**Software Test Life Cycle (STLC)**

STLC is a part of SDLC.

STLC is a procedure to test the software and it has different stages

|  |  |
| --- | --- |
| SDLC | STLC |
| 1. It is a procedure to develop the s/w 2. It has different stages like requirement collection, analysis, design, coding and testing, installation and maintainance. | 1. I t is the procedure to test the s/w 2. It has different stages like system study, prepare test plan, and write test cases, traceability matrix, test execution. |

System study

Prepare test plan

Write test cases

Prepare traceability matrix

Test execution

Defect tracking

Test execution report

Or

Test summary report

Retrospect meeting

Or

Project close meeting

Or

Postmortem meeting

1. System study

Here we read the requirement and try to understand the requirement if we have any quarries interact with developer or business analyst or customer and get it clarified.

1. Test plan

Test plan document which drives all the future testing activities.

This is where we decide

* How many engineers are needed
* What each engineers should do in each stages
* How we should install the product and set up the test environment
* This is where we decide what are the types of testing we should conduct on the product.
* What should be the testing approach
* This is where we decide what are the features to be tested and features not to be tested
* Schedule for every activity

1. Write test case

Once the planning job is done we start writing the test cases. When I say write test cases this itself involves different stages like

Identify all possible scenarios

Write test cases

Review test cases

Fix the review comments

Verify the test cases

Test case approval store it in test case repository

1. Prepare traceability matrix

Once the test cases are ready the biggest question is what is the guaranty that every requirement have got at least one test cases, to ensure that we prepare traceability matrix.

1. Test execution

Once we are sure that every requirement has got at least one test case we start our execution.

Test execution is a stage where in we test the product for more than 40-60 times this is where we conduct all the types of testing and this is where all the test engineer are productive for the organization and this is where the test engineer spend maximum time .

1. Defect tracking

We are executing the test mean defiantly we catch defect every defect that we find that should be track in a very organized manner which is nothing but defect tracking.

1. Test execution report or test summary report

It is a document which contains how many test cases are there, how many test cases are executed, what is the percentage of passed test cases and what is the percentage of failed test cases.

We prepare the document at the end of every test cycles and send it to development team customer and manager.

This is the end of project from testing point of view; from customer point of view we have one more activity that is retrospect meeting.

1. Retrospect meeting or project close meeting or postmortem meeting

Here the entire testing team meet and discuss about List of activities or good activities and list of mistakes or wrong activities followed in the entire test life cycle.

We document it and the document is called retrospect document when next project or the same project or in a next sprint when we are preparing test plan we refer the old retrospect document and prepare the test plan in such a way that old mistakes are not repeated and old good activities are again adopted.

At the end 2nd release once again we do retrospect meeting you come to know that number of mistakes that we are done are very less

Now we will look in detail each step of the STLC.

##### SOFTWARE TEST LIFE CYCLE (STLC)

Testing itself has many phases i.e is called as STLC. STLC is part of SDLC

Defect Life Cycle is a part of STLC

##### REQUIREMENTS

**Write Test Plan**

**System Study**

**Identify all possible test scenarios**

**Write Test Cases**

**Write test cases**

**Traceability Matrix**

**Review Test Cases**

**Test Execution**

**Fix Review Comments**

**Defect Tracking**

**Test Case Approval**

**Prepare test execution report(TER) and send it to customer**

**Store it in repository**

**Retrospect Meeting**

**Requirement** is the input for testing.

**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.

**Write test case** – we write test cases for each feature. These test cases are reviewed, and after all mistakes are corrected and once the test cases are approved – then they are stored in the test case repository.

**Traceability Matrix** – it is a document which ensures that every requirement has a test case .

Test cases are written by looking at the requirements and test cases are executed by looking at the test cases. If any requirement is missed i.e, test cases are not written for a particular requirement, then that particular feature is not tested which may have some bugs. Just to ensure that all the requirements are converted, traceability matrix is written. This is shown below,

***TRACEABILITY MATRIX***

|  |  |
| --- | --- |
| **Requirement Number** | **Test Case Name** |
| 1 | … |
| 2 | … |
| 3 |  |
| 4 | … |
| 5 | … |
| 6 |  |
| 7 | … |

For the requirements (3 and 6) for which test cases are not written, the cells are marked in thick border so that they are distinct and then test cases are written for them.

The Traceability Matrix is also known as RTM(Requirement Traceability Matrix) or CRM(Cross Reference Matrix).

**Defect Tracking** – any bug found by the testing team is sent to the development team. This bug has to be checked by the testing team if it has been fixed by the developers.

**Test Execution Report** :- Send it to customer – contains a list of bugs(major, minor and critical), summary of test pass, fail etc and when this is sent, according to the customer – the project is over.

TER is prepared after every test cycle and sent to development team, testing team, management and customer(depends if it is a *fixed bid project* or *time & material bid project*).

The last TER of the last test cycle is always sent to the customer. And this means that the project is over- according to the customer.

**Retrospect meeting** – (also called Post Mortem Meeting / Project Closure Meeting)

The Test Manager calls everyone in the testing team for a meeting and asks them for a list of **mistakes** and

**achievements** in the project.

|  |  |
| --- | --- |
| ***MISTAKES***(Mistakes in the Process) | ***ACHIEVEMENTS***(good process/procedure  followed) |
| 1) Review process is not good | 1)Last day of each cycle generally swap modules and do ad-hoc testing |
| …. | ….. |
| ….. | ….. |
| ….. | …. |

This is done by test lead or test manager. Here, the manager documents this retrospect meeting and stores it in QMS (Quality Management System). It is a folder, where inside this folder, there is another folder called Retrospect folder and here this excel sheet document is stored. When we get new project, while we write the test plan – we will open this retrospect file and will try and implement the good practices and correct the mistakes.

***Interview Questions***

***Q) What is test life cycle (OR) What is STLC (OR) I‟ll give you a product. What is the process you‟ll follow to test it (OR) In your current project, what is the process you are following to test the product.***

***Ans) everything has only one answer – STLC***

***Start from system study upto retrospect meeting. Briefly explain every stage and link every stage by saying “and then we move on”.***

***After retrospect meeting, we store it in QMS(Quality Management System). When a new requirements or project comes – during test plan stage – the test lead takes retrospect meeting excel sheet from the Test case repository and implements the good process followed and avoids the mistakes of the previous release. This procedure continues in the next release. Again the 2nd release retrospect meeting excel sheet is seen during test plan of a new project or during the 3rd release.***

***THUS WE FINE TUNE THE TEST LIFE CYCLE. So mistakes are reduced and also good procedures are followed, thus increasing the quality of the product.***

Now we will look in detail each step of the STLC.

## 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 :-***
   1. **Features to be tested**

For ex, Compose mail Inbox

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 time available for the project.

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

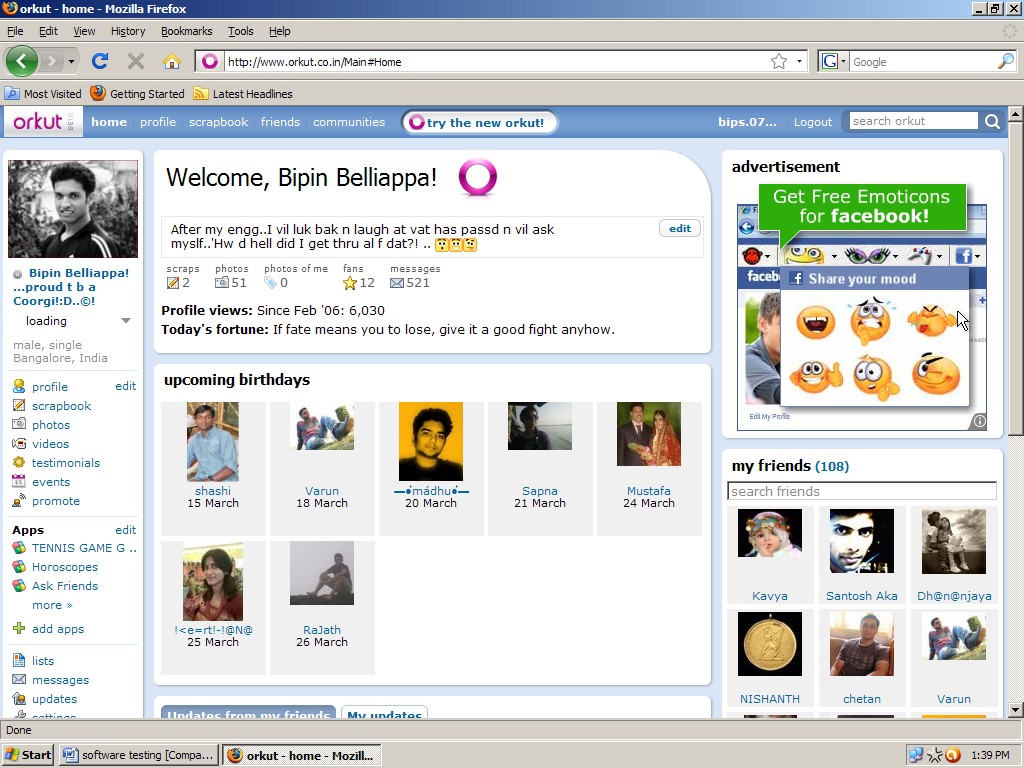
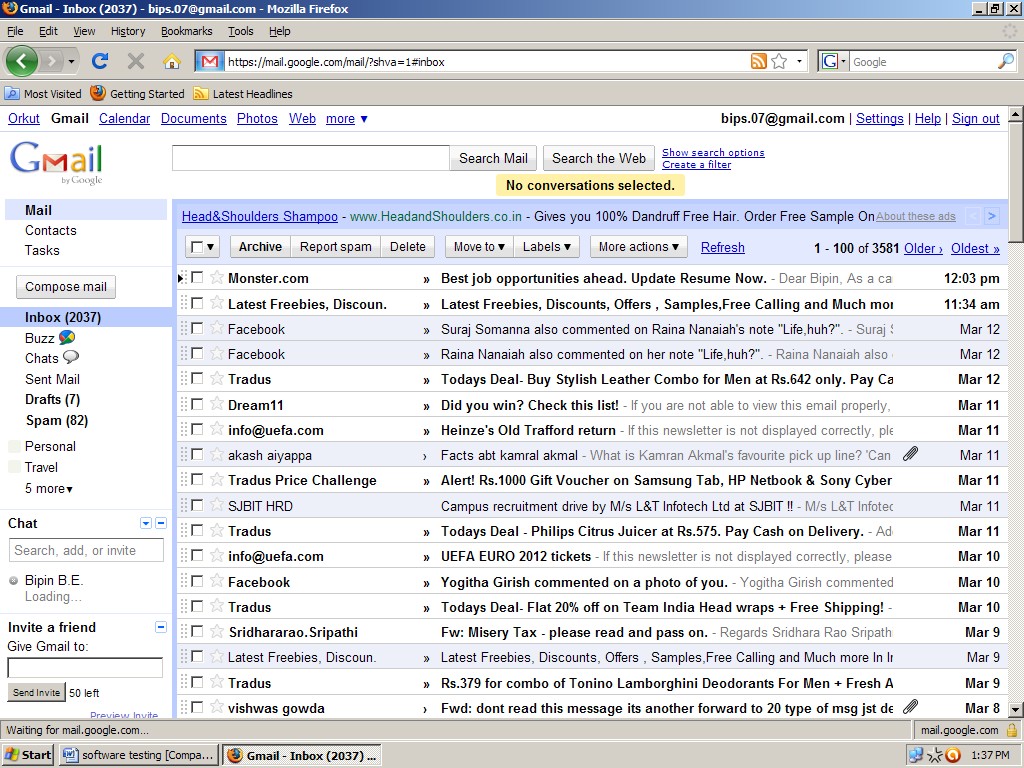
1. “HELP” is a feature developed and written by a technical writer and reviewed by another technical writer. So, we‟ll not test this feature.

##### 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)



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

* orkut, picassa, youtube etc. when we logged into Gmail and when we click on the orkut link – it must take us to Orkut‟s homepage.

Such features are called *Single sign-on feature* – it is a feature wherein 1 login allows access to multiple applications.

**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 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.

###### 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](http://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 | | |

1. ***APPROACH***

The way we go about testing the product in future,

1. By writing high level scenarios
2. By writing flow graphs

###### By writing high level scenarios

for ex, we are testing [www.yahoo.com](http://www.yahoo.com/)

1. Login to Yahoo – send a mail and check whether it is in Sent Items page
2. Login to …….
3. …..

…..

…..

…..

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.

1. ***By writing flow graphs***

##### Yahoo

**Finance Sports Mail Greetings**

##### Login

**Compose mail Sent Items Inbox Read mail**

##### Reply Reply all Forward

**Logout**

We write flow graphs because of the following advantages,

* 1. Merging is easy
  2. Coverage is easy

Flow graphs are written because writing high level scenarios is time consuming.

1. ***ASSUMPTIONS***

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

1. ***RISKS***

If the assumptions fail, risks are involved

1. ***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**

**B**

**A**

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.

**C**

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

1. ***ROLES AND RESPONSIBILITIES***

##### Test Manager

##### Test Lead Test Lead Test Lead Test Lead

**Senior Test Engineer Junior Test Engineer Fresher**

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.

##### 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(if it 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
    - …
    - …

###### 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.

**Systen study write test cases execute test cases release date**

**NEW INSURANCE**

**CUSTOMER NAME CUSTOMER ADDRESS**

**CUSTOMER TELEPHONE CUSTOMER VEHICLE**

**…..**

**…..**

**…..**

**…..**

Date 00/00/0000 Date 00/00/0000 Date 00/00/0000 Date 00/00/0000

This date is given by This date is given by customer development team

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.

1. ***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

* + 1. *Blocker(or Showstopper)*

….

…. (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 because this blocker has blocked the other modules.

* + 1. *Critical*

…

… (define it with an example)

Bugs which affects the business is considered critical

* + 1. *Major*

…

… (define it with an example)

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

* + 1. *Minor*

…

… (define it with an example)

##### Priority

* + 1. *High – P1*

*…*

* + 1. *Medium – P2*

*…*

* + 1. *Low – P3*

*…*

*…*

*… P4*

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

1. ***Test Environment***

##### Hardware

* + 1. *Server* :- Sun Starcat 1500

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

* + 1. *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

* + 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)

* + 1. *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 the test plan, we just write it as TBD – to be decided)

1. ***Entry and Exit Criteria***

**EXIT 1)Based on %age test execution 2)Based on %age test pass**



**3) Based on severity**

**ENTRY**

**FT**

1. **WBT should be over …..**
2. **Test cases should be ready**
3. **Product should be installed with proper test environment**
4. **Test data should be ready**
5. **Resources should be available**

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 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 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 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!!!.

1. ***TEST AUTOMATION***

##### Features to be automated

**…**

##### …

**…**

##### Features not to be automated

**…**

##### …

**…**

##### Which is the automation tool you are planning to use

**…**

##### What is the automation framework you are planning to use

**…**

We automate the test cases only after the 1st release (*we have studied this earlier*).

* 1. On what basis do we decide which feature to be automated ?

Very Important Features

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 tedious job.

* 1. How to decide which features are not to be automated ?

 For ex, “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.

1. ***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

##### Graphs and Metrics

Here, we will just mention what are the types of graphs we will deliver and also give a sample of each graph we will be delivering.

##### (Defect Distribution Graph)

**(Build-wise Graph)**

**(Defect Trend Analysis Graph)**

*Graph 1* :- in this graph we depict – how many bugs have been found and how many bugs have been fixed in each module.

*Graph 2 :-* in this graph, we depict – how many critical, major and minor bugs have been found for each module and how many have been fixed for each module.

*Graph 3 :-* in this graph, we depict – build wise graph i.e, in each build how many bugs have been found and fixed for each module. According to the module, we have found defects. Adding C has introduced a lot of bugs in A and B. Adding D has introduced a lot of bugs in A, B and C.

*Graph 4 :-* Defect Trend Analysis graph depicts – this graph is prepared every month and we must send it to management. It‟s a kind of forecast. By the end of the project, “rate of fixing defects” curve must have an upward trend. Test Lead prepares this graph.

*Graph 5 :-* Test manager prepares this graph. This graph is prepared to understand the gap in estimation of defects and the actual defects that have occurred. This graph helps in better estimation of defects in the future.

###### Metrics

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

***Defect Distribution Metrics***

We generate the *defect distribution graph*(graph 1) by looking at the above data. Similarly we can generate many such metrics.

##### For ex,

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Engineer Name** | **Critical** | | **Major** | | **Minor** | |
| **Found** | **Fixed** | **Found** | **Fixed** | **Found** | **Fixed** |
| Bipin | 40 | 36 | 80 | 30 | 90 | 15 |
| Rajath | .. | … | … | … | … | … |
| Amith | .. | … | … | … | … | … |

In the above graph, we are maintain a record of all the test engineers in the project and how many bugs have been caught and fixed etc. We can use this data for future analysis. When a new requirement comes, we can decide who to give the complex feature for testing based on the number of bugs they have found. We will be in a better position to know who can handle the complex features very well and find maximum number of bugs.

***Interview Questions and Tips***

1. ***What is Metrics ?***

***Ans) We can tell any of the above.***

1. ***On the last day of the project i.e on release date, we find a critical bug. Then what will you do? Will you release the product or fix the critical bug?***

***Ans) 1st say – Testing team prepares a report and sometimes also provides a suggestion on what can be done. But, it‟s the Management team which takes a decision on whether to release the product or not.***

***But, now the Interviewer asks you – “I am asking what will you do, not what the management will do?” Then answer like this – “I will not release the product with critical bug because I want to deliver a high quality product”.***