TEST PLAN

A test plan is a detailed document which describes software testing areas and activities.

Test Plan – is a document which derives all future activities of the project. All future testing activities is planned and put into a document and this document is known as Test Plan. It contains – number of engineers needed for the project, who should test which feature, how the defects must be communicated to the development team, when we should start and finish writing test cases, executing test cases, what are the types of testing we use to test for the application etc.

The test plan is a template for conducting software testing activities as a defined process that is fully monitored and controlled by the testing manager. The test plan is prepared by the Test Lead (60%), Test Manager(20%), and by the test engineer(20%).

Test plan varies from company to company as well as project to project

Test plan components or attributes

1. **Objectives:** It consists of information about modules, features, test data etc., which indicate the aim of the application means the application behavior, goal, etc

2. **SCOPE :-**
   It contains information that needs to be tested with respect to an application. The Scope can be further divided into two parts:

   In scope: These are the modules that need to be tested in-detail
   Out scope: These are the modules, which need not be tested rigorously.

   2.1 Features to be tested
   For ex,
   Compose mail
   Inbox
   Sent
   Items
   Drafts

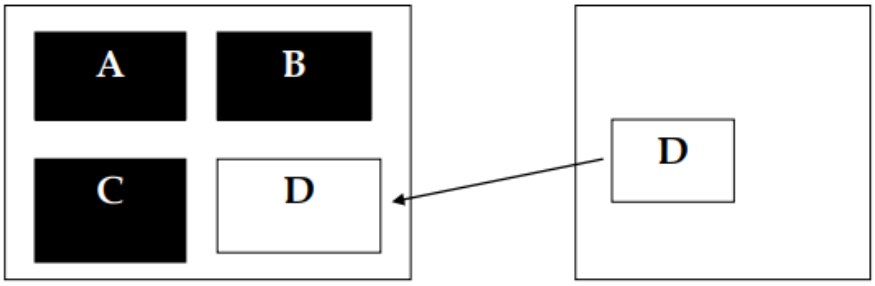
2.2 Features not to be tested
For ex,
Help
…
…

i.e, In the planning stage, we decide which feature to test and which not to test due to the limited time available for the project.

How do we decide this (which features not to be tested) ?

a) “HELP” is a feature developed and written by a technical writer and reviewed by another technical writer. So, we‟ll not test this feature.

b)



Let us consider that an application with features A, B, C and D are to be developed as per requirements. But then, D has already been developed and is in use by another company. So, the development team will purchase D from that company and integrate with the other features A, B and C.

Now, we will not do functional testing on D because D is already in use in the market. But we will do integration testing and system testing between A, B, C and D because the new features may not work with D properly

c) In the 1st release of the product – features that have been developed are – a, b, c, d, e, f, g, h, … m, n, o. Now, the customer gives requirements of new features to be built for enhancement of the product during the 2nd release. The features to be developed are – p, q, r, s, t.
During test plan, we write scope,

Scope
Features to be tested
P, Q, R, S, T (new features) A, B, C, D, E, F

Features not to be tested
G, H, I, J, … N, O

Thus we first test new features and then test old features which might be affected by building the new features i.e, impact areas. We do regression testing for A, B, C, … F.

3) TESTING METHODOLOGIES (Types of Testing)

Depending upon the application, we decide what type of testing we do for the various features of the application. We should also define and describe each type of testing we mention in the testing methodologies so that everybody (dev team, management, testing team) can understand, because testing terminologies are not universal.

For example, we have to test www.shaadi.com, we do the following types of testing,
Smoke testing Functional testing Integration testing System testing
Adhoc testing Compatibility testing Regression testing
Globalization testing Accessibility testing Usability testing
Performance testing

For standalone applications, like AutoCad, we do the following types of testing,
Smoke testing Functional testing Integration testing System testing
Adhoc testing Compatibility testing Regression testing
Globalization testing Accessibility testing Usability testing
Reliability testing Recovery testing Installation / Uninstallation testing

4) APPROACH

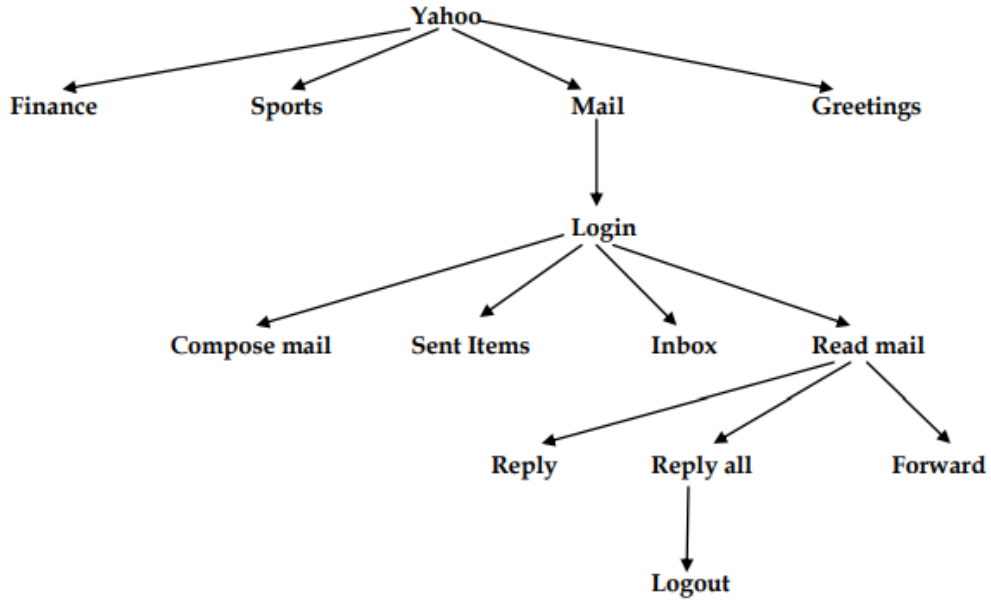
The way we go about testing the product in future, a) By writing high level scenarios b) By writing flow graphs

a) By writing high level scenarios for ex, we are testing www.yahoo.com

i) Login to Yahoo – send a mail and check whether it is in Sent Items page
ii) Login to …….
iii) ….. …..

This is written only to explain the approach to be taken to test the product. Only for the critical features, we will write a few very high level scenarios. We don"t cover all scenarios here. That is the job of the respective Test Engineers for whom the features have been allocated.

b) By writing flow graphs



We write flow graphs because of the following advantages, i. Merging is easy ii. Coverage is easy Flow graphs are written because writing high level scenarios is time consuming

The approach can be classified into two parts which are as following:
Top to bottom approach
Bottom to top approach

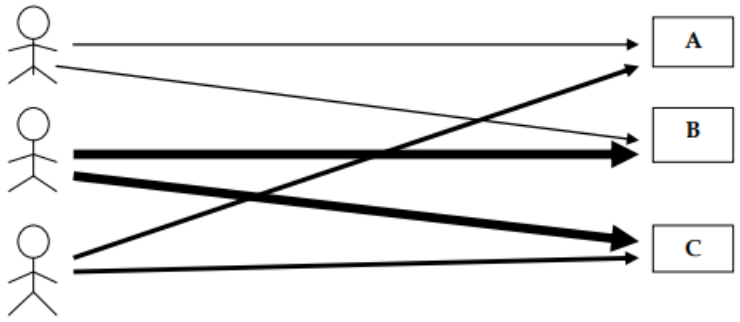
5) ASSUMPTIONS
It contains information about a problem or issue which maybe occurred during the testing process and when we are writing the test plans, the assured assumptions would be made like resources and technologies, etc.

6) RISKS
If the assumptions fail, risks are involved
For example, the effect for an application, release date becomes postponed.

7) CONTINGENCY PLAN OR MITIGATION PLAN OR BACK-UP PLAN

To overcome the risks, a contingency plan has to be made. Atleast to reduce the percentage from 100% to 20%

Let us consider an **example for 5, 6, 7**



In the project, the assumption we have made is that all the 3 test engineers will be there till the completion of the project and each are assigned modules A, B, C respectively. The risk is one of the engineers may leave the project mid-way.

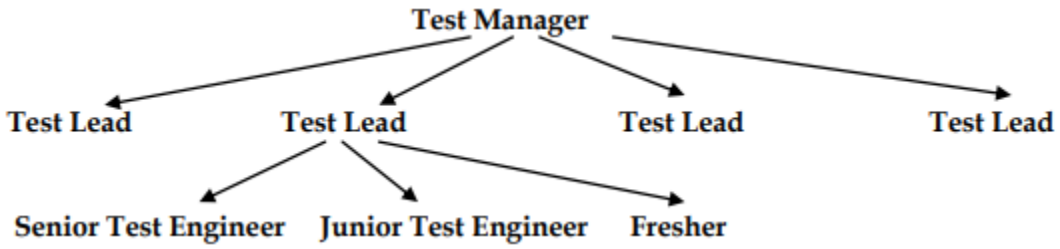
Thus, the mitigation plan would be to allocate a primary and secondary owner to each feature. Thus, one engineer quits – the secondary owner takes over that particular feature and helps the new engineer to understand their respective modules.

Always assumptions, risks, mitigation plan are specific to the project.

The different types of risks involved are,

- Resource point of view
- Technical point of view
- Customer point of view

8) ROLES AND RESPONSIBILITIES



When a Big project comes, it"s the Test Manager who writes the test plan. If there are 3small projects, then Test Manager allocates each project to each Test lead. The Test lead writes the test plan for the project which he is allocated.

8.1 Test Manager

- Writes or reviews test plan
- Interacts with customer, development team and management
- Sign off release note
- Handle issues and escalations

8.2 Test Lead

- Writes or reviews test plan
- Interacts with development team and customers
- Allocates work to test engineers and ensure that they are completing the work within the schedule
- Consolidate reports sent by Test Engineers and communicate it to development team, customers(if it is a time & material project) and management

8.3 Test Engineer 1

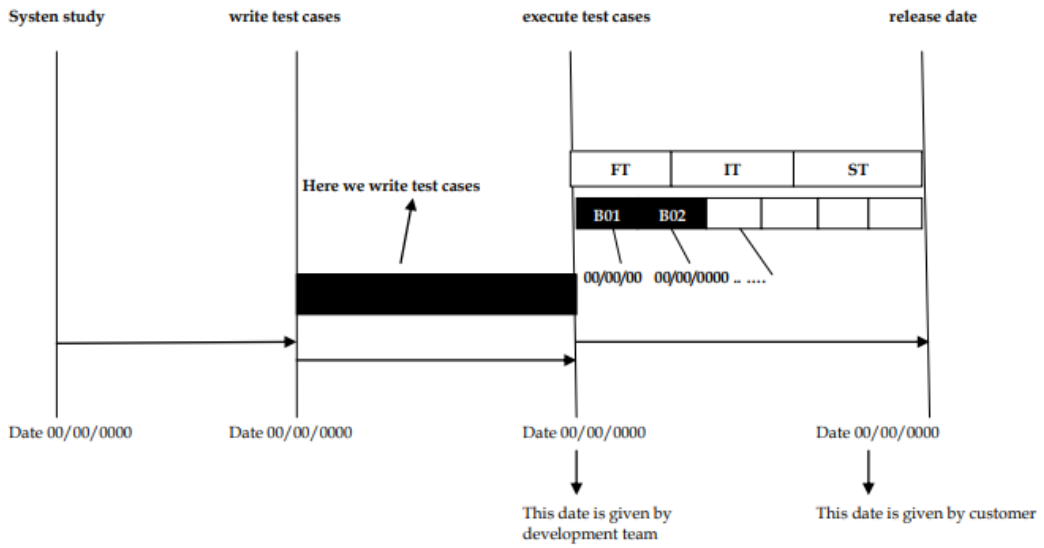
- Review test plan
- Write test cases for trend analysis
  Asset survey
- Write traceability matrix
- Review test cases written for sales and purchase modules
- Execute test cases written for trend analysis, asset survey, registration (old module developed in previous release. Adding trend analysis and asset surveys has affected. Old module has been affected. So do regression testing)
- Perform compatibility testing using Internet Explorer, Mozilla Firefox and Google Chrome in Windows XP and Windows Vista
- Prepare a test execution report and communicate it to the Test lead.

8.4 Test Engineer 2

- Set up and install the product
- Identify test cases to be automated
- Automate identified test cases using QTP
- Execute and maintain automation scripts

9) Schedules:-

- This section contains – when exactly each activity should start and end? Exact date should be mentioned and for every activity, date will be specified.



Thus, as we can see from the above figure – for every specified activity, there will be a starting date and closing date. For every build, there will be a specified date. For every type of testing for each build, there will be a specified date.

10) Defect Tracking

In this section, we mention – how to communicate the defects found during testing to the development team and also how development team should respond to it. We should also mention the priority of the defect – high, medium, low.

11) Test Environment (mentioned configuration is just for an example)

11.1 Hardware

11.1.1 Server :- Sun Starcat 1500 (this is the name of the server from which testing team take the application for testing)

11.1.2 Client :- 3 machines with following configurations,

Processor : Intel 2GHz

RAM : 2GB

…

…

(this gives the configurations of the computers of the Test Engineers i.e, the testing team)

11.2 Software

11.2.1 Server

OS : Linux

Web Server : TomCat

Application Server : Websphere

Database Server : Oracle (or) MS – SQL Server

(the above servers are the servers which the testing team will be using to test the product)

11.2.2 Client

OS : Windows XP, Vista, 7

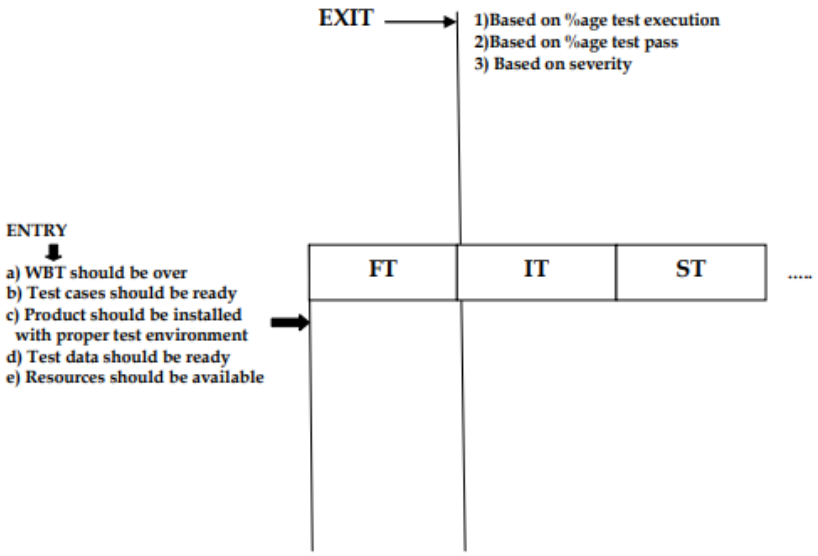
Browsers : Internet Explorer, Internet Explorer 7, Internet Explorer 8, Mozilla FireFox, Google chrome

(the above gives the various platforms and browsers in which the testing team will test the product)

11.3 Procedure to install the software

(Development team gives instructions on how to install the software. If they have not yet given the procedure, then in the test plan, we just write it as TBD – to be decided)

12) Entry and Exit Criteria



Before we start with Functional Testing, all the above entry criteria should be met.
After we are done with FT, before we start with Integration Testing, then the exit criteria of FT should be met. The percentage of exit criteria is decided by meeting with both the development and test manager. They compromise and conclude the percentage. If the exit criteria of FT is not met, then we cannot move onto IT. Based on severity of defects means,

The testing team would have decided that in order to move onto the next stage, the following criteria should be met,

- There should not be more than 20critical bugs
- There should not be more than 60major bugs
- There should not be more than 100minor bugs
-

If all the above are met, then they move onto the next testing stage.
But the problem with the above method was,
21 critical, 50major, 99minor – cant exit because there are more than 20 critical bugs.
10critical, 90major, 200 minor – can exit. But the 10 critical bugs can affect the product.
Thus, they came up with the concept of “weight of defects”. i.e, 3major = 1 critical, 5minor – 1critical and total critical should not be more than 60.

So, for, 21 critical – 21
50 major – 16critical
99 minor – 19critical
Totally there are 56critical bugs, so we can move onto the next stage. But for the 2nd example, we cannot move on.

Entry criteria for IT :
- should have met exit criteria of FT
(remaining all are same as entry criteria of FT)

Exit criteria for IT :
All points are the same as exit criteria for FT.
But if the %age pass for FT is 85%, then the %age pass for IT should be 90% - because as we reach the later stages of testing, we expect the number of defects to be less.

Entry criteria for ST :
- exit criteria of IT should be met
- minimum set of features must be developed
- test environment should be similar to production environment
(remaining all are same as of IT)

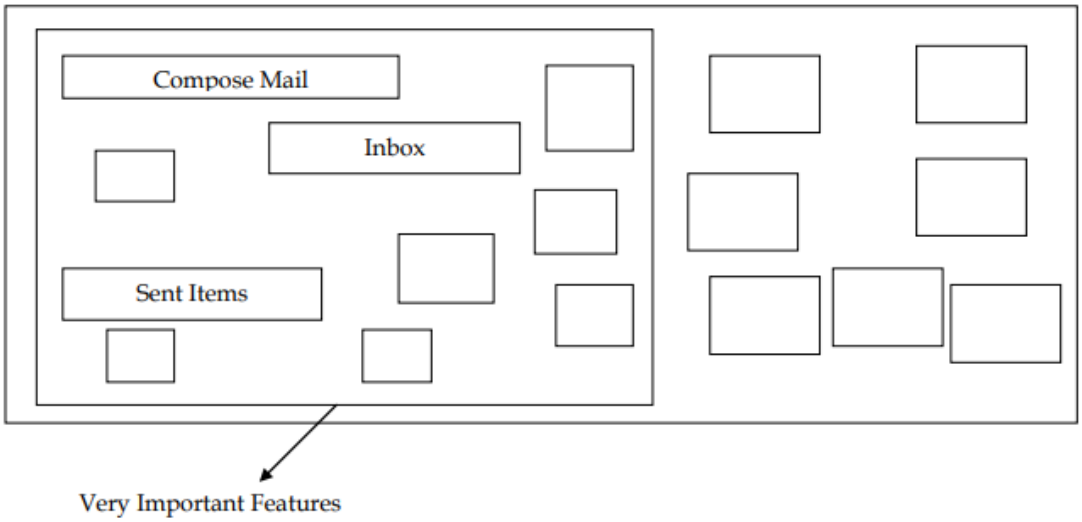
Exit criteria for ST :
- everything remains same as of above, but the pass %age is now 99% - there should be 0 critical bugs. There could be some 30major and 50minor bugs. If all this is met, then product can be released.

Note : All the numbers given above are just for example sake. They are not international standard numbers!!!.

13) Test Automation
13.1 Features to be automated
13.2 Features not to be automated
13.3 Which is the automation tool you are planning to use
13.4 What is the automation framework you are planning to use

We automate the test cases only after the 1st release

13.1 On what basis do we decide which feature to be automated ?



If the features are very important and need to be repeatedly tested, then we automate that feature. Because manually testing the feature takes longer time and also becomes a tedious job.

13.2 How to decide which features are not to be automated ?

- For example, “HELP” is a feature that is not repeatedly tested – so we don‟t have to automate it.
- If the feature is unstable and has lot of defects – we will not automate because it has to be tested repeatedly manually
- If there is a feature that has to be repeatedly tested, but we are predicting a requirement change for that feature – so we don‟t automate it as changing the manual test case is easier than changing the automation script.

14) Deliverables

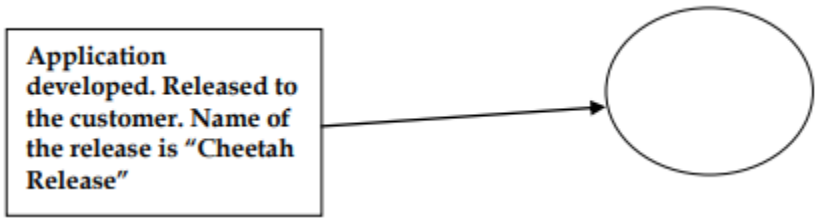
It is the output from the testing team. It contains what we will deliver to the customer at the end of the project. It has the following sections,

14.1 Test Plan
14.2 Test Cases
14.3 Test Scripts
14.4 Traceability Matrix
14.5 Defect Report
14.6 Test Execution Report
14.7 Graphs and Metrics
14.8 Release Note

Metrics

Module Name	Critical		Major		Minor	
	Found	Fixed	Found	Fixed	Found	Fixed
Sales	40	36	80	30	90	15
Purchase	..	…	…	…	…	…
Asset Survey	…	…	…	…	…	…

14.7 Release Note



The product is developed and tested and released to the customer. The name of the release is "Cheetah Release".
The release note contains,
1) List of pending/open bugs
2) List of features added, modified or deleted
3)Platforms(OS, Browsers, Hardware) in which the product is tested
4) Platforms in which the product is not tested
5)List of bugs fixed in current release which were found in previous release production

Let us consider that Cheetah release is the 2nd release of the product after the 1st release Tiger release. Some of the bugs found in the 1st release has been fixed in the 2nd release. Also a list of features which have been added, modified and deleted from the 1st release to the 2nd release will be mentioned here.

6) Procedure to install the software
7) Version of the software

Release Note is a document prepared during release of the project and signed by test manager

15)TEMPLATES

This section contains all the templates for the documents which will be used in the project. Only these templates will be used by all the test engineers in the project so as to provide uniformity to the entire project. The various documents which will be covered in the Template section are
Test Case
Traceability Matrix
Test Execution Report
Defect Report
Test Case Review Template

This is how a Test Plan document looks like,

Pg 1

CBO_Testplan

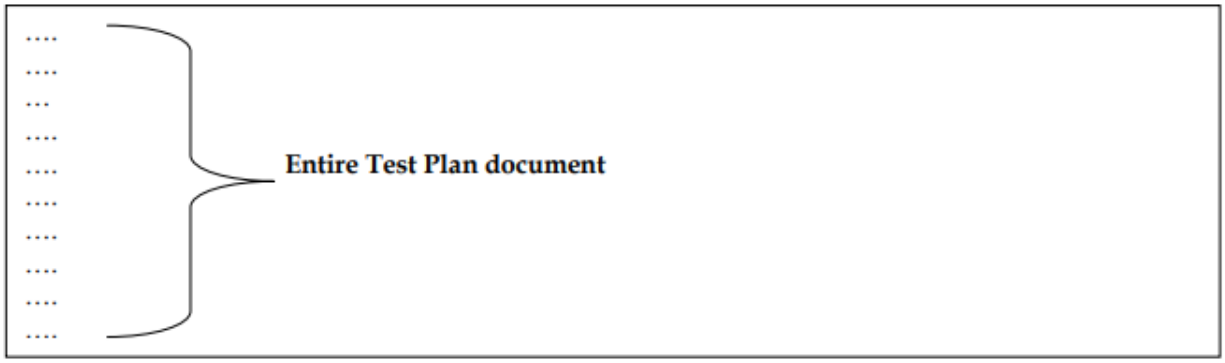
Revision History

Version	Author	Reviewed By	Approved By	Comments	Approval Date
1	..	...	Name of manager	Version 1.0 is developed	dd/mm/yyyy
1.1	..	..	" "	Version 1.1 is developed. XYZ feature is added	dd/mm/yyyy
...	...	...	...	...	...
...	...	...	...	...	...

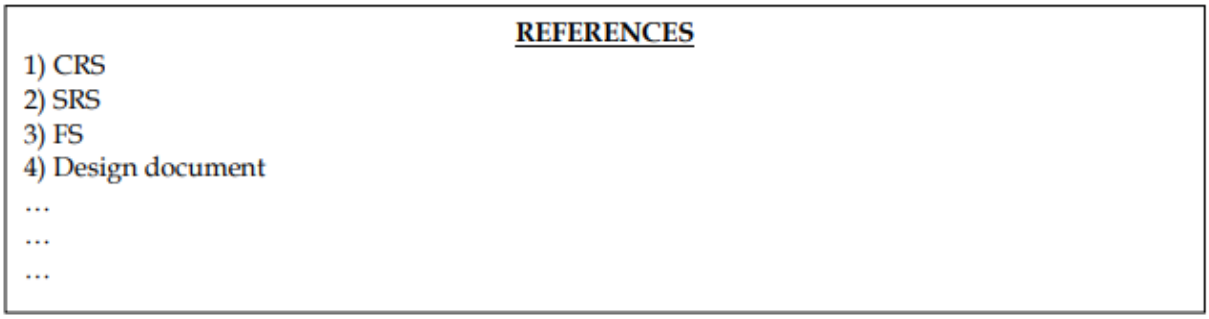
Pg 2

TABLE OF CONTENTS

Pg 3 – Pg 19



Pg 20



In the 1st page – we initially fill in only the version(write as Draft 1.0), author, comments and reviewed by. Later on when the manager approves it, we fill in the Approved by and approval date and also remove (Draft) written in the version column.

Generally Test Engineers only review it and the test plan is approved by the Test Manager.

When new features come, we modify the Test Plan. Change the version and whichever features need to be changed. Again it is reviewed, updated and approved by the manager. Test Plan must be updated whenever changes happen
References (Pg 20) contains all the documents that are used to write the test plan document.

Who writes the Test Plan ?
Test Lead – 60%
Test Manager – 20%
Test Engineer – 20%

Thus, we can see from above – in 60% of projects, Test plan is written by Test Lead and so on as shown above.

Who reviews Test Plan ?
Test Engineer

Test Lead
Test Lead
Customer
Development team

Test Manager looks at the Test plan from the customer point of view.
Test Engineer looks at the Test plan from his module point of view.

Who approves the Test Plan ?
Test Manager
Customer

Who writes Test Cases ?
Test Engineer
Test Lead

Who reviews Test Cases ?
Test Lead
Test Engineer
Development Team
Customer

Who approves Test Cases ?
Test Lead
Test Manager
Customer

Test Plan Guidelines

- Avoid overlapping and redundancy.
- If you think that you do not need a section that is already mentioned above, then delete that section and proceed ahead.

- Be specific. For example, when you specify a software system as the part of the test environment, then mention the software version instead of only the name.
- Avoid lengthy paragraphs.
- Use lists and tables wherever possible.
- Update plan when needed.
- Do not use an outdated and unused document.

Importance of Test Plan

- The test plan helps those people to understand the test details that are related to the outside like developers, business managers, customers, etc.
- The test plan helps in determining the necessary efforts to validate the quality of the software application under the test.
- Important aspects like test schedule, test strategy, test scope etc are documented in the test plan so that the management team can review them and reuse them for other similar projects.