

ActivTime

ActivTime: - It is a time tracking software which is used to track the time assigned to the user for the better management. (Software)

User- Admin

Password- manager

There are 7 Modules:-

- Projects & Tasks
- Reports
- Work Schedule
- Users
- Settings
- My Account
- Time Tracking

Customer → Project → User → Task → Assign the task to the User

(Functionality of the application)

Steps to create a Customer

- Click on the Project & Tasks
- Click on the Project & Customers
- Click on the Add New Customer Button
- Enter the Customer name & Click on the create customer

Steps to create a Project

- Click on the Project & Tasks
- Click on the Project & Customers
- Click on the Add new project
- Select the customer name from the drop down box
- Enter the Project name & Click on the create project

Steps to create a User

- Click on User Module
- Click on the Add new user button
- Enter the mandatory field
- Check the check box of enter time track
- Click on the create users

Steps to create a Task

- Select the Customer & Project from the drop down box
- Enter the task name, Dead line, Billing time & Click on the create table

Steps to assign the task to the user

- Click on the time track
- Select the user
- Click on Add task to list
- Select the Customer & Project & click on show task
- Select the check box of the task to the list

Note: - Here Manager track the time assign to users at any point of time

Manual Testing

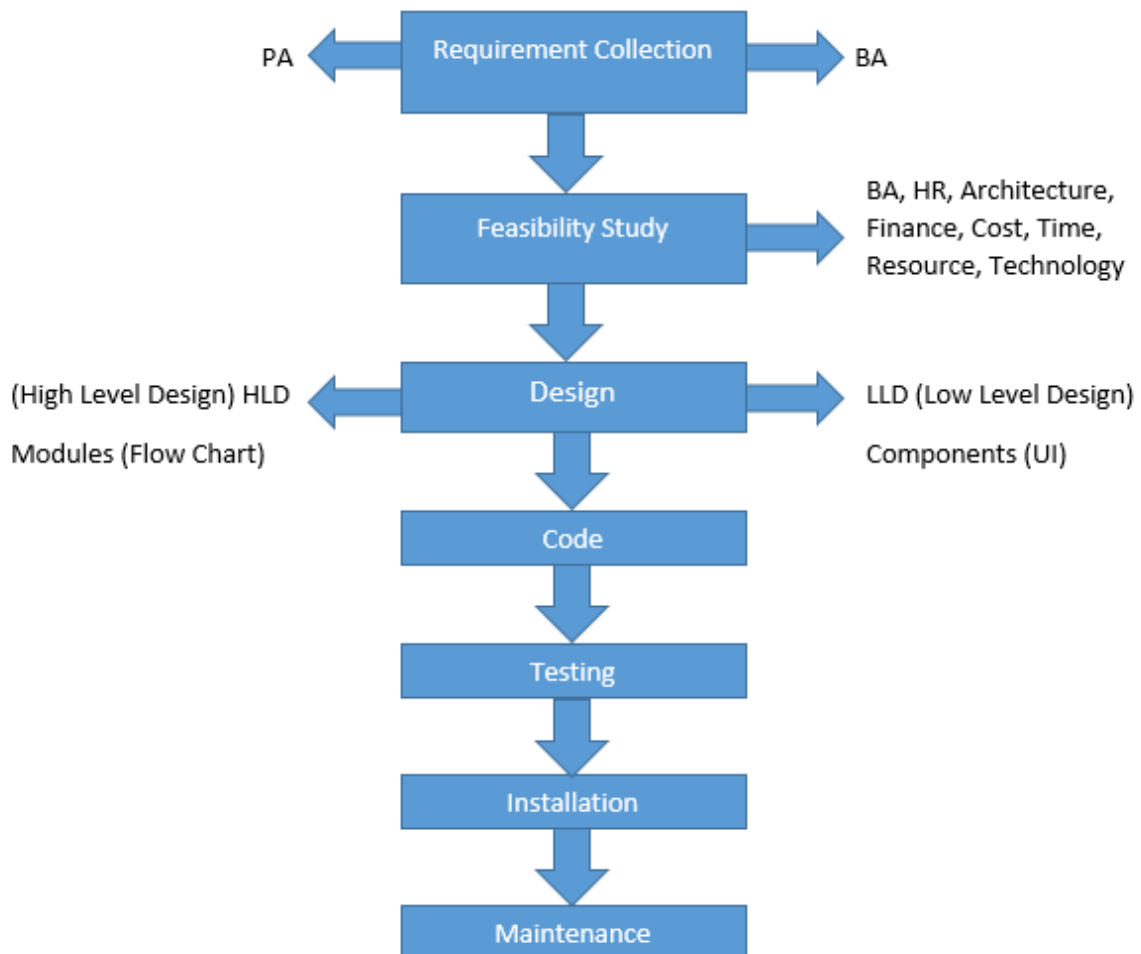
SDLC: - Software Developing Life Cycle

Service based S/W Company – BA (Business Analysis)

- Collect information from the client to develop S/W

Product based S/W Company – PA (Product Analysis)

- Collect information dependent on the market analysis



SDLC Simple Points

- It's a step by step procedure to develop S/W
- Its consists of various stages like
 - Requirement collection
 - Feasibility study
 - Design
 - Coding
 - Testing
 - Installation

Requirement Collection: - In this phase we collect the business need of the customer in the form of requirement documents.

Feasibility Design: - Based on the requirements. A set of people sit and analysis if the project is Do-able or not Do-able.

Design: - One feasibility is done we prepare a blue print of the application. Based on the design the developer start writing code using the particular program language.

Once coding is completed application is handed over to the test engineers where they start checking the functionality of an application according to the requirement.

During testing process we may encounter some bugs which need to be fixed by developers and retested by the test engineers.

This process continuous until the application is bug free/ stable/ works according to the customer needs. The stable application is handed over to the customer in the form of installation or deploying or roll out.

Maintenance: - When customer start using the S/W they may place some issues which need to be in-detail tested & fixed and handover back to the customer this is done under maintenance phase.

Different Types of SDLC:-

- Waterfall Model
- Spiral Model
- Prototype Model
- Verification & Validation Model
- Hybrid Model
- Agile Methodology

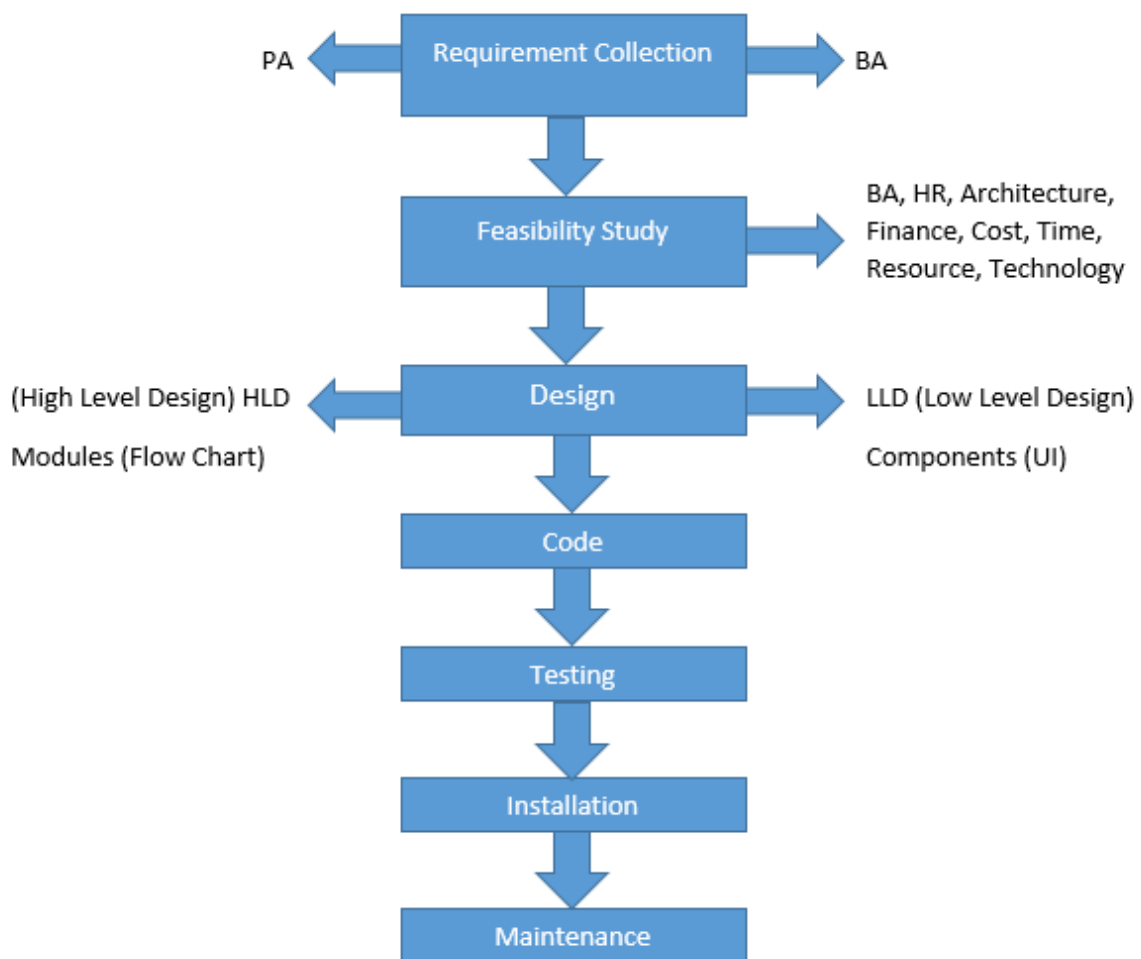
Waterfall Model: - It is a basic model of SDLC.

Execution happens in sequence order (O/P of one phase is equal to I/P of other phase).

Disadvantages

- ✖ Developers used to test application.
- ✖ No parallel deliverable.
- ✖ Requirement review not exit.
- ✖ Reversible flow of bug flow (Customer side bug will be found).
- ✖ Requirement changes not allowed.

This Model is used when we have – **Life Critical & Machine Critical** projects.



Note: - Apart from waterfall model requirement changes are allowed in all other models.

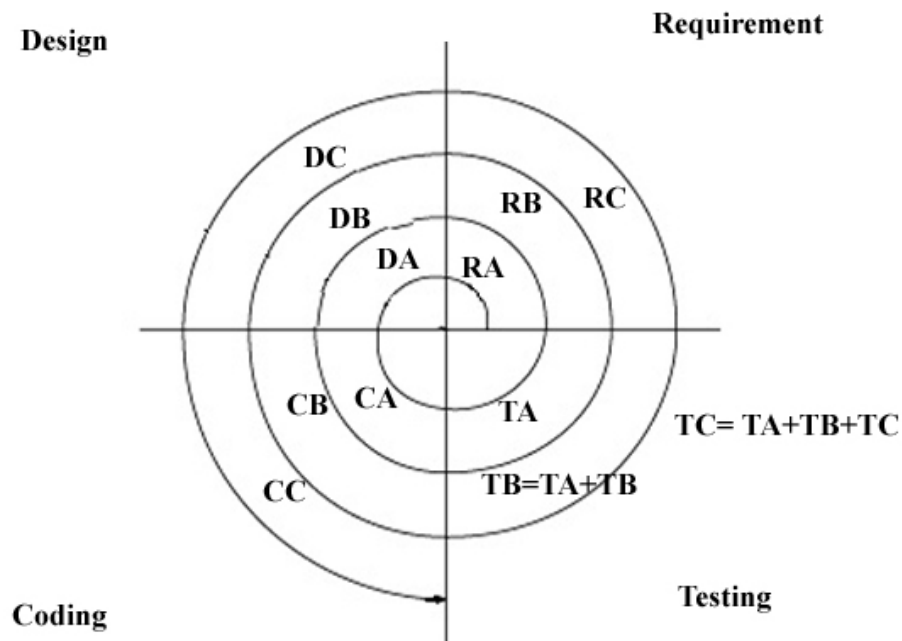
Apart from waterfall model test engineers test the application in all other models.

Spiral / Iteration / Cyclic Model: -

We go for this model whenever the modules are dependent on each other.

In this type we develop application models wise & hand it over to the customer.

Ex: - Excel

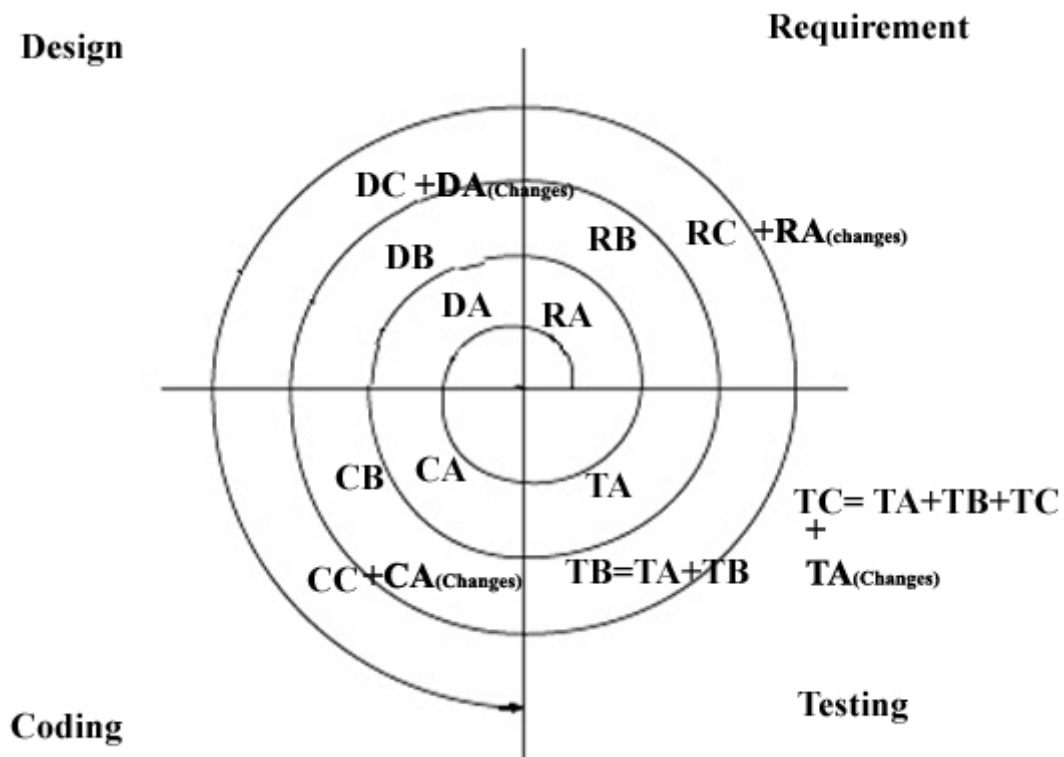


Advantages & Disadvantages

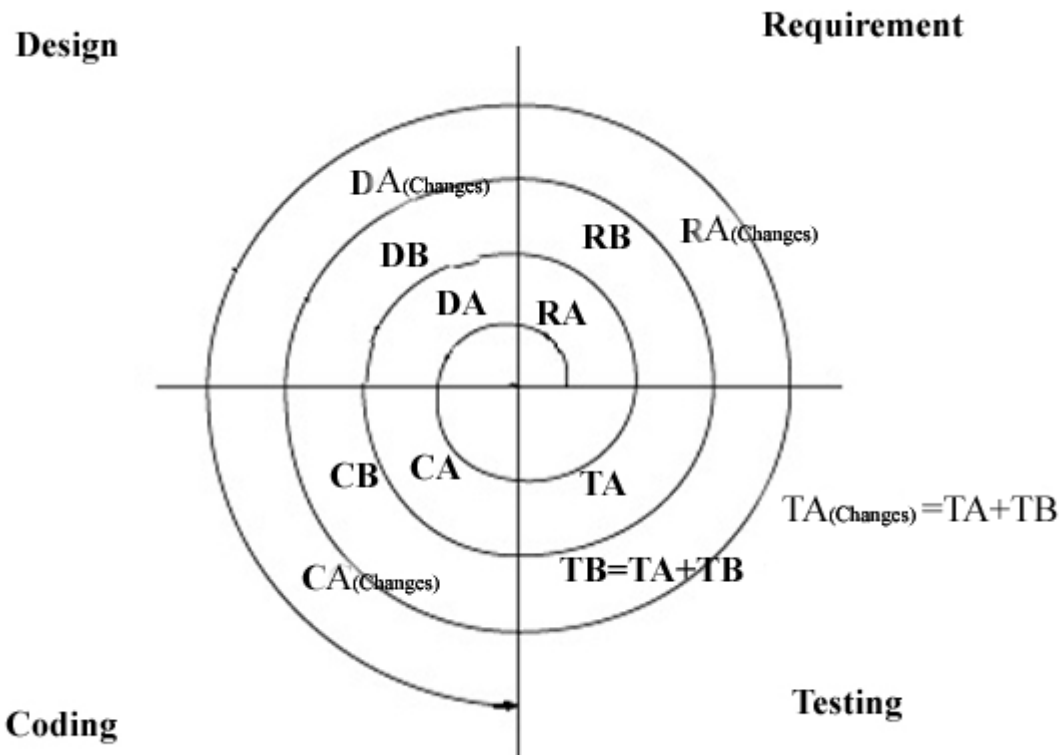
- ✓ Flexible for changing.
- ✓ Customer use S/W at early stage.
- ✓ More clarity for Developers & Test engineers
- ✗ No parallel deliverable.
- ✗ No Requirement review process.

Changes can be of 2 types: - Minor & Major changes

Minor changes: - In case of minor changes we never go for extra circular whereas there changes are accommodated with the same cycle with the next modules.

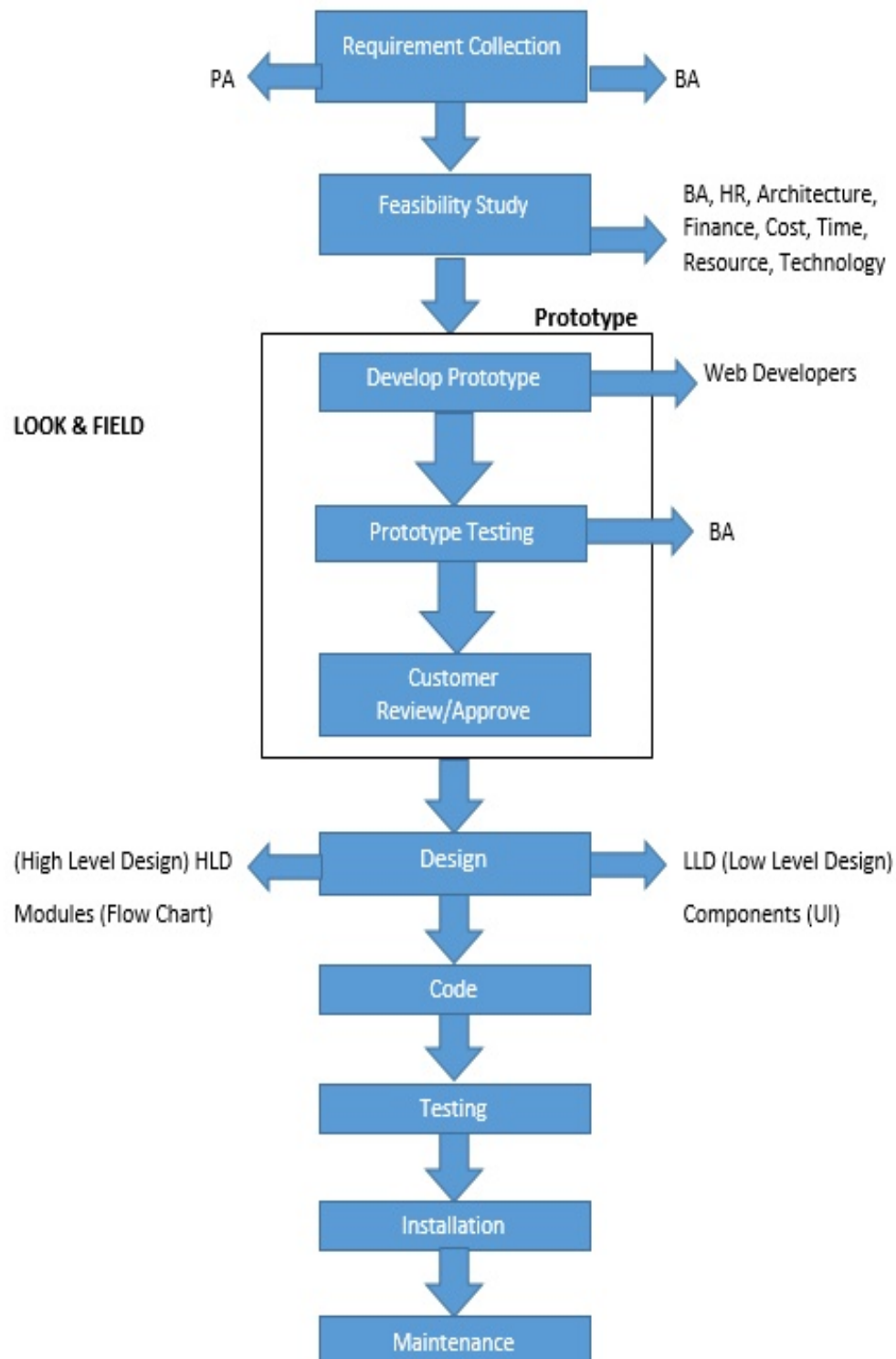


Major Changes: - In case of major changes we always go for one Extra cycle because it may affect the existing modules.



Prototype Model: -

Since the customer rejection was more in previous model the company started collecting the requirements to prepare a Prototype (Sample) show it to the customer and over it was approved. They started working on the original projects so that there won't be any customer rejection or customer rejection low.



Advantages & Disadvantages

- ✓ Customer satisfaction
- ✓ Customer review possible
- ✓ Customer rejection less
- ✓ Prototype reused
- ✗ No parallel deliverable
- ✗ No Requirement review
- ✗ Time consuming (if customer changes in prototype)

When do we go for Prototype Model?

Whenever the customer not clear about the requirement or when he is new to software industry this case we go for prototype model.

Requirements

- **CRS** – Customer Requirement Specification
- **BRS** – Business Requirement Specification

Note: - For CRS the details will in the form of high level language. This details brought by BA from the client (what client need)

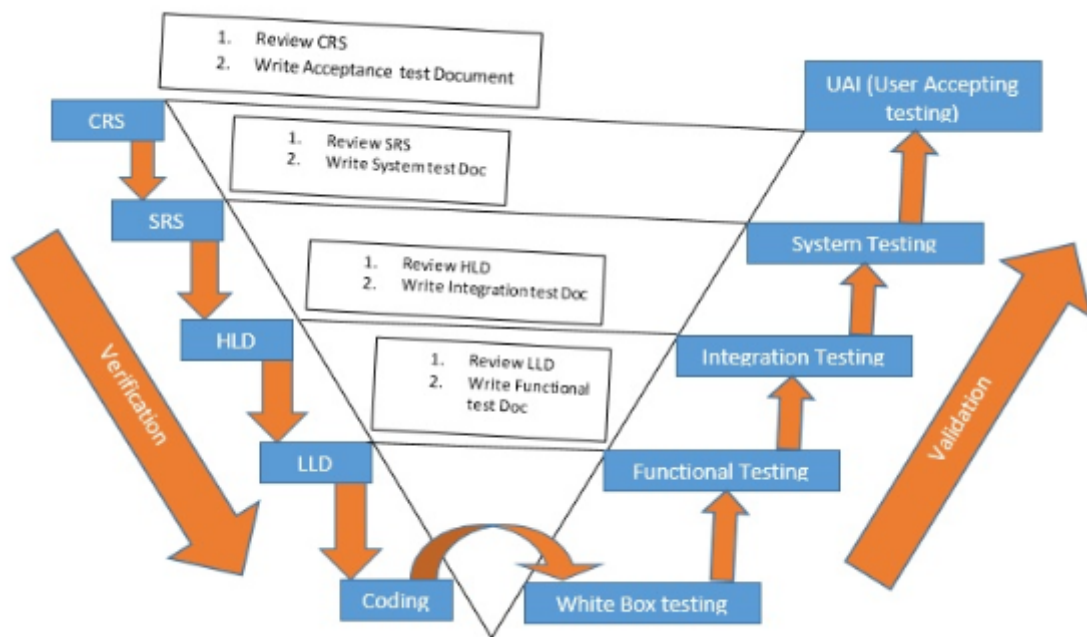
For SRS the details is converted to detail file which can be understand by the Developers & test engineers.

- **SRS** – Software Requirement Specifications (Converts to In-details language)
- **FS** – Functional Specification

Characteristics of a good requirement: -

- In-Detail & Proper flow
- Easy to understand (Simple language)
- Measurable (It should in Numbers) or Countable.

Verification & Validation Model



Errors in Verification

1. Improper flow
2. Conflict requirement (Similar)
3. Missing requirement

Advantages & Disadvantages

- ✓ Review Exist in every phase (less number of bugs in application)
- ✓ Parallel deliverable exist
- ✓ Robust product (Stable)
- ✓ Test Engineers has more knowledge about product
- ✓ Text document can be re-used
- ✗ Costlier (Expensive)
- ✗ Time consuming (if requirement changes, we need to change every text documents)

Why it's called as V Model?

Since the execution happens in v shape. I.e. first downward flow goes on for verification process & one point of time to upward flow go on for validation process so it's known as V Model.

Why it's called Verification & validation?

Since this model executed in two phases i.e. first verification process takes place and once the application is ready the validation parts stable. So it is called Verification & validation.

When do we go for V model?

Whenever the application are clear & complex these cases we go for V model.

KAPREDDY

Hybrid Model

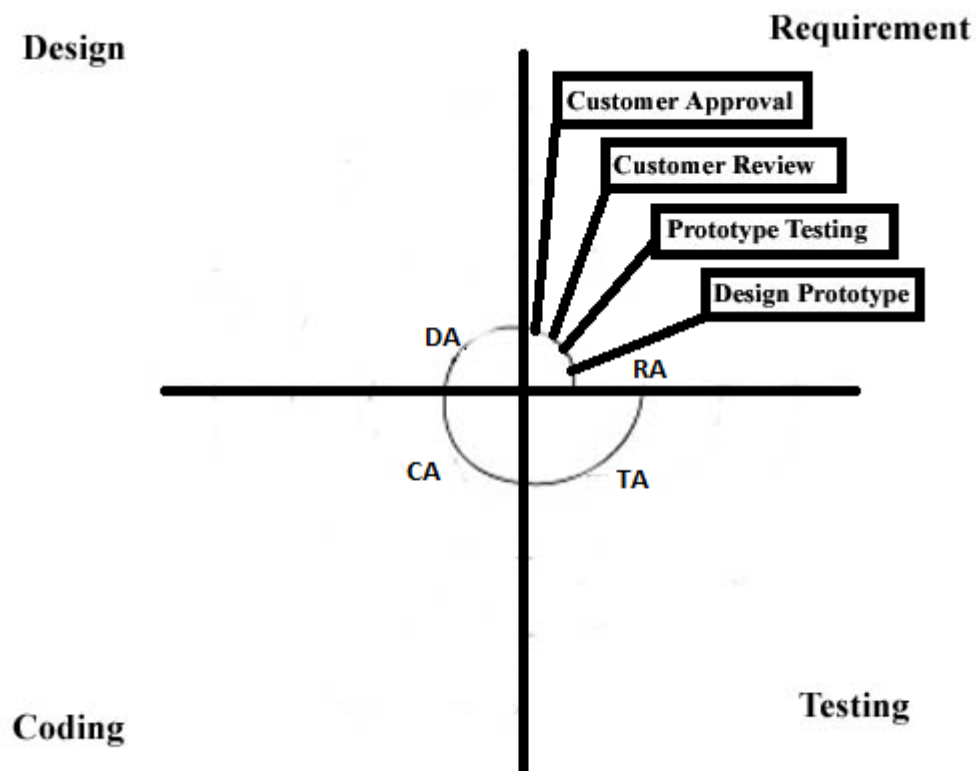
Hybrid Model is nothing but combination of two basic (traditional) models. So in hybrid model we can combine any two basic models basically

I) Spiral & prototype

ii) V and prototype

These 2 types are fall into hybrid category.

Spiral & Prototype



The team will disregard the features that they do not find beneficial.

Advantages & Disadvantages

- ✓ Highly Flexible
- ✓ Adopts Best of Breed
- ✓ Customer rejection is less because of prototype

- ✗ Does not Conform to Typical Standards
- ✗ Every Hybrid Model is Different

Model are dependent & customer new to industry

We go for hybrid model whenever we want to obtain the characteristics of two model in a single model

Interview Questions: -

1. What is SDLC? Explain the phases of it?
2. What are the Alter modes?
3. Explain V model in detail?
4. Advantages and disadvantages of V model?
5. Why is it known as V model V&V Model?
6. What is a hybrid model 7 which all models can be combined to form hybrid model?

Agile Methodology

It is a method to develop software at a very high speed.

It is an iteration & incremental method of software to develop.

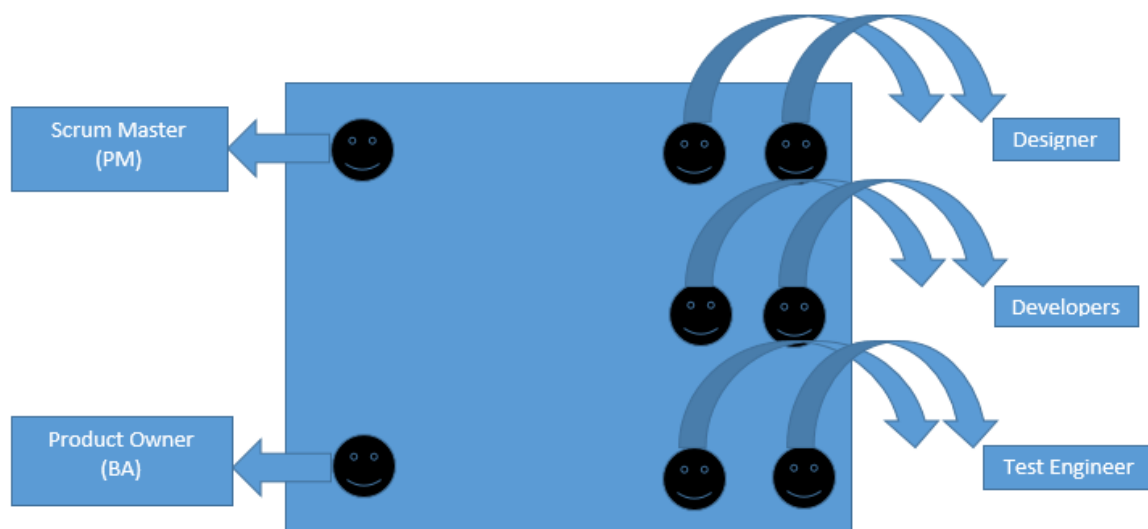
Advantages & Disadvantages

- ✓ More clarity is there
- ✓ Requirement changes exits
- ✓ Less commit gap between team members
- ✓ Customers are involved frequently
- ✗ If 2 or more members leave job it will lead to project failure.

Types of Agile methodology:-

1. Scrum Method
2. Extreme programming

Scrum Method: - Collection of people in an Organised way



Team Size= 7(+/-) 2

Time Duration = 7days / 15 days / 30 days

Requirement are given by the customer is in priority order

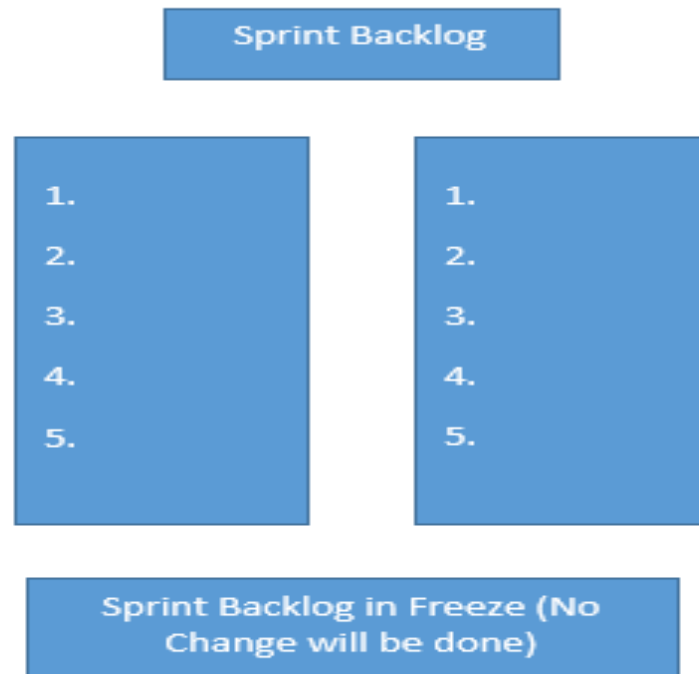
Product Backlog: - All the requirement given by customer in priority order

Sprint Backlog: - Certain set of requirement used for sprint.

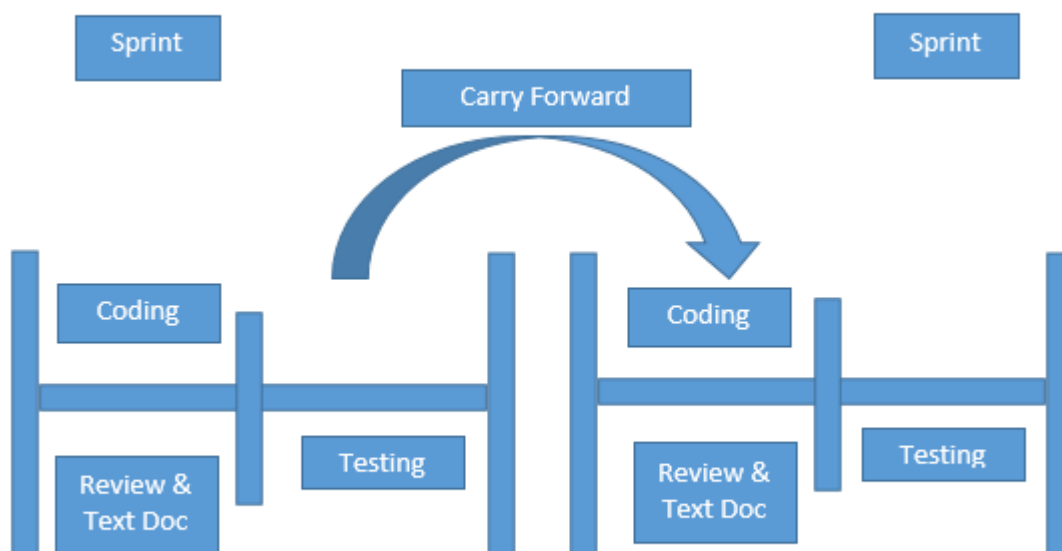
Sprint: - Time taken to develop certain set of requirement

Carry Forward: - The movement from one spring to other spring.

The requirement which are in priority order are divided into parts and it is developed in sprint sections by Group of people called as Scrum Team



Life Cycle for developing the sprint



Regression Testing: - After developing the new feature it will effect old feature then we go for the Regression Testing.

Or

When a bug is fixed by the development team than testing the other features of the applications which might be affected due to the bug fix is known as **regression testing**.

It is always done to verify that modified code does not break the existing functionality of the application and works within the requirements of the system.

Testing Efficiency: - It is nothing but the number of bugs found by the customer.

Cross Skill: - There is no separate team of designers, developers, and test engineers so there is only one team having all the skills & handle.

Scrum master: - He is the Captain/Lead of the team who is responsible for the development of the product/software under agile methodology.

There are 3 types of meetings

- Sprint planning meeting
- Daily Start-up meeting
- Sprint Retrospect meeting

- **Sprint planning meeting:** -
 - Scrum master explain the requirement to the team
 - Team identifies there tasks
 - ✓ Designers with their knowledge
 - ✓ Developers with their knowledge
 - ✓ Test Engineers with their knowledge
 - Scrum master will assign the task to the respective teams
- **Daily Start-up meeting:** -
 - About the Things which we have done yesterday/ Today/tomorrow.
- **Sprint Retrospect meeting:** -
 - Wrong, write, any improvement for the next scrum or next release

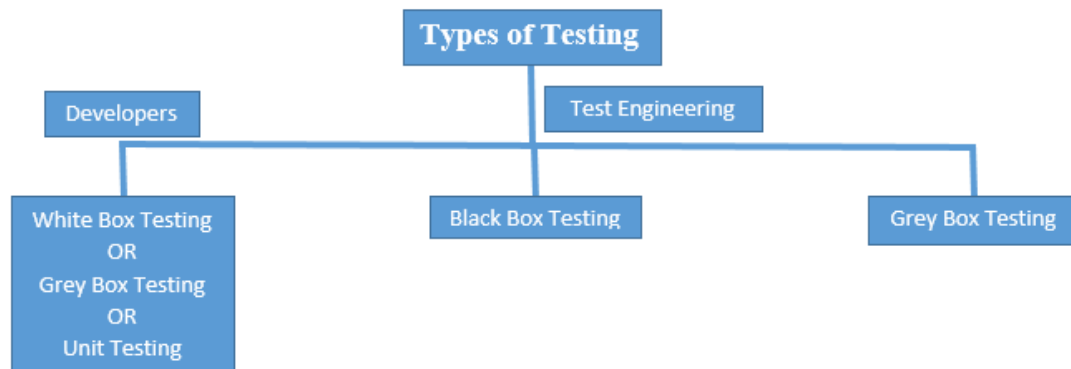
Experience fellows

- ✿ Worked in Agile methodology – Scrum method
- ✿ Active participant in SPM
- ✿ Involved in DSM
- ✿ Involved in SRM
- ✿ Collaborated with developers
- ✿ Team during any bug fix in the script

Fresher

- ✿ Good knowledge of Agile methodology – Scrum method
- ✿ Good knowledge of SPM
- ✿ Good knowledge of DSM
- ✿ Good knowledge of SRM

Types of Testing



If single person team done (Grey testing) both white box & grey box testing it is called grey box testing

Done by Developer where they check each and every line & code before handing over to the Test Engineer.

Since code is visible for developer during testing it is also known as WBT

BBT: - Type of testing done by the Test Engineer where they check the functionality of an application according to the customer requirement.

Since code is not visible during testing process it is known as black box testing

GBT: - The combination of WBT & BBT is known as GBT

It is done by a person who knows the knowledge of coding as well as testing

White Box Testing Process: -

After collecting the requirements in SRS format the developers will start writing code depends upon the requirement after that they cross check whole code by using the unit testing tools.

Advantages & Disadvantages using Software

- ✓ Re-use
- ✓ Accuracy

Advantages & Disadvantages by Manual Testing

- ✗ Time consuming
- ✗ Accuracy not maintained
- ✗ Teddies process

How to do WBT? Explain

- Once the coding is done the developers does one round off code checking which is known as WBT.
- During WBT has some draw backs like time consuming, teddies process, Accuracy.
- Hence WBT should be done with the help of unit test program
- Unit test program are small program which intern check the main program and generally they are return with the help me unit test tools.

Advantages of unit test programs

- ✓ They can be reused.
- ✓ Accuracy will be maintained.
- ✓ Time saving since execution is faster than manual.

Note: - Do not speak about WBT process until unless ask

Testing

Check the functionality of an application according to the customer requirement.

The process of verification and validation the system.

Checking the application like a user to make it bug free or stable.

Note: - In order to start testing we should have requirement, application ready, necessary resources available, to maintain accountability we should assign a respective model to different test engineers,

Main Testing is of 3 types: -

- Functional Testing
- Integration Testing
- System Testing

Functional Testing: -

Checking each and every component of a module or module of an application independently is known as functional testing.

Requirement

1. Account Transfer

1.1 FAN – Text Box

1.1.1 FAN – Accept the 10 digits

1.2 TAN – Text Box

1.2.1 TAN – Accept the 10 digits

1.3 Amount

1.3.1 Amount 4 digits

1.4 Transfer

1.4.1 Enable Transfer

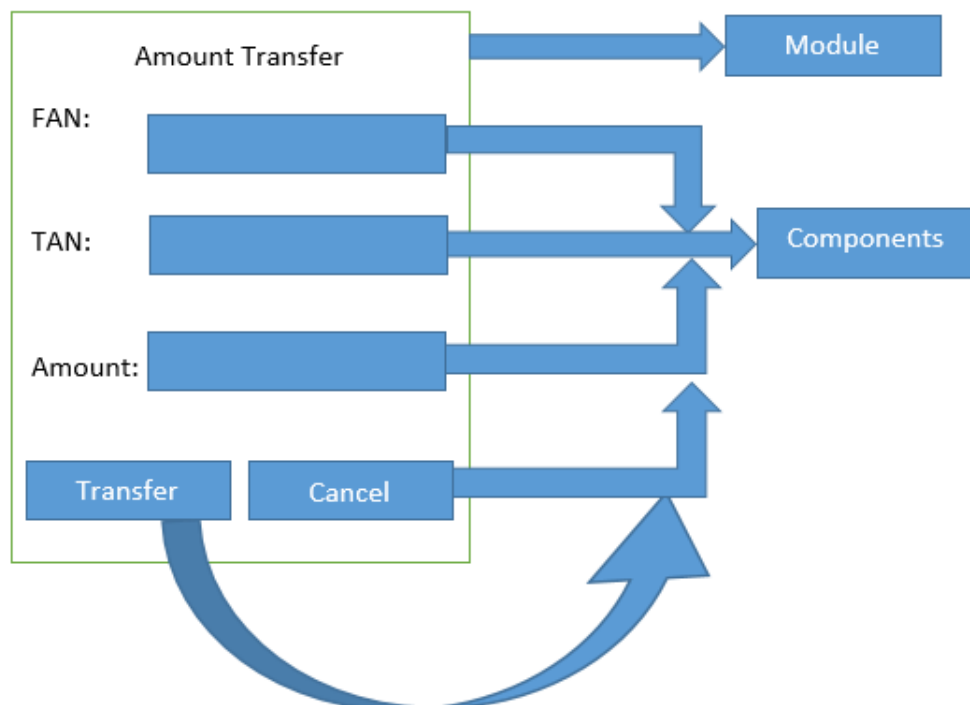
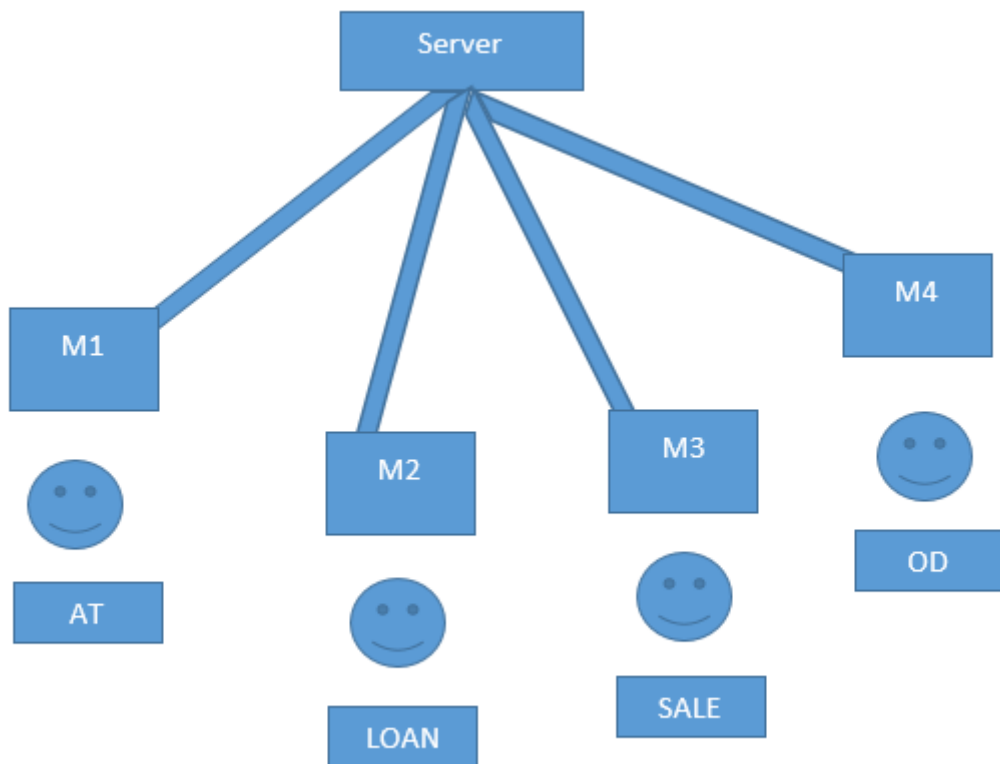
1.5 Cancel

1.5.1 Cancel Enable

Process: - URL

U/P→ok

Loan – sales – OD – at (Modules)



Note: - To start functional testing at least we should have one module.

→Positive Testing

→Negative Testing

→No over testing

→No Assumption required

(Feel that maximum test coverage Achieved)

Checking the FAN: - (10 Digits it should work)

1234567890 – Accept

1234567899 – Error →A/C Valid or not

Blank – Err→Enter some values

9 Digits / 11 Digits – Error Msg

Alphanumeric – Error Msg

Blocked Account No – Error Msg

Same as FAN & TAN – Error Msg

Copy & paste – Error Msg

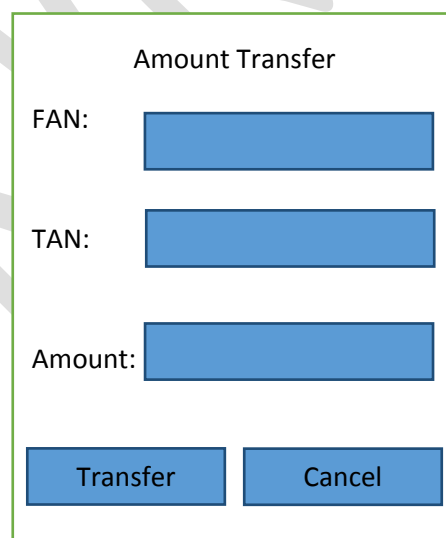
Similar Values for TAN also

For Transfer – +ve FAN / +ve TAN / +ve Amount / Transfer

Cancel – Refresh the fields. Any data we can enter

Integration Testing: -

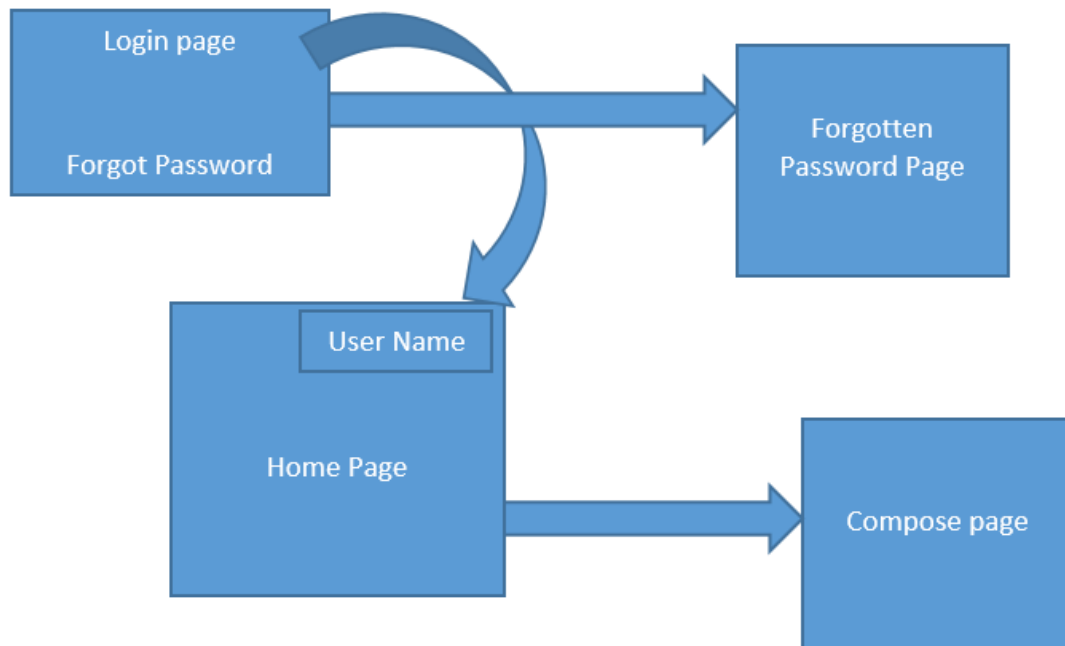
Checking the data flow between dependent modules.



The screenshot shows a web form titled "Amount Transfer". It contains three input fields: "FAN:", "TAN:", and "Amount:". Each field has a blue rectangular input area. Below the input fields are two buttons: "Transfer" and "Cancel", both with blue backgrounds and black text. The entire form is enclosed in a green border.

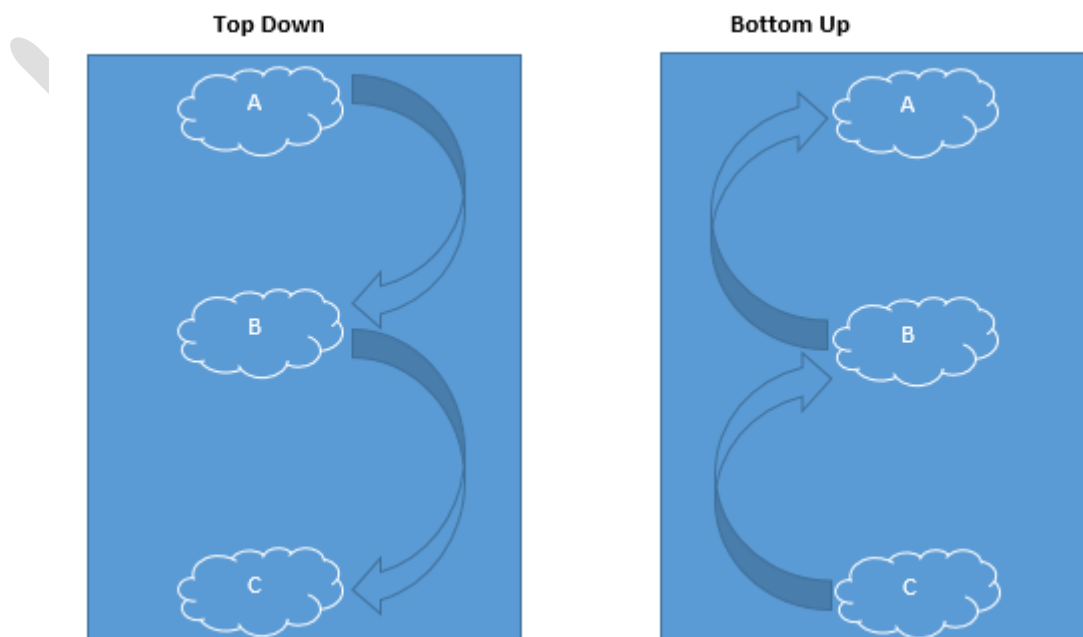
Login as a user check for a balance in that account = Present balance + Received Balance

Ex: - Dinga check for a balance in that account.



Types of Integration: -

1. Incremental testing
 - a. Top Down Approach
 - b. Bottom Up Approach
2. Non – Incremental testing or Big Bang theory



Incremental Integration testing: -

We go for this approach whenever there is a clear relationship between modules.

Ex: - Booking Ticket, shopping

There are some approaches: -

→ Top down approach

→ Bottom up Approach

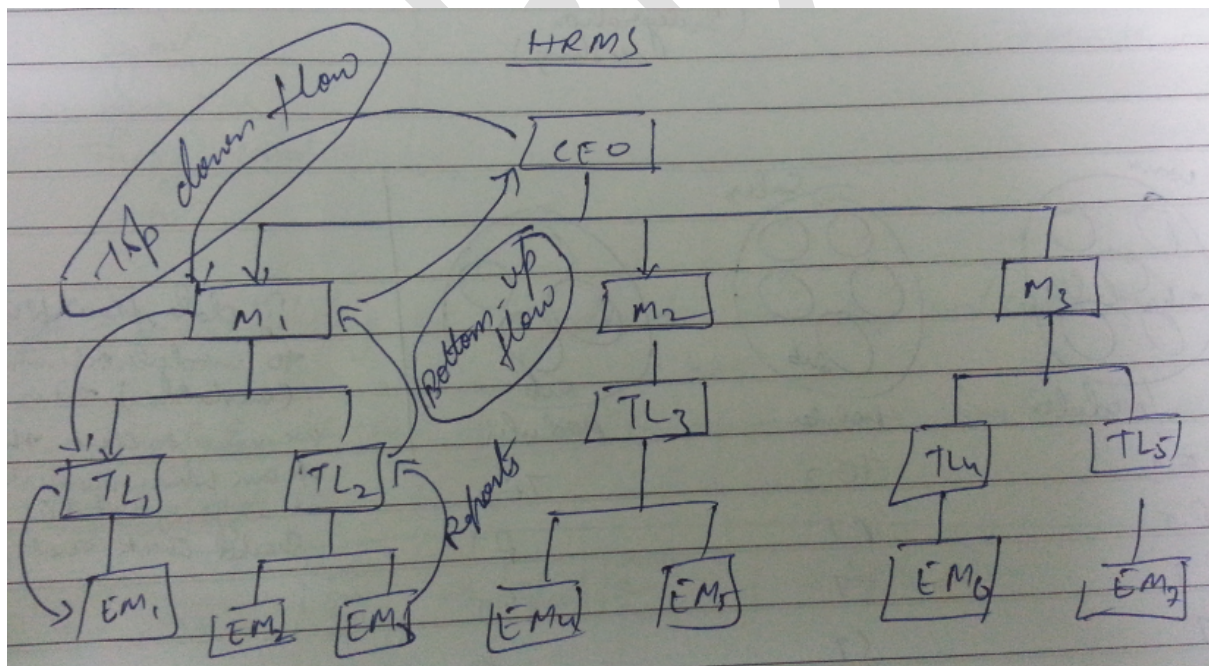
TDA: - In this case we add the modules incrementally from top to the bottom & check the data flow in same order.

BUA: - Here the modules are add incrementally from Bottom to top and data flow is checked in the same order

Non Incremental Integration testing: -

We go for this approach whenever there is no clear relationship between the modules.

In this case we create data in one modules & check for the saved data by banking on all other. Ex: - Gmail



IIT + NIIT → Sandwich Testing

Note: - Do not speak about types of integration testing until & unless asked.

Before integration testing each & every module working independently

→ Pick module by module for Integration testing so that we don't skip any scenarios (point) and a proper sequence is followed

→ For IT the data should be same in corresponding modules

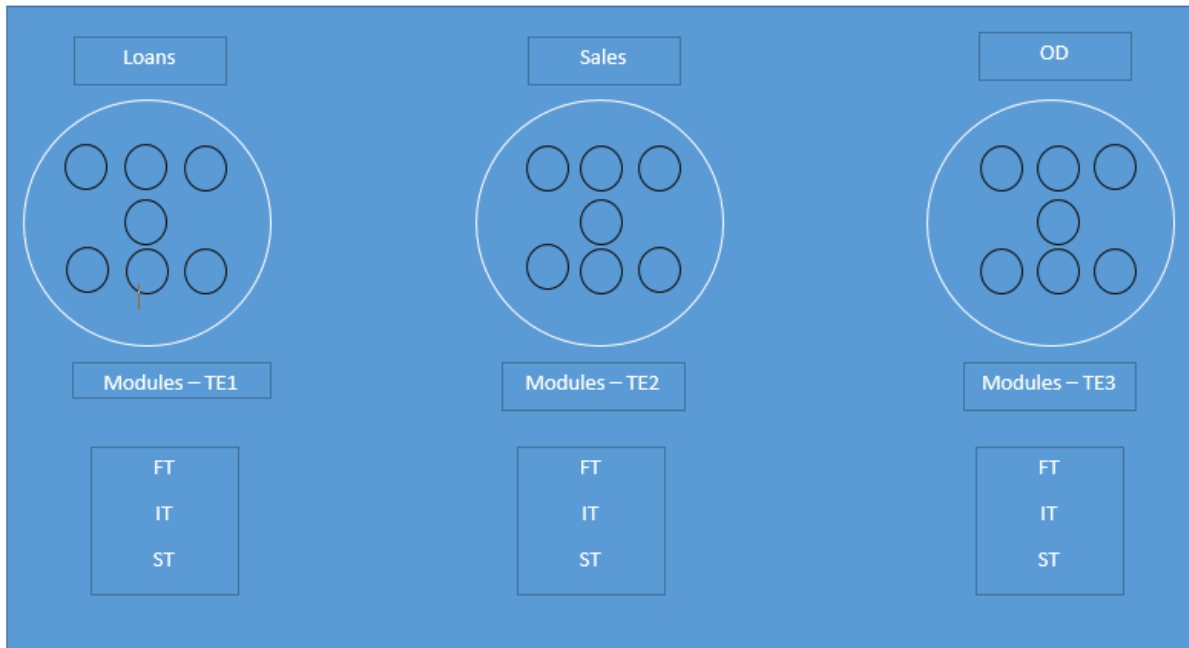
→The data flow can be within same modules as well

→Functional testing is compulsory on each & every module whereas IT can be done only if the data flow exist

System testing: - (Customer will use any modules)

Checking the end to end flow of an application.

In system testing we navigate through all the require modules of an application and check the system as a whole.



(Respective to their Assign Modules)

If data flow between to modules or scenarios (Mistake) then we need to clear that in which module it is going and that TE should check that thing.

Note: - functional testing – testing whether application functionally as per requirement or not

Non- functional testing – In this testing we go for load, stress, performance, security, configuration, compatibility.

Testing Any Application

Gmail: - Login, Compose, Draft, Inbox, Sent Item, Spam, Chat, Help, Logout

(We do Functional Testing on all Modules First)

Compose: - To, CC, BCC, Subject, Attachment, Body, Sent, Save to Draft, Close

Test **TO** as Functional Testing

+Ve: - dinga@gmail.com → Accept

Dinga12@gmail.com → Accept

Dinga@yahoo.com → Accept

-Ne: - Blank → Error

Dinga@yahoocom → Error

Dinga@yaho.com → Error

Same as **CC & BCC**

Subject: -

+Ve: - Enter maximum character → Accept

Enter Minimum character → Accept

Blank Space → Accept

URL → Accept

Copy & Paste → Accept

-Ne: - Crossed maximum digits → Error

Paste images / video / audio → Error

Attachment: -

→ File size at maximum

→ Different file formats

→ Total No. of files

→ Attach multiple files at same time

→ Drag & Drop

→ No Attachment

→ Delete Attachment

→ Cancel Uploading

→ View Attachment

→ Browser different locations

→ Attach opened files

Subject: -

→ Maximum character

→ Minimum character

→ Flash files (GIF)

→ Smiles

→ Format

→ Blank

→ Copy & Paste

→ Hyperlink

→ Signature

Sent: -

Write all the field and click on sent (Conformation msg “Msg sent successfully”)

Saved to Drafts: -

Write all the field and click on SAD (Conformation msg)

Cancel: -

Write all field & click on cancel button (Window closed / SAD / All filed refresh)

Integration on Gmail: -

Login – Entering user/password in login & Check username in homepage

Compose – Compose mail, send it & check the mail in sent item (sender)

Compose mail, send it & check the mail in receiver (inbox)

Compose mail, send it & check the mail in self (inbox)

Compose mail, click on save as draft & check in sender draft

Compose mail, send it invalid id (valid format) check undelivered message

Compose mail close and check in drafts

Inbox – Select mail, reply & check in sent items or receiver inbox

Select mail, in inbox for reply SAD check in the draft

Select mail delete it check in trash

Sent Item – Select mail, SI, Reply/Forward & check in SI or receiver inbox

Select mail, SI, Reply/Forward, SAD & check in the draft

Select mail, delete check in the trash

Draft – Select mail draft, forward & check SI or inbox

Select mail draft delete & check in trash

Chat – chat with offline saved in the inbox of receiver

Chat with user check in the chat window

Chat with user check in the chat history

Testing Example-2

1. Login User – apply for OD – logout
2. Login Manager – approve OD – logout
3. Login User – check balance – logout
4. Login User – repay OD – logout

✿ Functional testing on each module ,then check for integration testing, system testing

OD – Overdraft

Apply for OD in Advance as 2 month salary.

Approve by the manager. Amount credit. Interest will be there. Process fee for 1st time

Note: - During testing we may need to wait for a particular duration of time.

In system testing the test environment (server) is similar to the production environment.

1. Login user → Homepage [loan, sales, od] → OD page [amount OD, apply OD, repay OD] → Application
2. Login manager → Homepage [loan, sales, od] → OD page [amount OD, apply OD, repay OD, approve OD] → approve Page → approve application
3. Login user → Homepage [loan, sales, od] → OD page [amount OD, apply OD, repay OD] → approved OD → amount OD
4. Login user → Homepage [loan, sales, od] → OD page [amount OD, apply OD, repay OD] → repay OD → with process fee + interest amount

Parts of Application

UI (User Interface) → front end → it is the user interface of the application.

BL (Business Logic) → back end → these are the logic or code return by the developers which makes the functionality of an application to work.

DB (Database) → back end → it consist of the complete data of the users which is stored in the form of tables.

UI → GUI (Graphical User Interface)

→ CLI (Command Line Interface)

Types of applications: -

→ Stand-alone application

→ Client-server application

→ Web based application

Stand-alone application: -

In this type the entire software is installed at end user machine & only one user can access at a time (.exe files)

- ✓ Faster to access
- ✓ Security → data hacking not possible → virus
- ✓ We don't require any server for this application
- ✗ System resource is occupied
- ✗ Installation is required
- ✗ Data sharing not possible
- ✗ Single user access at time
- ✗ Maintains is very high or tough (application is installed in user machine if any issues happens we need to contact respective person for resolve)

Client-server application: -

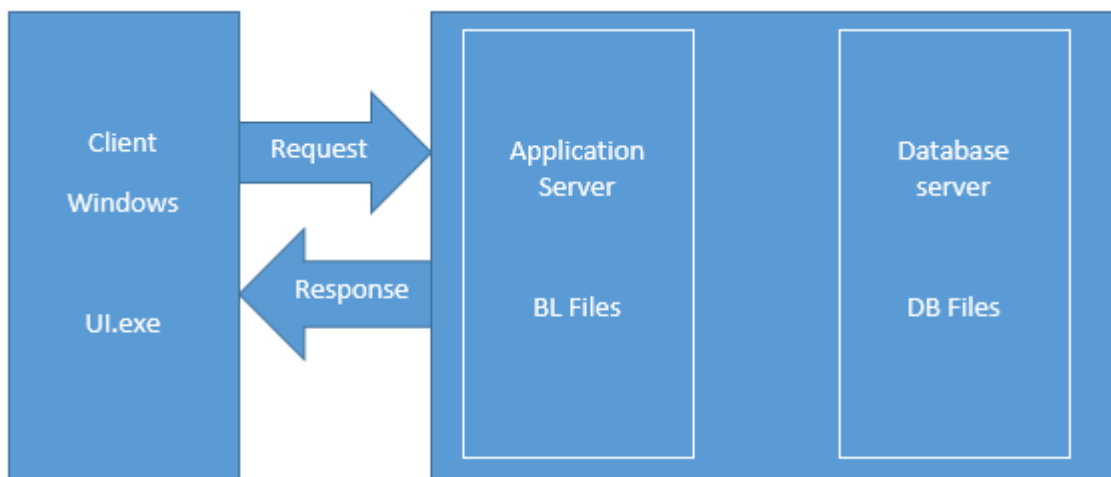
- ✓ Faster in access
- ✓ Secured → data sharing (security)
- ✓ Maintains is easy
- ✓ Multiple user can access
- ✓ Data sharing is possible
- ✗ Installation require
- ✗ System resource will occupied space (less than SAA)
- ✗ If server is down (no one can access)

It consist of two parts client & server. Client part (UI.exe) is installed in user machine & server part is installed (BL, DB)

In this type the software is divided into two parts

1. Client software (.exe)
2. Server software (BL, DB)

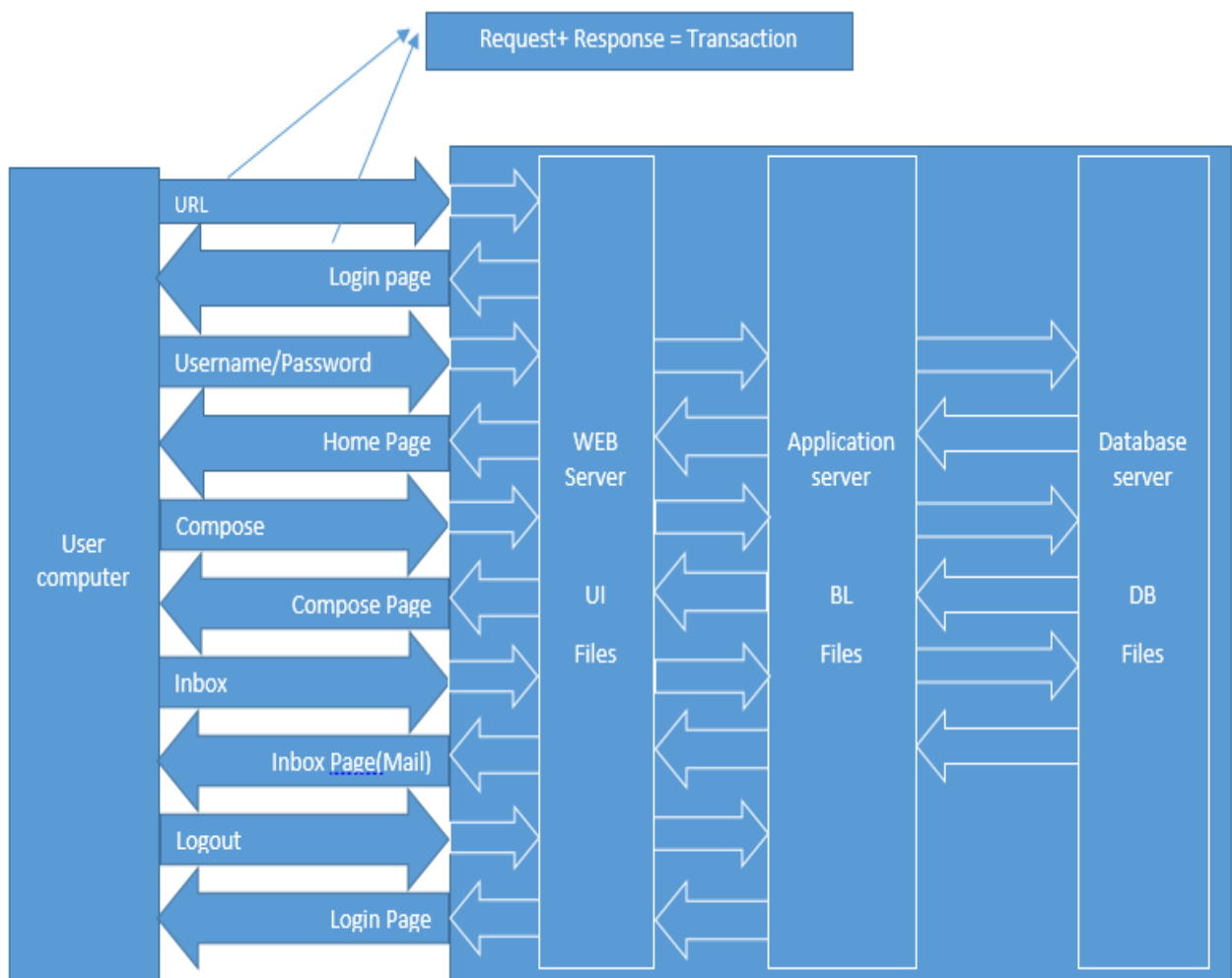
The client software is installed at end users machine whereas the server software is installed at server machine & both client & server communicate with each other for the application to work.



Web application: -

In this type the entire software is installed at server & the end users access the application with the help of URL

- ✓ Data sharing possible
- ✓ Multiple user can access at a time
- ✓ No maintains
- ✓ No installation
- ✓ Faster in access
- ✓ Security → data sharing (security) → No virus (antivirus)
- ✗ Server down (No access)

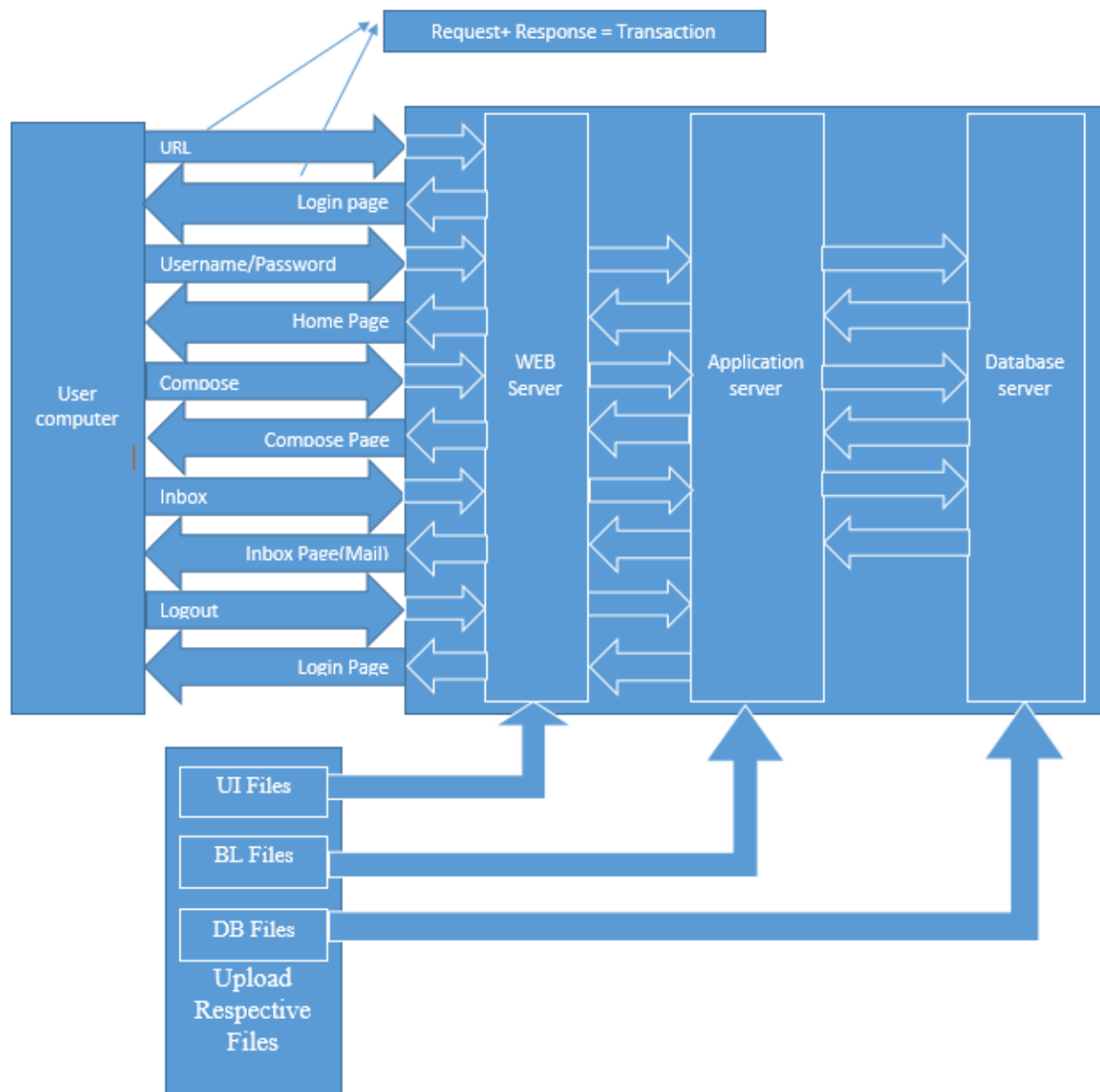


Web server: - it is the software consist of all the pages related to the application

Application server: - here the entire business logic return by the developers which makes the functionality to work is stored.

Database: - it consist of tables where the entire data of end user will be stored in table form

How to upload files to the respective locations?



Whenever we converted to build the build contain (UI, BL, DB) files. So we need to dump the respective files in the respective location.

Environment

Server consist of software/hardware is called as production server.

This server is used by the end user or real user.

The customer want an application then he go to Software Company and give necessary requirements then the company give a requirement to developer to develop the code in the development server after completing the code the application is installed in the test server so that the test engineer will test the application until the application is stable then it will dumped into the production server.

→Security given to production server since it contain real user data

→Highly configure system since used by N number of user

→Production server is access by production URL

→Once the application is developed/tested/stable then it is dumped to the production environment

Test Environment: -

After collecting the requirement from the customer the developer start developing the code after completing the application that application should be installed in the test server.

That application can be accessible using URL

[HTTP://COMPUTER NAME/PROJECT/COMPANY NAME.LOCAL](http://COMPUTER NAME/PROJECT/COMPANY NAME.LOCAL)

The tester find bugs and that should be told to developer and the developer should reproduce the bug in the test server using the test URL. This process goes until the application stable.

Separate server for both of them.

If both uses same server there won't be any parallel deliverable.

We take another server to check the compatibility of the server because the production sever is highly configure so we need a similar server to go for the system testing. Check the application compatibility bugs and some more. We call this server as staging server.

Why test environment similar to production server?

Since if we move the application from low configuration to the production user may find some issues to avoid this we do one round off end to end (system testing) on an environment which is similar to production environment.

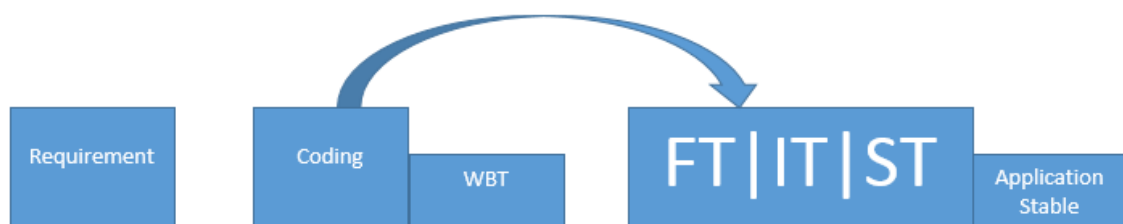
Waiting period execution: -

While testing we have to wait for some time to complete some condition for that sake we need to access the database and change the rows date as we required

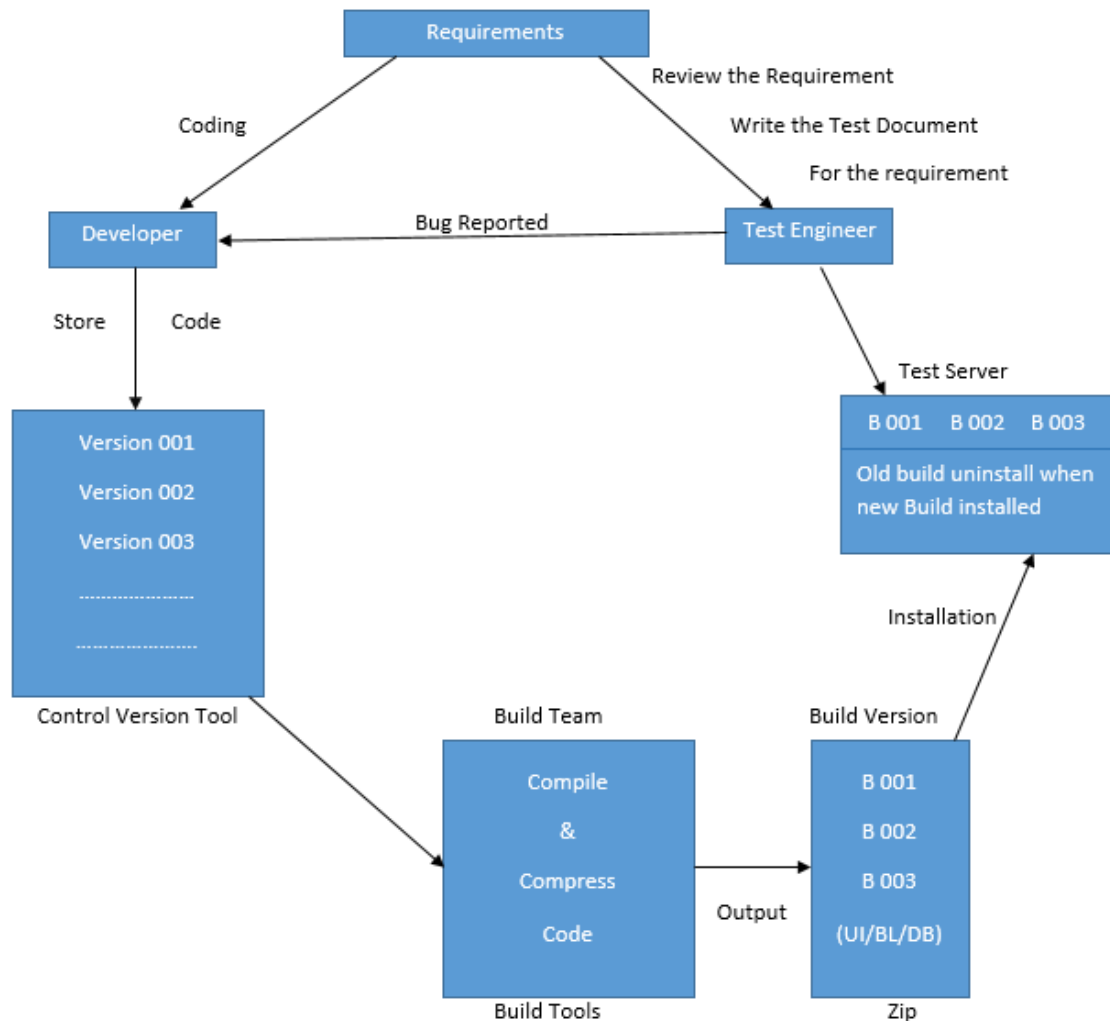
Note: - what is the use of test engineer in sql?

During testing process same time we need to wait for a particular transaction to be satisfied. This cases we access the database and do the necessary changes to proceed smoothly with testing. (UPDATE)

Many times during testing we want to see the data stored in the table for our reference. These cases we use SELECT or SQL QUERIES.



Build Process



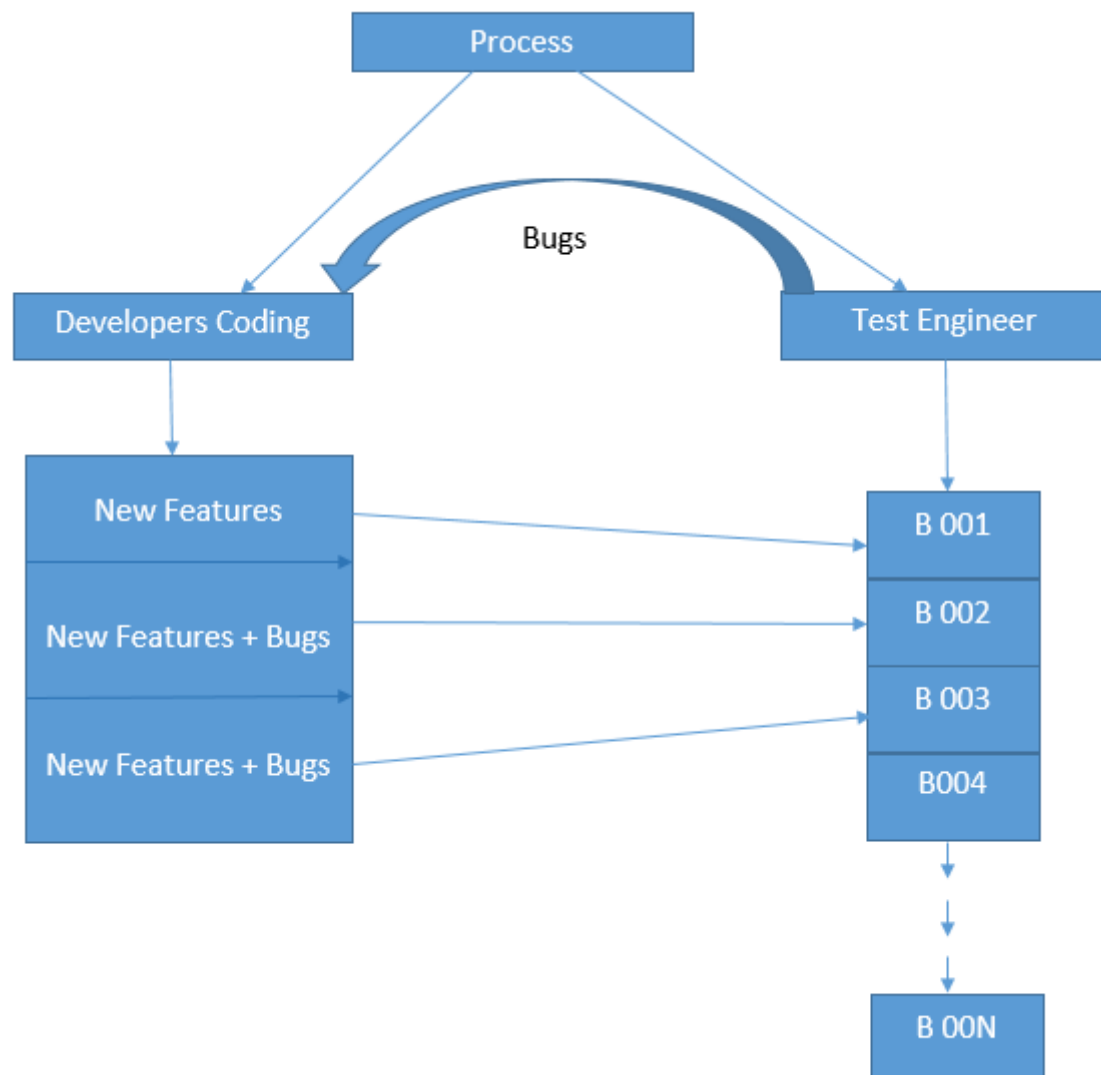
Whenever we get the requirement it is given to the developers & test engineer. After getting the requirements to the developers. They will start writing the code mean while test engineer review the requirement and write necessary documents which are require up to now the developers may completed the code that code is stored in the control version tool. Then build team take the code and compile the code & compress the code (ZIP) using the build tool. Every build will have some number. Then that build will installed in the test server. Then test engineer will test the build if they found any buy that will be reported to the developer then developer reproduce the bug in the test server then if it need any code change then they will do necessary change. This process goes until the builds stable.

Build: - Which is used to convert the code into application format

Control version tools: - The software which are used to store the necessary documents as well as to track the changes.

Note: - After collecting the files from the control version tool we use the build toll to compile the code from high to machine level language after compile the file size increase so we need to compress the file

After unzip we have some files which will be dumped into the test server in their respective servers (web, app, db)



After coding is converted to build which is installed on the test server.

Test engineer will test build & report the bugs to the developer at same time this is done in test cycle (duration).

After collecting the new bugs & new features coding again it will converted to build and previous build is uninstalled from the test server and new build is installed on the test server in

the form of new build like (B002, B003 etc.) With the new bug fixed documents to the test engineer. This will go until the build is stable.

Which bug is to be fixed that will be decided by TE. Since they know the product.

Build is a software that consists of some set of features/bug fixes & installed on test server which need to be tested for a stability.

Test the particular build in a particular time is known as test cycle.

Each new build will be the modified version of the new build.

Frequency of build: - In initial stages we used to get weekly builds but in later stages of testing when the build was becoming stable we used to get once in 3 days, 2 days, or daily build as well.

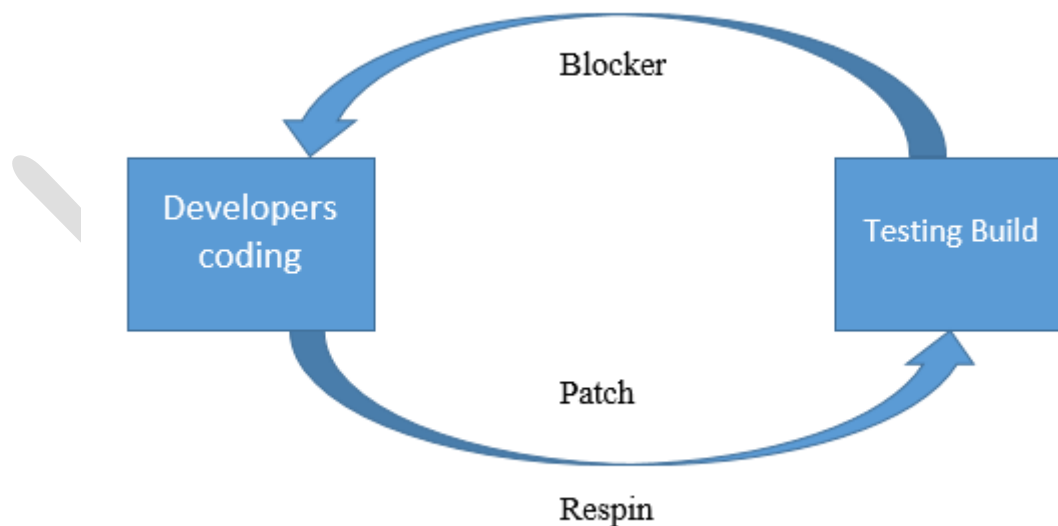
Respin: - When we start testing on build we can encounter a blocker bug it should be fixed and reinstalled in the same test cycle by replacing the old build (more than one build in test cycle).

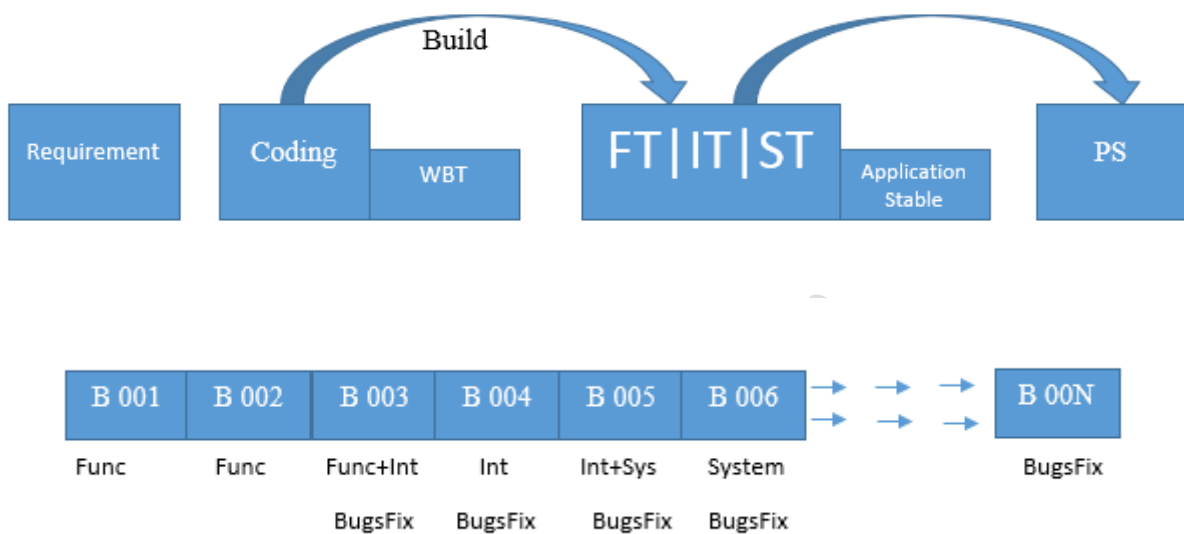
Patch: - It is a temporary solution for a blocker bug.

It is software that acts as an immediate fix for urgent issues.

Respin & Patch is decided by the developers & Build team

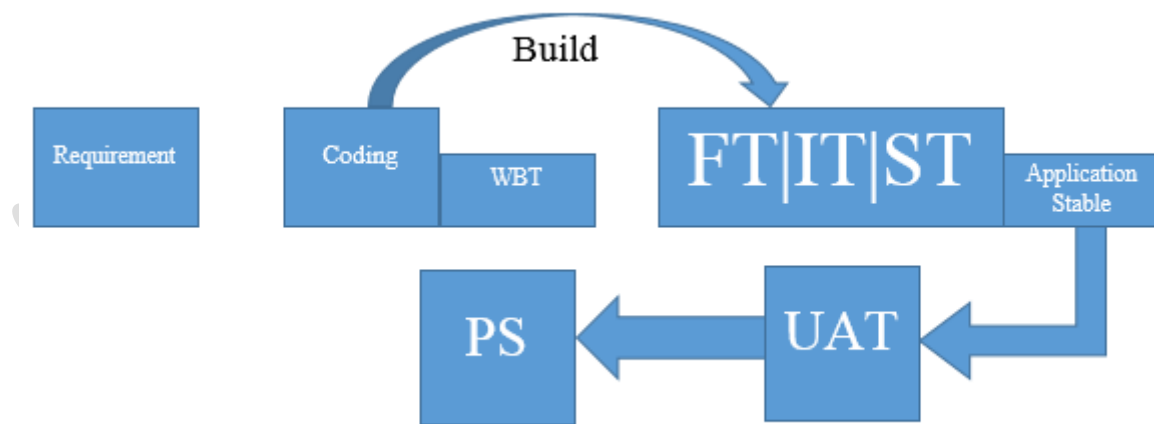
Patch is time saving / Respin time consuming.





After installing the build in the test server first we go for functional testing on each and every component of a module then we do integration and system testing as well as while doing testing developer fixing the bugs which we got while testing that bug need to be retested by the test engineer so that build will be stable.

Release the product: -



Release: - It is a final project/product which is handed over to the customer

A release consist of entire activities write from Requirement, Developer and Testing till it is handed over to the customer.

UAT: - We check the application before UAI for customer satisfaction and we check for business scenarios, real time scenarios.

On the separate environment called as UAT environment.

UAT is done by separate team called as domain expert who is having the knowledge on the application. UAT is done by customer (Domain expert).

Smoke Testing/Sanity/Skin/Dry Run/BVT (build verification test): -

Checking the basic & critical feature of an application before processing with deep regressing testing is known as smoke testing.

Here we only check the positive flow of an application to check if the build is testable or not.

When we do smoke testing we can identify the blocker bug at early stage so that test engineer won't sit idle. They can proceed testing other modules. Developers are not under pressure. It is time saving process.

Login→Home Page→Compose→Sent Item→Inbox→Chat→Help→Logout

Scenario – 1: - Build 001

Testing the build

Blocker Bug = later stage

- ✗ Time consuming
- ✗ Developers under pressure
- ✗ Test Engineer sit idle

Scenario – 2: - Build 001

Basic & Critical feature (+flow)

We should have some product knowledge

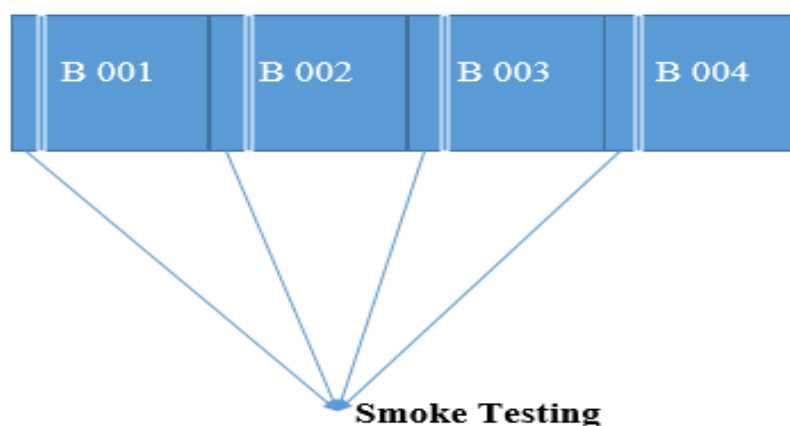
If blocker bug = Early stage

- ✓ Test Engineer can proceed testing
- ✓ Developer won't be in pressure
- ✓ Time saving

We go smoke testing when the build is installed on the test server so that test engineer may found some blocker bug if possible.

If they found we can proceed with independent modules.

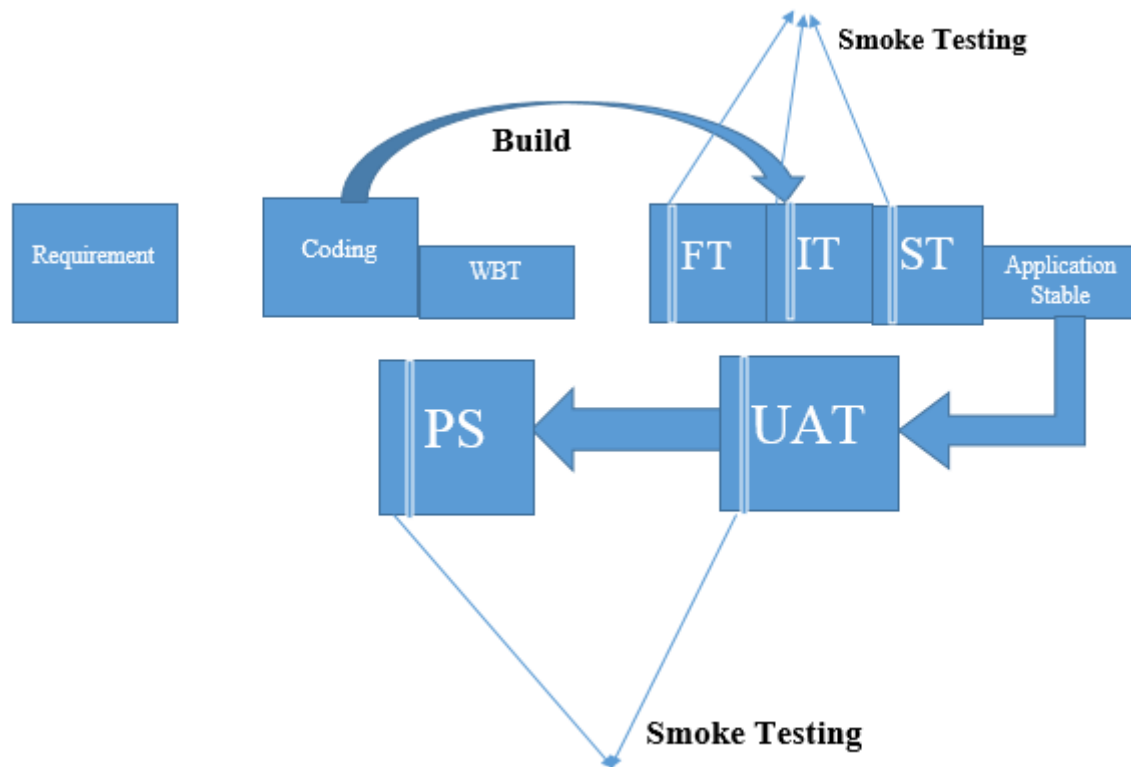
When we go for smoke testing on every build?



Whenever new build is installed we make shore that build is testable or not

Build Verification Testing: -

In smoke testing we verify the build is testable or not hence it is also known as Build Verification Testing.



Whenever there is a installation we go for a smoke testing every time

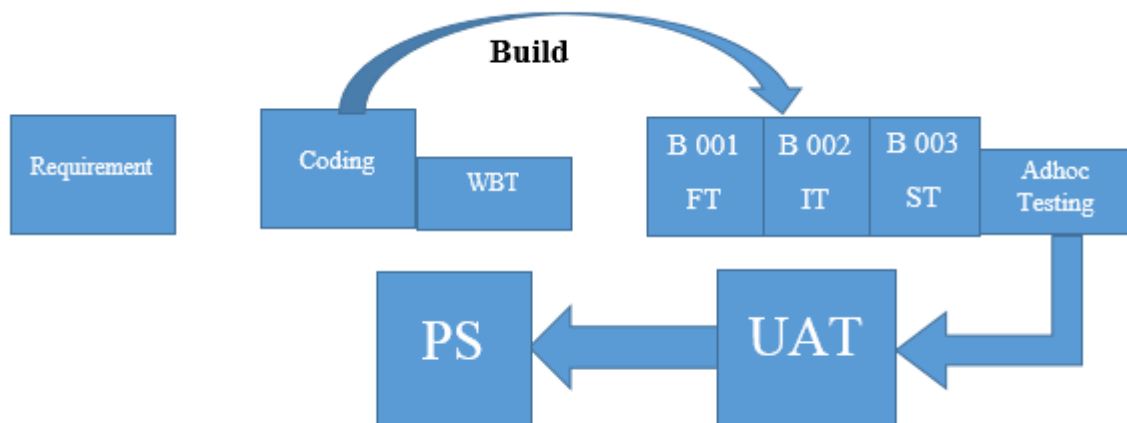
In production server Smoke Testing done by → BA/DM/TM/BT/Customer

Some Testing we go if it is not told by the Team Lead/manager for a good approach

Formal → Informed to do smoke testing

In-formal → No information give then also we have to go for Smoke Testing

Adhoc Testing: -



This testing we do when the build is checked sequence then we go for Adhoc testing by checking the application randomly.

When the product release to the market we go for Adhoc testing because the customer never uses the application in sequence/systematically for that sake we check the application by going for a Adhoc testing by checking randomly.

Checking the application randomly without following any sequence or procedure since user don't know how to use the application they may use it in a random way and find some issues to cover this we do one round off of Adhoc testing.

Application working in a systematically then we go for Adhoc

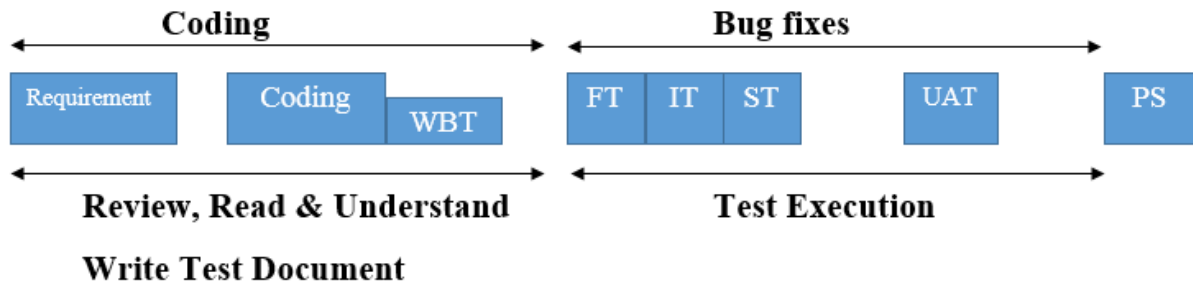
We go for Adhoc when we have time to do.

Smoke Testing	Adhoc Testing
As soon as build installed	After application working systematically
Always as good approach to start	Only if time permits
Checking in positive flow	Most of scenarios negatively

If we don't have any doc →time consuming, test engineer forget, test engineer, test consistency (they won't be sequence, they may forgot some scenarios).

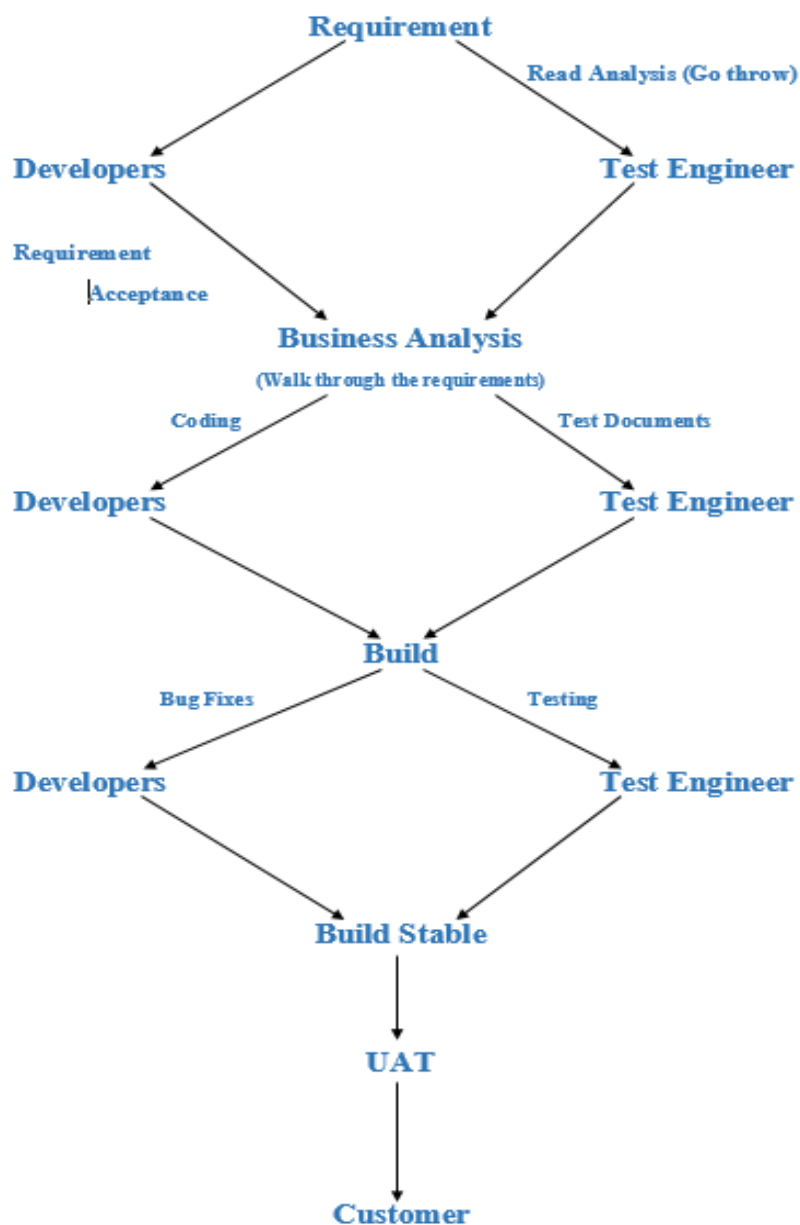
If we go doc →No time consuming, test engineer won't forget, other person review & execute, Accuracy maintained.

Test Document



These are the necessary documents that need to be prepared by test engineer before the test execution process.

We do this when developers are busy in coding.



When the requirements are collected by the business analysis that is stored in the requirement repository (storage) where the team lead will assign the modules for the good accuracy and we go through this requirement for some days as per given time. Then we have a meeting with the BA where BA walk through the requirement until all the requirements are explained if we don't understand anything in this meeting we have to clear that and this meeting we call as requirement acceptance.

Then we go for writing the test documents mean while the developers are written the code after that build installed on the test server we undergo testing the build using the test documents which we have prepared if we encounter any bug that will be sent to developers they fix and we need to re-test that bug this process goes until the build stable then customer undergo for UAT for their satisfaction.

How to write Test Scenarios: -

- ➔ Enter valid user name / password & check for home page
- ➔ Enter invalid user name / password & check for home page
- ➔ Leave user name / password blank & check for error
- ➔ Enter valid / invalid & click cancel & check for fields reset
- ➔ Enter data & check for data in home page

- ➔ Compose mail send it & check for conformation message
- ➔ Compose mail click on save as draft & check for conformation message
- ➔ Compose mail click on close & check for conformation save as drafts

We go for test scenarios because to check the flow & how we can go for multiple ways

- ➔ Simple language it should be
- ➔ Not in a paragraphs (should be in 1-2 lines) easy to understand
- ➔ Module by module – we don't miss any scenarios & proper flow we follow
- ➔ Every sentence must have Do's & checks

Test Scenarios: -

It is a document that defines the multiple ways or combinations of testing the application generally it is prepared to understand the flow of an application.

Test cases: -

It is an identical document that describe step by step procedure to test an application.

It consists of the complete navigation steps/inputs and all the scenarios that need to be tested for an application.

Template for Test cases						
	Test case Name/ Id					
	Test case Type					
H	Requirement #					
E	Module Name					
A	Status					
D	Sevarity					
E	Precondition					
R	Test Data					
	Release					
	Version					
	Brief Description					
	Steps	Description	Inputs	Expected Output	Actual Result	Status
B						
O						
D						
Y						
F						
O	Author					
O	Created Date					
T	Review By					
E	Approved By					
R						

Note: -

1. Release is the major changes done to any project.
Version are the update for the any application (Minor Changes).
2. Depend on the severity (status) we go for testing
3. Data should be created before testing any thing
Pre-condition changes test cases to test cases
No rules to have pre-condition for all test case
Any data which is not created by the test engineer not a pre-condition

Test Case Template Sheet No 1: - [Test Case Template.xlsx](#)

Test Case Example Sheet No 2: - [Test Case Template.xlsx](#)

Note: - Change the Location of the respective HYPERLINKS

Test Execution: - The process of testing the application by executing the test cases generally we start execution only after the cases are return, review & approved.

Status of the Application: - pass/fail

Pass: All steps should pass according to the excepted output.

Fail: If one step also fails we go for fail status.

Note: - Do not speak about partial until & unless they ask.

Test case design technique: - These are the rules or technique that should be followed by the test engineer while writing the test cases to achieve maximum test coverage.

Different TCDT are: -

1. Error Guessing
2. Equivalence Partitioning
3. Bonding Value Analysis

- **Error Guessing:** - In this type we simply start guessing the values based our understanding of the requirement.
- **Equivalence Partitioning:** -
 1. If the requirement is range of values then derive the test case for 1 valid & 2 invalid inputs.
 2. If the requirement is set of values then deriving the test case for 1 valid & 2 invalid inputs.
 3. If the requirement id Boolean (true/false) then derive the test case for both true / false values.
- **Bonding Value Analysis:** - If the requirement is range of values that is A,B then derive the test case as follows i.e. A, A+1, A-1, B, B+1, B-1.

Drawbacks: -

Error Guessing: - Depends on the person to person

Minimum test coverage may not be achieved

Boundary values not covered

Equivalence Partitioning: - Boundary values not covered

Range: - Whenever we want to find range values we go for equivalence partitioning we go for this to achieve minimum test coverage after that we go for error guessing to achieve maximum test coverage.

Set: - Whenever we have to test a set of value we go for 1 +ve, 2 -ve then we go for error guessing we need to check all the set of values as the requirement says to us.

Boolean: - Here we go for true & false values

Sno	Description	Input	Expected	Note
1	Select Valid	NA	TRUE	values changes according to the req we cant go for next question only one can select at a time
2	Select InValid	NA	FALSE	
3	Do not select	NA	Do not select any thing error msg should show	
4	Select Both	NA	We can select any one Radio button	

Boundary Values: - Here we cover all the boundary values

A, A+1, A-1

B, B+1, B-1

500,501,499

5000, 5001, 4999

Ex: - User Name Accept 2-5 Characters

- ✓ ab
- ✓ abc
- ✓ abcde
- ✓ abcd
- ✗ a
- ✗ abcdef

Boundary values will accept Alphanumeric, Numeric, and Special Character etc.

Fixed / Straight forward values we go for error guessing like password accept 3 values.

Checkbox: - We have to go for a set [Equivalence Partitioning]

Equivalence Partitioning: - Pressman Method

1. Amount Transfer (100000-700000)
1 lakh Accept

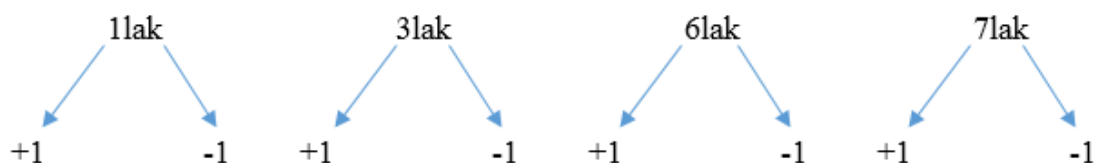
General Practise method: -

Range + Percentage given (1 lakh – 7 lakh)

Like: - 1lak – 3lak – 5.60%

3lak – 6lak – 3.66%

6lak – 7lak – free



If have loans types things we should go for general practise and divided the things to intervals to achieve minimum test coverage after that error guessing.

General Practise method: - Whenever the requirement is range + criteria divide the range into intervals & check for all these values.

Amount Transfer Requirement

Amount Transfer

FAN:

TAN:

Amount:

Test Case for Amount Transfer Sheet 3: - [Test Case Template.xlsx](#)

Advantages / why Test Documents:

- To document knowledge of Test Engineer (he may Forget).
- To Achieve Maximum Test Engineer (+ve, -ve values)
- To have Test Consistency
 - Proper flow
 - Same set of scenarios are tested across different Release if required.
- Time Saving
 - Re-Used by Test Engineer as well as New Test Engineer
- It becomes process oriented rather than person oriented.
- Test Cases are re-used by auto-machine Test Engineer to create test scripts.

Test plan

Test plan: - It is a document which is prepared by the managers or test lead. It consists of all information about the testing activities.

Test plan components or attributes: -

Aim: - It indicate the aim of the application means the application behaviour, goal etc.

Objective: - It consists an information about modules, features, test data etc.

Scope: - This contains an information which need to be tested with respective of an application.

Test methodology: - It contains an information about performing a different kind of testing like FT, IT, ST etc.

Schedule: - It contains information about the timing with respective to particular task.

Ex: - Writing test cases

Execution process

Template: - It is a format used to maintain Generic.

EX: - different types of templates

1. Test case Template
2. RTM Template
3. Bug Report Template

Test Environment: - It consist of software and hardware configuration

S/W configuration means information about OS/Browsers

H/W configuration like Ram, Rom, Processors

Assumption: - It contains an information about a problem may be occur during the testing process.

Risk: - The effect of the problem during testing process.

Ex: - Effect for an application, release date become post pond

Mitigation Plan or Contingency Plan: - It is a back-up plan prepared to avoid the assume risks

Assume: - There are the problem that may encounter during testing process

Test deliverable: - The test documents which is given to the customer while giving the product to him.

Test Cases – Test Scripts – RTM – Test Execution Report – Release Notes like

User manual/Installation process/List of platform on which Tested/list of available bugs in current release/list of fixed bugs in previous release

Role & Responsibility: - It defines the complete task which need to be performed by the entire testing team.

Role: Test Manager

Name: _____

Responsibility: Write/Review test plan

Meeting with Development Team/Test Team/Customer

Handling Escalations

Role: Test Lead

Name: _____

Responsibility: Write/Review test plan

Approve test cases

RTM & Reports

Assign Modules

Daily status meeting

Role: Test Engineer 1, Test Engineer 2, Test Engineer 2

Name: _____

Assign Modules: M1, M2, M3

Responsibility: Read, Review & Undergo requirement

Write/Review/Execute Text Documents

RTM for Respective Modules

Defect Tracking

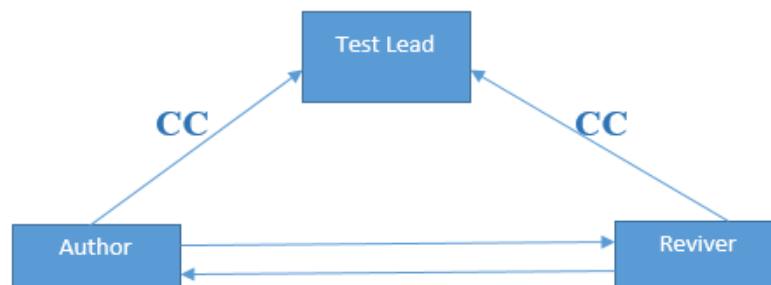
Test Case Review

If we don't go for this we – Miss out some scenarios – Accuracy won't be there – Test Engineer won't be serious

This is done after writing all test case and before the test execution of the test case

Note: Once the Test Cases are written it should be Review to check for the proper flow and maximum test cover before the execution so that every test engineer will be serious about writing the test cases.

After Author wrote the Test Cases that will be sent to the Reviver and review the Test case allowing with the corresponding requirement and check the correctness of Test Case by proper flow & maximum Test coverage and he found any mistake that will be written separate document called as Review Document and Team Lead be in the loop this process goes until the test case stable and it finally sends to the Test Lead to Approve the Test case.



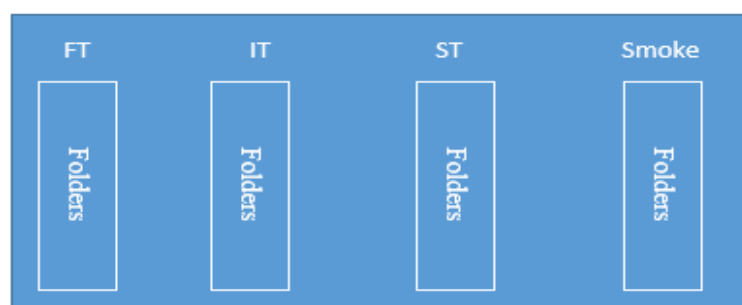
Once the Test Cases are Written, Review & Approved they are stored in the centralized location know as Test Case Repository.

These based line Test Cases (Return, Review, and Approved) can be used for Test Execution Process.

Review Process:

- Look for flow & Maximum Test Cases
- Find Mistake not solutions
- Both Author & Reviver are responsible for Test Case
- Review content not Author

Test Case Repository: - File Storage location (we call it as base lined Test Cases)



RTM (Requirement Traceability Matrix)

CRM (Cross Reference Matrix)

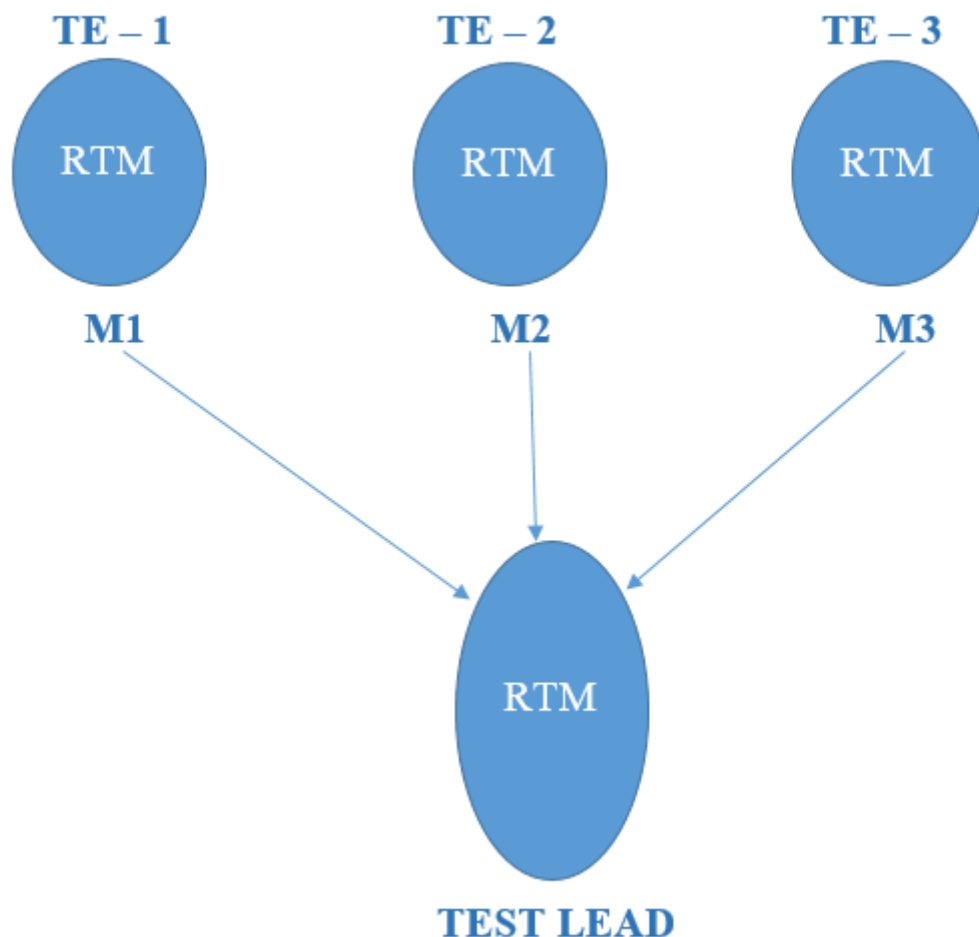
We go for RTM after approval & before execution so that we don't miss out any Test Case for any requirement.

We don't write RTM while writing the testing because it can be incomplete and after writing the test case we don't go here because the test case can be reject.

RTM: - It is a Document which is prepared before the test execution process to make shore that every requirement is covered in the form of Test case so that we don't miss out testing any requirement.

Test engineer will prepare RTM for their respective assign modules and then it will be sent to the Test Lead. The Test Lead will go repository to check whether the Test Case is there or not.

Finally Test Lead consolidate and prepare one necessary RTM document.



RTM Template: -

Requirement Traceability Matrix Template				
Requirement No	Module Name	High Level Requirement	Low Level Requirement	Test Case Name

KVREDU

Defect: - Deviation from requirement

Bug: - Informal name of Defect

Error: - Mistakes in coding

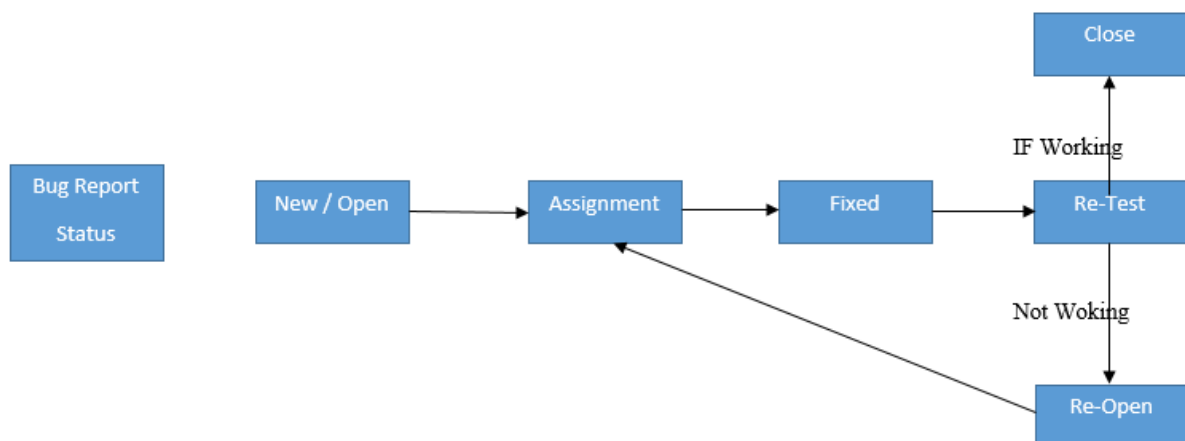
Issues: - Problem found by customer

Failure: - Many Defects leads to failure of application

Why do we get Defects?

- Wrong Implementation → Not working as per Requirement
- Missing Implementation → Not There
- Extra Implementation → Extra things as per Requirement

Defect / Bug Life Cycle: -



It indicate the life cycle of bug write from the stage it was found – Fixed – Re-tested – Closed.

As soon as we get a bug the status is new or open. This bug is reported to the concert person by changing to assign. The developer first go-through the bug then reproduce and they do the necessary code changes & change the status to fixed.

The Test Engineer need to Re-test the fixed bug and change the status according i.e. close if the bug is fixed properly or Re-open if the bug is still exist.

This process continues until the bug is fixed & closed.

Note: - Whom to assign the bug

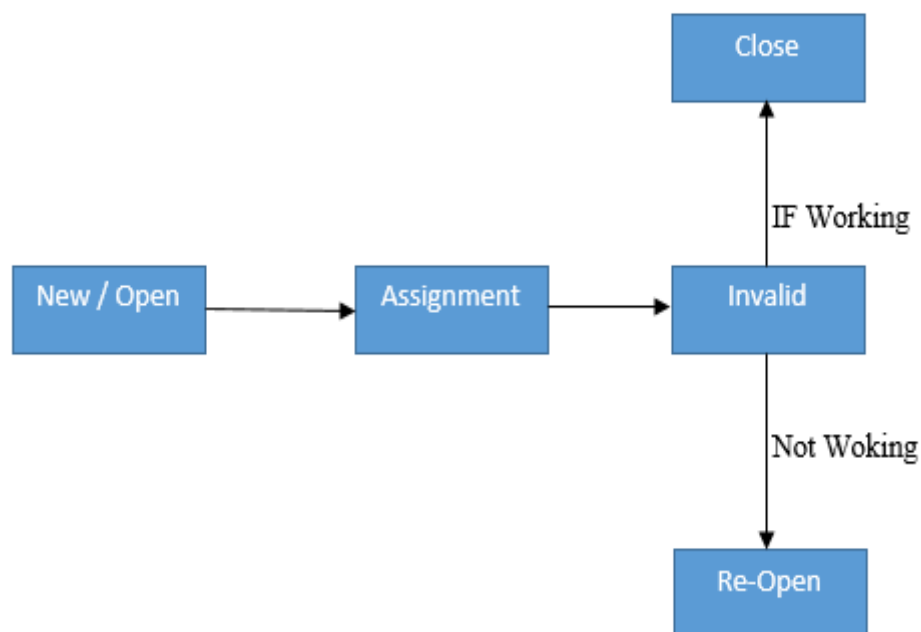
1. **Developers:** - If we know who has developed that module.
2. **Developer Team:** - If we don't know the developer who has developed the module.
3. **Test lead:** - We don't have any contact with the development team.

If bug is fixed and closed then if it is having any impact on the other module then we go for new bug report.

If the status of the bug is Re-open (not fixed) if it has effect on other module then we have to prepare new bug report.

Other Status of bug:

1. **Invalid / Rejected:** - Developers doesn't accept it as a bug.
 - ✖ Test Engineer Misunderstanding Requirement
 - ✖ Developer Misunderstanding Requirement



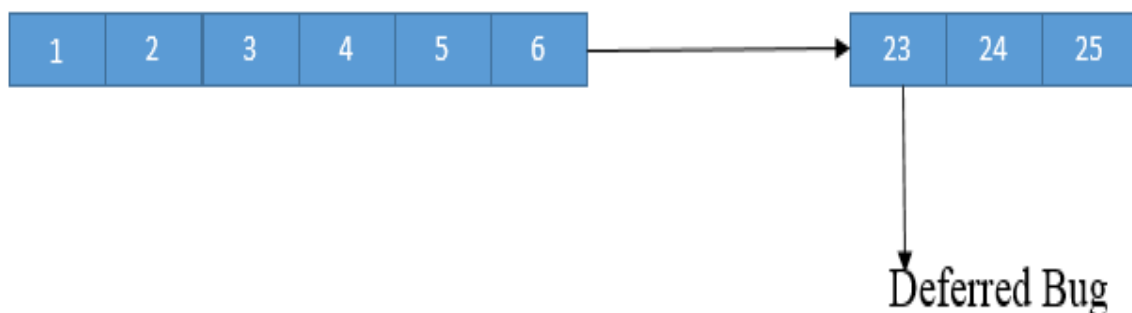
When the Test Engineer wrote a wrong Bug Report because of Misunderstanding Requirement then the developer will give status as Invalid and send it back. (Some time Developer can)

2. **Duplicate:** - Same bug has been reported multiple times
 - Dependent Modules
 - Common features
 - First go to bug repository search – if bug exit do not lock

If bug does not exit (Lock a new bug)

- If the status of bug (fixed) No need of lock a same bug (That need to be Tested Again)
- If bug found that need to be send to Developers and Test Engineers adding them in [CC].

3. **Not Reproducible:** - Developer accept the bug but not able to Reproduce due to some reasons.
- **Incomplete Bug Report**
 - ✖ Navigation steps
 - **Environment Mismatch**
 - ✖ Server Mismatch: - TE – Test Server / Developer – Development Server for Reproducing.
 - ✖ Platform Mismatch: - TE – Different Platform / Developer – Different Platform
 - **Data Mismatch:** - Different Values used by TE while testing & Developer uses different values.
 - **Build Mismatch:** - TE tested in one Build and developer testing same bug in another build. The bug may be automatically fixed while fixing another bug.
4. **Inconsistent:** - Bug found at some time and some time it won't happen.
5. **Can't fixed:** - When developer accepting the bug and able to reproduce as well but can't do the code changes due to some constraints.
- No technology support
 - Bug is in core of code
 - Cost of fixing bug is more than keeping it. (This bugs is called as Minor Bug).
6. **Deferred / Postponed:** - The bug has been postponed to the future release due to time limitations.



7. **RFE (Request for Enhancement):** - Suggestion given by the Test Engineer for the improvement of the application.

Requirement

User Name
 Password
 Cancel
 Ok

Developed

User Name:
 Password:

Cancel
 →
←
 OK

Severity: - Impact of bug on application as (Blocker/Critical/Major/Minor)

Priority: - Which bug is to be fixed. (P1, P2, P3, P4) (Urgent/High/Medium/Low)

Blocker/P1	Critical/P1	Major/P3	Minor/P4
Click on Amount Transfer Blank Page Displayed	Click on Amount Transfer Transfer a conformation msg Login User Check Balance	Click on Amount Transfer Transfer No conformation msg Login User Balance There	All UI Issues

Bug Report Template

Bug Report Template	
Test Case Name	
Module	
Requirement	
Date	
Repoter	
Assign To	
CC	
Status	
Release	
Version	
Sevarity	
priority	
Server	
Platform	
Test Data	
Build	
Attachment	
Brief Description	
Description	
Steps to fallowed	
Observation	
Expected Result	

Bug Report in Sheet 7: - [Test Case Template.xlsx](#)

Test Execution Report

It is the document prepared by leads after the execution is completed which define the stability of the product.

It consist of information like the number of cases return, executed, pass, fail and their percentage.

Every module have separate spread sheet of their respective module.

Test Execution Report						
Module Name	Total No. of TC written	Total No. of TC Executed	Total no. of TC Pass	Total No. of TC Fail	Pass %	Fail %
Total						

Entry & Exit Criteria: -

These are the condition that need to be satisfied before starting and stop the testing.

The entry criteria starts when Coding should be completed necessary Test Document should be ready Build should be installed on Test Server & Resources must be ready similarly the exit criteria would be

All Test case need to be executed

Majority of the case should be pass

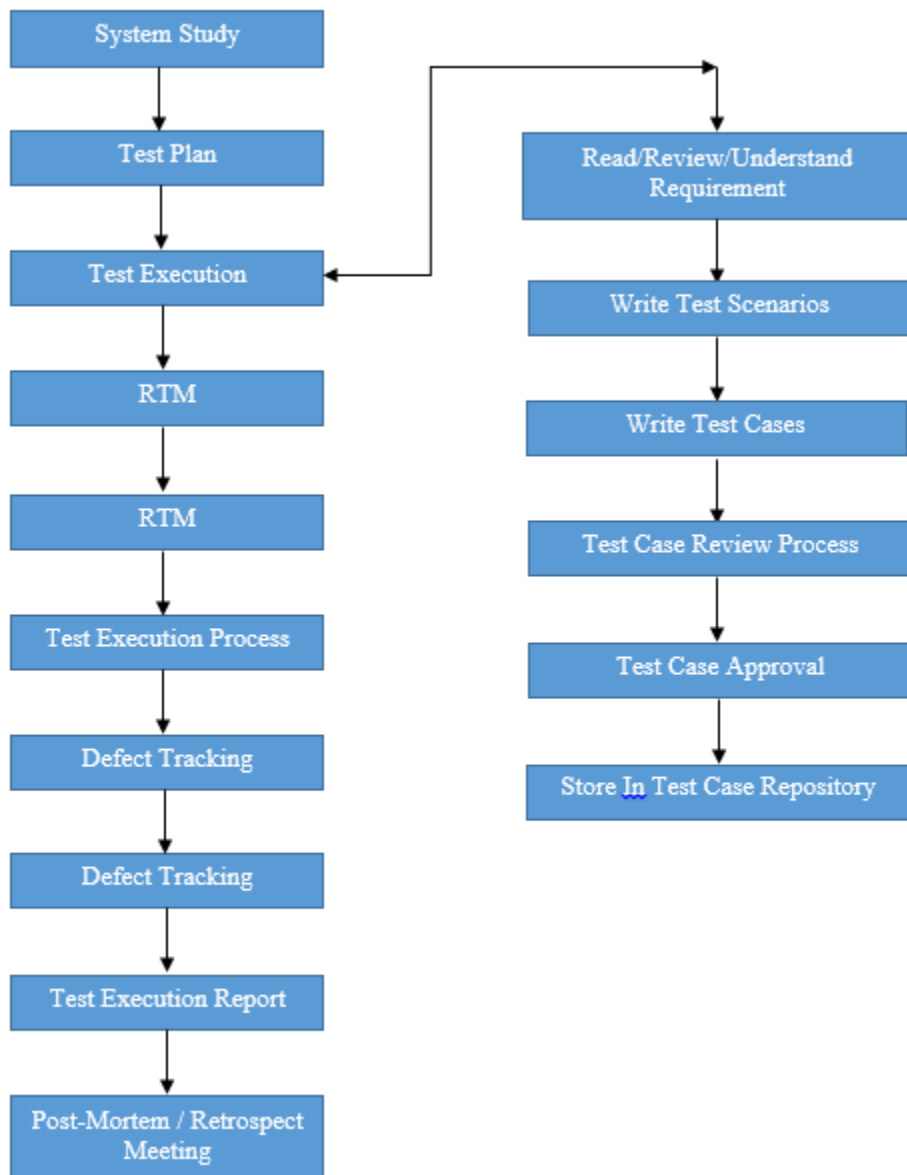
There can't be any Blocker, Critical, Major Bugs

Whereas some minor bugs can exist.

% of Test Case Execution		90	95	100	
% of Test Case Passed		85	90	95	
Based on Severity bugs					
	Blocker	0	0	0	
	Critical	1 .. 5	1 .. 3	0	
	Major	7 .. 8	5 .. 6	0	
	Minor	8 .. 9	6 .. 8	2 .. 5	
			Exit FT & Entry IT	Exit IT & Entry ST	Exit FT & Entry IT
		F T	IT	ST	
	Coding Should Ready				
	Test Document Ready				
	Build Installed Test Server				
	Resource Ready				

Note:	FT - Functional Testing
	IT - Integration Testing
	ST - System Testing

STLC (Software Testing Life Cycle)



Whenever Client gives requirement that will go through by the Manager/Test Lead. To understand the complicity of the requirement.

Depending on that they will prepare Test Plan in which they write all necessary things required while testing.

These document is handed over to the Test Engineer in which they have mentioned what things need to be done while testing.

We go for Test Execution and we require some document which need to be prepared and we go for Read/Review/Understand requirement depending on that we write Test Scenarios where we write end to end flow of an application which is high level Test Document.

This Document need to be converted to Low Level Document called as Test Case in which we mention all the navigation steps then it send to other Test Engineer for Review this

process goes until both (Author & Review) satisfies. Then it send to the Test Lead for Approve and it is stored in the centralize location called as Test Case Repository.

Then we prepare RTM so that we don't miss out any Test Case for any Requirement up to this time Developers completed coding and Build installed on the Test Server.

Now we start testing execution using Test Case Document.

While testing we may encounter some bugs that need to be fixed by the developers this process goes until the product stable. After completing the execution we prepare Test Execution report which talk about the stability of the product which is handed over to the customer.

Then Manager/Test Lead call a retrospect meeting where we discuss that do the Test Engineer got any problems in Test Plan, things need to be done in the next release.

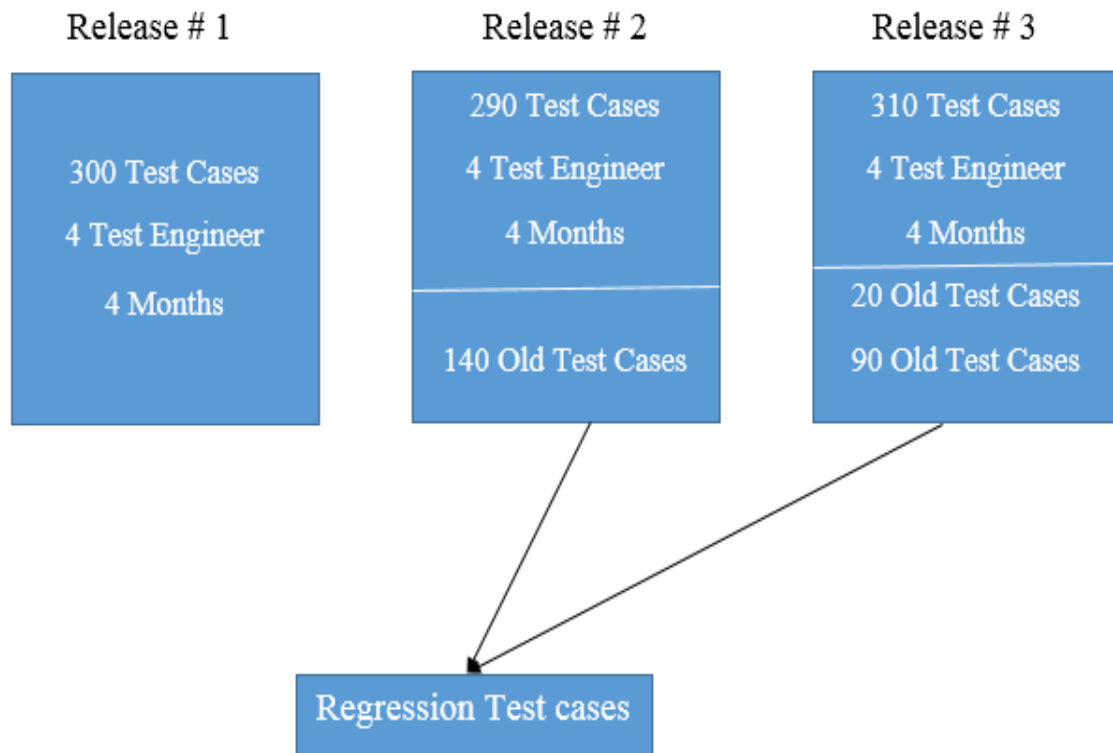
Regression Testing

Regression Testing type 1: -

Whenever there are new Release for some project then we go for Regression Testing because the new feature may affect the old features in the previous releases.

Regression Test Cases:-

These are the Test Cases of the old releases Text Document which need to be Re-executed.



Challenges of Regression Testing

- ✖ Identify Impact Area
- ✖ Time Constraint
 - Test Cases Increases Release by Release
 - Less Resources
- ✖ No Accuracy
 - Repetitive Task
 - Monotonous Task

Types of Regression Testing

- Unit Regression Testing: - Check any changed unit [No Impact Area]
- Regional Regression Testing: - Check Changes + Impact Area

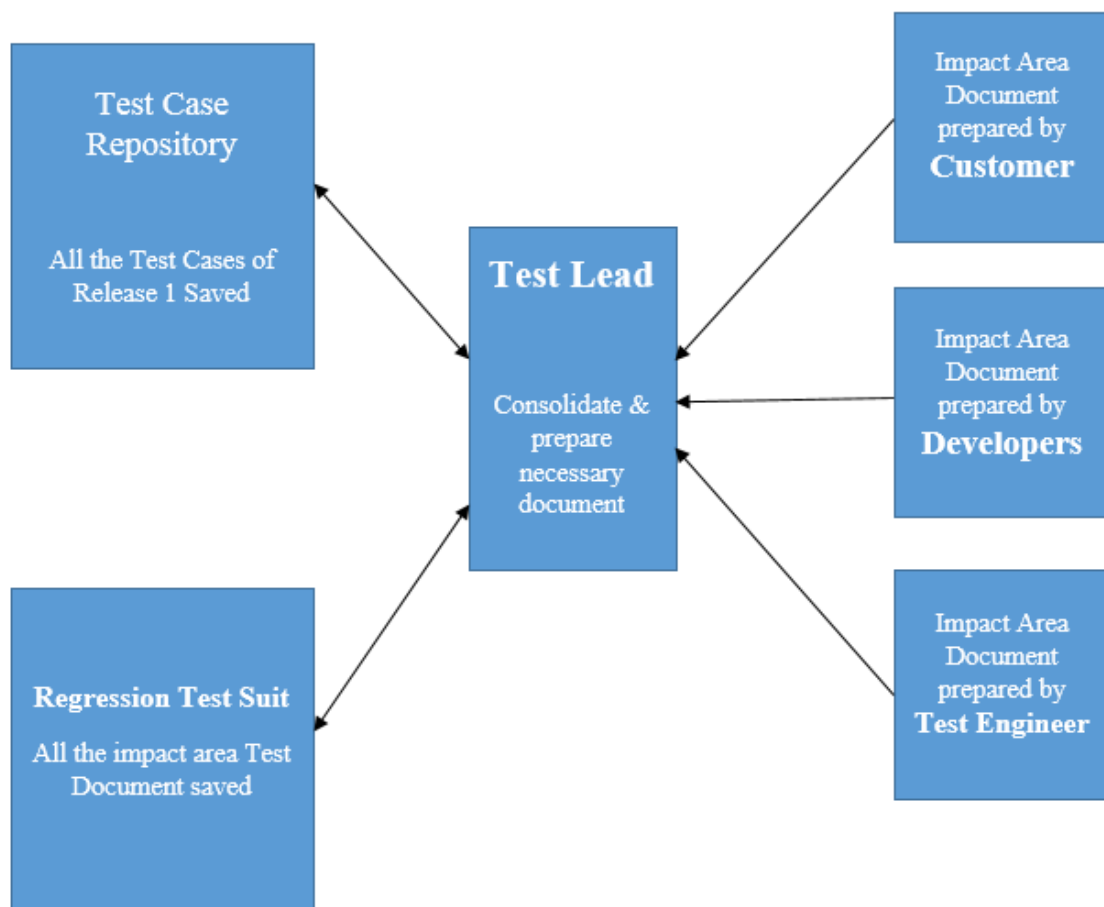
Impact Area is analysed by

- Customer (Business Knowledge)
- Developer (Code Knowledge)
- Test Engineer (Product Knowledge)

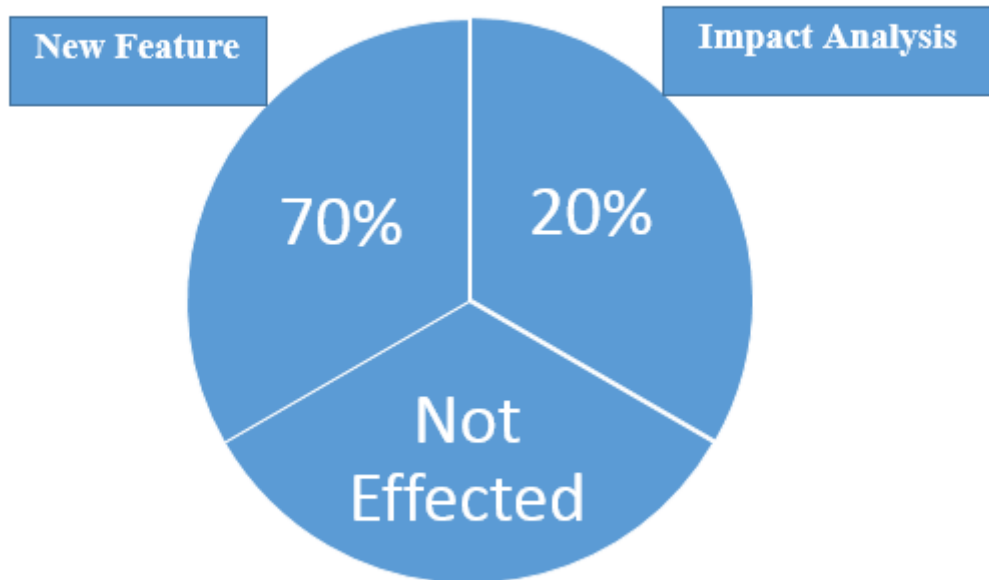
Because if single person do he may not cover all the impact area so we include all this person so that we may cover maximum impact area.

Note: - Impact Analysis should be done at early stages of the releases to avoid any major risk.

Whenever we get new features (Modifying Features) we will analysis the impact area while same time the Developer also prepare the impact area (document) and customer also given impact area document so that we can achieve maximum impact analysis. All this document given to the Test Lead to consolidate. Then Test Lead take the help of RTM and pick the necessary Test Case from the Test case Repository and those files placed in the Regression Test Suit. Meanwhile we are working on the new features when the new features are stable then check the impact area using the test case until it is stable for old features + new features.

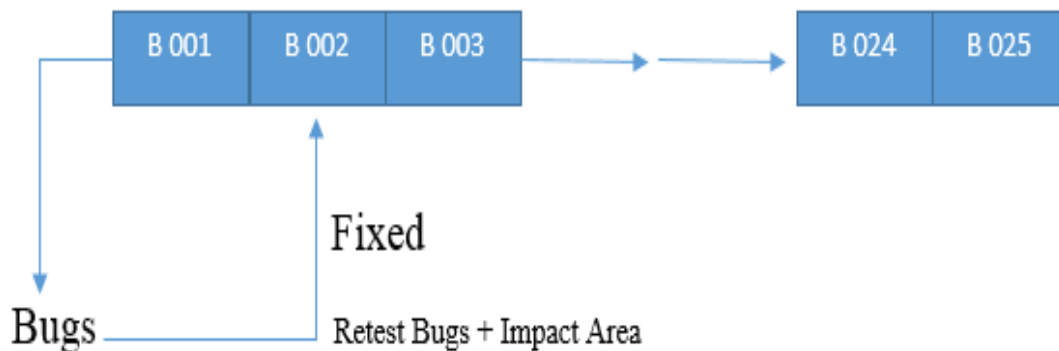


- ✓ **Full Regression Testing:** - Check changes + Complete Old Features
- Major Changes
 - Life Critical / Machine Critical



Note: - Don't speak about types of Regression Testing until unless asked.

Regression Testing Type 2: -



Whenever Bug fixed we retest the bug and if there is any dependent module we go for a Regression Testing.

What is Regression Testing?

Retesting or Re-executing the old features or test cases to make sure that changes due to enhancement or bug fixes has not affected the existing functionality is known as Regression Testing.

When Regression Testing?

Whenever there are code changes due to enhancement or bug fixes.

Why we do Regression Testing?

To make sure existing features are not effecting due to changes.

How to do Regression Testing?

We prepare Regression Test Suit then Auto-machine Testing Process.

Who will do Regression Testing?

Auto-machine Test Engineer.

Re-Testing: - Bug + Fix → Re-Testing bug only (No impact Area)

Regression Testing: - Bug + Fix → Re-Testing + Impact Area

Usability Testing

Checking the user friendliness of application is known as Usability Testing.

User Friendly

- ➔ Easy to understand – all the features must be visible to end users.
- ➔ Easy to Access.
- ➔ Look & Field should be good.
- ➔ Faster to Access (Response time should be fast)

Why Usability Testing?

Customer must be comfortable with your application with the following parameters.

- ➔ Flow of an Application be good.
- ➔ Navigation steps should be clear.
- ➔ Content should be simple.
- ➔ Layout should be clear.
- ➔ Response time.

What features has to be tested in Usability Testing

- ➔ How easy it is using the application.
- ➔ How easy to learn application.

Components of Usability Testing

1. **Learnability:** - End user takes minimum amount of time to learn the basic task.
2. **Efficiency:** - End user who is an expert takes minimum amount of time to perform his/her basic task.
3. **Memorability:** - When you are not asking an application for a span of time & return to the application & trying to do the basic task without any help then the memorability of an application is good.

When we are not able to perform basic task without any help after a duration of time period & then your memorability for an application is poor.

4. **Errors:** - Customer are prone to make errors and here we try to help the end users to resolve those errors and perform his task.
5. **Satisfaction:** - Customer must be satisfied with the application he should feel free to use the application.

Advantages of Usability Testing

- ➔ Better quality product
- ➔ Customer eager to use the application.

Bugs in Usability Testing

Path holes & potential bugs – which are visible to the developers and the test engineers (when we do Testing we can get) Usability Testing is a wide Testing where we need to have a application knowledge.

Check list – It contains all the document related to usability testing

- ➔ Create check list
- ➔ Review check list
- ➔ Execute check list / Approve check list
- ➔ Derive check list report (Execution Report)

Auto-machine Testing Process

Test Management Tools: - Quality Centre, Test Manger

Bug Reporting Tools: - Jira, Bugzilla, memtis

Functional Testing Tools: - Selenium, QTP, win runner, silk test

Performance Testing Tools: - Load runner, ineter, web stress tool, E-load, new-load

Whenever there is a repetitive or monotonous or we have multiple regression life cycles or multiple releases we go for Auto-machine.

- ✓ Time Saving
- ✓ Re-Usable

Whenever we get a requirement for a new release we check for impact area and prepare regression test case suit and this regression test cases are given to Auto-machine team and we check the new feature mean while Auto-machine team write the Test Scripts on the Regression Test Case (Using Tools, Stable Application means previous release).

After completing the new feature testing manually and it is stable then the Auto-machine Team will test the application using the Test Script if they find any bug that will be check by manual Test Engineer and prepare Bug report after code fixed the manual Test Engineer need to test it again and check for any impact area is there or not.

Then Auto-machine team again run the Test Script and check for the bugs this process goes until the new feature & Regression Test Case working properly and product is stable.

Difference of Smoke Testing & Sanity Testing

Smoke Testing	Sanity Testing
As soon as build installed	As soon as bug fixes done
It is shallow & wide	It is narrow & deep
It can be documented / scripted	It can't be documented / Scripted

Difference of Functional Testing & Non-Functional Testing

Functional Testing	Non-Functional Testing
Some Functional Testing like Unit Testing,	Some Non - Functional Testing like Usability Testing,
Integration Testing,	Performance Testing,
System Testing,	Compatibility Testing
UAT Testing	

Difference of Static Testing & Dynamic Testing

Static Testing	Dynamic Testing
Here we check Code/Application without Executing/Testing	Here we check Code/Application by Executing/Testing
It includes activities like Review , Walk-Through Etc.	It includes activities like UT,IT,ST & UAT
It is Verification Process	It is a Validation Process

Difference of Quality Assurance & Quality Control

Quality Assurance	Quality Control
We check if we are developing right product	Check if we have developed produce right
It includes activities like review , Walk-Through Etc.	It includes activities like UT,IT,ST & UAT
It is Verification Process	It is a Validation Process
It is Process Oriented	It is Product Oriented
It is Preventive Method	It is Detective Method

Alpha & Beta Testing is done in Product based companies.

Note: - Generally Alpha & Beta testing is done in product based companies

Alpha Testing: - It is a type of testing done by the Test Engineer at developer site (where the Application was developed)

Beta Testing: - It is done by the End users before Accepting the product at customer site where uses the Application.

Exploratory Testing

- Exploring Application: - use in all possible ways
- Understanding flow application
- Prepare Test document
- Test Application

We go for this approach when there are no requirement but requirement exist but it is not clear in these cases first we need to explore the application by using in all possible ways. By doing this we will understand the application we can prepare necessary text document and then start testing the application this approach is known as Exploratory Testing.

Dis-Advantages of Exploratory Testing

- ✖ Time Consuming
- ✖ Bugs can Miss-Understand as feature
- ✖ Feature can Miss-Understand as bug

As a Test Engineer I would never suggest to go for Exploratory Testing but if this situation arrives where requirements are not there we may go for Exploratory Testing.

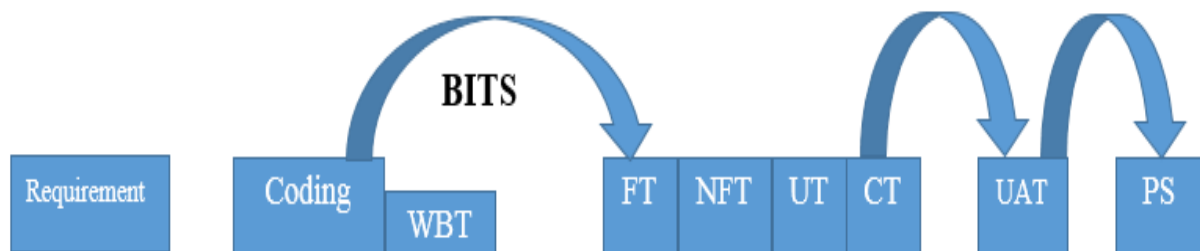
Domain Expert will tell us about the Application & we explore, understand and write the necessary document. This document is sent to the domain expert. This fellow will go through the document.

Compatibility Testing

Checking application on the different software & hardware platform.

Software: - Check on different Operating System & Browser

Hardware: - Different sizes & Makes.



Whenever the functionality of an application is stable then we go for compatibility testing.

We pick the operating system & browser based on most commonly used.

Note: - VMware software – it is a software with the help of one physical machine can be divided into multiple Virtual machine can be access simultaneously.

For doing compatibility testing we use VMServer where we install all necessary operating system and browser and we access that server using Remote Desktop connection.

Bugs found in compatibility testing like

Alignment issues

Scattered objects

Overlap object

Colour & Font sizes

Forward Compatibility Testing: - Latest Version of platform (s/w)

Win 7 → Win 8 → Win 8.1

Backward Compatibility Testing: - Xp → Vista → Win 7 → Win 8 → Win 8.1

Performance Testing

- Checking the response time
- Checking the load – N no of users using
- Checking the stability – N no of users using

Checking the behaviour of an application by applying load is known as performance testing

Here we check for various factors like Response time, Load, Stability

Response Time: - It is the time taken by the server to response to the client request.

Load: - N – No of users using the application simultaneously

Stability: - N – No of users using the application for a certain period of time.

Functionally stable application then only we go for performance testing.

N – No of users using simultaneously & they may find some issue

Note: - performance testing not be done manually since it costly & accurate result can't be maintained.

How to do Performance Testing

- Identify performance scenarios
 - Most commonly scenarios
 - Most critical scenarios
 - Scenarios huge data
- Create script according scenarios with the help of tools
- Distribute load according to [Usage Patten]
- Execute the scripts
- Result in graph

We install tools in the Test Engineer Machine and access the test server and write some script and give some virtual user in the tool and run the tool.

If the response is not meeting requirement time response

- Analysis Result
- Identify Bottle Neck (Performance Issue)
- Tuning { Code | H/W – S/W | Network }
- Re-Run Script
- Result = Requirement

Types of Performance Testing

1. **Load Testing:** - Checking the behaviour of an application by applying load less than or equal to desired load is known as Load Testing.
2. **Stress Testing:** - Checking the behaviour of an application by applying load greater than the desired load.
3. **Scalability Testing:** - Checking the behaviour of an application by increasing or decreasing the load in a particular scales.

Upward Scalability Testing

1000 – 4.8 /sec

1100 – 5.1 /sec

1200 – 5.8 /sec

1300 – 6.2 /sec

1400 – Crash

Downward Scalability Testing

1000 – 6.2 /sec

900 – 6 /sec

800 – 5 /sec

700 – 4.9 /sec

4. **Stability Testing:** - Checking the behaviour of an application by applying load for a certain duration of time is known as Stability Testing.