case link

Spawn-if that functionality was working in previous release but not in this release

Hot fix-The term hotfix is generally used when a client has found an issue or market feature within the current release of the product and can not wait to be fixed until the next big release.needs to be addressed by scrum master

Environment-

Cross browser,device,resolution compatibility.

In real life we mark a bug as verified instead of closed and close it after the release.

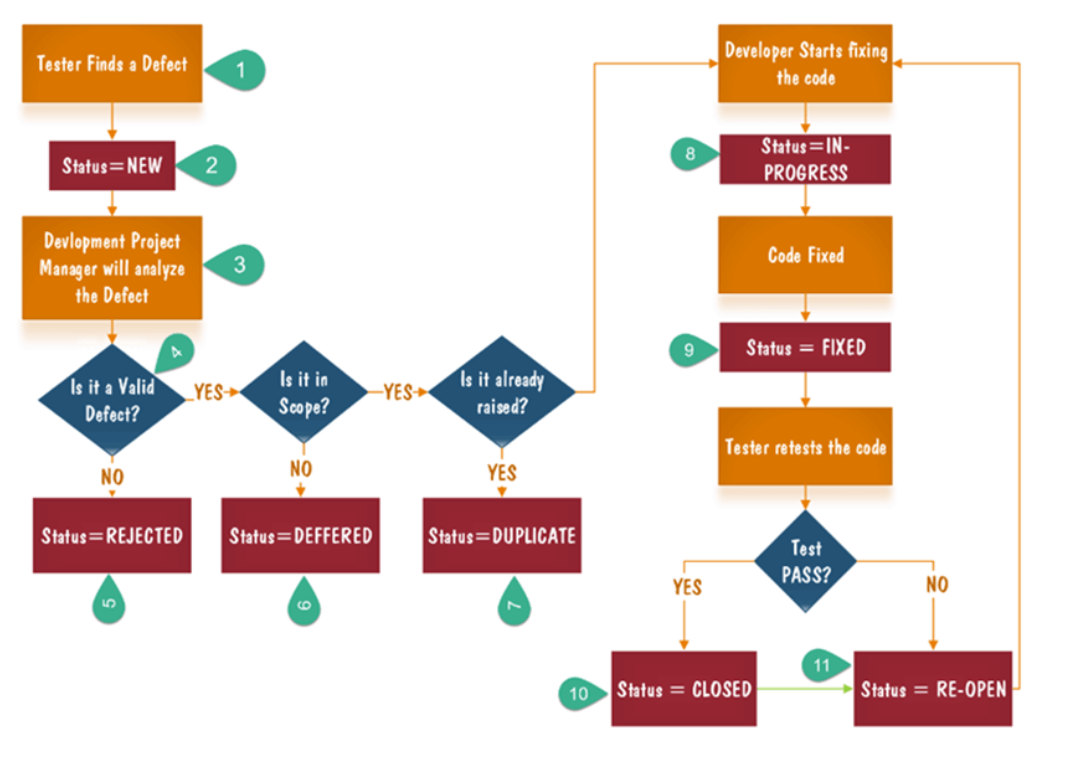
Phptravels.com

Think of having the bugs and not as everything is fixed

Never lose p1 or p2 bugs

Day 4

Bug life cycle



Deferred- not part of current release or we need to analyse it

**Day 5**

Sql commands

Stored Procedure

Query response time should also be noted,

API testing-soap API and REST API

Postman

API are more secure than ….

POST PUT GET PATCH

API testing ----> we should first check for http code

Can do load test(Never run load test prod. Env.)

Proxy Tools -

Business logic

**Day 6**

Bugs not that are not working fine….but not working for you

Source Control-Version management system

The 2 types of bugs are -

1.Reproducible bugs- occuring in all condition

2.Happens once in a time-

Application Logistics or log || focus on effect of bug in revenue can be used to prove it is a bug during conflict of tester and developer

Product owner has final say in whether its a bug or not

Memory Usage-

Know about mitigation risk(plan to overcome risk, plan to avoid risk) - Know about the s/w crashing alternatives : as stated in test plan

**Learn To Do Critical Testing**

Look for critical function of the apps

Know about the competitors also

Test plan

Test case

Bug report

Test report: what areas you have done.

Critical Feature **|** Benefit of testing this feature **|** test Cases

Try to identify and link these 3 heading to make the test cases more effective.

**DAY 8**

Bootstrap loader-More formally called an Initial Program Loader (IPL), a small section of executable code, permanently stored in a ROM chip within the computer, whose sole function is to load into memory from DISK

Wrapper classes

Reflection

Assertions and verification

Soft verification

Hard verification

Verify username by using regular expressions (pattern and matcher)

How selenium works internally

Runtime polymorphism

Javascriptexecutor

1.type your name on google

2.Fetch all url from page,find count and print it in notepad or any editor

3.Using (Http client) find which links are broken don't forget verification,avoid using thread.sleep,use try and catch (your coding will be judged)

testng(optional)

getTagNameby ‘td’ not ‘tr’

Install new software

Robot class,select classes

Avoid hard coded and do soft coded reference

**DAY 10**

Alert api by selenium

DOM  
Git checkout(tag)(tag)

Git checkout -bNewBranchNameT

master---> develop

Git stash

Git fetch and pull

Uuid

**Day 7**

Demo env

↓

Beta env

↓

Production env

Insprint automation

Try to write script that is skillable,robust and can work for every device,require less maintenance

Create pipeline in jenkins ,

Can we do work of put by post and vice versa

**POSTMAN**

Automation life cycle

↓

Automation feasibility analysis

↓

Test plan and design

↓

Environment set

↓

Test script creation

test script exe

Generate Test results/analysis

Xpath

Locators

Create regular exp for login and email

Selenium webdriver-

Json wire protocol

<https://selenium.dev/>

<https://www.toolsqa.com/>

gekco driver for firefox

Page load time out for web page loading

Implicitly wait is applied globally for which driver is interacting,it is dynamic wait ie if timeout is mentioned for 20 seconds, and page will load within 2 seconds , it will ignore 18 seconds

Webdriver wait is actually explicit wait

**EXPLICIT WAIT**

Public static void sendkeys(WebDriver driver,WebElement element,int timeout,String value){

new WebDriverwait(driver,timeout).untilExpectedConditions.visibilityOf(element));

element.sendKeys(values);

}

WebElement firstname=byelementbyid(“firstname”);

sendKeys(driver,firstname,10,’Tom’);

Avoid using implicit wait for large applications

DAY 11

Data driven testing

Tool tip (do on tools qa)

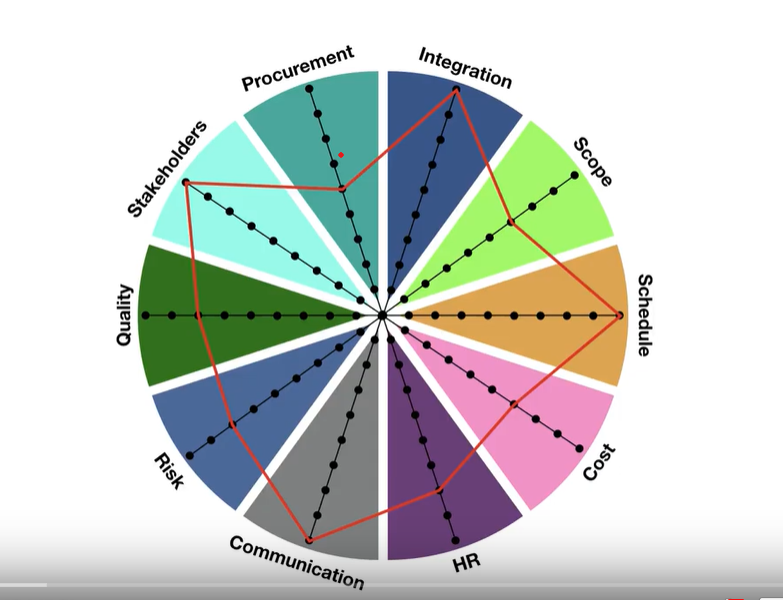
Constructor chaining

getAttribute

Captcha

POM  
testNg

**Project Master**



Scrum Master (should be influential,collaborative,leadership,knowledgeable)facilitator and provide assistance for team and the product owner

* Resolution of conflict, obstacles
* Tracking and monitoring
* Maximizing productivity of the team
* lead the meeting resolve the issues
* Communication and reporting results

Daily Scrum or stand ups

* To develop strategy
* Maintain transparency
* To get on same page
* What is completed since last meeting
* What have you desired to complete till next meeting
* Any impediments

## What Does the Product Backlog

* New features
* Infrastructure updates
* Changes to existing functionalities
* Bug fixes

## Who Owns the Backlog?

While the entire cross-functional agile team works together on the backlog, the [product owner](https://www.productplan.com/product-owner/) owns it. In most cases, the product owner (or product manager) holds responsibility for organizing and maintaining the product backlog.

Product Backlog (Everything that needs to be done in project )

Sprint Backlog (Every that needs to be done in a sprint)

User Story

A user story is a tool used in Agile software development to capture a description of a software feature from an end-user perspective

As a user<type of user>

I want to<Actions>

So that <objective>

* Burn up Charts - amount of work completed ------|
* Burn down Chart - amount of work remained to complete the project ------|

Scrum Retrospective Meeting

* Discuss current sprint activity
* Plan further improvements
* Identify what went well, what could have been better