SOFTWARE TESTING

Syllabus

Theory - Software overview - CMM level of companies - project designations - SDLC - Testing basics and principles - Levels of Testing - Testing Process - Testing stratergies - Test design techniques - Test design documents - STLC - Bug life cycle (defect Tracking Process) - SDLC methodologies - Verification documents - Validation process.

DOMAINS:

1. Finance.
2. ERP (Enterprise Resource Planning)
3. BFSI (Banking Financial Service & Insurance)

TOOLS:

1. QTP (QuickTest Professional) - AutomationTool
2. Qc (Quality center) - Test Management tool but used for defect Tracking process.
3. Selenium

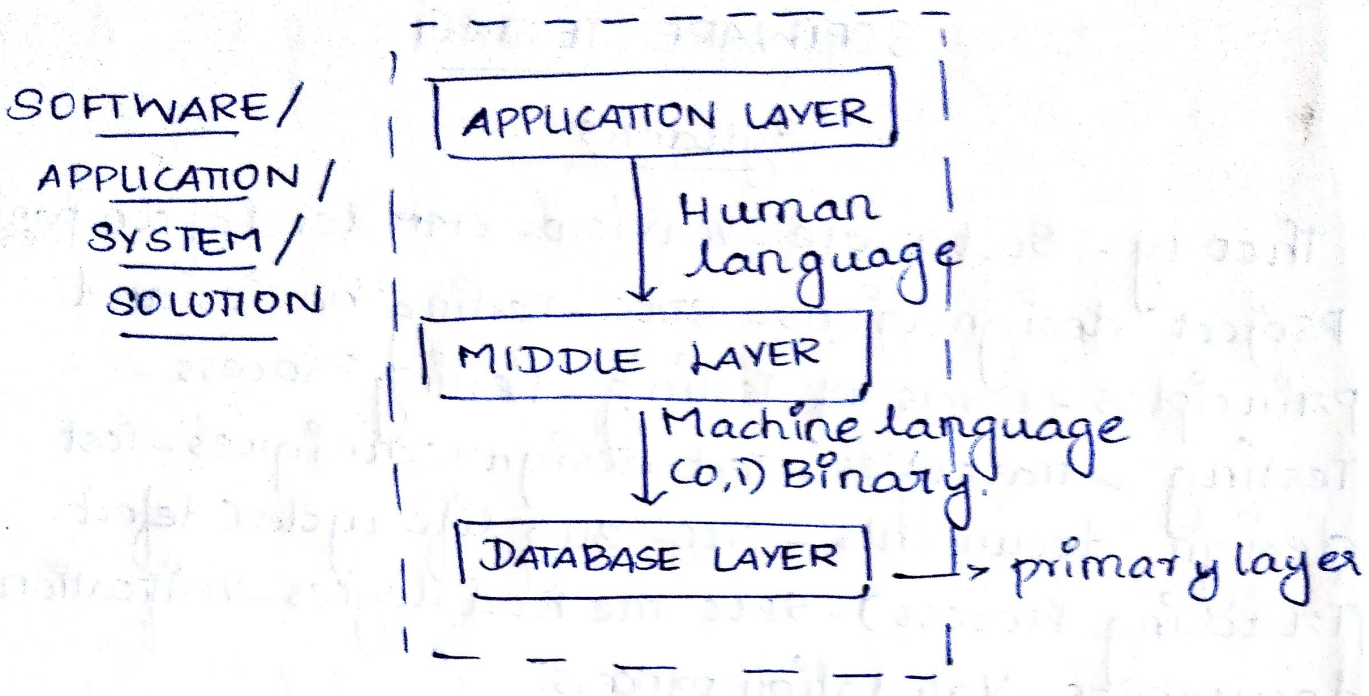
(1.) SOFTWARE OVERVIEW:

MS-office is a Setup/.exe

→ 2 Main Sections 1. OS
2. Harddisk

→ 2 drives in HD
- → one used for os
- → second for database layer

→ 3 main layers
- → Application layer
- → Middle layer
- → Database layer.



APPLICATION (screen)/client

→ Front end language
→ Eg: C, C++, .net, PHP, HTML, VB.
→ Coding

DATABASE

→ Backend language
→ Eg: SQL, MYSQL, Oracle, DBase, access
→ Query

(C) + (Q) ↓ Program (P)

② CMM level of companies:

→ CMM - Capability Maturity Model.
→ It has 5 levels

Level 1 company: Initial stage
[Info ↑ Risk ↓ Quality]

Level 2 company: Repeatable stage
[No procedure but task will be completed]
[Info, Risk, Quality - Medium]

Level 3 company: Defined stage
[structured] [2cmpies - service Prov & client]

Level 4 company: Managed stage
[even critical situation]

Level 5 company: Optimized stage
[Product based cmpy]

Hint: 2 types of companies
1. Project based cmpy - Specific requirement
2. Product " " - Global req
Service provider.

③ PROJECT DESIGNATIONS:

cmpy 1: client → Responsible to provide his need as req.
→ also known as financier / Project Provider.

cmpy 2: Service Provider

① BDE (Business Development Executive)
→ Responsible to market his company profile through his Ad / through door to door marketing.
[MARKETING]

② BA (Business Analyst)
→ Responsible to analyse the client's business and gather the req. from client and his environment to prepare documents called BRS and Usecase. [Technical]

②① BRS (Business Req. Specification)
→ It has general info about client business and his environment.

(2) Usecase:

→ Pictorial representation of docu which has info abt hw the app is going to be used by every user.

→ It is mainly used by testers to identify the scenarios & conditions.

(3) Software Architect:

→ Responsible to consolidate info collected from technical team to prepare a doc. called SRS (Soft. Req. Spec)

→ Baseline doc of project which is used for entire proj. development activity.

↳ It has detailed description abt func. & non fnc. activities of the app.

→ It is used by testers to understand req of the client.

→ Once the SRS is concluded SLA (Service Level Aggrement) will be signed off & SRS will be freezed.

(4) Account Manager:

→ Responsible to manage the entire project aggrement activities and also to share info related to status of project.

(5) Project Manager:

→ Responsible to manage entire team

→ other responsibilities are

1. cost estimation
2. Team Recruitment

3. Training & task allocation
4. Environment creation
5. client interaction & approvals
6. co-ordinating with team.

④ SDLC

→ 6 Phases (stos)

1. Req. gathering
2. Planning & analysis
3. designing
4. coding
5. Testing
6. Implementation, support & maintenance

1. Req & gathering:

→ It will be gathered from client. & doc like BRS, SRS and usecase will be created.

2. Planning & analysis:

→ Planning will be done reg. how Project has to be started team req. deployments phases & mile stone approvals etc.,

→ Risk will be analysed and mitigation activity will be performed

3. Designing:

→ Longest phase.

Types
1. S/m design
2. Architecture design
3. Module design

S/m design:

→ Here SDD (S/m design doc) will be created

→ It contains H/w selection, s/w selection

Server info, s/m config addons etc.,

Architecture design:

→ structure given to the application according to its usage.

1. Single tyre tier → 3layers
2. Two tyre tier → FIFO
3. Three tyre tier → Web based Application
4. Multi tier → various (bank domain)

Module design:

→ Here the req. will be splitted into high level and low level design.

→ Imp(x) for the developers to know the functions.

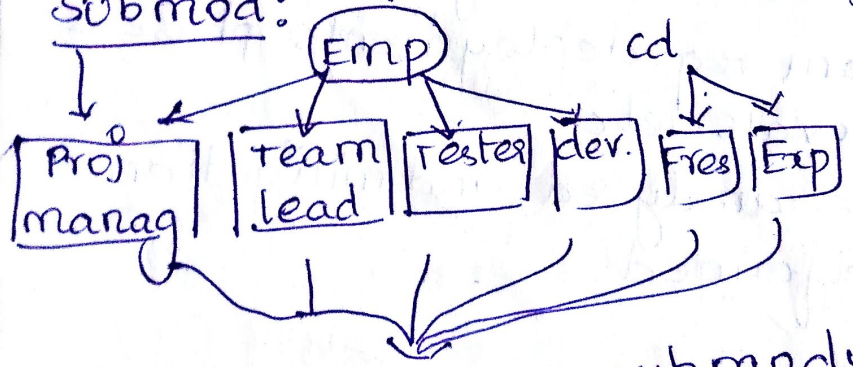
Hierarchy

SRS
↓
Modules
↓
Sub Modules
↓
Screens
↓
Scenarios
↓
Functions
↓
object.

Eg: A company.

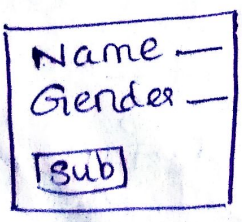
mod: To recruit emp & student candidate.

Submod:



All are submodules.

Screen:



→ creating a profile
↓
(Scenario)

object: Inside a screen all are objects

Function: It is performed by object.

High + Low level

will be handled by tech writers

↓

FRS

(Functional Req. Spec)

Develop/design | Tester

↓ Top down | ↑ bottom up approach

FRS:

→ It is prepared by Technical writer

→ It has detailed description about every individual objects and their functions

↔ It is mainly used by developers for developing coding.

Scenario:

→ A brief description about a event or an event what screen is to perform.

4. Coding: Developer will develop coding by using FRS doc.

5. Testing: Testers will test the application by using SRS and Use case.

6. Implementation support & maint:

Project will be implement in customer place. Support will be provided to end users & maintenance will be done by maint. Team.

TESTING BASICS & PRINCIPLES:

Testing:

→ It is an inspection done to identify the defects

→ It is done to ensure the quality of an application.

→ It is done to check whether all the req. are working properly.

Why Tester is necessary?

→ developer will develop the application in positive attitude whereas testers will test the app in -ve attitude for +ve result.

→ Tester will also validate the application in customer point of view.

Eg: Name : [ ]

condition: only caps Alpha.

Developer: +ve attitude

Input	Acceptance	Status
ABCD	✓	Pass

Tester: -ve attitude

I/P	Acceptance	Status
abcd	✓	Fail (defect)
1234	×	Pass
*//	×	Pass

Test case:

Step by step procedure for client is test case

1. Valid Test case (Pass)
2. Invalid Test case (Fail).

Define Error/defect/bug/failure/fault:

Error:
Human made mistakes
→ The logical mistake done by developer in coding.

Defect:
Due to the logical mistake there willbe some descripancy b/w expected & actual result

BUG:
The defect identified by testing team is accepted by development team.

FAILURE/FAULT:
A Fixed bug persist in customer place after shipping.

PRINCIPLES OF TESTING:

1. Testing shows presence of defects
2. Early Testing (X)
3. Exastive testing is impossible.

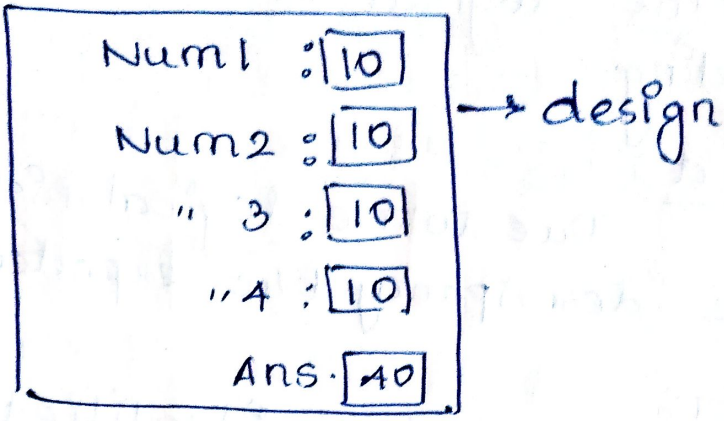
Exastive Testing → Validating a field with max no. of possible s/ps

4. Pesticide Paradox → developer is executing the test over and over gain, same test case will no longer find any new changes or defects.
5. defect clustering
6. Testing is context dependent.
7. Absence of error - fallacy
(X)8. Prioritize tests so that whenever you stop testing, you would have done best testing in time available.

View over application by Tester:

Eg:

Addition



Procedure:

→ Get a pop up msg when the box is empty
→ Add a button to cross check I/p.

Eligibility to be a Tester:

1. Understanding the requirements clearly.
2. Good communication skill
3. Good listening skill
4. Good Negotiation skill.
5. Good in basic domain knowledge.
6. Good Team player.
7. Easy adapt to New environment.
8. Preparation of documents for all formal and informal discussions & maintaining it properly for future support.

LEVELS OF TESTING:

1. Unit level of Testing.
2. Integration level of Testing.
   - 2.1 2types of Testing
     - 2.1.1 Structural Int. Testing
     - 2.1.2 component /module Testing

2.2 Approaches
- 2.2.1 Top down approach
- 2.2.2 Bottom up "
- 2.2.3 Sandwitch "
- 2.2.4 Bigbang "

3. S/m level of Testing.
4. Acceptance level of Testing.
(Its done in 2 Plcs)
1. Alpha Testing
2. Beta "

Unit Level of Testing:

→ It is the 1st level of testing performed by developers by using FRS doc.

→ Here the internal codings will be compiled with I/p's to validate its O/p.

Integration Level:

1. Structural Integration:

→ It is done by developers along with unit testing.

→ Here the communication interface b/w the codings will be validated. It is done through TOP-down approach with the help of stubs & drivers.

STUBS: Set of programs in the form of dummy database which occupies the incompleted portion of a build. It is also known as calling program.

Drivers: Set of programs which drives the datas from original module to the dummy module.

→ BUILD: Set of program in the form of exe. file which is ready for validation in developing team after unit testing is completed. It is also known as AUT (APP. Under Test).

2. Component / module Testing:

→ It is performed only after unit testing is completed. It is done by testers by using SRS & use case.

→ Here the conditions, functions & communication Interface b/w fields, screens and modules will be validated respectively. It is done through bottom-up approach.

Sandwitch approach:

→ combination of both top down & bottom up approach is sandwitch approach.

→ It is performed where the size of the project is very huge inorder to identify the missing links b/w modules.

(X) Bigbang approach:

combining all the components together to form a single application & check commu. Interface b/w entire modules without disturbing its functions.

S/M Level of Testing:

combining all the components together to form a single application & check whether the app. is working satisfied for the client req. or expectations.

Acceptance Level of Testing:

It is done to check whether the app is working properly acc. to

every individual users acceptance criteria.

Alpha testing: Validation of the app will be performed in developing environment with real time I/p's. (done by deve, Testers & team L)

Beta Testing: Validation of the app will be performed in client environment with real time I/p's. It is performed only by client.

TESTING PROCESS: known as It is also, VV process (verification & validation)

verification:
→ Process of cross-checking the contents, concepts, procedures, designs, doc etc,
→ It says are we going to do the product right?

validation:
→ Process of cross-checking conditions and functions of a product.
→ It says are we doing the right product.

These Process includes two types of testing.
1. Static testing
2. Dynamic testing.

STATIC: It comes under verification process.
Here the internal codings will not be executed but it will be cross checked for its structure, trees, branches, links, loops
It is done through inspections, audit, walk-throughs, reviews, meetings etc.

DYNAMIC:

It comes under validation process. Here the internal codings will be executed both internally & externally to validate its conditions & functions by using test case & test data.

Test case:

Step-by-step procedure followed to validate an application, conditions & fncs by using test data.

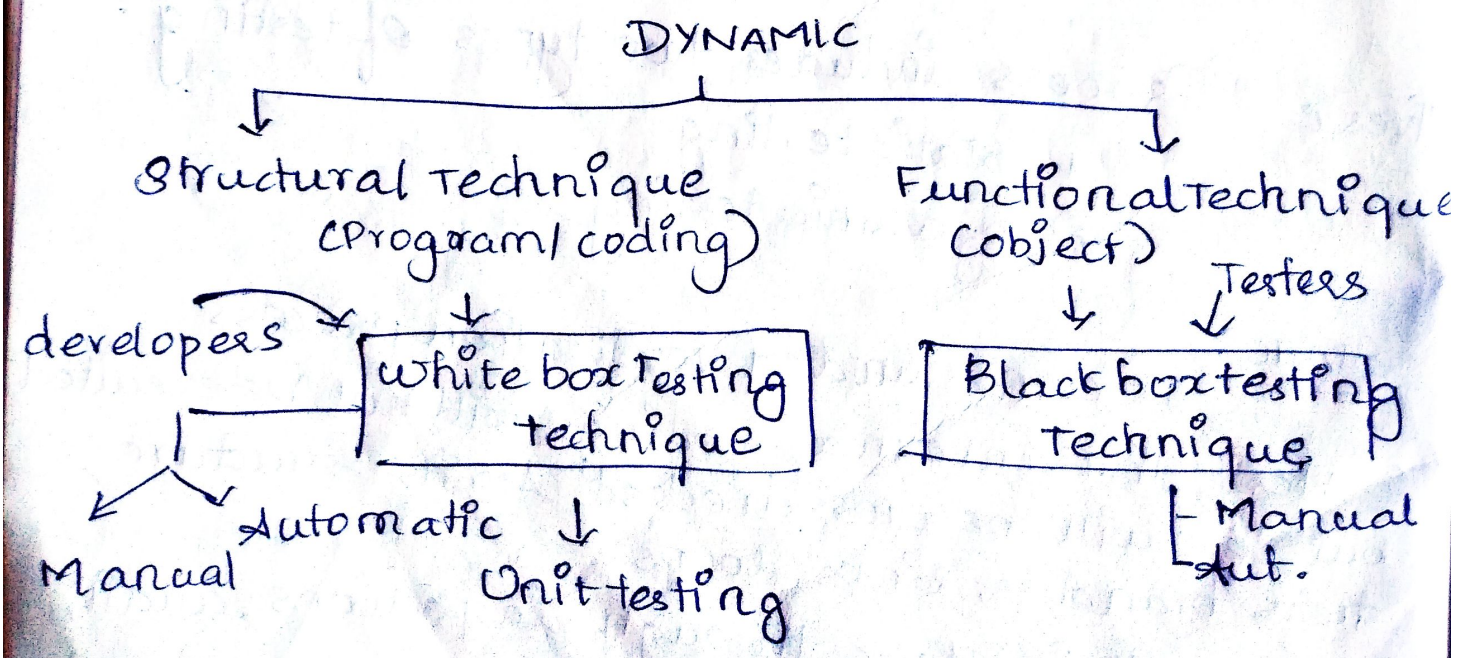
It has modules, sub-modules, scenarios, conditions, Test data, expected result, action, actual result, status, comments etc.,

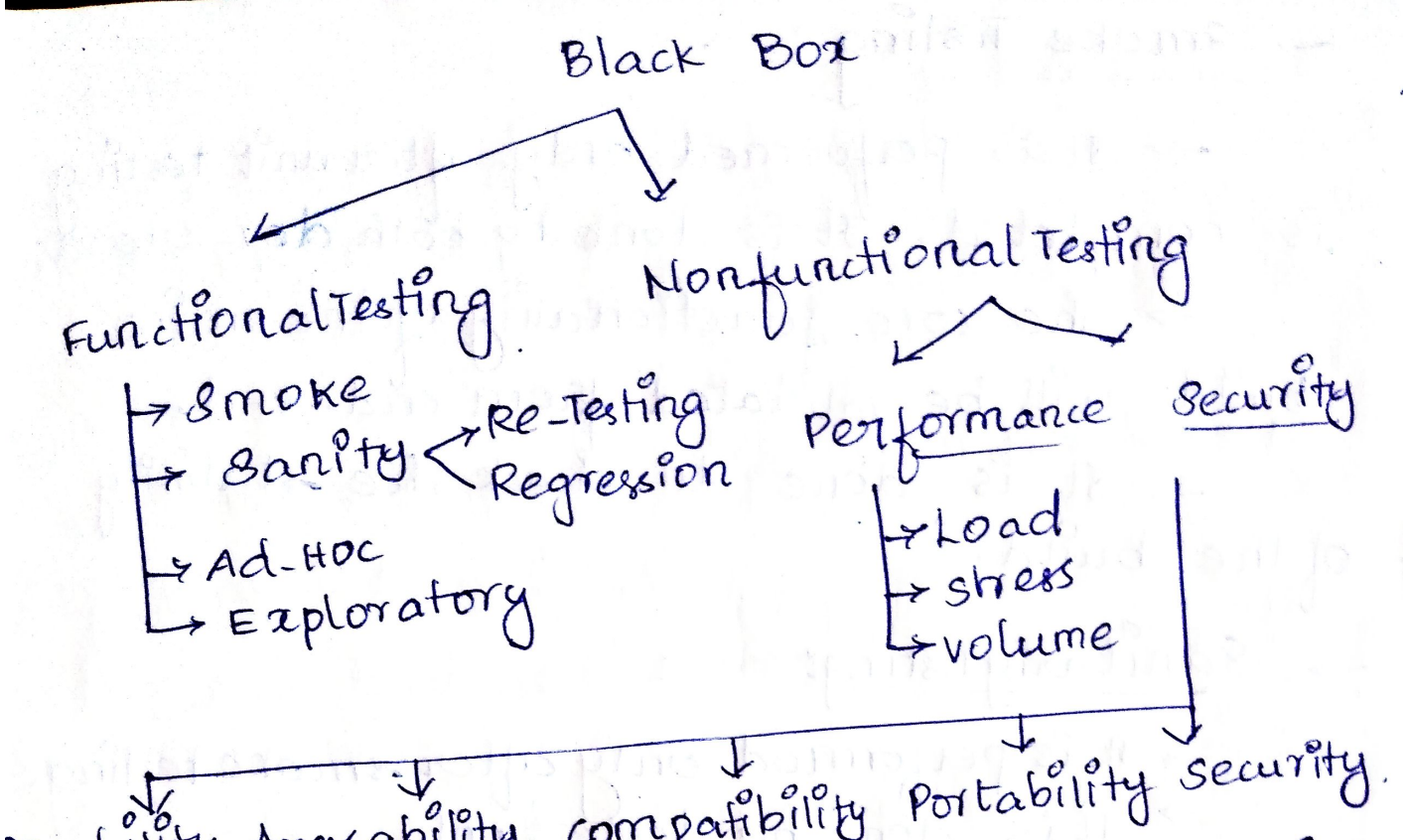
Test data:

The I/ps given in each field to validate its conditions.

Types: 1. valid
2. Invalid
3. Illegal
4. blank

TESTING TECHNIQUES:





WBT (White Box Testing)

1. It is done in internal perspective. Here internal codings will be compiled with I/Ps to validate its O/P by using test case and test data.
2. A Unit testing is Performed
3. It is done by developers
4. DOC - FRS
5. Top-down approach
6. Programming Knowledge is mandatory

BBT (Black Box Testing)

1. It is done in external perspective. Here the conditions and functions will be validated externally using TC & TD.
2. Fnc. & non-fnc testing is performed.
3. It is done by testers.
4. DOC - SRS & Usecase
5. Bottom-up approach
6. Prog. know. Is not necessary. But will be considered as added advantage

FUNCTIONAL TESTING:

It is done to check the functional behavior of the app by giving I/Ps

→ Smoke Testing:

→ It is performed only after unit testing is completed. It is done by both dev. & testers.

→ The core functionality of the entire build will be validated from end-end.

→ It is done to check the stability of the build.

→ Sanitary Testing:

→ It is performed only after smoke testing.

→ It is done by testers only.

→ Here the basic functionality of every individual module of the build will be validated indepth.

→ It is done to check the rationality of the build.

→ It also includes the new functionality and the functionality after the bulb is fixed.

Re-Test:

Executing the existing test case in the existing application after the bug is fixed.

Regression:

It is done to check whether the changes are working properly and also to check whether there is any impact in the existing functionalities due to the changes made in the application.

It is performed whenever there is a change in coding or client req.

Ad-HOC:

→ It is a non-linear informal testing performed randomly at the end of each phases / milestones

→ It is done to check whether the product is stable for further proceedings

→ Also known as final level testing of each phases.

Exploratory:

→ It is performed wherever there is less info/ no info about the project.

→ It is done by exploring the source file through installing it and understand the req. of the client.

→ Support also will be taken from experts to improve the quality of the informations to validate the application

Non-functional Testing:

1. Performance Testing: → Tools: Jmeter, loadRunner

→ It is done to check how fast the app is working under normal and abnormal loads

Load Testing:

It is performed to check how fast the app is working under anticipated load. (expected)

Stress Testing:

It is performed to check how fast app is working under unanticipated load. (unexpected)

Volume Testing: It is done to check how fast the app is working while increasing the volume of data in database.

Compatability:

It is done to check the compatibility of app in different computer platforms & browsers

Portability: (upgrade)

It is done to check whether the app is working properly when it is ported from one envir. to another environment.

Usability:

User friendly.

Accessability:

It is done to check whether the data's are shared only according to the priviledge provided to users.

Security:

It is done to check how well the app securing itself from unethical activities.

Penetration testing

It is done to check how well application protecting itself from penetration of unauthorized persons & hackers into the server.

Security concepts:

1. URL manipulation (Uniform Resource locator)
2. Secured socket layer
3. SQL injection
4. Cross site scripting
5. Bio-metric security concept. (virtual keyboard)
6. Brute force attack (Password lock)

7. Kryptographic security concept. (encrip/decrip)
8 Password extension
9. Cookies testing
   - bridge (b/w server & IP address (user))
   1. session cookies
   2 Persistent "

TESTING STRATEGIES:

1. GUI (Graphical and User Interface)
   - 1.1. OBJECT Testing
   - 1.2. UI (User Interface) Testing
2. FT (Functionality Testing) strategy
3. IT (Integration Testing) strategy.
4. SIT (System Int. Testing) strategy.

1. GUI - OBJECT: It is done by developer & testers. Here the graphical activities of the objects will be validated.

2. GUI - UI: It is done by testers. Here the conditions of every individual fields of a screen will be validated by giving I/Ps

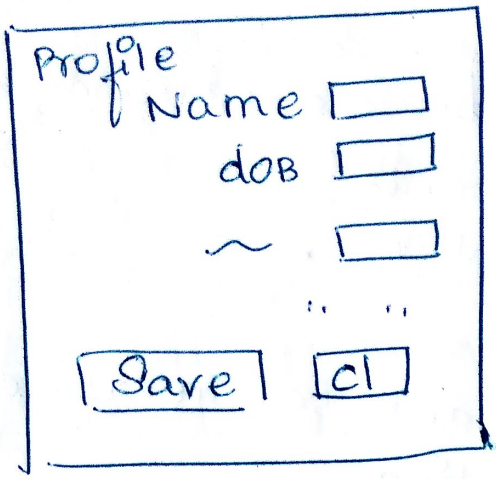
3. FT: It is done to check the functional behaviour of the screen and its scenario.

4. IT: It is done to check the communication interface b/w two diff screens/modules of an application. (viewing doc saved in account at report)

5. SIT: checking communication interface b/w two diff s/ms or applications.

Eg: mod1 mod2 mod3

ACCOUNT | PORTFOLIO | REPORT



1. OBJ: Eg: check object of tab, arrow

2. UI: Test data chect

3. FT: Eg: Save function OK

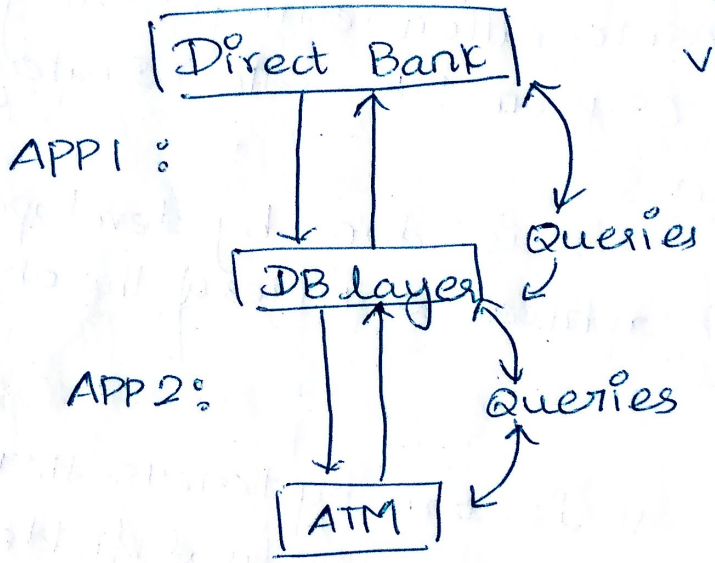
4. IT: two diff mod

Account ↔ Report

Save as (~.ext) | Search (~.ext)

view in report

5. SIT:



If ATM is not given then

DB → SQLTool.

↓

Basic queries

↓

data retrieve

(can view data).

1) SIT
2) FI confirm
3) duplication

TEST DESIGN DOCUMENTS: 5 doc

(Deliverables)

1. Test Plan
2. RTM (Req. Traceability Matrix)
3. Test case doc.

4. Bug report
5. Test summary report.

1. Test Plan:
It is prepared by Testing team lead. It contains entire plan for testing activity

2. RTM: It is prepared by testing team lead & updated by testers. It is mainly used to trace whether all the req are covered.
It is done through by cross checking the every requ. & scenarios contains minimum single test case to validate.
It is used by testers to map req. & scenarios with test cases & test cases with bug & its status
It is also used to manage test case coverage, schedule, Impact analysis and chect whether all the requirements are met
It is used in 2 ways
1. ~~Forward~~ ~~action~~ direction
2. Backward ~~action~~ "

forward: It is followed by testers. Here the req ID will be mapped with scenario ID & testcase ID which represents bug ID & its status.

backward: It is followed by TL & client. Here the status of bug & bug ID will be mapped with test case ID & scenario ID which with respect to req. ID.

3. Test case doc: It is prepared & updated by testers.

4. Bug Report:

It is prepared & updated by testers
It contains bug ID, test case ID, buildversion, bug description, priority, Similarity Status, type of bug, release date, comments etc., / tester name

(situation) Priority: How fast bug has to be fixed. It is based on the urgency of bug fixing & need of that bug to get fixed. It is validated in customer point of view

It is categorized as high medium low

(impact) Severity

How badly bug affects the app. It is based on the impact of the bug in the functions of the app & percentage of data loss. It is validated in app point of view

It is categorized as high, med, low.

Test summary Report:

It is prepared by testing team lead. It contains qualitative & Quantitative measurements for activities performed in testing.

Test design techniques / validation techniques / Test case tech / testing phase techniques:

1. BVA (Boundary Value Analysis)

It is used to validate the left & right boundaries or integers of the limit.

It is used to analyse the boundaries of the values.

2. ECP : (Equivalence class Partition).

It is used to validate the lower & upper classes of the limits as well as its middle value.

3. Cause and Effect Graphic:

By giving I/ps, O/ps will be validated.

4. Error Guessing:

It can be performed only through experience.

Alpha:

Name [ ] 3 to 80

$a \Rightarrow 0.5$

$n \to 3 \quad n_1 \to 80$

Form: $(n-a, n, n+a ; n_1-a, n_1, n_1+a)$

$(3-0.5, 3, 3+0.5 ; 80-0.5, 80, 80+0.5)$

$(2.5, 3, 3.5 ; 79.5, 80, 80.5)$

× ✓ ✓ ✓ ✓ ×

Numeric:

$n \quad n_1$

(Age → 1 to 100)

$a = 1$

BVA:

$(0, 1, 2 ; 99, 100, 101)$

× ✓ ✓ ✓ ✓ ×

ECP: middle also

$(LC-a ; MC ; UC+a)$

$1 \quad 50 \quad 100$

$(1-1 ; 50 ; 100+1)$

$(0, 50, 101)$

× ✓ ×

→ one only valid.

Name:

[ ] 3 to 80

$a = 1$

$(3-1, 3, 3+1 ; 80-1, 80, 80+1)$

$(2, 3, 4 ; 79, 80, 81)$

× ✓ ✓ ✓ ✓ ×

Cause & effect:

If this got success then only BVA & ECP is validated.

Age: 1 to 100

V → 0, 100, 2, 101

Inv → 0.5, 0.1

Illegal → ABCD#*

Blank →

STLC:

Phases - 7

1. Req. Analysis:

→ Req will be analysed from SRS & usecase.

2. Test Plan:

Test Plan doc & RTM will be created.

3. Test Analysis:

→ KT (Knowl. Transfer) will be provided

→ cost will be allocated

→ RTM, SRS & usecase will be circulated

4. Test environment:

→ Envir. for testing activity will be created.

5. Test design:

Req. will be understood. Scenarios will be identified. Test case & test data will be prepared and forwarded for reviewprocess

6. Test execution:

→ Reviewed testcases will be executed.

→ defects will be identified

→ defect tracking process will beperformed.

7. Test closure:

Test summary report will be

prepared & proj will be ready to release.

DESCRIPTION

→ After receiving SRS & usecase, testing TL will prepare test plan doc. & RTM. With help of test plan, tasks will be allocated RTM, SRS & usecase will be circulated after KT is provided.

→ Tester will understand the req. from SRS & identify scenarios from usecase & forward scenario for review process

→ Once review is completed, tester will prepare test case and test data for identified scenarios and forward test cases for review process

→ Once test cases are review, it will be forwarded back to testers, simultaneously testing team will receive the build from developing team through VPN (Vir Private area Network) after unit testing is completed.

→ Testers will access the application through allocated URL & execute the review test cases in order to identify the defects

→ once a defect is identified tester will perform defect tracking process & the status will be updated in RTM & bug report

→ once all the functional, non-fnc req activities are completed testing TL will prepare Test summary report & project will be ready to release.

Test case Reviews:

3 Types of Review.

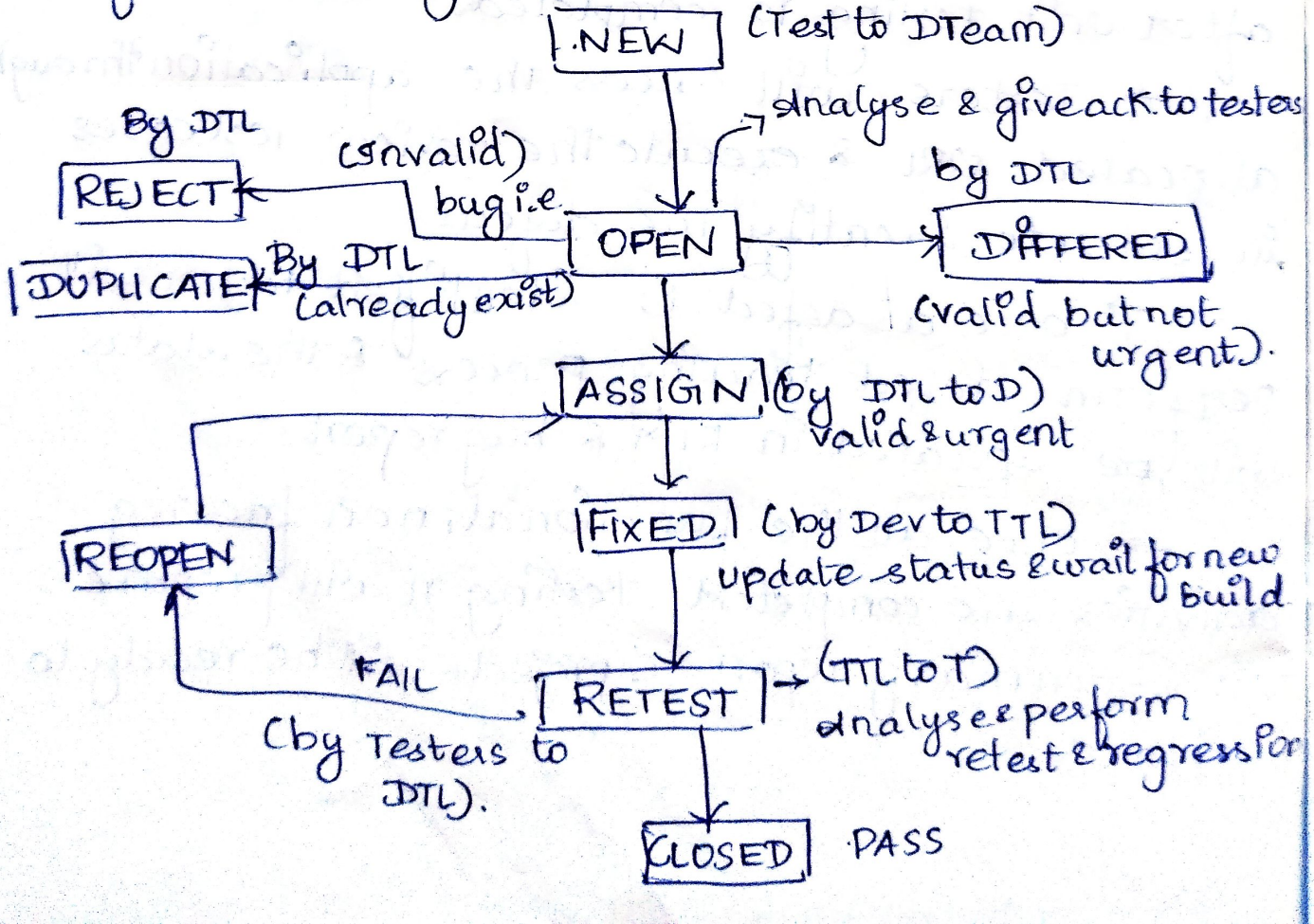
1. Personal review
2. Peer review
3. Superior review.

BUG LIFE CYCLE / DEFECT TRACKING PROCESS:

Once the defect is identified, tester will prepare defect ~~tracking~~ / bug description doc. & forward the doc to test TL to check the genunity of the bug.

Once the bug is identified as genuine the doc will forwareded back to testers with exact priority & siverity.

With help of this doc, tester will perform defect tracking process in Qc (Quality center)



SDLC methodologies:

Types:
1) Waterfall
2) Spiral
3) Agile & scrump
4) V-model

Waterfall: It is a traditional methodology which is used where the req. is constant without any changes

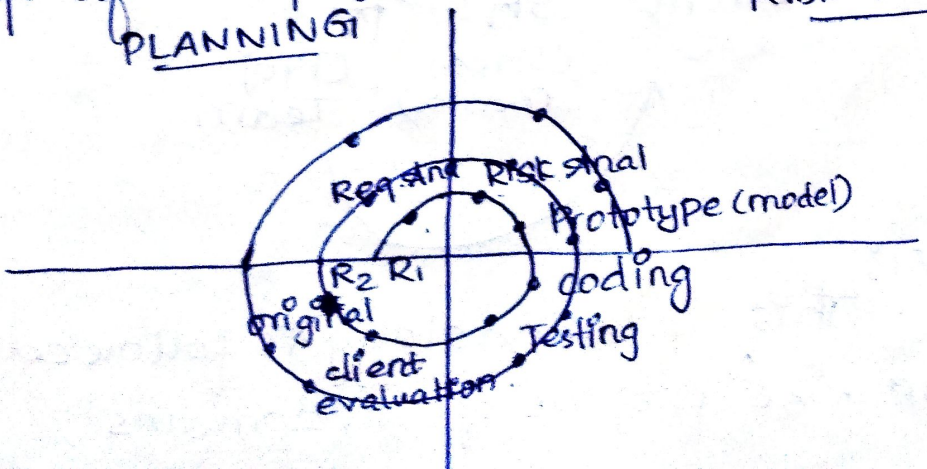
Here each and every phases willbe started only after 100% completion of previous phase.

Req. gath → constant (100%).
↓
P&A → if plan is 100%)
↓
Design (100%).
↓
coding
↓
Testing
↓
Tsl

Note: if any changes then entire process willbe affected & need to start from 1st

→ Rarely used.

SPIRAL: This methodology is followed where the req. of the project is very huges where risk is high.

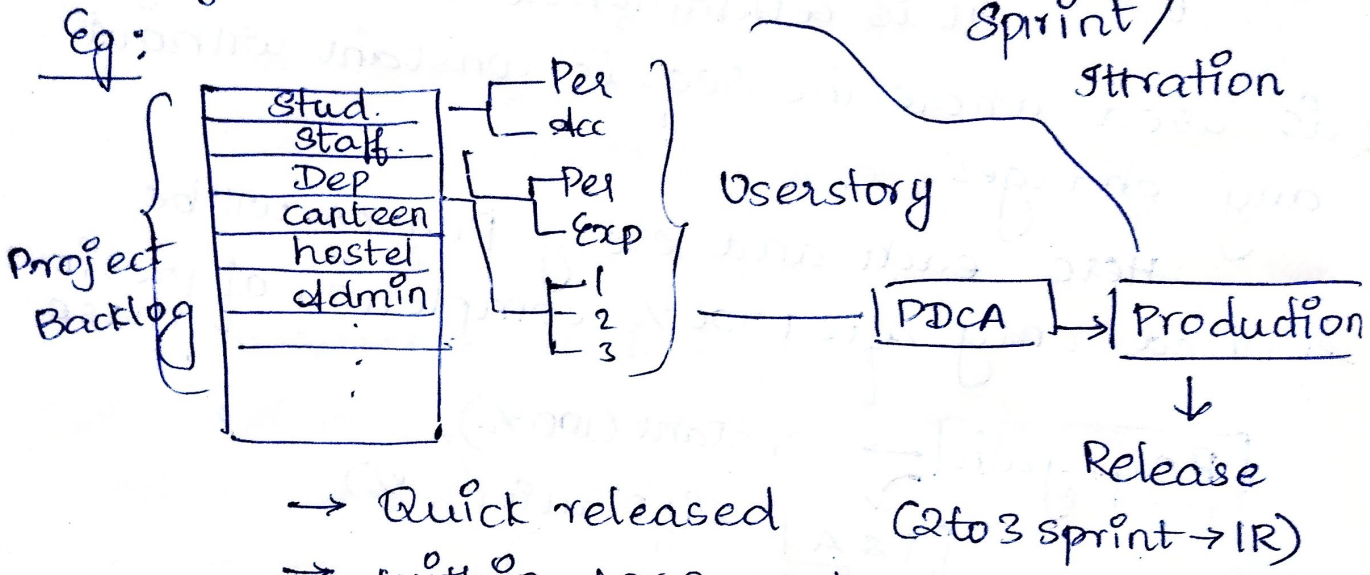


AGILE & SCRUM: (Best)
(quick)

This methodology is followed where client doesnot have clear idea about his req, where the req has freq. changes & where uncertainity is more in req.

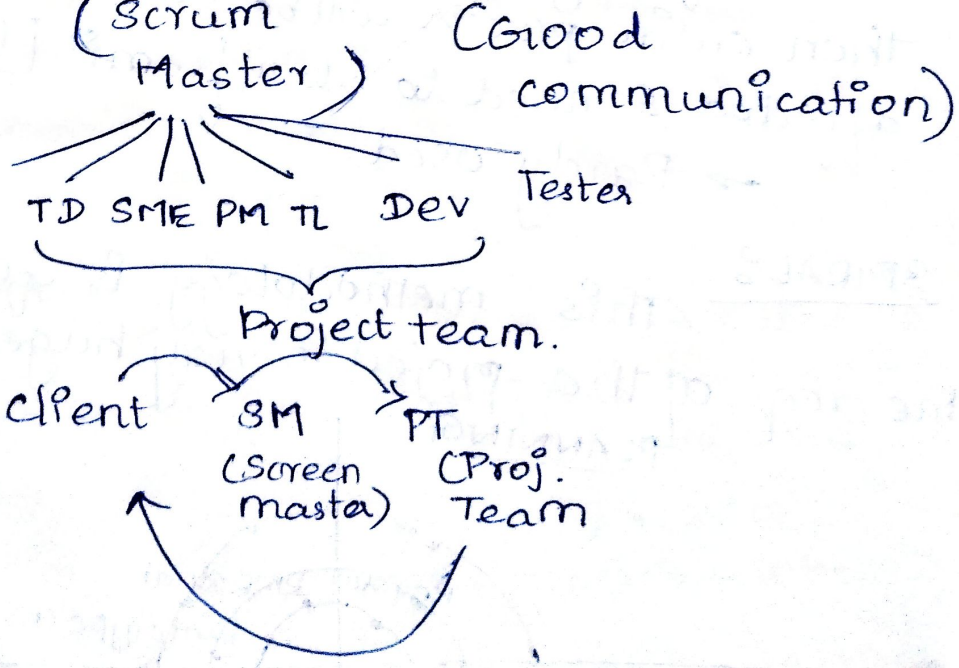
Agile:

Eg:



→ Quick released
→ within 1 or 2 weak
→ Bit by bit.

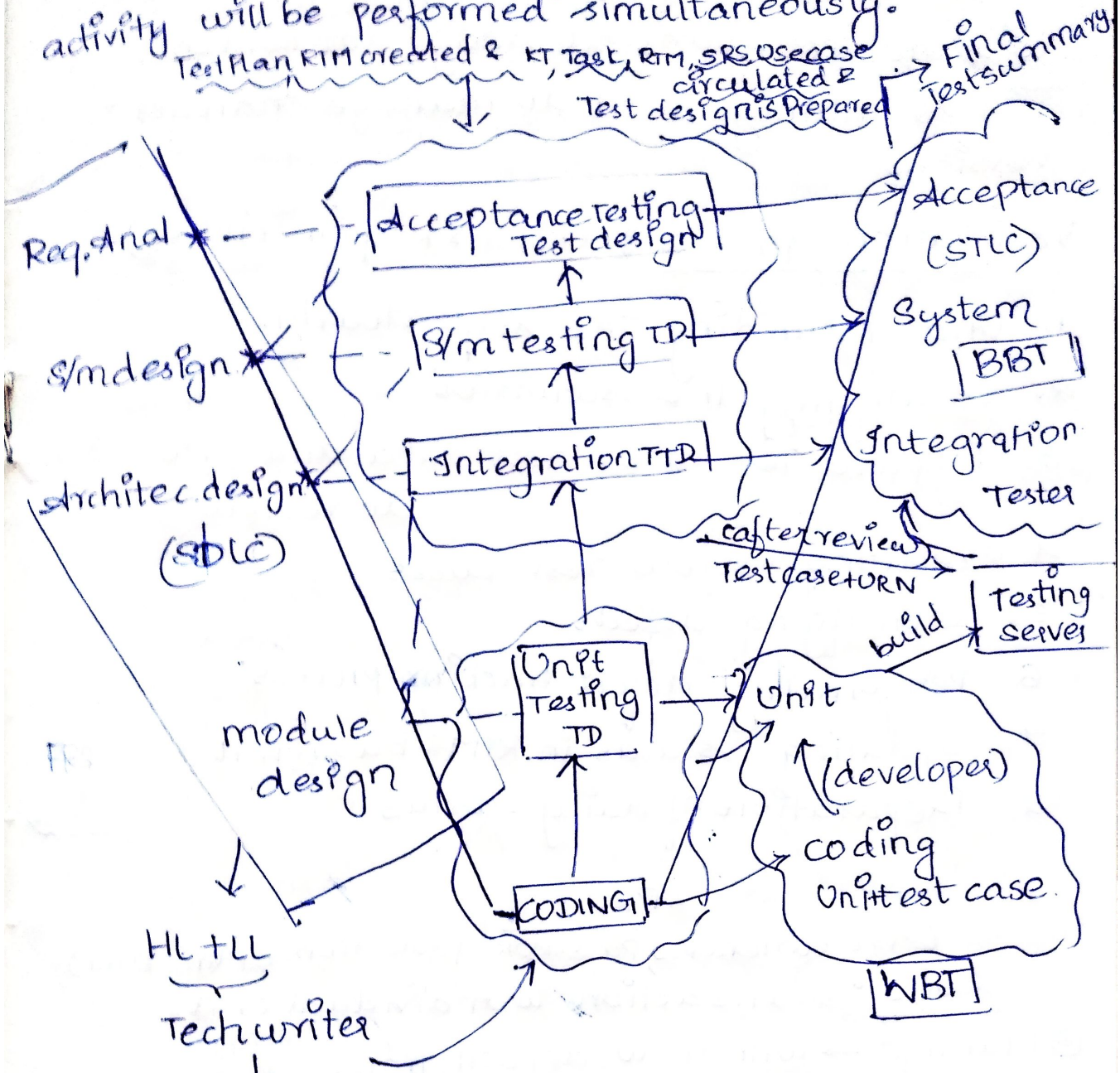
Scrum:



V-model:

This methodology is followed in medium size projects of all domains.

Here project design activity & test design activity will be performed simultaneously.



Verification documents:

1. BRS
2. Usecase
3. SRS
4. FRS
5. SDD
6. Test specification
7. Test Plan
8. RTM
9. Testcase doc
10. Bug Report
11. Test summary report

12. CMT: (config. management TOOL)

CMT:

It is a safe depository which is used to store entire project design, doc content. It is mainly used to manage changes & versions.

Validation process: (Roles & Responsibilities)

1. Understanding the req. clearly.
2. Identifying the scenarios
3. Prepare Test case & test data and forward to review
4. Executing review test cases.
5. Identifying defects
6. Performing defect tracking process.
7. Updating status in RTM & bugreport.
8. Preparation of daily status.

PROJECT

(ISO currencies, Password protection, online bank)

Scope: ① transaction to individual or a cmpy

② Manage by → withdraw, deposit, transfer

③ Payee info

④ holder → payee or viceversa

∓ mod / New account, paybills ∈ add edit skip,