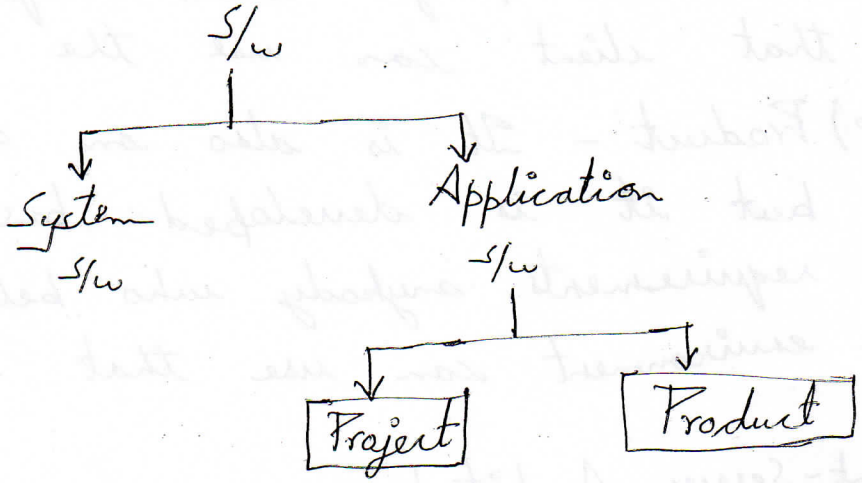
5/10/2018

Software

→ A software is a program (or) combination of programs used to perform some operation in the computer.

→ In general in IT industry we are having two types of software. They are

* System software
* Application software



System S/w

→ It is used to interact with the hardware components of a computer such as display cards, sound cards etc.

→ They will be developed by using languages like C, C++, Unix etc.

→ In order to use these softwares we must have technical skills.

Ex: * OS
* Display drivers
* Sound drivers

Application software

→ These are the GUI (Graphical user interface) tools developed for non-technical people called end-users / clients.

→ These softwares will be developed by using technologies like Dot net, Java etc.

→ These are further divided into two ways

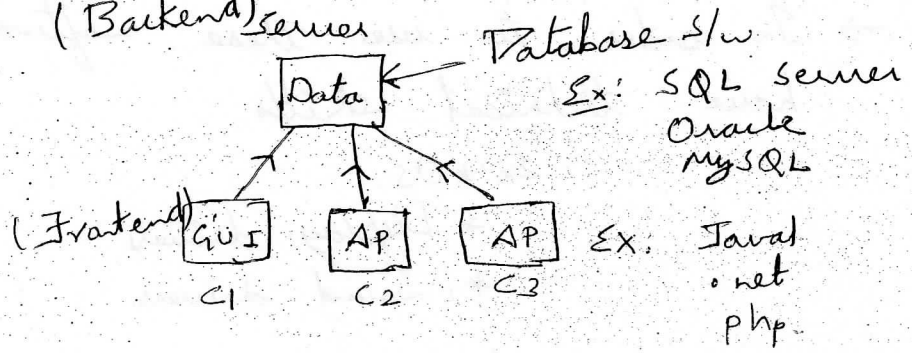
a.) Project - It is a application s/w developed based on specific client requirement, only that client can use the s/w.

b.) Product - It is also an application s/w but it is developed based on market requirements, anybody who belongs to the environment can use that s/w.

Client-Server Architecture

→ This is the widely used architecture model to develop a project (or) product

→ In this model there will be a main computer called server which is connected with multiple other computers called clients



Project/Product = 1FE + 1BE

→ The server will be used to store the data in an organised manner with a help of a database s/w
→ Inorder to access and modify the data each client system contains a GUI tool called application program.
→ A project is a combination of one frontend application & one backend database.
→ In a project development the backend developer (or) database developer will the develope the database and then the frontend developer will design the user-interface & and also implements communication b/w Frontend application & backend database
→ After developing the project it is the responsibility of a test engineer to test the s/w before delivering to the client.

Software testing

→ It is the process of verifying & validating whether the developed software product is satisfying all the client requirements or not.

22/10/2018

Software Environment

→ In general what ever software we are developing will be based on a client-server architecture model. In this model there will be three major components They are

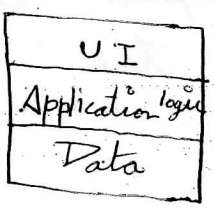
* 1-Tier
* 2-Tier — Presentation layer – User Interface
* 3-Tier — Business logic layer – Application logic
* n-Tier — Data layer – Database

→ The client-server model is divided into the following categories. They are

1-Tier
2-Tier
3-Tier &
n-Tier

1-Tier

→ In this model all the three layers will exist in a single computer



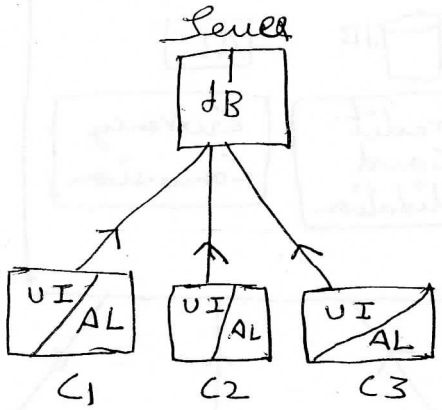
Ex: Excel applications

2-Tier model

→ This is the frequently used model, in this model there will be two types of computers. They are

* Server & Client

→ Server contains database whereas client contains user interface & application logic.

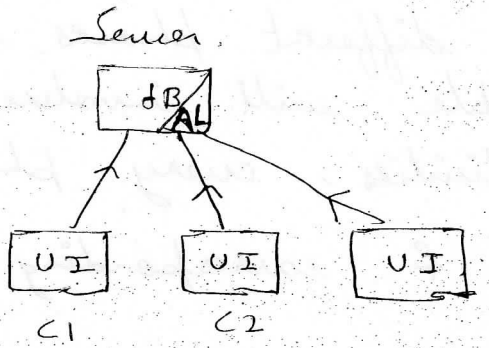


Note:→ The drawback of this model is since user interface & application logic are integrated, this application logic is not reusable

3-Tier model

→ This model also contains two computers They are Server & client

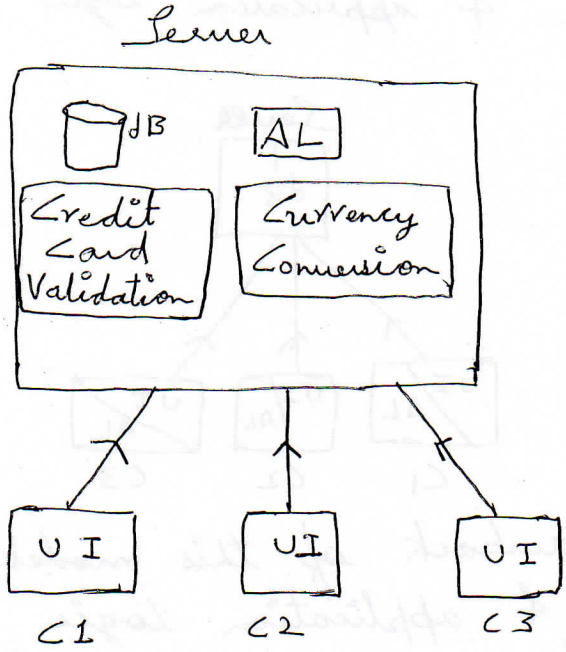
→ Server contains database as well as application logic and client contains only user interface



→ Since the application logic is seperated & kept on the server it can be reusable for different kinds of clients.

n-tier model

→ It is similar to the 3-tier model, the difference is in this case the server contains different types of Application logic. These logic can be used by different clients.



23/10/18

Software Development Life cycle (SDLC)

→ Every development company will follow some standard procedure to implement a software. that process is called SDLC.

→ It contains different phases, in each phase different people will involve & perform different activities; every phase will have some input & corresponding output.

→ The following are the different kind of people involves in the SDLC process. They are

* Clients,
* Business Analyst
* System Analyst
* Technical Architect
* Developers
* Testers &
* CCB (Change control board) / Production support team

→ The following are the different phases in SDLC

*. Requirements Gathering
*. System Analysis
*. System Design
*. Coding / Implementation.
*. Testing
*. Release / Deployment &
*. Maintenance

Requirements Gathering

→ In this phase the business analyst will go to the clients place, studies the current system the client is following, gathers all the requirements of the client & also his expectations.

→ Based on all these information the BA will prepare a document called as BRS (Business Requirement specification) (or) FRS (Functional requirement specification) or CRS (Customer Requirement specification) documents.

System analysis

→ In this phase the system analyst will analyse the BRS document & conducts feasibility study to identify the s/w & H/w needed and then prepares all the requirement in a technical format with the help of SRS (Software requirement specification) document.

System design

→ This is the starting stage of actual project implementation.

→ In this phase the technical architect will study the SRS document & starts designing the s/w.

→ He prepares two documents. They are
  * HLD (High level design) document
  * LLD (Low level design) document

→ The HLD contains information about main modules for the software

→ The LLD contains info internal details of each module

→ These documents will be prepared in the form Dataflow diagrams, flowcharts, Use-case diagrams & ER diagrams (Entity-Relationship diagrams).

Coding

→ In this phase the developers will start developing the s/w with the help of HLD & LLD document

→ In this phase the database developers will develop the database, the frontend developers will design the Users Interface (UI) for the screens/phases & also implements the code that has to be executed in the background.

→ The combination of entire process will result software product.

Testing

→ In this phase a test engineer will execute the software & verifies each functionality to ensure that whether all those functionalities are satisfying client requirement or not, it means the tester will test the quality of the software by performing testing at different levels using different techniques called testing methodologies.

→ The outcome of this phase is Quality software

Release/Deploy

→ After completing the testing phase the company will form a team consists of some developers & Testers who are responsible to deploy the project on to the clients place

Maintenance

→ While client is using the s/w the CCB team will maintain the s/w by rectifying the problems of the client as well as performing the changes requested by the client.

24/10/18

SDLC models

→ Based on the SDLC process the development companies will develop the softwares by using different SDLC models. They are

* Waterfall model
* Increment "
* Prototype "
* Spiral "
* V- model
* Agile model

Waterfall model

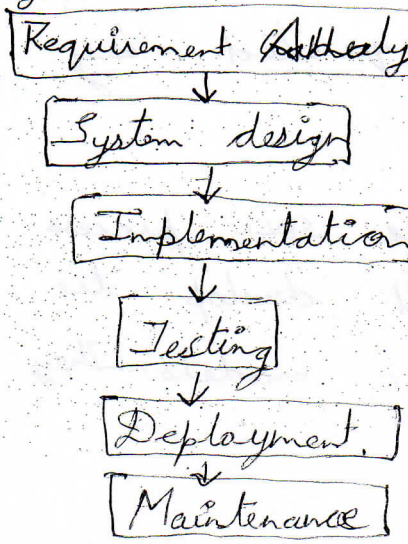
→ This is the very old model & first ever model used by the companies to develop the s/w.

→ When the s/w is small & all the requirements are available then we use this model.

→ In this model after completing of one phase then only the next phase will be implemented.

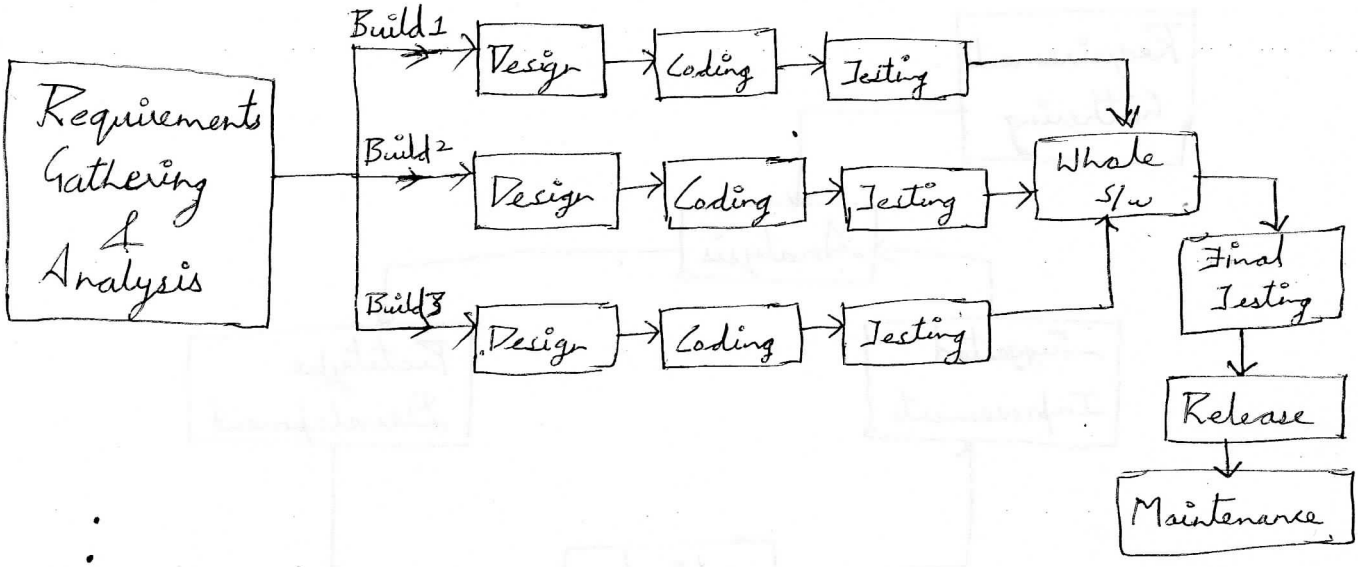
→ The advantages of this model are
* It is easy to implement.
* 100% clear requirements are available.

→ The drawbacks of this model are
* Resource wastage will happen.
* Testing is happening in the later stages, so if any defect is identified cost of defect fixing will be increased.
* Changes to the existing system is difficult because every stage has to modified.



Incremental model

→ When the requirements are clear but they are huge in size then we use this incremental model.



→ In this model all the requirements are divided into different builds & those builds will be implemented parallely. in each build implementation we will follow design, coding & testing phases, after all builds are implemented they will be combined as a whole S/w which can be finally tested & released to the client.

*Advantages

→ Since the builds implementation is happening ||ely the resources can be engaged.

Drawbacks

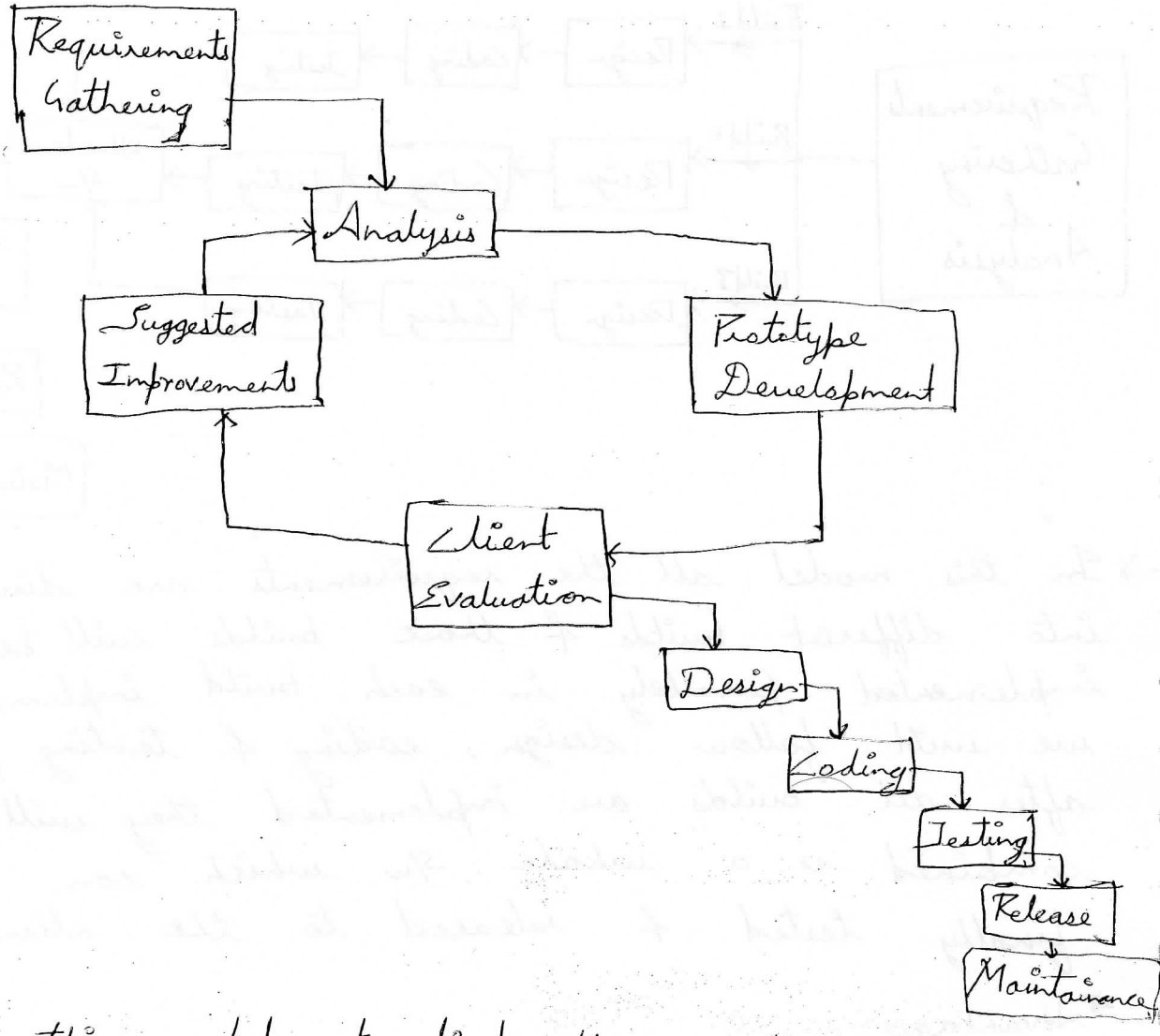
→ Until all the builds are implemented client cannot use the S/w

→ Changes to the existing S/w is difficult.

→ Testing is happening in the later stages which makes defect fixing as costly

Prototype model

→ When the requirements are not clear then we use this model



→ In this model at first the company will develop a prototype model for the client, the client will evaluate the model & suggest the improvements. these improvement will be applied on the prototype. This process will be continued until the client satisfaction.

→ After client approval the actual design, coding testing etc. phases will be implemented.

Advantages

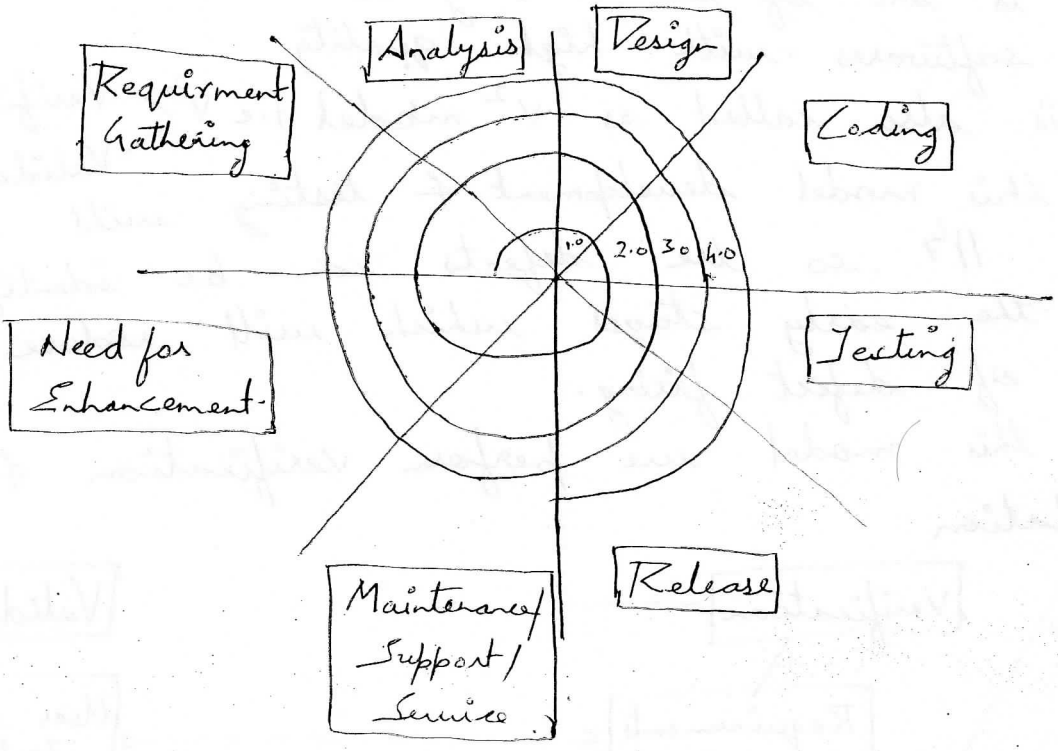
→ Since the client is interacting in the initial stages itself there might be no changes in the later stages.

Loopholes

→ Testing is happening in the later stages that causes increase in the cost of defect fixing.

Spiral model

→ This model is mainly used for product development



→ In this model the s/w implementation will be done version by version, in every version entire SDLC process will be followed

→ At first some features will be selected they will be implemented as one version & then some new features will be implemented & added to the existing version make it as a new version and so on.

Advantages

→ The s/w will be released fastly to the customer.
→ No pressure from the client.

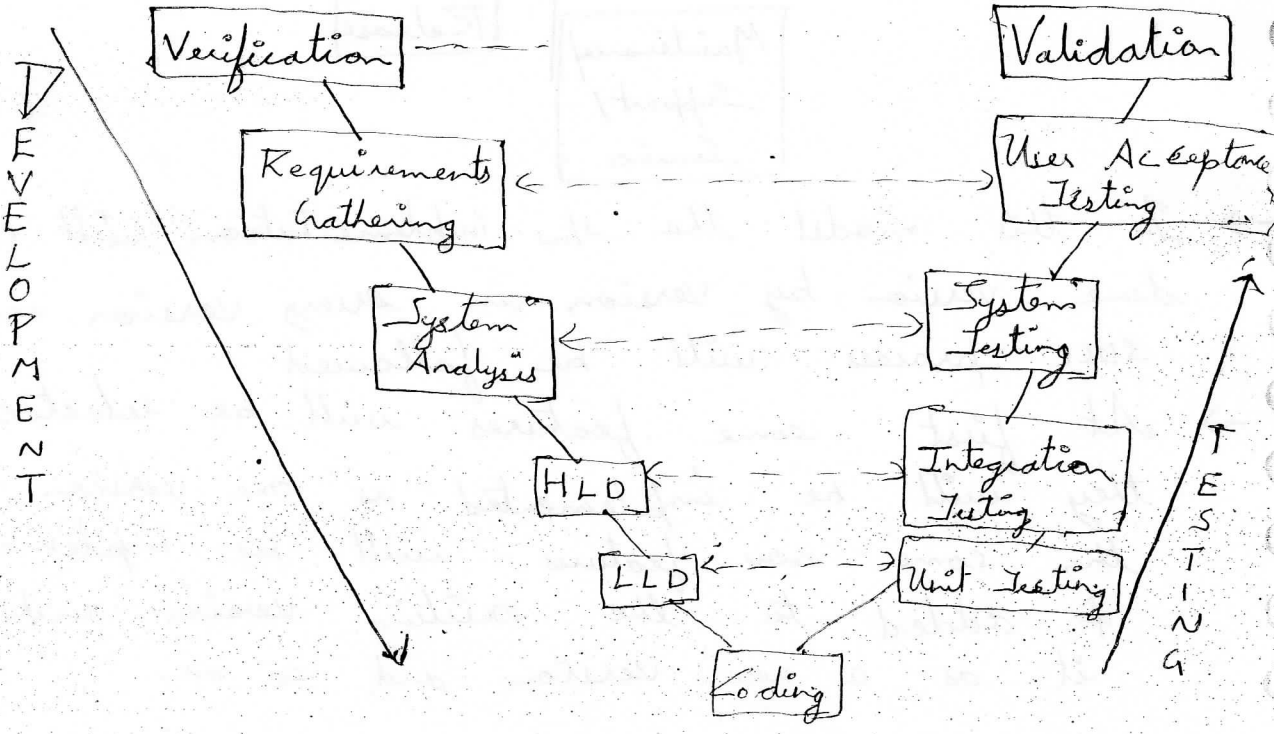
Loophole

→ There is no certain ending to the product.
→ Everytime giving training to the new employees is a difficult task

25/10/18

V-model

→ This is one of the widely used model to implement the softwares with high quality.
→ It is also called as V^2 model i.e V^2 < Verification / Validation
→ In this model development & testing will be done 11 by 7 so the defects can be identified in the early stages which will reduce the cost of defect fixing.
→ In this model we perform verification & Validation.



→ Verification is the process of evaluating whether the development process is happening properly or not, it will be done with the help of reviews.

→ Validation is the process of testing the developed product about whether it is satisfying the client requirements or not.

→ The disadvantage of this model is the difficulty to change the existing system for new enhancements

Agile model

→ Agile stands for fastness, this is the most frequently used software development model by the companies.

→ The main advantage of this model is client collaboration will be done in each & every stage so that the client will understand the process of development and provide a feedback so that any changes to be done can be performed in the early stages itself.

→ In agile model customer requirements are called "customer(user) stories"

→ A large user story is called as "epic".

→ In this model the s/w implementation will be done as "sprint by sprint".

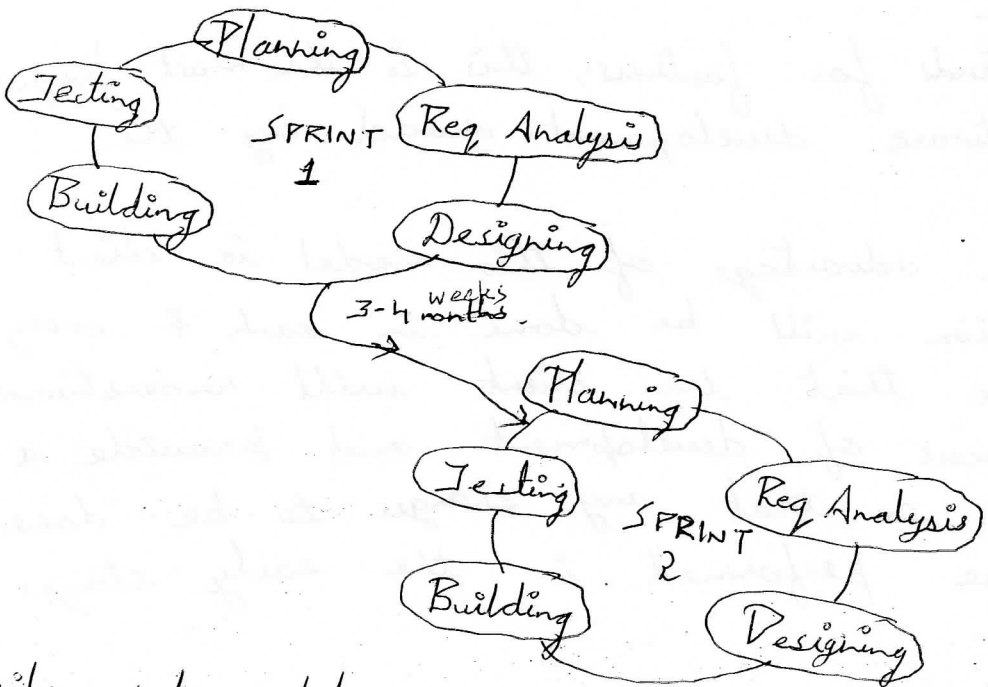
→ A sprint is a small part of the s/w.

→ At first entire s/w is divided into equivalent sprints, in general a sprint implementation time is 3-4 weeks.

→ Based on the priority some user stories will be selected as a sprint & start developing that sprint.

→ Once it is completed it can be released to the client so that the client can use that sprint.

→ After completion of one sprint we can start implementing next sprint and so on. In every sprint implementation we will follow SDLC phases.



Agile sub-models

→ The following are the different kinds of internal models in agile

* Scrum
* Extreme programming
* Adaptive s/w development
* Dynamic system development model.

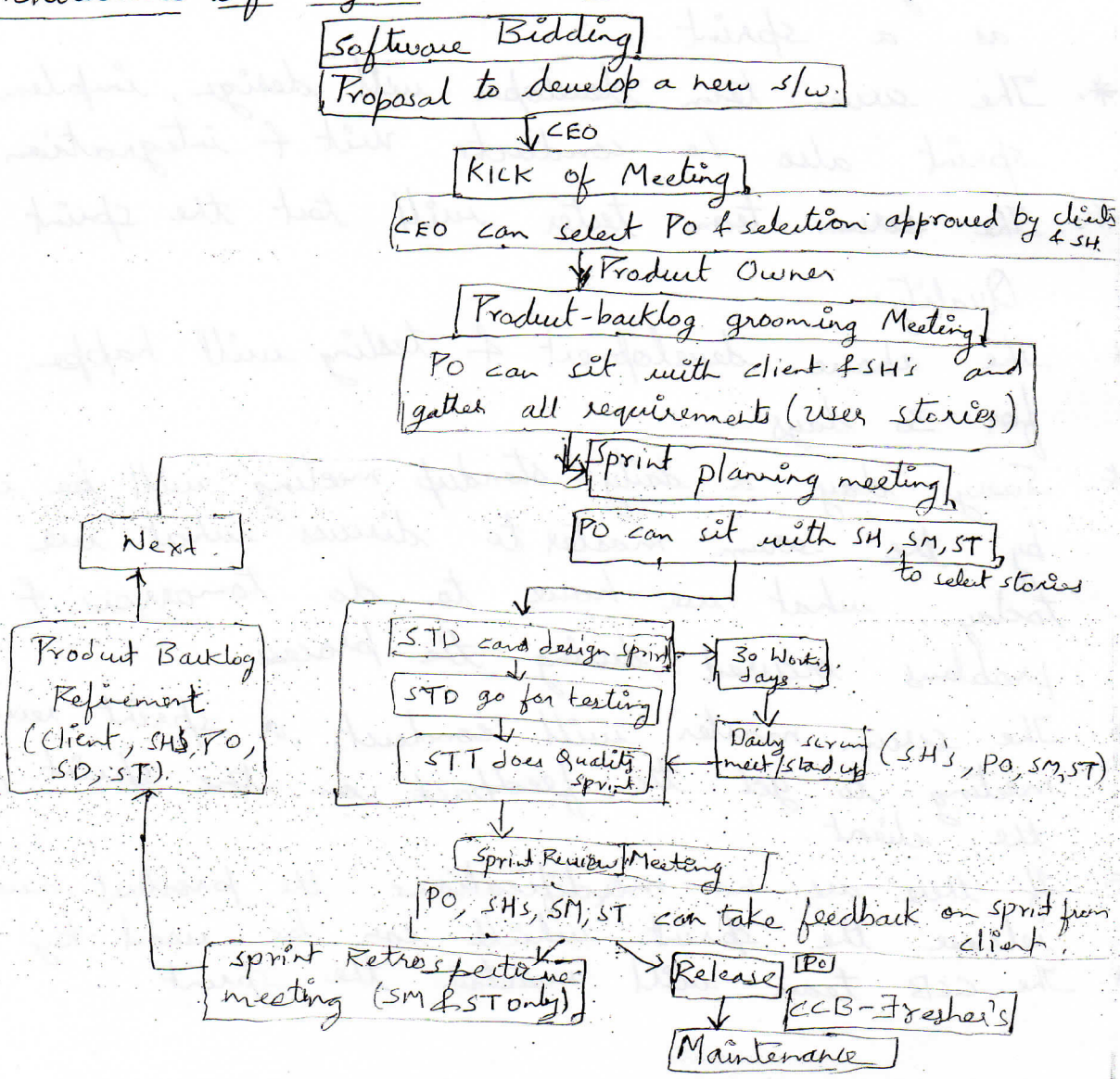
Agile scrum model

→ It is the most widely used agile model in the companies. The following are the different kinds of people who involves in the agile scrum development process.

*. Client
*. Stake holder
* CEO
* Product owner
* Scrum master
* Scrum Teams.
  - Scrum Team Developers
  - Scrum Team Testers
*. CCB Team

26/10/18

Architecture of Agile scrum development process



→ The following are the different activities that can be developed in the agile scrum development process.

Steps

* The client will provide a proposal to the development company about a s/w to be developed.
* The CEO of the organization will select a person as product owner.
* This selection should be approved by the client & stake holder.
* The product owner will gather all the client requirements (user stories) by conducting product Backlog grooming meeting.
* The scrum master will conduct sprint planning meeting where some user stories will be selected as a sprint.
* The scrum team developer will design, implement the sprint also he conducts unit & integration testing.
* The scrum team tester will test the sprint for Quality.
* The above development & testing will happen around for 30 days
* Every day a daily standup meeting will be conducted by the scrum master to discuss what we did today, what we have to do tomorrow & the problems occured during the process.
* The scrum master will conduct a sprint review meeting to get the feedback on the sprint from the client.
* If there are no modifications the product owner will release the sprint which can be used by the client
* The CI/B team will maintain the sprint.

* If client feedback is negative SM will conduct sprint retrospective meeting to resolve the problem.
* The product owner will conduct product backlog refinement meeting to identify the enhancement/changes to be performed for the sprint.

Agile scrum components

→ The agile scrum model mainly consists of three components. They are

* Jobs (Roles).
* Ceremonies (Meetings).
* Artifacts (Documents).

29/10/18

Software testing

→ It is the process of verifying & validating whether the developed s/w is satisfying specific client requirements or not.

(or)

→ Software testing is a process used to help to identify correctness, completeness & quality of the developed s/w.

(or)

→ IEEE defination is "Testing is the process of executing (or) evaluating a system or system component by manual or automated means to verify that it satisfies specific requirements.

Importance of s/w Testing

→ The following are the different reasons why we have to perform s/w Testing.

→ To deliver a bug free s/w to the client.
→ To deliver more reliable s/w to client.
→ To deliver a s/w which should be easy to maintain.
→ Early finding bugs will reduce the cost of fixing the bug.

Quality

→ The main purpose of s/w testing is to provide quality s/w to the client.
→ Quality will be determined in different perspectives in different ways.
→ In a development company perspective quality means the software should satisfy all the client requirements.
⇒ whereas in a customer point of view quality means the developed s/w should be fit for use.
→ The following are the different factors that will ensure the quality of the s/w.

- Bugs
- Delivery time.
- Budget.
- Reliability
- Maintainability

→ When the developed s/w is bug free and delivered within the client specified time & Budget and also easy to maintain and works well in all condition then we call it as a "Quality s/w".

Quality Management

→ It is the process of preventing defects in the s/w development process to ensure that there are no more defects in the developed s/w, it will be done in two ways.

- Quality Assurance (QA)
- Quality Control (QC)

Quality Assurance

→ It is an activity that is based on process, here we will verify each and every development process to find any weakness so that it can rectified which internally prevents defect to occur.

Ex: Reviews & Auditing.

Quality Control

→ It is an activity that is based on product, here we execute the s/w product & ensure that all the functionalities are working properly & it is satisfying all the client requirements.

Ex: Software Testing is a QC activity.

*** Differences b/w QA & QC

QA	QC
* Process Oriented (producing the product)	* Product Oriented.
* Preventive approach	* Detective approach.
* Usually done throughout the lifecycle.	* Usually done after the product built.
* Audit & Review are the examples.	* Testing is part of Quality Control.

Error

→ It is a mistake which will occur at development time, in general developers will find these mistakes.

Defect

→ While testing the s/w if the tester identifies any mistake in the s/w functionality then it is called as a "Defect"; it is a deviation b/w expected & actual results.

Bug

→ If the tester identified defect is accepted by the developer then that defect is called as a "Bug".

Failure

→ While using the s/w if the client identifies any mistake in the s/w functionality then it is called as a "Failure".

Test Engineer Roles & Responsibilities

→ The following are the different activities that are to be performed by the test engineer.

- Understanding functionality of application with the help of BRS (or) FRS.
- Identify test scenarios for allocated module.
- Prepare test cases.
- Prepare automation test script.
- Executing test cases & automation test script.
- Defect reporting and prioritize what you report.
- Participating in re-testing & Regression testing

Note: Test-scenario is an activity to be tested.

Ex: Login process verification.

→ Test-case is a condition used to test that activity

Ex: Testing the login process with valid login details & with invalid login details.

31/10/18

Types of Software testing

→ Software testing is broadly divided into two ways. They are.

* Static testing
* Dynamic testing

Static testing

→ It is the process of examining or evaluating the process of software development.

→ It is mainly used to prepare an environment to develop a quality software.

→ It will be conducted with the help of Reviews & Audits.

→ In this method we are going to test the documents at different levels; it is also called as "documents testing".

Reviews

→ A review is the process applying on the document to check its correctness & completeness.

→ The main purpose of reviews are

* To identify the defects in the early stages
* Because of early finding defects the cost of defect fixing will be reduce.
* Evaluation as a team will happen because of lot of communication b/w team members & teams.

Types of people in Reviews

→ The following are the different people who involves in the review process.

* **Author** :- The person who prepares the document is called as Author. He is the owner of that document.
* **Moder Inspector** (Reviewer) :- The person who performes actual review (inspection) on the document is called as "Inspector or Reviewer".
* **Moderator** (Leader for all the Reviewers) - He controls entire review process.
* **Scribe** (Recorder) - The person who documents the review process such as defects identified, defects fixed etc is called as Scribe.

Types of Reviews

→ The following are the different types of reviews that can be done in static testing.

* Informal reviews.
* Walkthroughs.
* Technical reviews
* Formal reviews.

Informal reviews

→ It is nothing but oral communication b/w the peers, there is no need fof documentation in these reviews.

→ It is mainly used to get the confidence about Review.

Walkthrough

→ It is the step-by-step explanation of the document by the author to the review team.

→ 100% documentation is required.

→ Author will conduct this process.

Technical Reviews

→ These reviews will be conducted b/w the technical people.

→ The technical people of the organization will communicate with each other to study the technical documents like HLD & LLD to understand what process to be used, what functionalities to be developed etc.

→ Moderator is the leader of the review

→ 100% documentation is required.

→ There will be no involvement of management people.

Formal Review

→ It is the most important review type, it is mainly used to identify the defects.

→ Moderator is the leader of this review.

→ 100% documentation is required.

The following are diff phases in the formal reviews.

* Planning
* Kick of meeting
* Preparation
* Review meeting
* Rework
* Follow-up

Planning

→ The author of the document will send a formal request to the moderator to review his document

→ The moderator prepares a checklist consisting of what to be reviewed.

→ The moderator also prepares entry & exit criteria

⇒ Note:- Entry criteria specifies pre-conditions to start the review & whereas exit criteria represents post-conditions to stop the reviews.

→ The moderator schedules the review process

*. Kick-off meeting

→ It is an optional phase, here the moderator will explain the check-list to the reviewer which should be documented.

*. Preperation

→ In this phase individuals will start review the document & identifies the questions to be asked to the author, later in the meeting.

→ The scribe has to document these questions.

*. Review Meeting

→ It is the important phase in formal review, here main review process will occur it is internal divided into three sub phases. They are

⊛ Logging - In this phase the scribe will document entire communication happening in the review meeting.

⊛ Discussion - In this phase the reviewers asks the questions to the author and some kind of discussion will happen about the defects

⊛ Decision - In this phase the author will take a decision about defects to be accepted (or) not

*. Rework

→ The author of document will verify how many modifications to be performed based on that he will decide whether has to be done on the document.

*. Followup

→ The entire review process should be monitored & status checking should be done by the moderator.

Dynamic Testing

→ It is the process of executing the developed s/w & verifying all the functionalities are satisfied and also customer requirements are reached or not

→ In this method we have to perform different levels of testing using different testing methodologies.

Test levels

→ While performing dynamic testing we have to test different levels as follows.

* Unit Testing.
* Integration Testing.
* System Testing
* User acceptance testing.
* Release testing.
* Maintenance testing.

Unit Testing

→ It is the process of testing independent components of the software product. Here the internal logic of each function, class, program etc should be tested.

→ We need to have technical skills to perform this test

→ In general it will be done by the developers.

Integration Testing

→ After developing independent programs the developers will start integrate one program with other

→ In this level of testing we have to verify the interconnectivity b/w the programs.

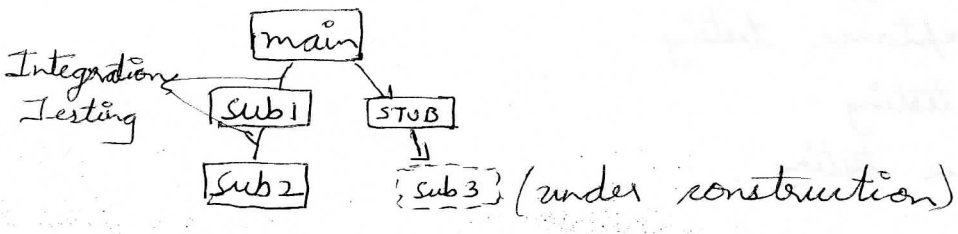
→ We will follow different approaches to perform this test.

* Top-down approach.
* Bottom-up approach.
* Sandwich / Hybrid approach.
* Big-Bang approach.

Top-down approach

→ When the main program is ready & some of the sub-programs are ready but some of the sub-programs are not ready then we use this approach.

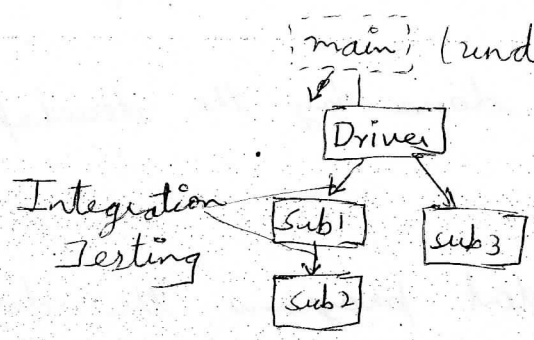
→ In this case ~~the~~ in place of sub program which is not ready we can use ~~dummy program~~ method known as "STUB".



Bottom-up approach

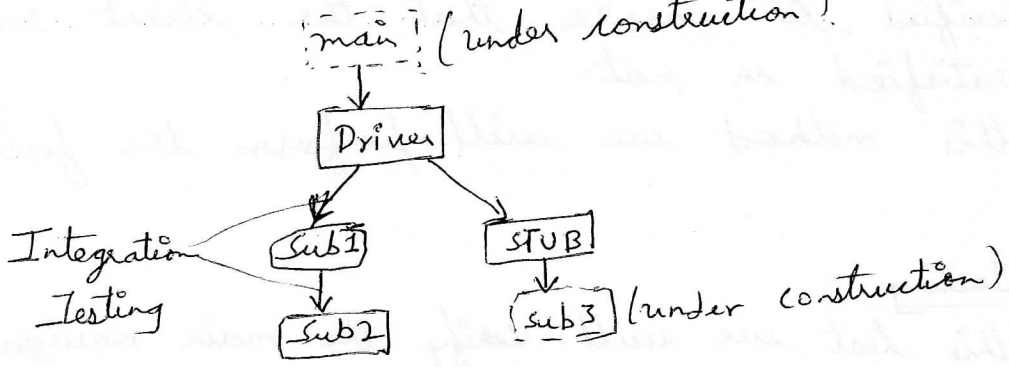
→ When all the sub-programs are ready but the main program is not ready then we use this approach.

→ Here in place of under construction main method we will use "driver method" (a dummy method).



Sandwich / Hybrid Approach [Top-down + Bottom-up]

→ When the main program & some of the sub-programs are under construction (not ready) then we use this approach.



Big-Bang approach

→ When all the programs are ready & no program is under construction then we use this approach.

→ To perform this integration testing we must have technical skills, so this test is done by the developers.

System Testing

→ It is the process of testing the s/w whether it is satisfying all the client requirements as well as client expectations.

→ This test will be conducted by the test engineer by executing the developed s/w product.

→ System testing is further divided into two ways.

They are

System Testing

- Functional (Requirements Tstg)
  - ⊛ GUI Testing
  - ⊛ Input-domain Testing
  - ⊛ Error-handling Testing
  - ⊛ Output testing
  - ⊛ Database testing
  - ⊛ Data-volume Testing
  - ⊛ Recovery Testing
  - ⊛ ~~Interfaces~~
  - ⊛ Inter-System Testing
- Non-Functional (Expectation Tstg)
  - ⊛ Usability Testing
  - ⊛ Compatability "
  - ⊛ H/w configuration "
  - ⊛ Performance Testing
  - ⊛ Security Testing
  - ⊛ Multilanguity Testing
  - ⊛ Parallel Testing
  - ⊛ Compilance Testing

2/11/18

Functional Testing

→ In this type of system testing the developed s/w will be executed each & every functionality will be verified, to ensure that the client requirements are satisfied or not.

→ In this method we will perform the following test.

* GUI Testing

→ In this test we will verify the main navigations in the application.

→ Here we test menus & menu items.

*. Input domain Testing

→ In this test we have to check each & every field in each & every screen or page to ensure that only valid input is accepting.

Ex: Name field should not allow numerics
Phone no. field should accept ten digits only.

*. Error handling Testing

→ In this test we have to verify run-time error are properly handled or not, it means when the user is providing incorrect input related error messages are displaying are not to be tested.

*. Output Testing

→ In this test we will verify for a valid input appropriate output is coming or not

Ex: If the screen contains date of birth & age fields when we enter valid date of birth correct age should be displayed automatically.

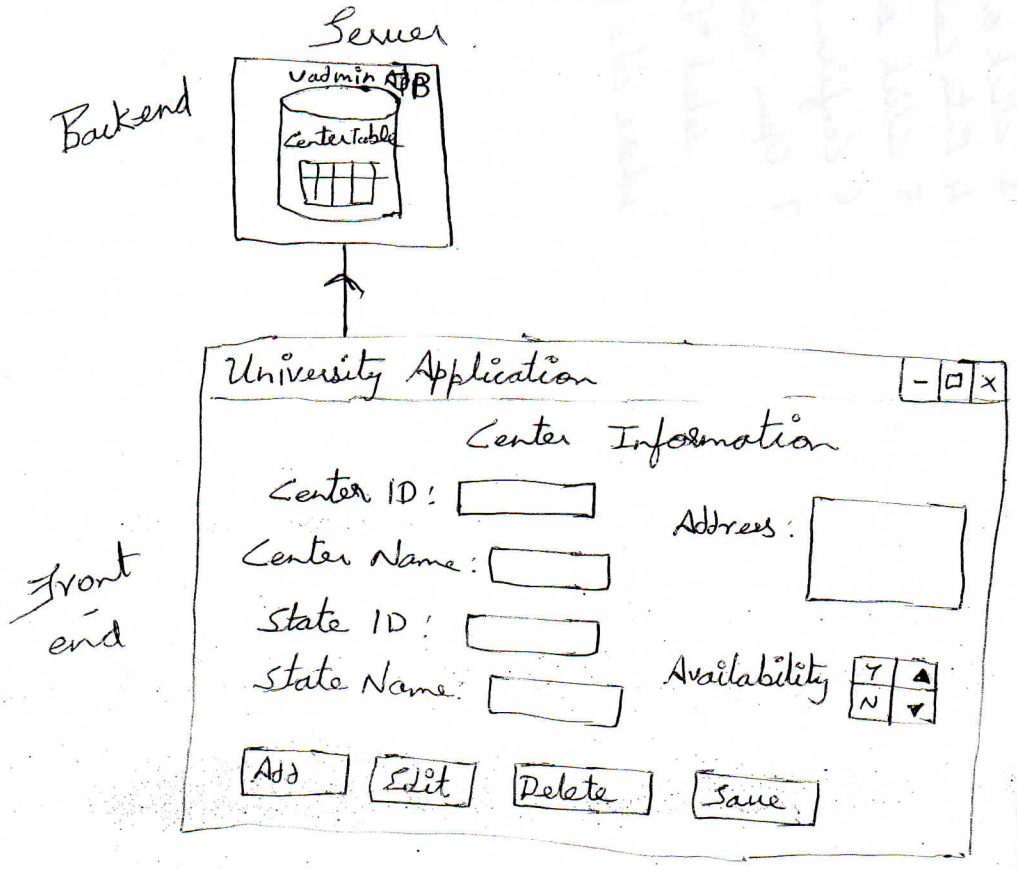
* Database testing

→ In this test we will perform an operation on the front-end application & verify whether it is correctly effecting the backend database or not.

→ In general while doing functional testing such as database testing we have to follow these steps.

- Prepare a test case
- Perform the test execution
- Fill the test result.

Ex:



5/11/2018

→ Write a test to verify the insertion operation in the center screen.

TestCase_ID	Test Scenario	TestScenario_ID	Priority	Test Procedure	Expected Result	Actual Result	Result
⊛ TC_01	→ Verifying insertion Operation in the center screen	TS_center_1	High	1. Open university App 2. Open center screen 3. Click on "Add" Button 4. Enter Center Information 5. Click on "Save" Button 6. Confirm as Yes in dioglogbox 7. Open xadmin Database Select * from center where cid = (select max(cid) from center)	→ Frontend entered data = Backend stored data	→ Matched	PASS

Datavolume Testing

→ In this test we will verify the capacity of the database.

→ Inorder to do this test we will enter some test data into the database repeatedly until the database is unable to accept any more data.

Note: The first four tests (GUI, Input domain, Error handling & Output) are called as front-end testing whereas Database & DataVolume testing are called as Backend testing.

Recovery/Reliability Testing

→ In this test we have to verify whether the application under test is converting from unstable state to stable state when the software is running in a unwanted environment, also the software should not effect the power failures as well as with other software installations.

Intersystem Testing

→ In this test we will verify the interconnectivity b/w the software to be tested & the other service calling by the software.

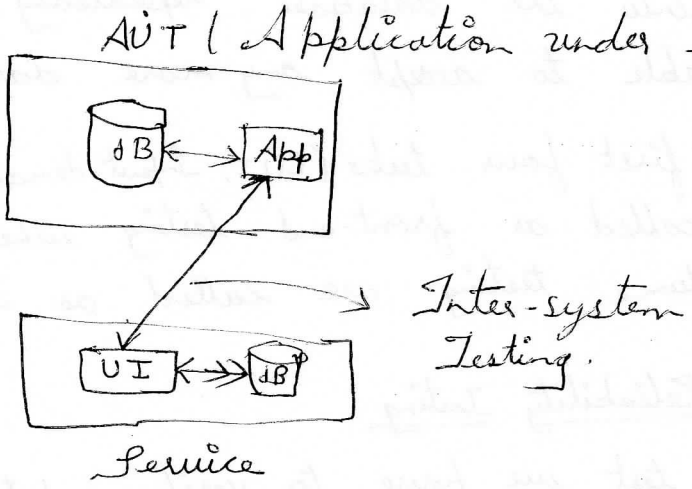
→ Now-a-days most of the softwares are using the services of other third-party softwares to get extra funtionality.

Ex: Using payment gateways in online shopping sites such as Amazon.in, Flipkart.com etc.

→ In testing such kind of websites we have to verify the following operations.

(*) When we click on "pay Now" from the shopping website we have to redirect to the payment gateway site.

* The payment process should be succed for (3) valid information.
* After successful payment we should redirect back to the shopping website to continue the ~~shopping~~



Non-Functional Testing

→ In this method we will execute the software and verify whether it is reaching clients expectations or not, that is why it is also called as "Expectations testing."

→ The following are the different test that can be conducted under this category.

(*) Usability Testing

→ In this test we will verify whether the screen (or) page is user-friendly or not, it means each & every option should be visible to the user in such a way that it is easy to understand.

(*) Compatability Testing

→ In this test we will verify the software is able to run on different networks, platforms & Browsers

For Ex: A web-application should able to run on any OS such as Windows, Linux, Mac & also Br any

browser like internet-explorer, Google-Chrome, Safari, Firefox et

⊛ Hardware configuration Testing

→ In this test we have to verify whether the software is able to run on any kind of network as well as is it supporting all kind of i/p devices such as keyboard, Mouse & o/p devices such as printers, scanners etc.

⊛ Security Testing

→ In this test we will verify whether the software is able to stop the unauthorized access or not.
→ This test can be conducted in the following ways.
* Authentication / Authorization Testing
* Access control Test
* Encryption & Decryption Test.
→ Incase of authentication testing we will verify only valid user is able to access the application or not
→ Incase of access control test we have to verify whether the users are able to perform only permitted operations are not.
→ To perform encryption testing we will check the database to ensure that sensitive data is encrypted or not.

6/11/2018

Performance Testing

→ In this test we will verify the speed & capaci of the software. This test will be conducted the following ways
* Load Test
* Stress Test
* Spike Test
* Endurance Test

* Load Test

→ In this we will verify the performance of the software with customer expected load.

Ex: In a web-application client requirement is to prepare a software with 1000 users to be logged-in at a time.

→ To complete this test as a test engineer we have to use 1000 logins to verify the connectivity.

*. Stress Test

→ In this test we will apply more than customer expected load to verify the capacity of the software.

Ex: In the previous application test with more than 1000 logins.

*. Spike test

→ In this test we will apply more than (huge) amount of load to find the server crash point.

Ex: In the previous application login with 4000/5000 users to check the server crash point.

*. Endurance test

→ In this test we will verify memory leakages in the long-run.

Ex: In the above application login with customer-expected load i.e 1000 users but (keep on login with more time)^{x} login and keep a side for more time to see whether there are any session disconnect.

⊛ Multilonguity Testing

→ In this test we will verify languages also support for the application. It is not a compulsory test to be conducted

→ If the web-site supports multilonguity, then only we perform this test.

→ This test can be done in two ways.

* Localization test - In this test we will verify the support of specific country language
* Globalization test - In this case we have to verify the support for international languages.

⊛ Parallel Testing

→ In this test we have to compare the current software with the competetive software to verify the weaknesses as well as the drawbacks of the current software.

⊛ Compilance Testing

→ In this test we will verify the standards followed by the developers interms of coding and testers interms of testing standards such as no. of test scenario's, test cases. Prepared per day & no. of test executions to be done etc.

User Acceptance Testing (UAT)

→ This test will be conducted by the user / client It will be done in the following ways.

* Alpha Testing (Factory testing) } Yellow Box Testing
* Beta Testing (Field testing) }
* Regulatory Acceptance Testing
* Contractual " "
* Operational " "

* Alpha testing ("Factory Testing").

→ In this test the client will visit the development company & perform the testing of the software in the same environment where development & testing happened.

→ While testing the software if any defects are found they will be rectified by the development team & then retesting & regression testing will be conducted by the testing team.

* Beta testing ("Field Testing")

→ The company will form a team consists of developers & testers to deploy the project on the client environment, after deploying the software the team will give training to the client.

→ In this type of test, the client will perform testing of the software in his own environment to check its correctness.

28/11/18

→ At the time of beta testing the following scenarios can occur.

* No defects - In this case the software is ready to release for usage
* If any minor defect is found it will be postponed to the next build for fixing the defect because it is not important to fix the defect immediately.
* If a major defect is found which can be fixed immediately then the development team will fix the defect & testing team will perform Re & Regression test, and then the

functionality will be tested again by the client

→ Whereas if the defect is not able to fix immediately then currently installed version of the software will be uninstalled & old version will be installed, after that "Roll-back" testing will be conducted to ensure that the old-version is working properly (or) not.

* Regulatory acceptance testing

→ For mission critical softwares such as airways & healthcare related regulation authory has to perform one more test called as "Regulatory acceptance testing".

Ex: In India for airways softwares will be tested by Indian Aviation authority & Health care softwares will be tested by FDA (Food & Drugs authority).

* Contractual acceptance testing

→ In the early stages of project some kind of agreement will happen b/w the client & the development company in the form of a contract. In this contract document the development company will describe the faulties they are providing in the software as well as to the client.

→ At the end of software development the client should verify the contract to identify whether all the faulties mentioned were implemented or not.

⊛ Operational acceptance Testing

→ In this test the system administrator will install the software in different networks with different hardware devices to check the functionality of the software.

Release Testing

→ After completion of user acceptance testing the company will form a team consisting of developers, testers and send them to the client place.

→ The developers will deploy the project in the client environment & Testers will perform main functionality testing to check the software working status

→ The testing will be conducted by observing the following activities

* Installing the software on different n/w's & check its behaviour.
* Test with different i/p & o/p devices.
* Verify co-existence with other softwares in the client's mission.

Maintenance Testing

→ This will be conducted by the CCB Team. In general there are two types of maintenance They are

* Corrective maintenance
* Enhansive maintenance

(*) Corrective maintenance

→ While using the software if the client identifies any failure then it will be reported to the CCB team.
→ The developers of the CCB team will perform "Route cause Analysis" to identify the problem area.
→ Once the development team fix the problem then testing team will perform Re & Regression test
→ Finally a patch file will be released to the client by the CCB team, the client can execute this patch file to get a solution to the problem.
→ The above process is called as 'Corrective maintenance

(*) Enhancing maintenance

→ Whenever client wants to have new features to the existing software then he will send a change request to the CCB Team.
→ The technical people of the CCB Team will perform "Impact Analysis" and prepares a document based on that analysis.
→ The management people will use this document to identify the cost & time to perform the change, the development team will identify the place where the changes has to be performed & the testing team will know where to perform Re & regression testing should be perform.
→ Finally the CCB team will release a patch file to the client which can be install to get the new features in the s/w.

Testing Methodologies

→ In order to perform testing at different level we have to make use of different testing technique based on different methods called as Testing methodologies / Test design techniques. They are.

Test Design Techniques

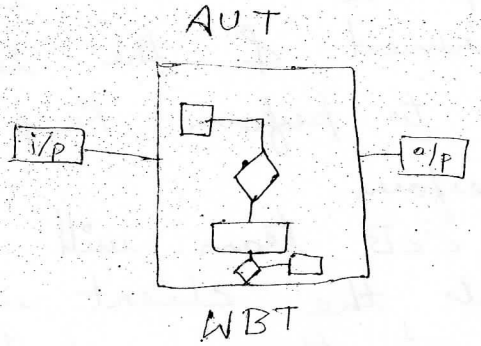
- Structure Based (White-Box Testing)
  - *. Statement Coverage Testing
  - *. Brand/Decision Coverage Testing
  - *. Path Coverage Testing
- Specification Based (Black-box Testing)
  - *. Equivalent partition (EP)
  - *. Boundary Value Analysis (BVA)
  - *. Decision-Table
  - *. State-Transition
  - *. Use-case
- Experience Based
  - * Error Guessing
  - * Exploratory.

2/11/18

Structure Based / White box Testing

→ In this methodology we will test the internal programming logic of a function or class or program.

→ In this case we are verifying structure of the program.



→ White box testing will be performed based on the following techniques. They are

*. Statement coverage
*. Branch / Decision coverage
*. Path coverage.

⊛ Statement coverage testing

→ In this technique we have to prepare a test case in such a way that most of the statements of the program should be covered.

→ The statement coverage % is calculated as follows

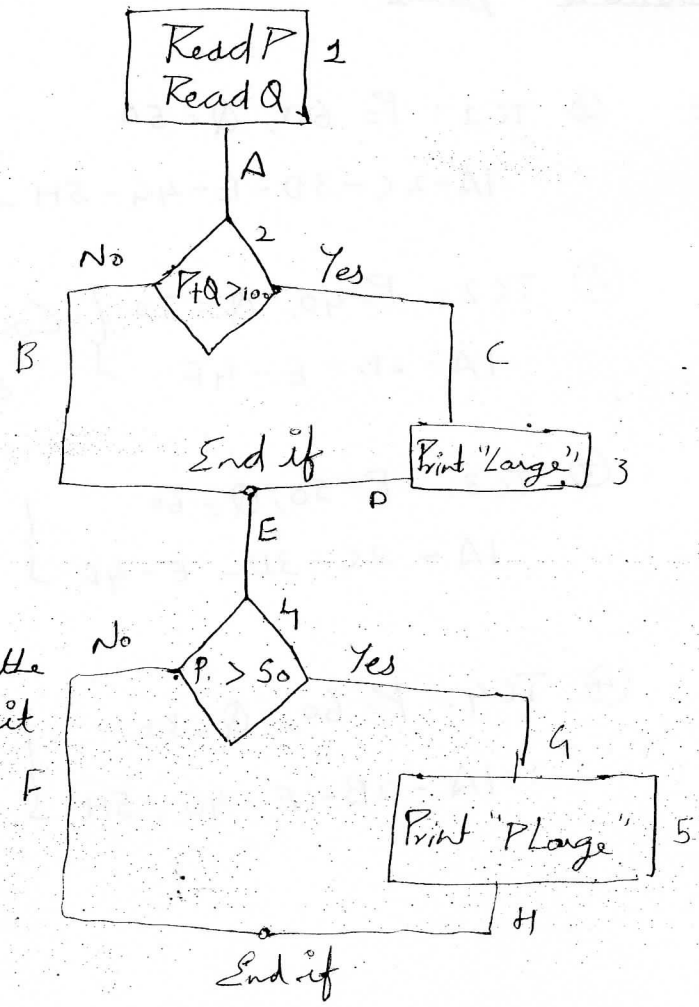
$$\text{Statement Coverage (\%)} = \frac{\text{No. of statements Covered}}{\text{Total no. of statements}} \times 100$$

Ex:

1. Read P
2. Read Q
3. If P+Q > 100 Then
4. Print "Large"
5. End if
6. If P > 50 Then
7. Print "PLarge"
8. End if

TC1 : P = 60, Q = 50

1A - 2C - 3D - 4G - 5H



→ The above test case is the best one to use because it is providing 100% statement coverage.

Decision & Branch Coverage.

→ In this method we have to prepare the test cases in such a way that they are covering the true conditions atleast once & false conditions atleast once.

Ex:

① TC1 : P=60, Q=50
1A-2C-3D-E-4G-5H } Covers all True criteria

② TC2 : P=40, Q=30
1A-2B-E-4F } Covers all False criteria

Path Coverage Testing

→ In this technique we have to prepare the test cases in such a way that it has to cover all possible paths.

Ex:

① TC1 : P= 60, Q=50
1A-2C-3D-E-4G-5H } Covers all TRUE criteria

② TC2 : P=40, Q=30
1A-2B-E-4F } Covers all FALSE criteria

③ TC3 : P=50, Q=60
1A-2C-3D-E-4F } Covers one TRUE & one FALSE criteria

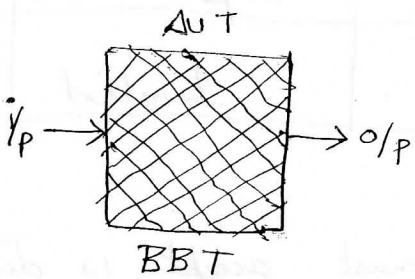
④ TC4 : P=60, Q=30
1A-2B-E-4G-5H } Covers one FALSE & one TRUE criteria

Note: White Box testing technique will be used to perform unit testing & integration testing. Generally done by "Developers".

Black-box Testing / Specification based Testing

→ In this method we have to perform the testing based on the client-specification.

→ In this method we don't check the internal logic of the program. We have to verify for a valid input appropriate output is coming or not.



→ This test can be conducted using the following techniques. They are

* Equivalent Partition (EP)
* Boundary Value Analysis (BVA)
* Decision-Table
* State-Transition
* Use-case

Equivalent Partition (EP)

→ In this method we have to divide the data into different partition based on the specification. The partition has to be done in such a way that all partitions are executing same type of behaviour.

→ We can make use of different data factors

such as *. Data Type
*. Data size &
*. Data range
based on the type of specification

Ex1: Empid should allow only numerics

Data Factor = Data Type

Partition 1	Partition 2	Partition 3	Partition 4.
Numbers	Alphabets	Alpha-Numeric	Spl Characters
Valid	Invalid	Invalid	Invalid

1/11/18

Ex2: Phone no. field must accept 10 digits

Data Factor = Data size

Partition 1	Partition 2	Partition 3
<10 digits	10 digits	>10 digits
Invalid	Valid	Invalid

Ex3: Transfer amount value should accept a value b/w 10,000 & 50,000

Data Factor = Data Range.

Partition 1	Partition 2	Partition 3
Below 10,000	B/w 10,000 & 50,000	Above 50,000
Invalid	Valid	Invalid

⊛ Boundary Value Analysis

→ This method is used to test the specification with extreme minimum values & extreme maximum values so that all possible test cases will be covered.

→ Generally for testing data range factor this technique is more useful

Ex: Transfer amount should be between 10,000 & 50,000

BVA	Value	Valid/Invalid
min-1	9999	Invalid
min	10,000	Valid
min +1	10,001	Valid
max -1	49,999	Valid
max	50,000	Valid
max + 1	50,001	Invalid

⊛ Decision table

→ Whenever we can specify the value is valid/Invalid based on some special condition then we have to this decision table technique.

Ex 1: Candidate age should be in b/w 21 and 35

Conditions

a.) For SC/ST Candidates 5yrs age relaxation

b.) For BC Candidates 3yrs age relaxation

c.) For ph Candidates 6yrs age relaxation

category	age	valid/Invalid
OC	20	Invalid
OC	37	Invalid
SC	37	Valid
Ph	40	Valid
ST	50	Invalid

Ex2: Banking system interest rates for fixed deposits

1 to 2 yrs	7%
2 to 5 yrs	8%
above 5yrs	10%

For senior citizens 0.5% extra interest for all ranges.

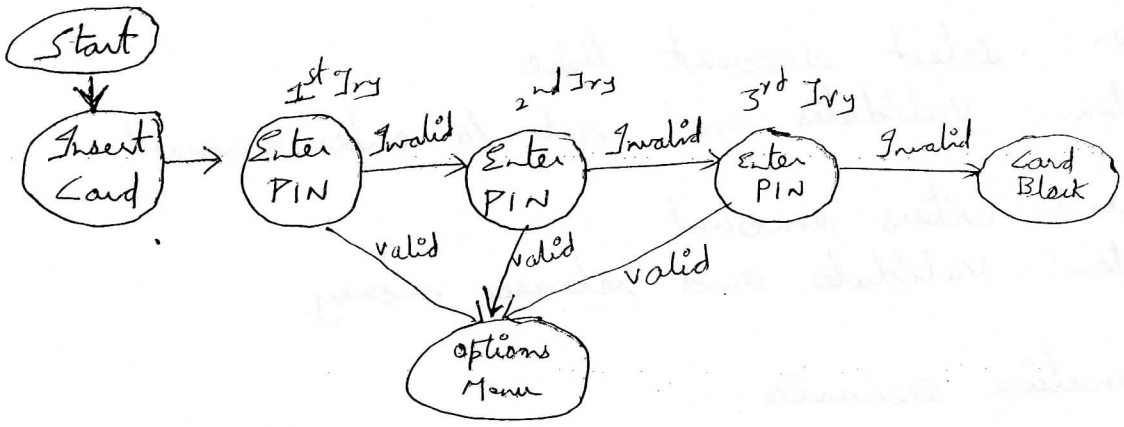
Age	period	Interest Rate	Valid/Invalid
25	1	7%	Valid
66	1	7.5%	Valid
35	6	8%	Invalid
75	3	7%	Invalid
65	2	7.5%	Valid

State - Transition testing

→ When there is a iteration of input to test a functionality then we use this technique.

→ In this case for some input b/w iterations the state will be changed from one way to other way.

Ex 1: ATM pin Entry



Ex 2: Amount withdraw in Banking system

Balance	Withdraw Amount	Valid/Invalid
50,000	20,000	Valid
30,000	20,000	Valid
10,000	20,000	Invalid

⊛ Use-case Testing

→ When there is frequent communication b/w the user and the system then we perform this kind of testing technique.

→ In this case we will have mainstream scenario & alternative scenario's.

Ex: ATM cash withdraw process

* Main stream scenario

1. User: Inserts ATM card
   System: Validates and ask for pin
2. User: Enters pin
   System: Validates pin and ask to select language

3. User: selects language
System: Validates and ask to select account type

4. User: select account type
System: Validates and ask to enter amount

5. User: enters amount
System: Validates and releases money.

* Alternative scenarios:

2-a. suppose if the user enters invalid pin

System: shows error message and ask to enter correct pin

User: enters correct pin

4a. if user select invalid account type

System: shows error message and ask to select correct account type

User: selects correct account type

5a. if user enters incorrect amount (more than the balance / more than daily limit)

System: shows related message and again ask to enter correct amount.

User: enters correct amount.

14/11/18

Experience based testing

→ This method will be used by the testers who are already having experience in the environment to be tested.

→ This test will be conducted in two ways. They are

* Error guessing testing
* Exploratory testing.

⊛ Error guessing testing

→ As a test engineer if we are already having experience of testing similar type of software previously then we can guess the errors where they might occur so that directly those functionalities can be tested.

⊛ Exploratory testing

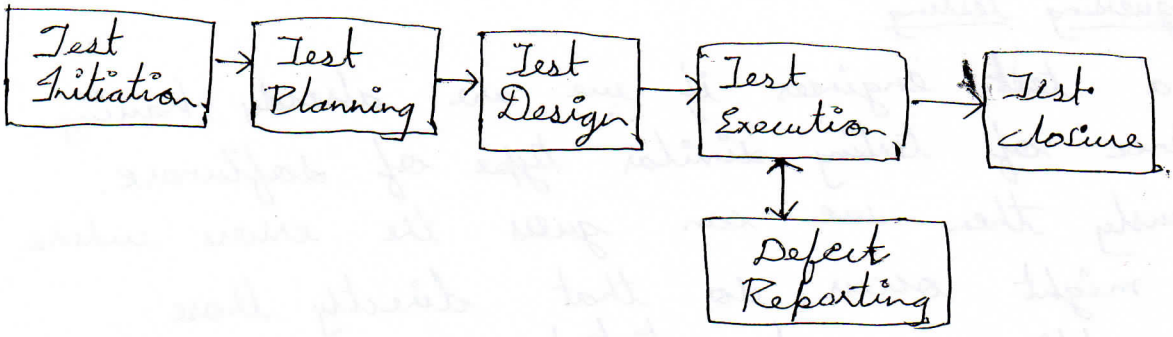
→ As a tester if we are not having the knowledge about the environment to be tested then the company will provide training from the experts.

→ After gaining the knowledge we can explore it by testing the current software.

Software Test Lifecycle (STLC)

→ STLC is the process we will follow in an organization to implement the testing process.

→ This STLC internally contains different phases. They are. Each phase will take some input & do some activity and delives some output; this output will become input to the next phase. The phases are



⊛ Test Initiation

→ In this phase we have to define the testing approach.

→ In general software testing can be done in the following ways.

*. Exhausive testing
*. Adhoc testing
*. Optimal testing

* Exhaustive testing

→ In this kind of testing we have to prepare the test cases in such a way that all possible values should be used to conduct the test.

Ex: In a screen transfer amount field should accept a value b/w 10,000 & 50,000

→ In exhaustive testing we have to write different test cases for the values 10,000, 10,001, 10,002 50,000
→ It is impossible to perform.

*. Adhoc testing
→ In this method testing will be performed based on random values.

Ex: Testing the above scenario with values like 5000, 20,000, 60,000 etc.

→ This method is not preferable because, it is not performing the complete testing.

* Optimal testing
→ In this method we will test with all possible conditions.
→ It is the best approache to be used.

Ex: In this method the above scenario will be tested with the following values.

min - 1 → 9999
min → 10000
min + 1 → 10001

max - 1 → 49999
max → 50000
max + 1 → 50001

Test strategy document

→ In the test initiation phase we will define the testing approach with the help of a test strategy document by following IEEE 829 standards.

→ The following are the different parameters contained by the test strategy document.

1. Test strategy document ID :- It is a unique name (or) numbers used to recognize the document.
2. Scope and Objective :- Here we specify the importance of testing in the current sprint / software.
3. Business issue :- Here how much budget should be allocated for testing will be mentioned.
   Ex: 70% for development, 30% for testing.
4. Test Responsibility Matrix (TRM)

→ Here we have to list out the tests to be conducted.

Ex:

Testing Topics	Yes/no (Comments)
* Functional Testing (8 Topics)	Yes
* Usuability Testing	Yes
* Compatability Testing	Yes
* H/w configuration Testing	Yes
* Performance Testing	No need, But lack of resou
* Security Testing	No need, But lack of skill
* Multilanguity	No, No unicode requir requirem

15/11/18

5. Roles & Responsibilities

→ The PO will define different roles and responsibilities to the scrum master & scrum team testers as follows.

* Scrum Master
  - → Prepare test plan
  - → Conduction meeting
  - → Responsible for status reporting.

* Sr. Testers (Quality Assurance)
  - → Prepare test scenarios, cases & data
  - → Review scenario, cases & data
  - → Responsible for defect tracking.

* Jr. Testers (Quality Control)
  - → Executing test case on sprint/software
  - → Shares Defect & Report to developers.

6. Test automation & Testing Tools

→ The PO will decide is there any need for implementing automation testing for the current sprint.

→ After taking a decision he has to specify what automation tool to be used

Ex: For functional Testing - Selenium / QTP

For Performance Testing - Jmeter / Load Runner

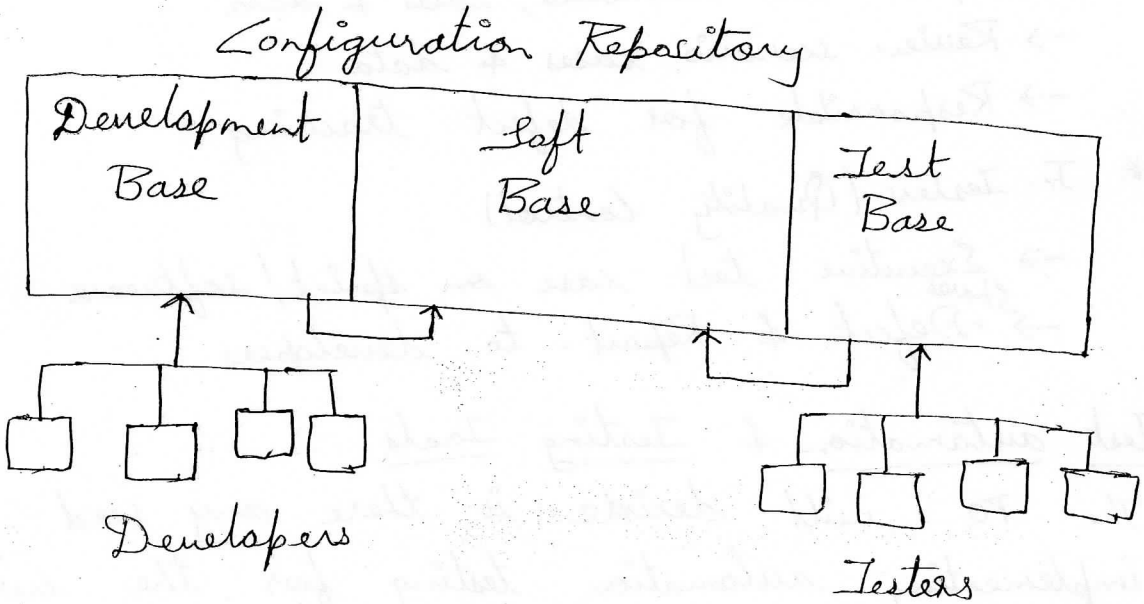
7. Defect report & Tracking

→ The product owner is responsible to prepare a communication channel b/w the developers and testers for the purpose reporting & tracking.

→

8. Configuration Mangement & Test Managment

→ It is the responsibility of the product owner to prepare a configuration repository on the server computer, this is the place where the developers and testers will store their information.



9. Testing measurements & Metrics

→ The PO is responsible to calculate the efficiency of the testers with the help of measurements & metrics

Ex: * A testers should prepare 15 to 20 test scenarios per day.

* A testers should able to implement 10 to 15 test case executions per day etc.

10. Training Need

→ By checking the efficiency of testers the PO can understand any training is needed for the testers or not.

Note: This document is prepared by the product owner either in excel or MS Word, it should be approved by the stakeholder.

Test Planning

→ After defining the test approach in the test initiation phase by the PO, scrum master/ Test lead will start this test planning phase.

→ In this phase the scrum master will define the following activities in the form of a test-plan document.

* What to test
* How to test
* When to test
* What to test.

→ The following are the different parameters that are to be specify in the IEEE 829 standard test plan document.

1. Test plan document ID

→ It represents a unique name or value to identify the document.

2. Introduction

→ It specifies some description about current sprint.

3. Features (or) Modules

→ Here we have to specify list of modules in the current sprint.

Ex: In a banking software the modules are as follows
* Customers
* Banking
* Loans
* Reports etc.

4. Features to be tested

→ Here we specify the modules to be tested

Ex: Customers, Banking.

5. Features not to be tested

→ Here we specify the list of modules we are not testing.

Ex: Loans, Reports.

Note: 3, 4 & 5 properties represents "What to test".

6. Approach (or) Strategy

→ Here we have to specify test strategy, document ID

7. Test Environment

→ It represents the list of hardware & software needed for the tester computer to perform the testing.

8. Test Deliverables

→ It represents the list of documents that should be prepared by the tester

Ex: * Test case document
* Test data document

9. Entry Criteria

→ Here we have to specify pre-conditions to start the test execution.

Ex: *. Test case & Test data document prepared by the testers should be available.

*. The sprint to be tested should be ready.

*. The test environment must be established with required S/w & H/w.

10. Suspension criteria

→ It represents when we have to interrupt the test execution process.

Ex: *. When a high severity & high priority (show stopper bug) is occured.

11. Exit criteria

→ It specifies when we have to stop the test execution.

Ex: *. When all the bugs are closed & all functionalities are satisfied.

Note: The properties 6 to 11 represents "How to test".

12. Staff

→ It specifies the names of the testers who are involving in the current sprint test

13. Responsibilities

→ Here we specify work allocation for the testers

Note: Properties 12 & 13 represents "Who to test".

14. Schedule

→ Here date & time to be followed by the tester will be specified.

Note: The above property (14) represents "When to test"

15. Risks & Assumptions

→ Here the list of risks & solutions for them will be represented.

16. Signatures of PO & Stakeholders :

16/11/18

Test Design

→ After completing the test planning by the team leader or scrum master, the test engineer or scrum team tester will perform this test design.

→ The main objective of this phase is to prepare test scenarios & test cases.

Test scenario

→ It is an activity to be tested.

Ex: Verifying Login process.

Test case

→ Test cases are different conditions that can be use to test an activity.

Ex: For the above test scenario we can prepare two test cases.

i. Testing the login process with valid username & password.

ii) Testing the login process with invalid username and password.

→ Inorder to prepare the test scenarios & test cases we must have knowledge about test design technique such as Black box testing techniques & Regular expression.

Regular expressions

→ It is a simple representation of complex programming logic. We can use different special symbols called wild card characters inside these regular expressions to allow certain no. of characters or digits

* [] - one position.
* [0-9] - one digit
* [a-z] - one small alphabet
* [A-Z] - one capital alphabet
* [a-z A-Z] - one alphabet
* [A-Z] [a-z] - Two positions with one uppercase alphabet & one lower case alphabet.

  Ex: Bc
  Xp

* [A-Z] [a-z]{2} - 3 positions, 1st uppercase alphabet followed by 2 lowercase alphabets.
* [A-Z] [a-z]{2} [0-9] - 4 positions, 1st uppercase alphabet, 2 lower case alphabets & then 1 digit

⊛ 10 positions, first 5 positions are uppercase alphabet and 4 digits and then 1 lowercase alphabet.

[A-Z] {5} [0-9] {4} [a-z].

⊛ 6 digits but not start with zero and not end with odd digit.

[1-9] [0-9] {4} [02468]

[^0] [0-9] {4} [^13579].

⊛ 1 to 10 digits

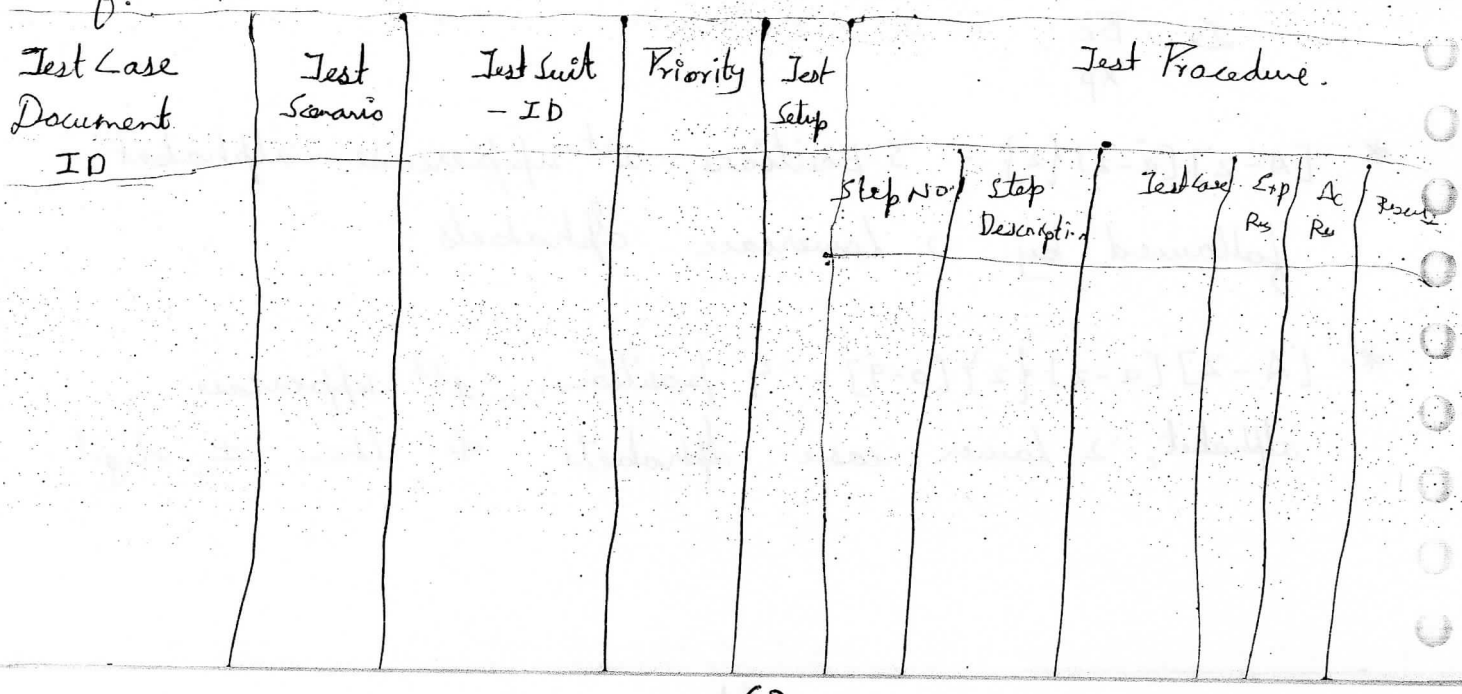
[0-9] {1,10}

⊛ 2 positions to 10 positions data with first as uppercase alphabet and remaining are digits.

[A-Z] [0-9] {1,9}

IEEE 829 standard test case design document.

→ As a test Engineer while doing the functional testing we have to prepare a test case document followed by IEEE 829 standards as follows.



⊛ Prepare a test case design document for the following user story.

User story id	As a	I would like to	So that	Acceptance criteria
⊛ Login	Existing user	→ Launch net-banking site → Enter customer id which is an 8 digit number but does not starts with 0 and 1. → Enter password which is a alpha-numeric with 4 to 16 positions → Click on "sign-in" button	→ If given data is valid. options page should be appeared → If given data is invalid error message should be displayed	→ Login operation is user-friendly → Login page should be possible to open by using chrome/fire-fox/safari/opera/ie on win/mac/linux systems → 1000 users can try to login at a time.

→ The test case for the above user story can be implemented as follows:

Project Name : Net Banking site
Module Name : Login
Tester Name : Sai
Designation : Test Engineer
Prepared Date : 19th Nov 2018
Review Date :
Reviewed By :

TestCase_ID	Test Scenario	Test case Description	Test suite_ID	Priority	Test setup	Test Procedure		
						Step / StepDesc	TestCase	Expected Result
TC_NBS Login 1	→ Verifying Login process in NBS site	→ Verifying the launching of login page	TS_login_1	High	NBS is ready to launch	1 → Launch NBS site 2 → Click on Login link	→	→ NBS site home page should be opened → Login page should be opened.
TC-NBS Login 2	→ Verifying Login process in NBS site	→ Verifying customerid field	TS_login_2	High	Login page should be ready	1 → Enter customer ID	→ Enter 7 digits → Enter 8 digits → Enter 9 digits → [^0 1] [0-9]{7} → [a-z A-Z]{8} → spl chars → Blank space	Rejected Accepted Rejected Accepted Rejected Rejected Rejected
TC-NBS Login 3	→ Verifying login process in NBS site	→ verifying password field	TS_login_3	High	Login page & Customerid should be ready	1 → Enter Password	→ Enter 3 positions → Enter 4 positions → Enter 15 positions → Enter 16 position → Enter 17 position → [a-z A-Z 0-9]{4,16} → spl chars → blank space	Rejected Accepted Accepted Accepted Rejected Accepted Rejected Rej

64

⊛ TC-NBS_Login4	→ Verifying Login process in NBS site	→ Verifying sign-in button	→ TS_Login4	→ High	→ Login page filled with Customerid & Password	1. Click on sign-in link	→ Valid customerid & password		→ Options page should be displayed
							→ Invalid customerid/ password		→ Related Error message should be displayed.

21/11/18

Usability test design document

→ In general in the usability testing we will verify how user friendly the application is and also the look and feel of the application.

→ Most of the applications will make use of the following usability testing document.

Test Case Document ID	Test Scenario	Test Suit ID	Priority
TCD_NBS_SPRINT1_Sai_UT_1	1. Validate spellings in all pages	TS_UT_01	Low
	2. Validate initially capital letter of labels in all pages.		
	3. Validate meaning full names are provides for the lables in all pages.		
	4. Validate lables font size/style is uniform in all pages.		
	5. Validate alignment of all elements		
	6. Validate linespace gap which should be uniform b/w text of the labels.		
	7. Validate linespace gap which should be uniform b/w elements		
	8. Validate mapping in b/w icon symbols and corresponding funtionality		
	9. Validate tool tips		

10. Validate format visibility for date & time fields
11. Validate meaningful error messages are displaying or not.
12. Validate confirmation for delete option.
13. Validate the existence of control box (min, max, restore) (for windows)
14. Validate the existence of status bar.
15. Validate grouping of related elements
16. Validate the existence of help content for critical options in the pages.
17. Validate short-cuts for the menu items.
18. Validate keyboard access in all elements.
19. Validate the availability of scroll option when the screen is changed in size.

Compatability Test design document

→ Based on the client expectations we have to verify whether the software is running in the client expected environment or not with the help of following document.

Test Case ID	Test Scenario	Test Suit ID	Priority	Test Environment		
				Component	Version	Y/N
TCD_NBS_SPRINT1 -SAI_CT_1	→ Valid login operation in customer expected platform	TS_CT_1	Medium	Browser	IE	Y
					mf	Y
					gc	Y
					safari	Y
				OS	opera	Y
					window	Y
					Linux	Y
					mac	Y
				OS(mobile)	Android	Y
					IOS	Y
					Win	Y

Test Procedure			Expected
Steps	StepDesc	TestCase	
1.	→ Launch site & do login	→ In Specified Environment	→ Options page should be displayed for valid login & Error message has to be displayed for invalid login

68

Hardware Configuration design document

→ The following document describes how the hardware configuration testing can be performed.

Test Case Description	Test Scenario	Test Suite ID	Priority	Test Environment			Test Procedure			Expected Result
				H/w Comp	Type	Y/N	Step	Step Des	Test Case	
1. TCD_NBS_Sprint1_SAI_HCT-1	→ Validate login operation in customer expected hardware configuration	TS_HC-1	Medium	n/w	bus hub Ring	Y Y Y	1 →	Launch NBS and do login	→ In specified Environment	→ Options page should be displayed for valid login & Error msg has to be displayed for invalid login
2. TCD_NBS_Sprint1_SAI_HCT_2	→ Validate money transfer in customer expected hardware configuration	TS_HC-2	Medium	n/w	bus hub Ring	Y Y Y	2. →	Perform money transfer	→ In specified Environment	** → Succesful message has to be displayed for valid data & error message has to be displayed for invalid data.

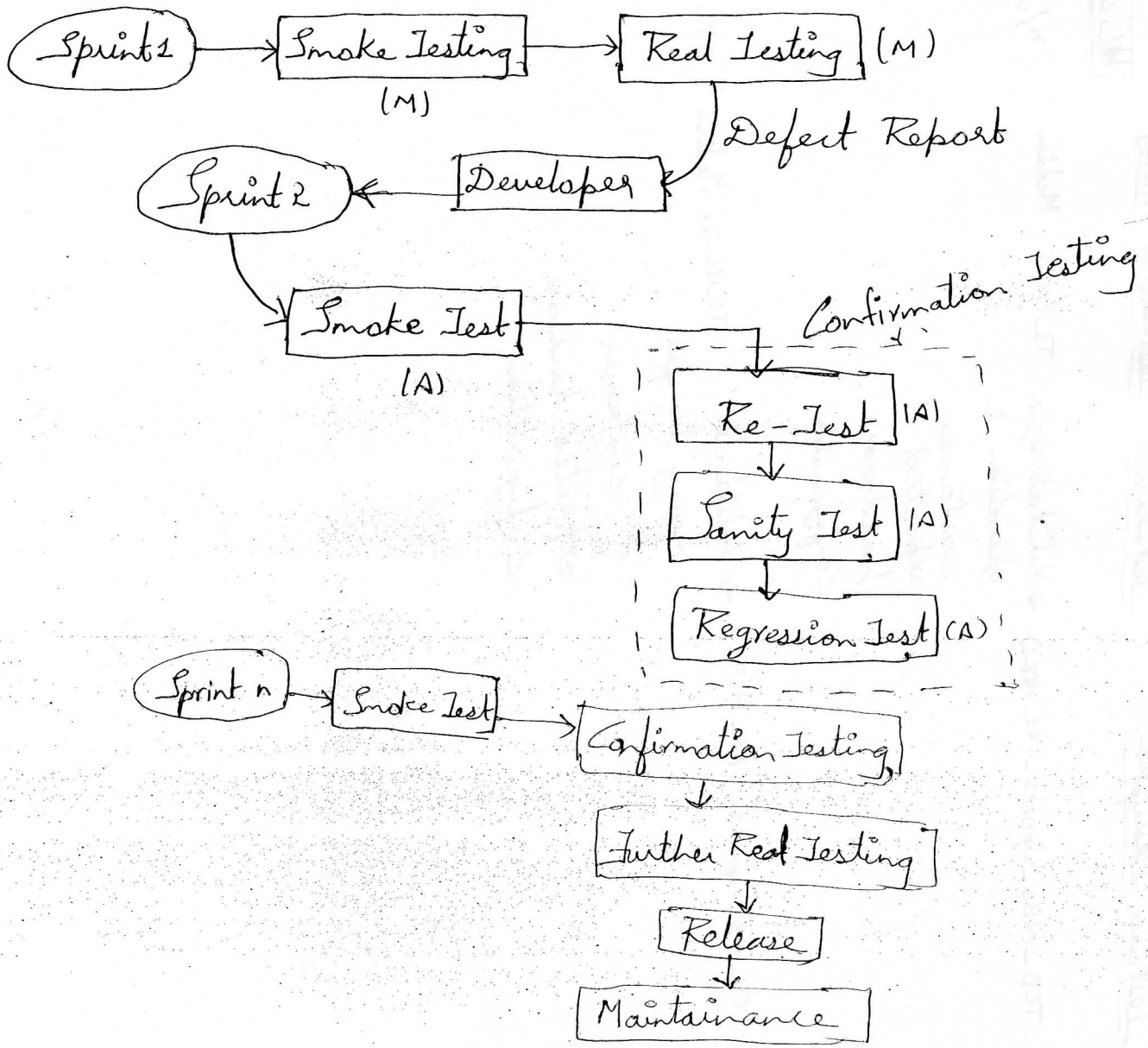
22/11/18

Test Execution

→ After completing test case designing we have to start execute one by one test based on the steps in the test case document. After test execution observe the actual result and fill that information in the test case document.

→ If the expected result is equal to the actual result then we can say the test is "PASS" otherwise the test is "FAIL".

→ The test execution process is as follows.



Smoke Testing

→ Instead of testing all the functionalities at first we have to identify most important functionalities and test them, when they are passed then only we have to start real testing.

→ This kind of testing is called as "Smoke Testing".

→ It is mainly used to verify whether the s/w is testable or not.

Re-Testing

→ After a bug is fixed it is the responsibility of a test engineer to verify whether the bug is fixed or not by testing the same test case again.

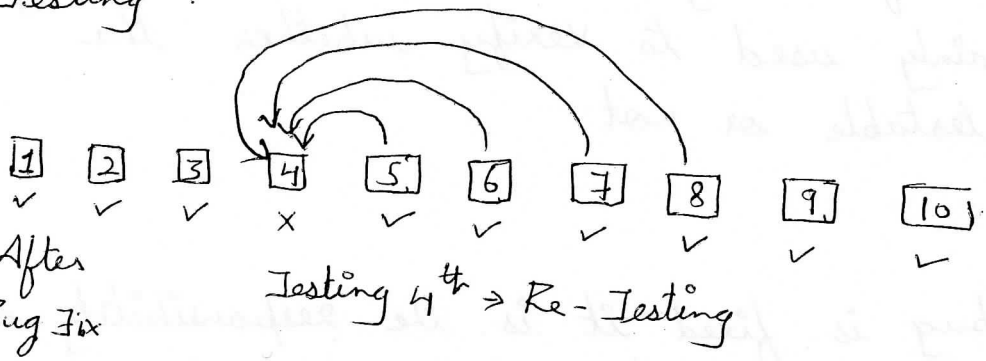
→ This process is called as "Re-Testing".

Regression Testing

→ In order to fix the bug the developers might change the code and releases a new build, once the build is modified the test engineer has to verify whether all the dependant functionalities are working or not. This is called "Re-gression" Testing.

Ex: Assume that there are 10 functionalities in the application, 5th, 6th, 7th & 8th functionalities are dependant on the 4th functionality, Suppose all the functionalities are working properly except the 4th, now the test engineer will send a defect report to the development team about the defective functionality.

→ After the development team fixed the defect then the tester will test the correctness of 4th functionality. this is called Re-testing followed by this he has to test the dependant functionalities 5th, 6th, 7th & 8th this process is called as "Regression-Testing".



Sanity Testing

→ It is a "subset of Regression Testing". When there is no-sufficient time to perform the regression Testing then we perform "Sanity Testing".

→ In this test instead of testing all the dependent functionalities we have to conduct the test on most important dependent functionalities

Defect Reporting

→ This phase is parallel to the test execution phase

→ While doing the test execution we might find the defects in the functionalities, once a defect occurs the test engineer has to prepare a defect report document and report that to the development team.

→ The IEEE829 standard defect report contains the following properties.

Defect Report Format

1. Defect ID : Number/Name as title for the report.
2. Defect description : Details of defect.
3. Sprint/software build version : Version of sprint where this defect was detected.
4. Feature/module : Name of the model in which the defect was detected.
5. Failed testcases document ID : ID of test case document in which this defect is detected.
6. Severity : The seriousness of the defect wrt Tester.

* High/Critical/Showstopper - not able to continue further Testing
* Medium/Major - Able to continue further Testing, but mandatory to fix this defect
* Low/Minor - Able to continue further Testing, but may or may not fix this Test.

7. Priority: (test lead)

The importance of defect fixing wrt customer

Ex: High, Medium, Low.

8. Reproducable: (Yes/no).

→ Is the defect occuring everytime or not?

→ When defect came everytime we will give "yes" and when defect occurs rarely we give "no".

9. If, yes, attach test cases:

→ When we get defect everytime we can send mail to developer.

10. If, no attach test case & screenshot.

11 Status - Represents current status of the defect

New - When defect came first time

Reopen - When developers are not responding until that bug is fixed.

12. Test Environment.

→ In which environment defect occured (details of tester computer).

13. Detected by

→ Name of the testers.

14. Detected on.

→ Date & time of defect detection & reporting.

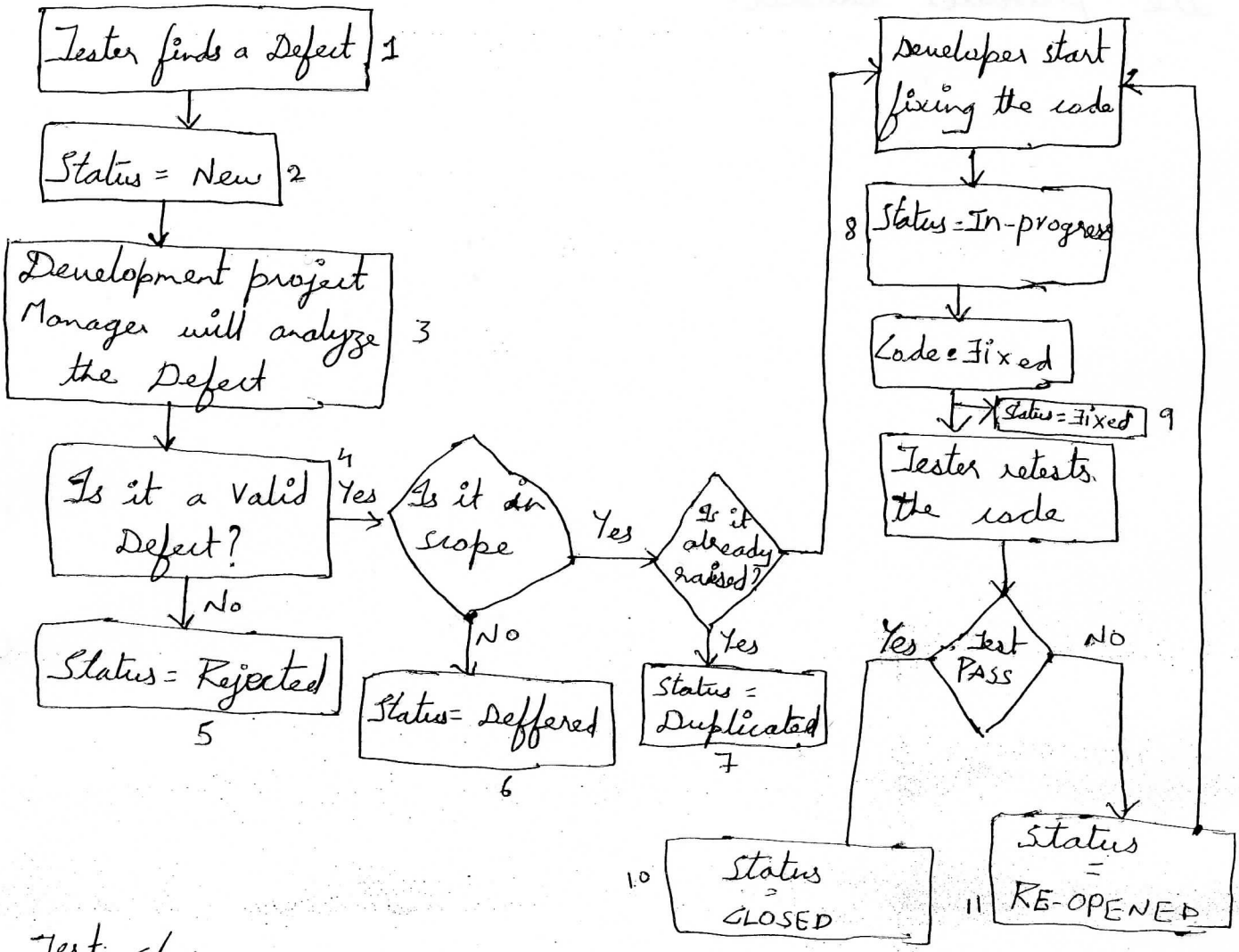
15. Suggested fix (optional):

Suggestions to be fix defect if you know.

23/11/20

Defect or Bug Lifecycle

→ From the point of finding the defect upto fixing the defect different stages will be there in the defect lifecycle which is as follows.



Test closure

→ After following the STLC process to test the software we will stop the testing when the following activities are satisfied

* All test cases should be passed
* Test cases with defects raised during the test execution should be either closed/deffered

* Showstoppers bugs in the last failed build must be rectified.
* All the functionalities must satisfy client requirement
* The entire information will be summarize in a document called test summary report prepared by scrum master/ Test lead & approved by the product owner.