## REQUIREMENTS COLLECTION / SYSTEM STUDY

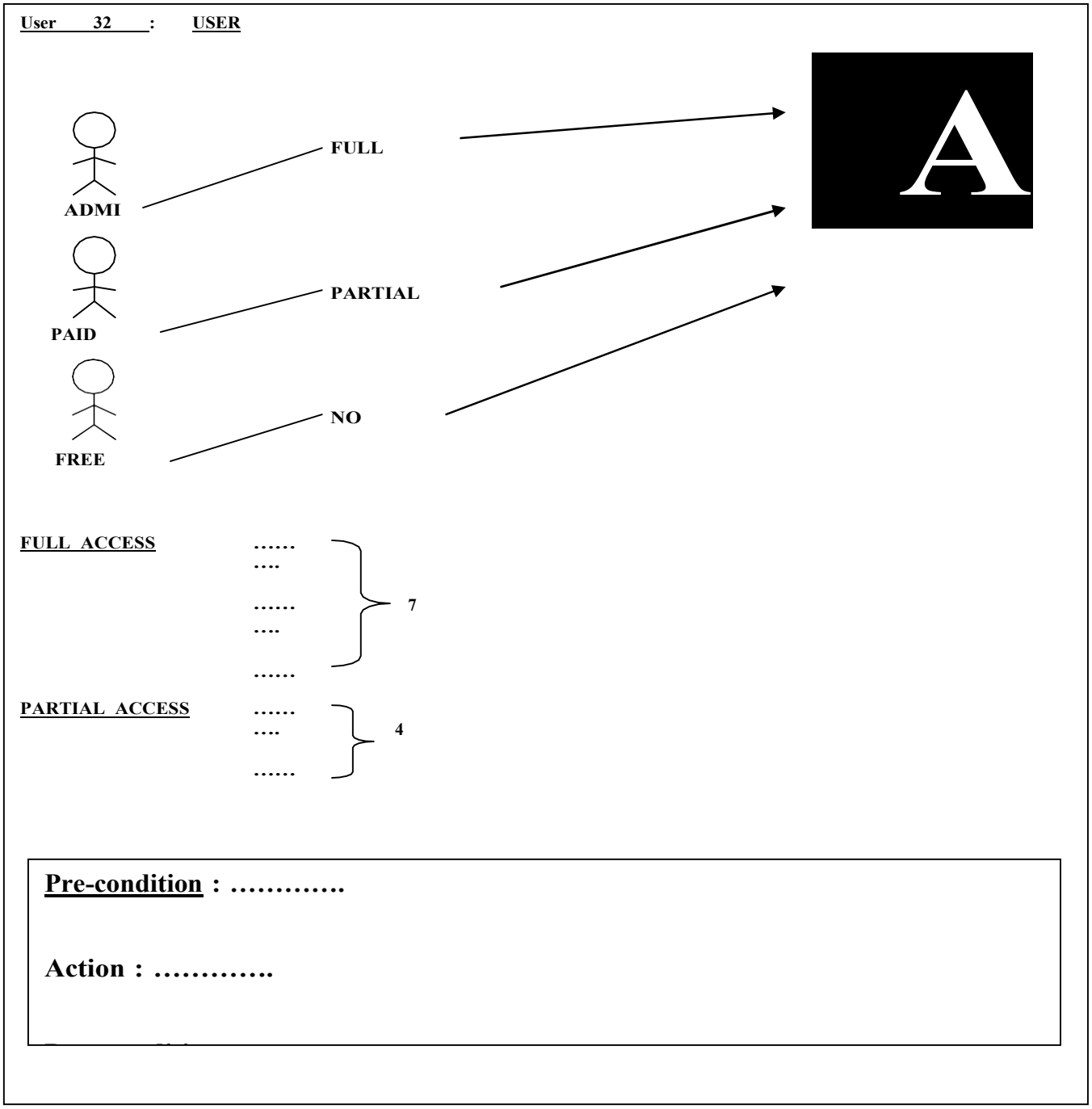
The requirements can be in any of the following forms,

- CRS (Customer Requirement Specification)
- SRS (System Requirement Specification)
- FS (Functional Specification)
- If we don‟t have requirements and if we are given only the application, then we do *exploratory testing*.
- Use case

***Use Case***

**Use case** is a pictorial representation of requirements. It explains how the end user interacts with the application. It gives all possible ways of how the end user uses the application.

Below is shown an example of how a **use case** looks like,




The above figure shows a sample **use case** of one of the requirements in the CRS.For the module A of the application, there are 7features.
Admin has access to all the 7 features. For a paid user – access to 4 features
For a free user – no access to any of the features.

**Ex** – for admin

*Precondition* – admin must be created

*Action* – login as paid user

*Post condition* – 4 features must be there

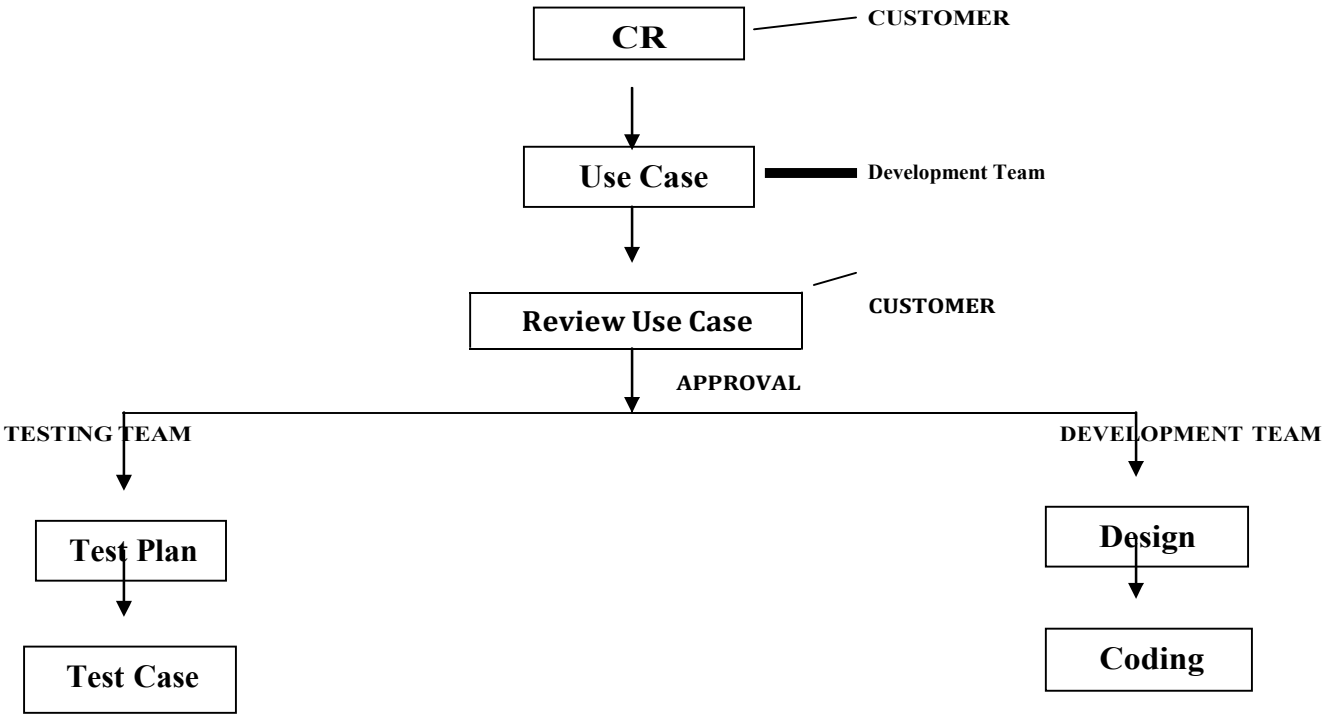
**Ex –** for free user

*Precondition* – free user must be created

*Action* – login as free user

*Post condition* – no features

**Who writes use cases**




Customer gives the CRS for the application to be developed. The development team write the **use case** for the CRS and the **use case** is sent to the customer for review. If the customer approves it, then the approved **use case** is sent to the development team for design and coding. The approved **use case** is also sent to the testing team who start writing test plan and later on start writing test cases for the features of the application.

**Difference between use case and prototype**

**Use case** – talks about how the product should work. It is a pictorial representation of the application and its various features and also how they should work.

**Prototype** – here, we will not see how the end user interacts with the application. It‟s just a screenshot of theapplication (exact image of the application)

**How developers develop *use cases***

Developers use standard symbols to write **use cases** for universal understanding. He uses UML – UnifiedModelling Language to develop **use cases**.
There are readymade tools to write use cases – like **Rational Rose**. It has readymade UML symbols – we can just drag and drop them to write **use cases** – developers use these symbols and write **use cases**.

**When you join a company, always ask for –**

- Requirements of the project
- Test plan of the project
- Test cases(existing) of the project
- Application which is to be tested.

## TEST PLAN

Test plan is a document which drives all future testing activities.
Test plan is prepared by **Test manager(20%)**, **Test Engineer(20%)** and by **Test Lead(60%)**.There are **15 sections** in a test plan. We will look at each one of them below,

***1) OBJECTIVE*** :- It gives the aim of preparing test plan i.e, why are we preparing this test plan.

***2) SCOPE :-*** This covers features to be tested and features not to be tested

**Features to be tested**

For ex, Compose
mailInbox
Sent Items
Drafts

**Features not to be tested**

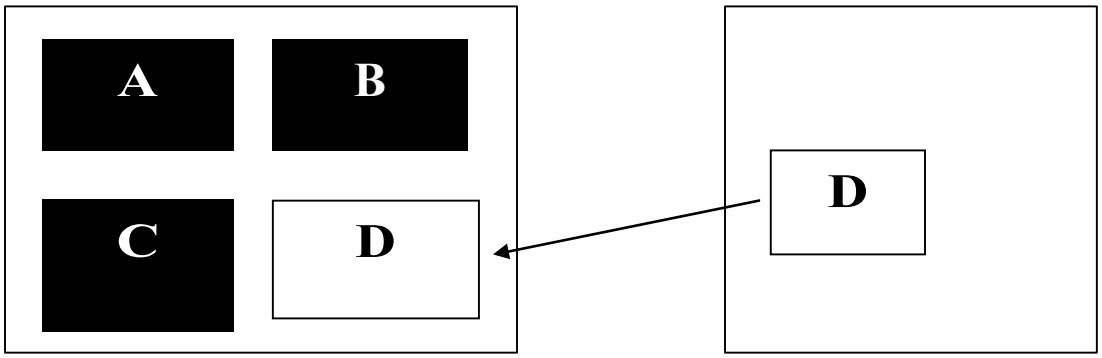
For ex, Help
…
…
…
…

i.e, In the planning stage, we decide which feature to test and which not to test due to the limited timeavailable 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 technicalwriter. So, we‟ll not test this feature.

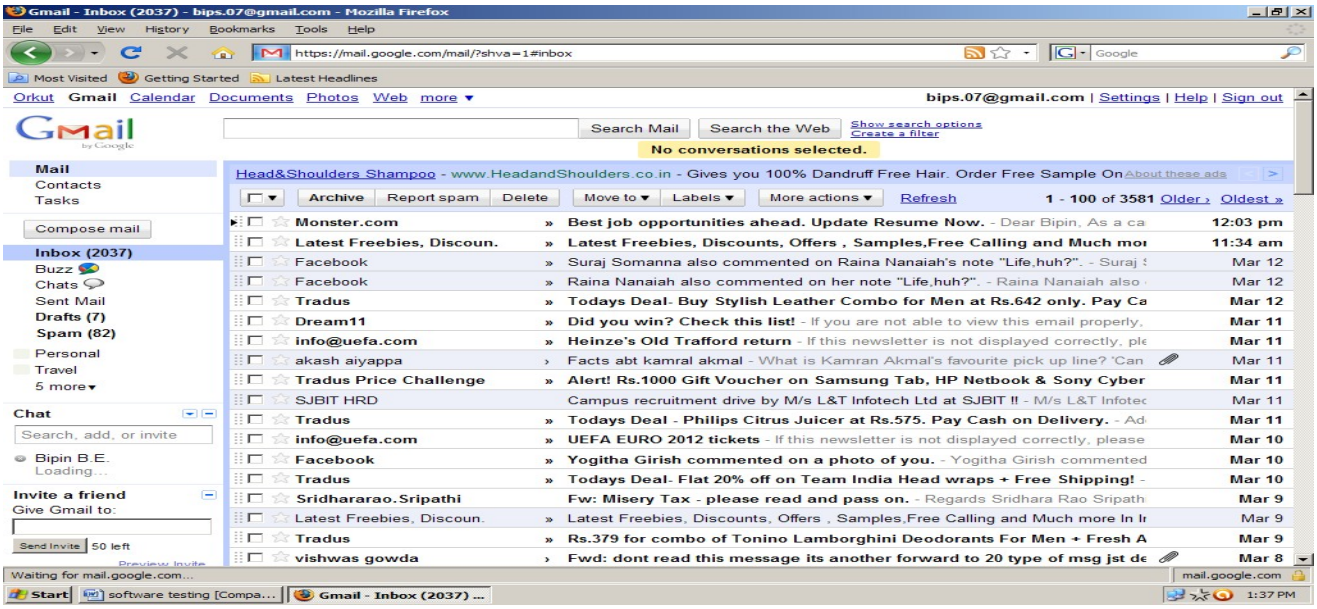
**b)**



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)**





The application might be having link to some other application. Here, our scope of testing is limited to,

- Whether link exists
- If it goes to homepage of the corresponding application when we click on the link.

Let us consider the **example** of Gmail. When we log into gmail, we see many links to other applications like

– FaceBook, LinkedIn, youtube, orkut etc. when we logged into Gmail and when we click on the orkut link – it must takeus to Orkut"s homepage.
Such features are called *Single sign-on feature* – it is a feature wherein 1 login allows access to multipleapplications.

**d)** 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 duringthe 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,

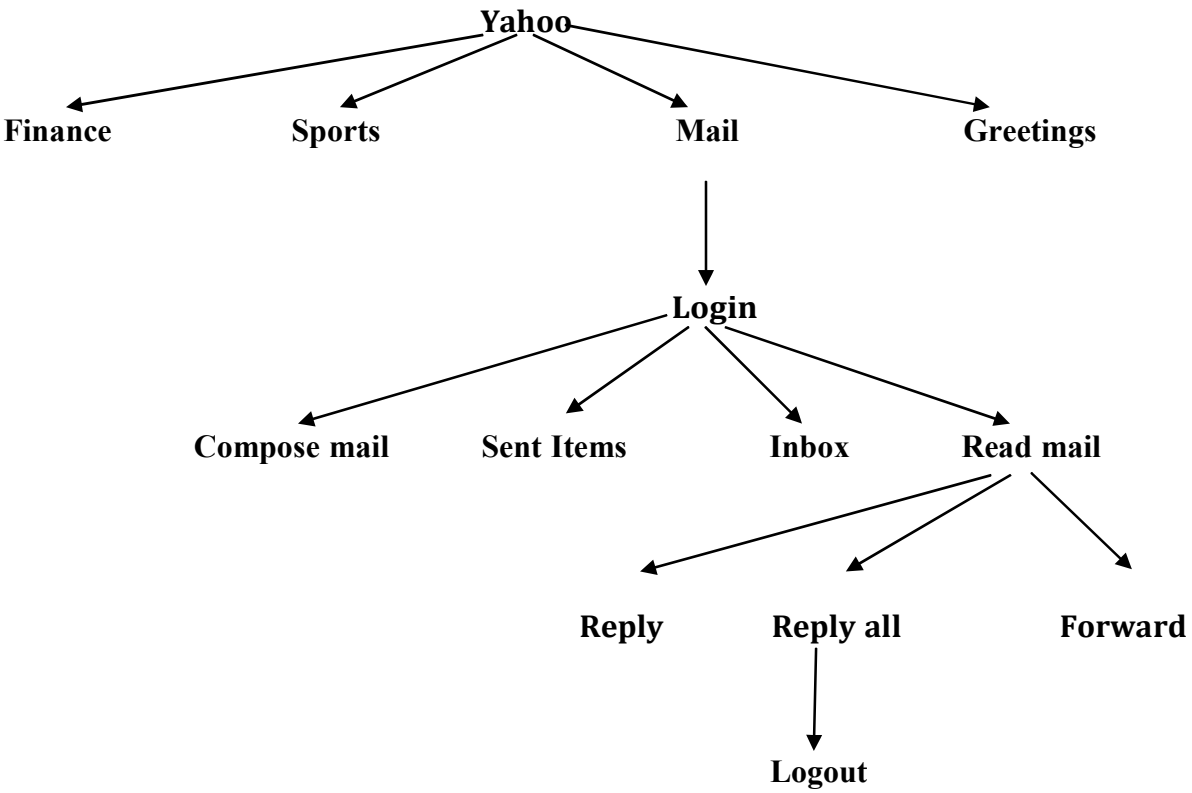
a) By writing high level scenarios
b) By writing flow graphs

***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.

***5) ASSUMPTIONS***

When writing test plans, certain assumptions would be made like technology, resources etc.

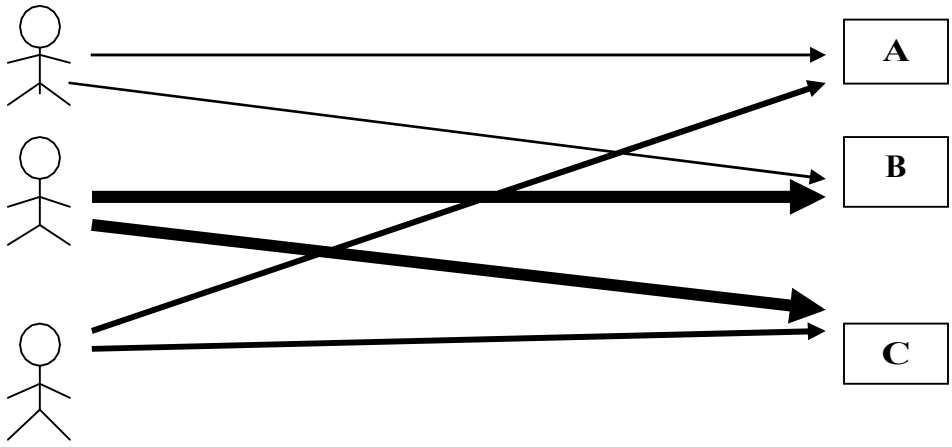
***6) RISKS***

If the assumptions fail, risks are involved

## *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%

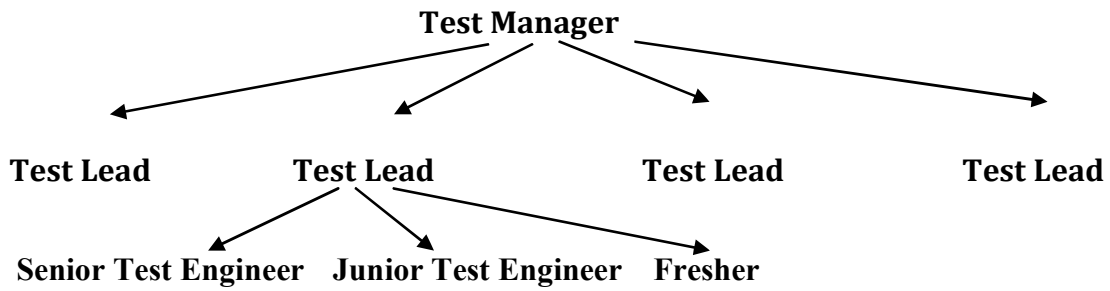
Let us consider an **example for 5, 6, 7**




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 writesthe test plan for the project which he is allocated.

**Test Manager**

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

**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(ifit is a time&material project) and management
- …
- …
- …

**Test Engineer 1**

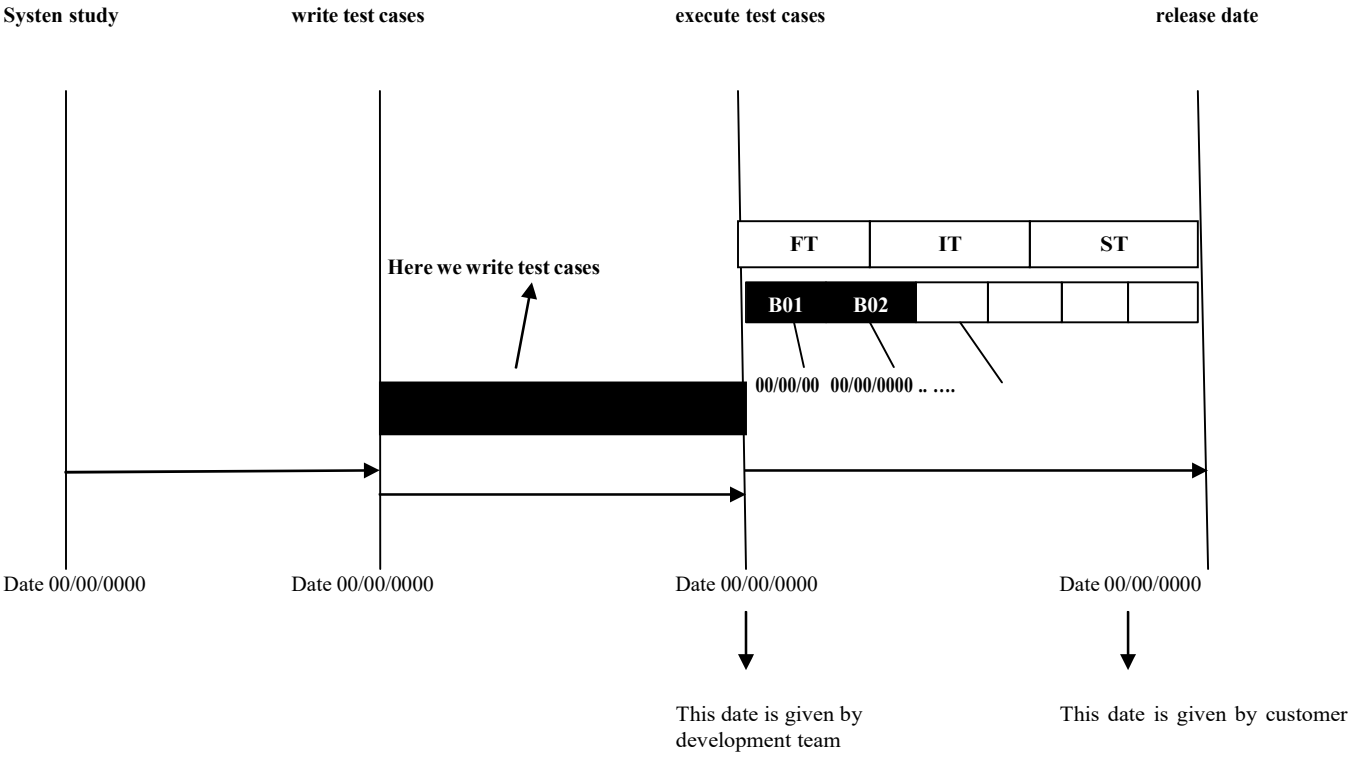
- Review test plan
- Write test cases for trend analysis
- Asset survey
- Write traceability matrix
- Review test cases written for sales and purchase modules
- Execute test cases written for trend analysis, asset survey, registration (old module developed in previous release. Adding trend analysis and asset survey 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 test execution report and communicate it to Test lead.
- ….
- ….
- …

**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 mentionedand 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.

**Procedure to track the defect**

….
….
….
….

**Defect tracking tool**

We mention the name of the tool we will be using to track the defects

**Severity**

*Blocker*

….

…. (Define it with an example in the test plan)

For ex, there will be bug in the module. We cannot go and test the other modules becausethis blocker has blocked the other modules.

*Critical*

…

… (define it with an example)

Bugs which affects the business is considered critical

*Major*

…

… (define it with an example)

Bugs which affects look and feel of the application is considered as major

*Minor*

…

… (define it with an example)

**Priority**

*High – P1*

…

*Medium – P2*

…

*Low – P3*

…

…

… *P4*

So, depending on the priority of the defect(high, medium or low), we classify it as P1, P2, P3, P4.

***11) Test Environment***

**Hardware**

*Server* :- Sun Starcat 1500

(this is the name of the server from which testing team take the application for testing)

*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)

**Software**

*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)

*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)

**Procedure to install the software**

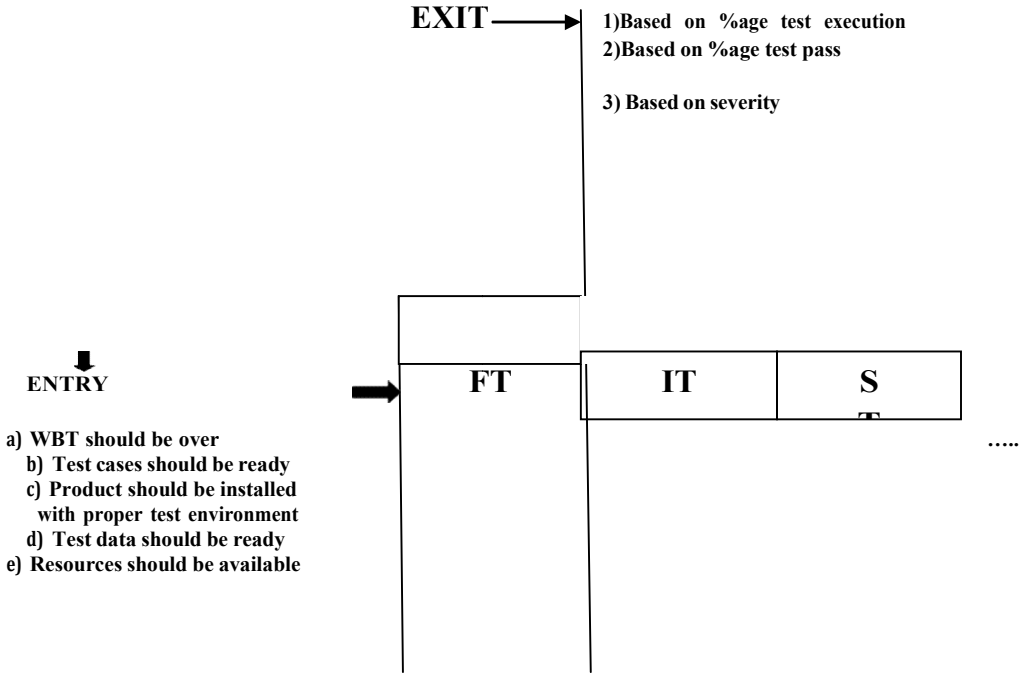
…

…

…

(Development team gives how to install the software. If they have not yet given the procedure, then in thetest 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 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 criteriashould 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 totalcritical should not be more than 60.

So, for,
21 critical – 21
50major –
16critical 99minor
– 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 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 laterstages 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. Therecould 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 standardnumbers!!!.*

***INTERVIEW QUESTIONS***

***Q) Customer gets 100% defect free product means,***
***a) Testing team is not good***
***c) Product is old***

***Ans) a) is correct. Testing team is not good – because – fundamentals of software testing says there is no product which has zero defects.***