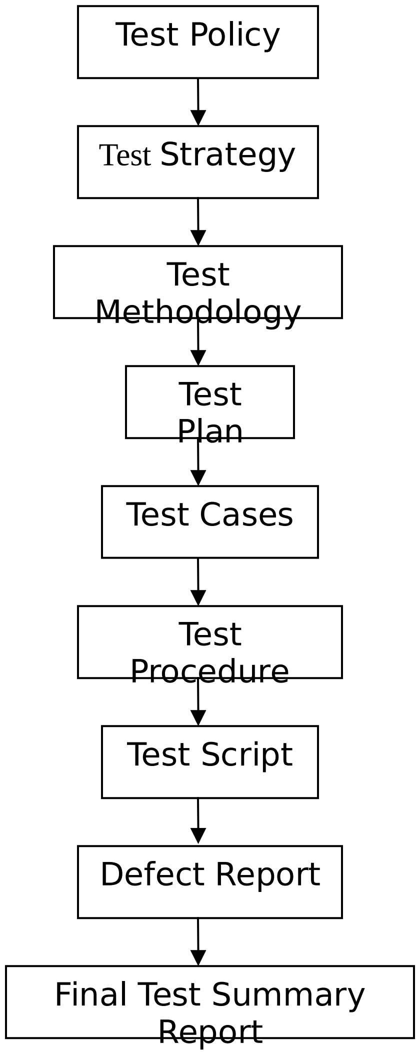
Testing Documents



Above Figure, shows the various levels of documents prepared at project testing. Test Policy is documented by Quality Control. Test Strategy & Test Methodology are documented by Quality Analyst or Project Manager. Test Plan, Test Cases, Test Procedure, Test Script & Defect Report are documented by Quality Assurance Engineers or Test Engineers.

Test Policy & Test Strategy are Company Level Documents. Test Methodology, Test Plan, Test Cases, Test Procedure, Test Script, Defect Report & Final Test Summary Report are Project Level Documents.

1) TEST POLICY:

This document developed by Quality Control people (Management). In this document Quality Control defines "Testing Objective".

Test Policy Document

Address of the Company

Test Definition : Verification & Validation

Testing Process : Proper planning before starts testing

Testing Standards : One defect per 250 lines of code or 10 FP (Functional points)

Testing Measurements : QAM, TTM, PCM

****
(C.E.O)

QAM: Quality Assurance Measurements, how much quality is expected

TTM: Testing Team Measurements, how much testing is over & is yet to complete

PCM: Process Capability Measurements, depends on old project to the upcoming projects.

---

2) TEST STRATEGY:

This is a Company level document & developed by Quality Analyst or Project Manager Category people, it defines "Testing Approach".

Components:

a) Scope & Objective: Definition & purpose of testing in organization
b) Business Issue: Budget control for testing

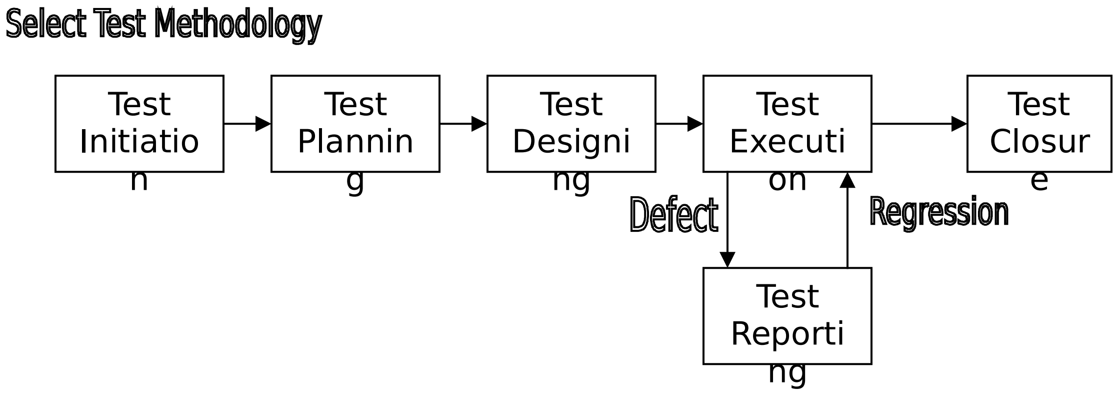
c) Test Approach: Mapping between development stages & Testing Issue.
d) Test Deliverables: Required testing documents to be prepared
e) Roles & Responsibilities: Names of the job in testing team & their responsibilities
f) Communication & Status reporting: Required negotiation between testing team & developing team during test execution
g) Automation & Testing Tools: Purpose of automation & possibilities to go to test automation
h) Testing Measurements & Metrics: QAM, TTM, PCM
i) Risks & Mitigation: Possible problems will come in testing & solutions to overcome
j) Change & Configuration Management: To handle change request during testing
k) Training Plan: Required training sessions to testing team before start testing process

Testing Issues:

1. Authorization: Whether user is valid or not to connect to application
2. Access Control: Whether a valid user have permission to use specific service
3. Audit Trail: Maintains metadata about user operation in our application
4. Continuity of Processing: Inter-process communication
5. Correctness: Meet customer requirement in terms of functionality
6. Coupling: Co-existence with other existence software to share resources
7. Ease of Use: User Friendliness of the screens
8. Ease of Operator: Installation, Un-installations, Dumping, Uploading, Downloading, etc.,
9. File Integrity: Creation of backup
10. Reliability: Recover from abnormal state
11. Performance: Speed of processing
12. Portable: Run on different platforms
13. Service levels: Order of functionalities
14. Maintainable: Whether our application build is long term serviceable to our customer
15. Methodology: Whether our tester are following standards or not during testing

3) TEST METHODOLOGY:

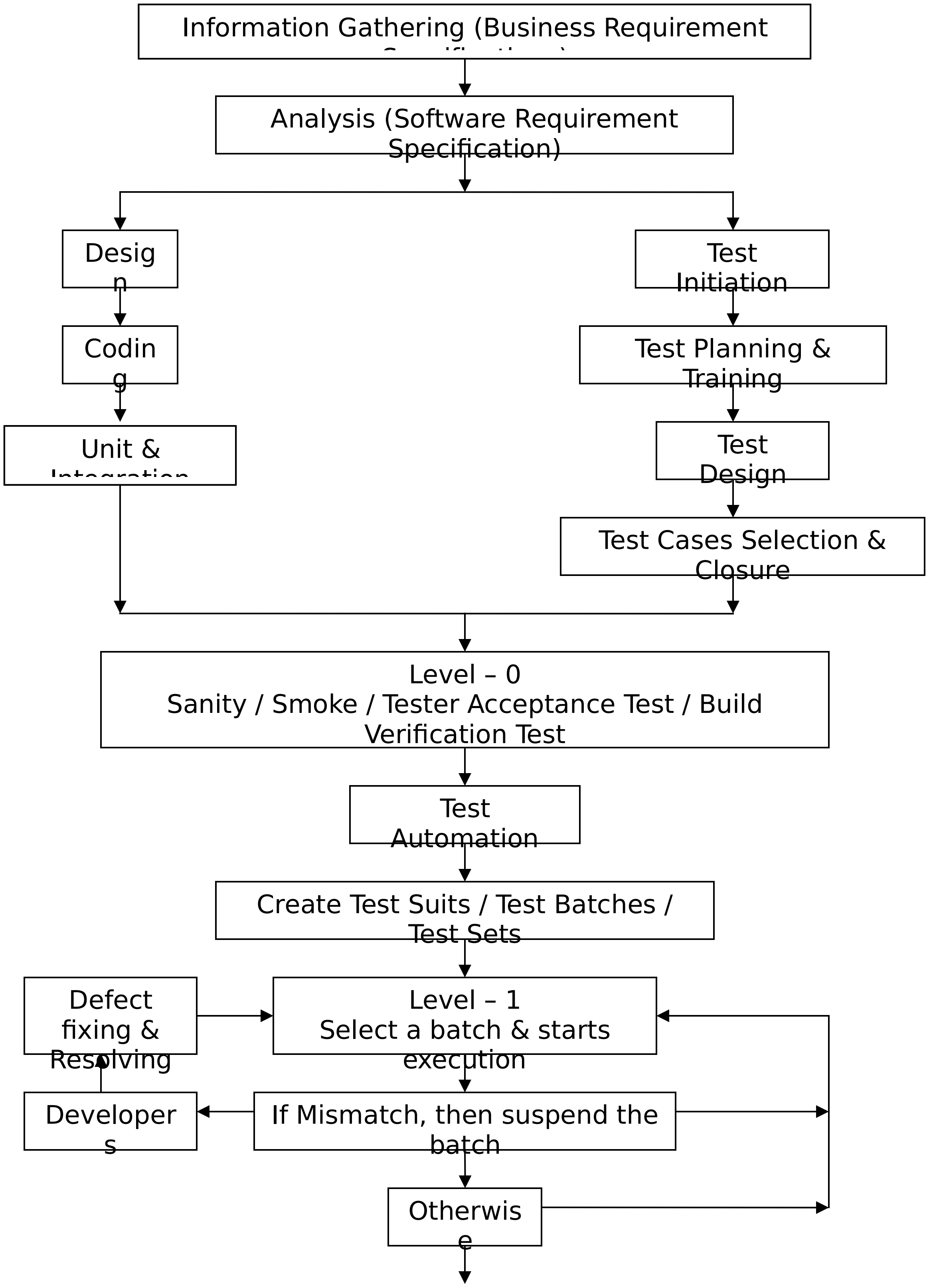
It is project level document. Methodology provides required testing approach to be followed for current project. In this level Quality Analyst select possible approach for corresponding project testing.

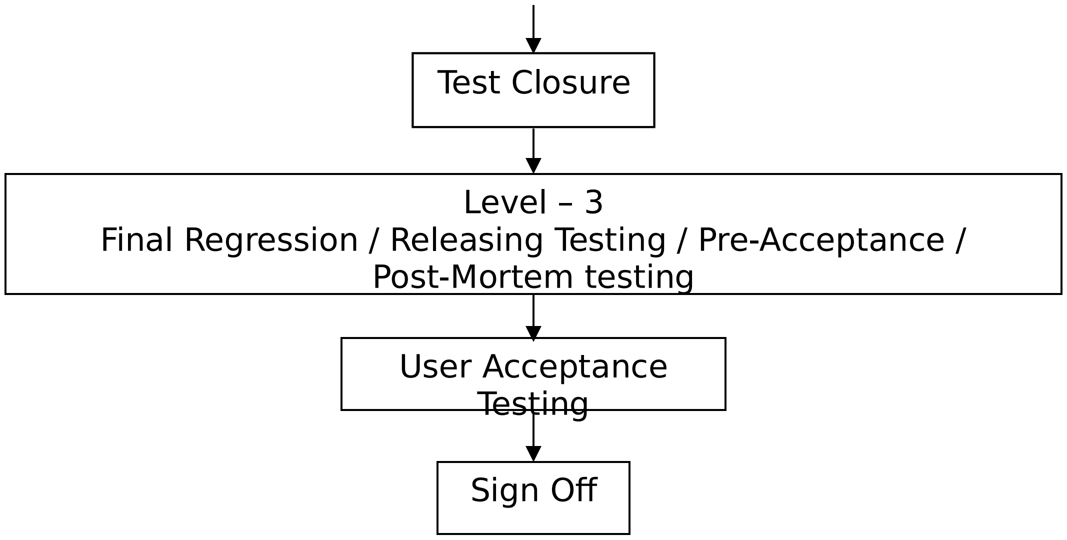


Pet Process:

Process involves experts, tools & techniques. It is a refinement form of V-Model. It defines mapping between development & Testing stages. From this model, Organizations are maintaining separate team for Functional & System testing & remaining stages of testing done by development people. This model is developed in HCL & recognized by QA Forum of INDIA.

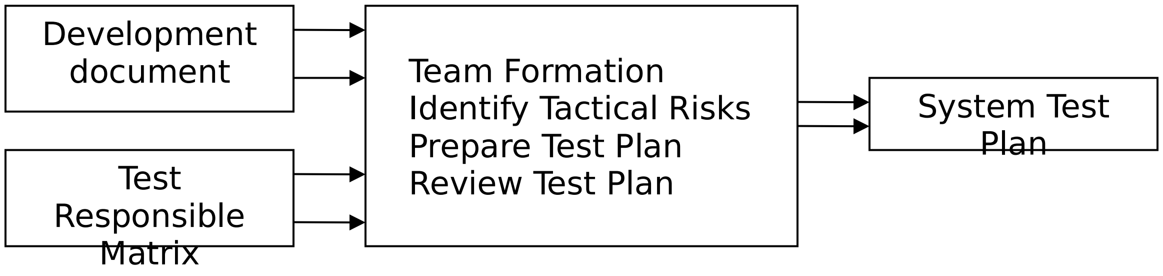
TESTING PROCESS


Information Gathering (Business Requirement
Analysis (Software Requirement Specification)
Design
Coding
Unit & Integration
Test Initiation
Test Planning & Training
Test Design
Test Cases Selection & Closure
Level – 0
Sanity / Smoke / Tester Acceptance Test / Build Verification Test
Test Automation
Create Test Suits / Test Batches / Test Sets
Defect fixing & Resolving
Level – 1
Select a batch & starts execution
Developers
If Mismatch, then suspend the batch
Otherwise


Test Closure
Level – 3
Final Regression / Releasing Testing / Pre-Acceptance / Post-Mortem testing
User Acceptance Testing
Sign Off

4) TEST PLANNING:

After finalization of possible test for current project, Test Lead category people concentration on test plan document preparation to define work allocation in terms of What, Who, When & How to test. To prepare test plan document, test plan order follows below approach;



1] Team Formation:

In general, Test planning process starts with testing team formation. To define a testing team, test plan author depends on below factors;

1. Availability of testers
2. Test duration
3. Availability of test environment resource

2] Identify Tactical Risk:

After Testing team formation Plan author analysis possible & mitigation (ad hoc testing)

Risk 1:   Lack of knowledge of Test Engineer on that domain
Sol^n 1:   Extra training to Test Engineers

Risk 2:   Lack of Resource

Risk 3:   Lack of budget {less no of time}
Sol^n 3:   Increase Team size

Risk 4:   Lack of Test data
Sol^n 4:   Conduct test on past experience basis i.e., ad hoc testing or contact client for data

Risk 5:   Lack of developer process rigor
Sol^n 5:   Report to Test Lead for further communication between test & development PM

Risk 6:   Delay of modified build delivery
Sol^n 6:   Extra hours of work is needed

Risk 7:   Lack of communication in between Test Engineer - > Test team and
Test team - > Development team

3]   PREPARE TEST PLAN:

After completion of testing team formation & Risk analysis, Test plan author concentrate on **Test Plan Document** in IEEE format.

01)   Test Plan ID: Unique No or Name e.g. STP-ATM
02)   Introduction: About Project description
03)   Test Items: Modules / Functions / Services / Features / etc.
04)   Features to be tested: Responsible Modules for Test design (preparing test cases for
added modules)
05)   Features not to be tested: Which feature is not to be tested and Why? (Due to test
cases available for the old modules, so for these modules no
need to be tested / no test case

Above (3), (4) & (5) decides which module to be tested – > What to test?

06)   Approach: List of selected testing techniques to be applied on above specified

modules in reference to the TRM(Test Responsible Matrix).
07) Feature pass or fail criteria: When a feature is pass or fail description
(Environment is good) (After testing conclusion)
08) Suspension criteria: Possible abnormal situations rose during above features testing
(Environment is not good) (During testing conclusion)
09) Test Environment: Required software & Hardware to be tested on above features
10) Test Deliverables: Required testing document to be prepared (during testing, the type
of documents are prepared by tester)
11) Testing Task: Necessary tasks to do before start every feature testing

Above (6) to (11) specifies -> How to test?

12) Staff & Training: Names of selected Test Engineers & training requirements to them
13) Responsibilities: Work allocation to every member in the team (dependable modules
are given to single Test Engineer)
14) Schedule: Dates & Times of testing modules

Above (4) specifies -> When to test?

15) List & Mitigation: Possible testing level risks & solution to overcome them
16) Approvals: Signatures of Test plan authors & Project Manager / Quality Analyst

4) Review Test Plan:

After completion of plan document preparation, Test plan author conducts a review of completion & correctness. In this review, Plan author follows below coverage analysis

- BRS based coverage (What to test? Review)
- Risks based coverage (When & Who to test? Review)
- TRM based coverage (How to test? Review)

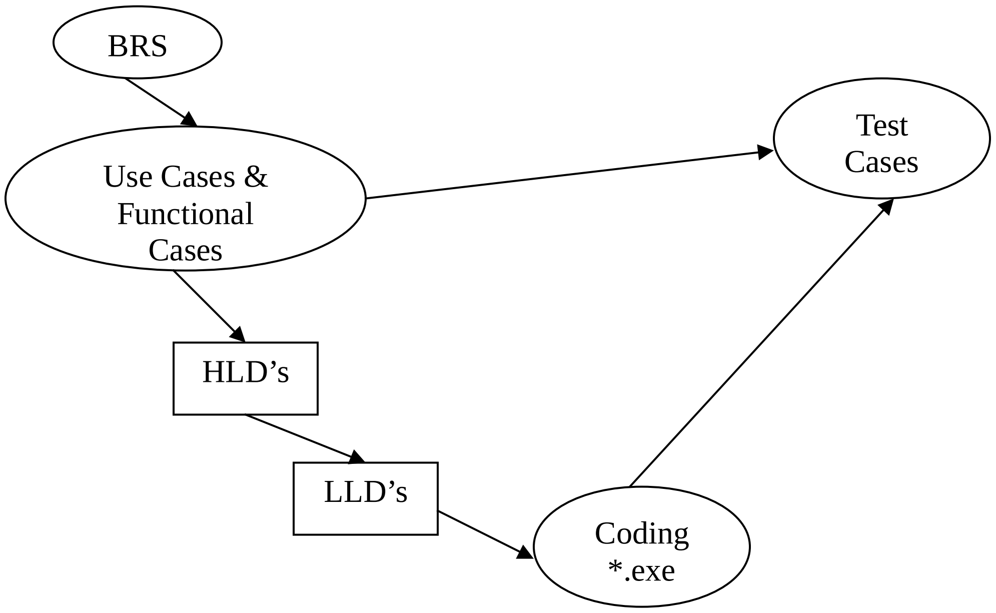
5) TEST DESIGNING:

After completion of Test Planning & required training to testing team, corresponding testing team members will start preparing the list of test cases for their responsible modules. There are three types of test cases design methods to cover core level testing (Usability & Functionality testing).

a) Business Logic based test case design (S/w RS)
b) Input Domain based test case design (E-R diagrams / Data Models)
c) User Interface based test case design (MS-Windows rules)

a) Business Logic based Test Case design (SRS)

In general, Test Engineers are preparing a set of Test Cases depends on Use Cases in (S/w RS). Every Use Case describes a functionality in terms of inputs, process & output, depends on this Use Cases Test Engineers are preparing Test Cases to validate the functionality



From the above model, Test Engineers are preparing Test Cases depends on corresponding Use Cases & every test case defines a test condition to be applied.

To prepare test cases, Test Engineers studies Use Cases in below approach:

Steps:

1) Collect Use Cases of our responsible module
2) Select a Use Case & their dependencies from the list
2.1) Identify entry condition (Base state)
2.2) Identify input required (Test data)
2.3) Identify exit condition (End state)
2.4) Identify output & outcome (Expected)
2.5) Identify normal flow (Navigation)
2.6) Identify alternative flows & exceptions

3) Write Test Cases depends on above information
4) Review Test Cases for the completeness & correctness
5) Goto step (2) until completion of all Use Cases completion

<u>Use Case I:</u>

A login process allows user id & password to validate users. During these validations, login process allows user id in alpha-numeric from 4 to 16 characters long & password in alphabets in lowercase from 4 to 8 characters long.

<u>Case study:</u>

Test Case 1) Successful entry of user id

BVA (Size)

min	-> 4 chars	=>	pass
min-1	-> 3 chars	=>	fail
min+1	-> 5 chars	=>	pass
max-1	-> 15 chars	=>	pass
max+1	-> 17 chars	=>	fail
max	-> 16 chars	=>	pass

ECP

Valid	Invalid
a-z A-Z 0-9	special chars blank

<u>Test Case Format:</u>

During Test design Test Engineers are writing list of Test Cases in IEEE format.

01) Test Case ID: Unique no or name
02) Test Case Name: Name of test condition to be tested
03) Feature to be tested: Module / Function / Feature
04) Test suit ID: Batch ID, in which this case is member
05) Priority: Importance of Test Case {Low, Med, High}

P_0 -> Basic functionality
P_1 -> General functionality (I/P domain, Error handling, Compatibility etc,)
P_2 -> Cosmetic testing (UIT)

06) Test Environment: Required Software & Hardware to execute
07) Test afford (person / hr): Time to execute this Test Case e.g. 20 minutes
08) Test duration: Date & Time
09) Test Setup: Required testing task to do before starts case execution (pre-requisites)
10) Test Procedure: Step by step procedure to execute Test Case

Test Procedure Format:

1) Step No:
2) Action:
3) Input required:
4) Expected:

-> Test Design

5) Actual:
6) Result:
7) Comments:

-> Test Execution

11) Test Case passes or fails criteria: When this case is pass or fail

Note: Test Engineers follows list of Test Cases along with step by step procedures only

Example 1:

Prepare Test Procedure for below test cases “Successful file save operation in Notepad “.

Step No	**Action**	**Input Required**	**Expected**

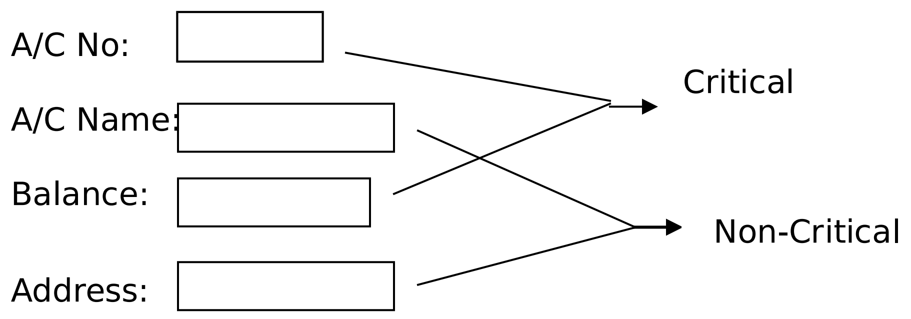
1	Open Notepad		Empty Editor
2	Fill with text		Save Icon enabled
3	Click Save Icon or click File menu Option & select save option		Save Dialog box appears with Default file name
4	Enter File name & Click Save	Unique File name	Focus to Notepad & File name Appears in title bar of Notepad

__Note:__ For more examples refer to notes

b) Input Domain based Test Case design (E-R diagrams / Data Models)

In general, Test Engineers are preparing maximum Test Cases depends on Use Cases or functional requirements in S/wRS. These functional specifications provide functional descriptions with input, output & process, but they are not responsible to provide information about size & type of input objects. To correct this type of information, Test Engineers study Data model of responsible modules E-R diagram. During Data model study, Test Engineer follows below approach: Steps:

1) Collect Data model of responsible modules
2) Study every input attribute in terms of size, type & constraint
3) Identify critical attributes in the test, which is participated in manipulation & retrivals
4) Identify non-critical attributes such as input & output type



5) Prepare BVA & ECP for every input object

	ECP		BVA(Size / Range)	
Input Attribute	**Valid**	**Invalid**	**Minimum**	**Maximum**
Xxxx	xxxx	xxxx	xxxx	xxxx

"	"	"	"	"
"	"	"	"	"
"	"	"	"	"

DATA MATRIX

Note: In general, Test Engineers are preparing step by step procedure based Test Cases for functionality testing. Test Engineers prepare valid / invalid table based Test Cases for input domain of object testing {Data Matrix }

Note: For examples refer to notes

c) User Interface based test case design (MS-Windows rules)

To conduct Usability Testing, Test Engineers are preparing list of Test Cases depends on our organization User Interface standards or conventions, Global User Interface rules & interest of customer site people.

Example: Test Cases

1) Spelling check
2) Graphics check (Screen level Align, Font style, Color, Size & Microsoft six rules)
3) Meaning of error messages
4) Accuracy of data displayed
5) Accuracy of data in the database are result of user inputs, if developer restrict the data in
database level by rounding / truncating then the developer must also restrict the data in
front-end as well
6) Accuracy of data in the database as the result of external factors. Ex. File attachments
7) Meaningful help messages (Manual support testing)

Review Test Cases:

After completion of all possible Test Cases preparation for responsible modules, Testing team concentrates on review of Test Cases for completeness & correctness. In this review, Testing team applies coverage analysis.

Test Case Review

1) BR based coverage
2) Use Cases based coverage
3) Data model based coverage

4) User Interface based coverage
5) TRM based coverage

At the end of this review, Test Lead prepare Requirement Traceability Matrix or Requirement Validation Matrix (RTM / RVM)

Business Requirement	Source (Use Cases, Data model)	Test Cases
xxxx	xxxx	xxxx xxxx xxxx
	xxxx	xxxx xxxx
	xxxx	xxxx xxxx xxxx
xxxx	xxxx	xxxx xxxx
	xxxx	xxxx xxxx xxxx

From RTM / RVM model, it defines mapping between customer requirement & prepared Test Cases to validate the requirement.

6) TEST EXECUTION:

After completion of Test Cases selections & their review, Testing team concentrate on Build release from development side & Test execution on that build.

a) Test Execution Levels or Phases:

See figure in next page for clear understandability of this levels

b) Test Execution Levels v/s Test Cases:

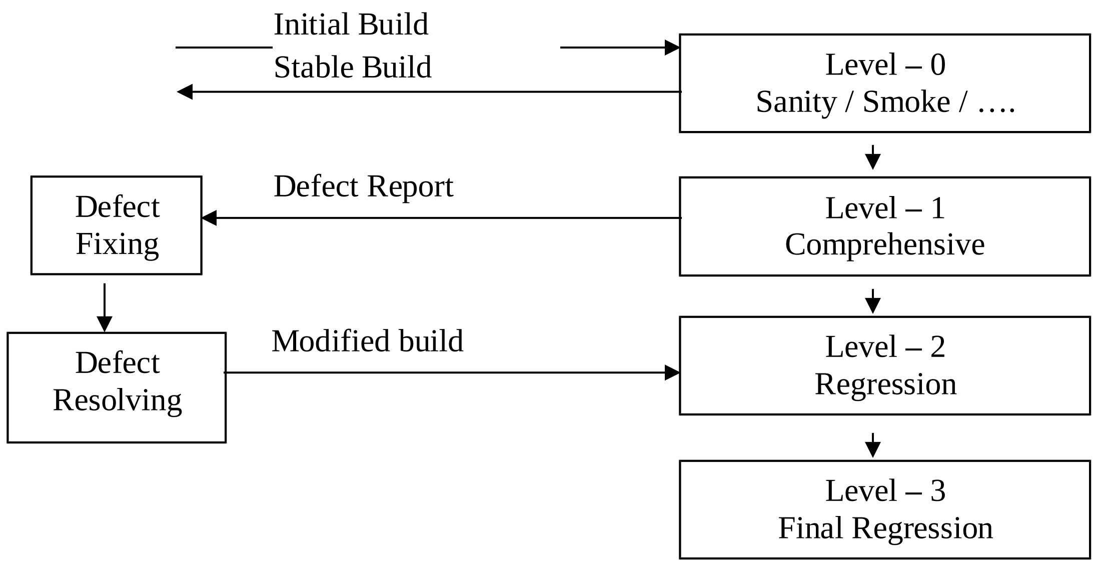
Level – 0 -> P_0 Test Cases

Level – 1 -> All P_0, P_1 & P_2 Test Cases as batches

Level – 2 -> Selected P_0, P_1, & P_2 w.r.t modification

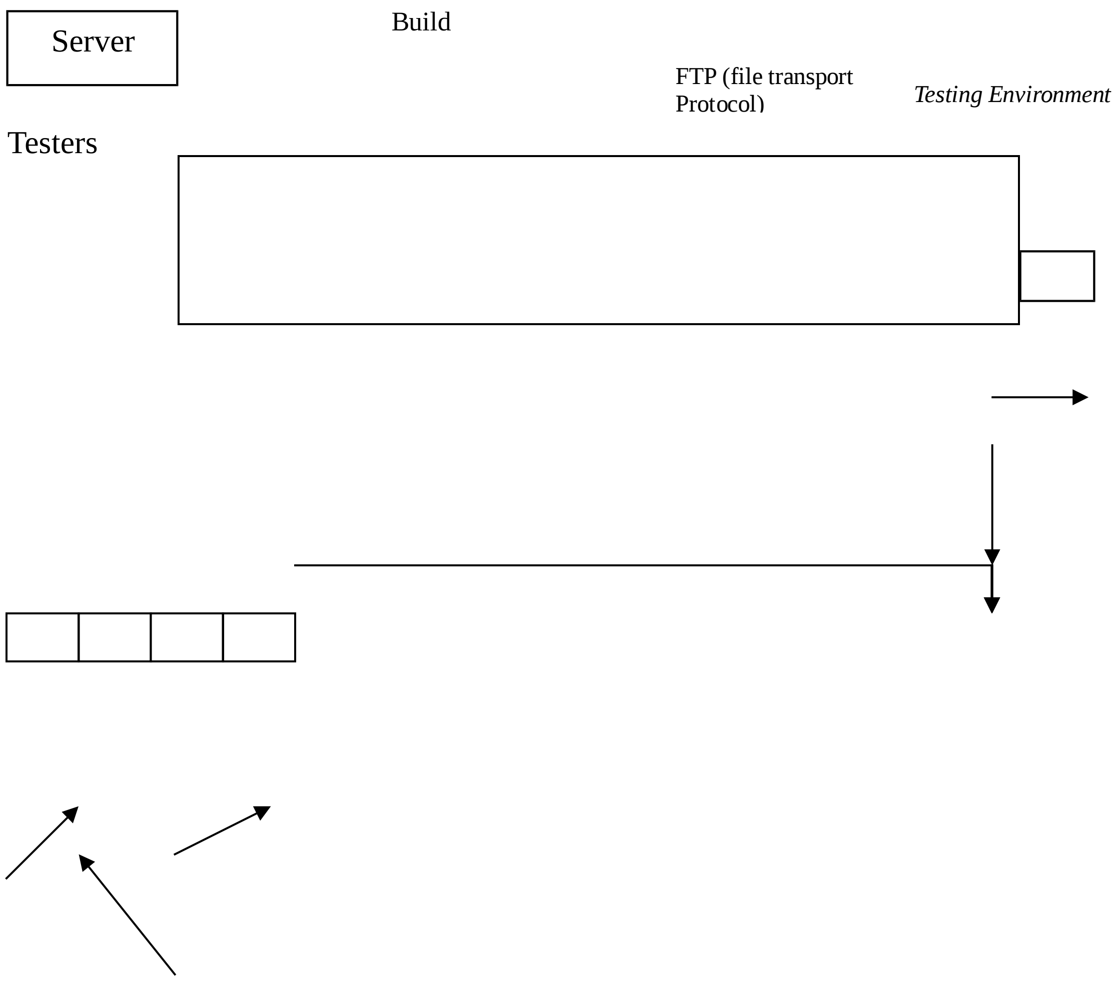
Level – 3 -> Selected P_0, P_1, & P_2 w.r.t critical areas in the master build

A) Test Execution Levels or Phases:



c) Build Version Control:

In general, Testing Engineers are receiving build from development in below model



During Test execution Test Engineer are receiving modified build from software. To distinguish old & new builds, development team maintains unique version in system, which is understandable to Tester / Testing team. For this version controlling developers are using version control tools (Visual Source Safe)

d) Level – 0 (Sanity / Test Acceptance / Build verification test):

After receiving initial build, Testing Engineers concentrate on basic functionality of that build to estimate stability for complete testing. In this Sanity testing, Testing Engineers tries to execute all P_0 Test Cases to cover basis functionality. If functionality is not working / functionality missing, Testing team rejects that build. If Tester decided stability then they concentrate on Test execution of all Test Cases to detect defects.

During this Sanity testing, Testing Engineer observes below factors on that build

1) Understandable
2) Operatable
3) Observable
4) Consistency
5) Controllable
6) Simplicity
7) Maintainable
8) Automation

From the above “8” testable issues, Sanity test is also known as Testability testing / OCT angle testing.

e) Test Automation:

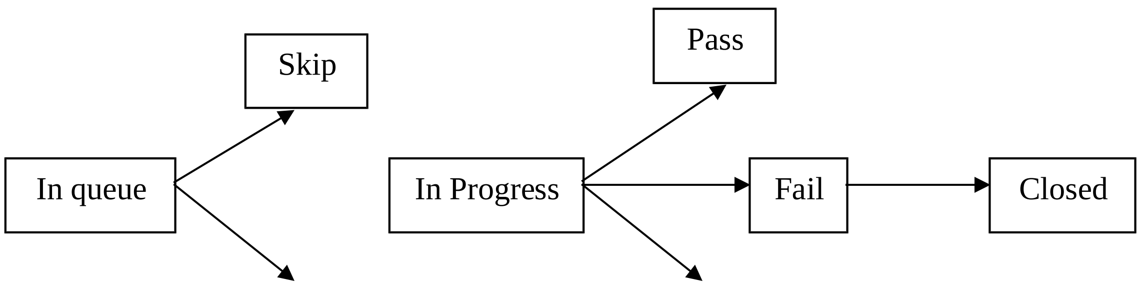
If test automation is possible than Testing team concentrate on Test script creation using corresponding testing tools. Every Test script consists of navigation statement along with required check points.

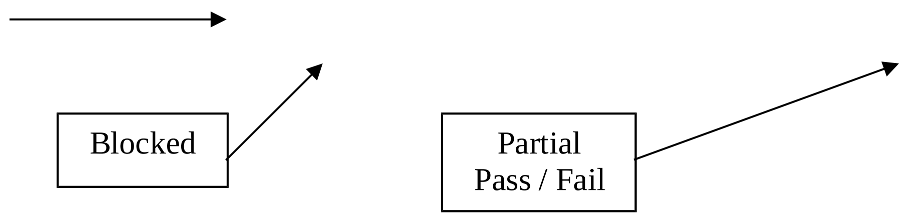
$$\begin{tikzcd}
\text{Stable Build} \arrow[d] \\
\text{Test Automation (Select Automation)} \arrow[d] \\
\text{(All } P_0 \text{ \& Carefully selected } P_1 \text{ Test Cases)}
\end{tikzcd}$$

f) Level – 1 (Comprehensive testing)

After completion of Sanity testing & possible test automation, Testing team concentrates on test batches formation with depended Test Cases. Test batches are also known as Test suits / sets. During these Test batches execution, Test Engineers prepare test log document, this document consist of three types of entries.

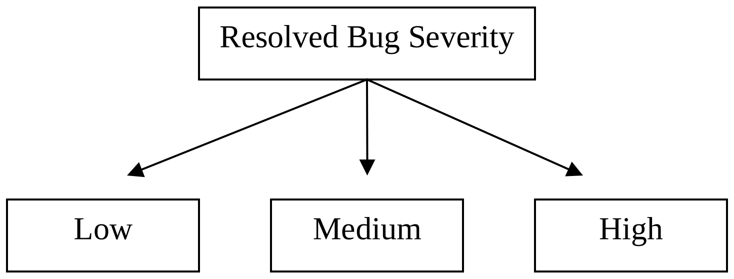
1) Passed (Expected = Actual)
2) Failed (Any one Expected != Actual, Any one Expected variants from Actual)
3) Blocked (Corresponding parent functionality failed)





g) Level – 2 (Regression testing)

During Comprehensive test execution, Test Engineers are reporting mismatches as defects to developers. After receiving modified build from developers, Test Engineers concentrate on Regression testing to ensure bug-fixing work & occurrences of side effects.



<u>Case I:</u>

If development team resolve bugs severity which is high, Test Engineers re-execute all P_0, P_1 & carefully selected P_2 Test Cases on modified build

<u>Case II:</u>

Bugs severity is medium, then all P_0, carefully selected P_1 & some of P_2 Test Cases

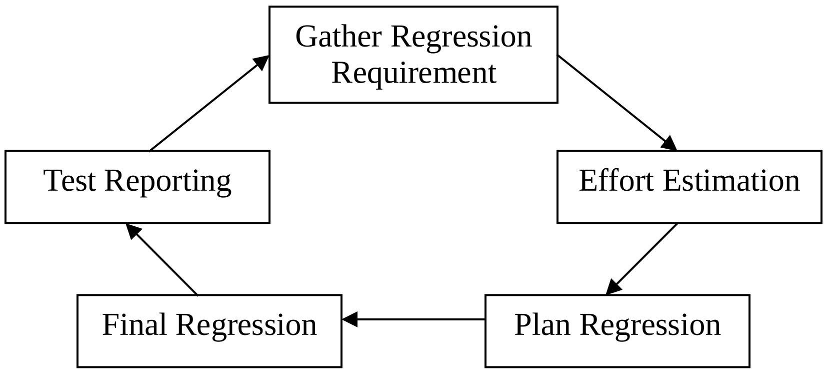
<u>Case III:</u>

Bugs severity is low, then some P_0, P_1 & P_2

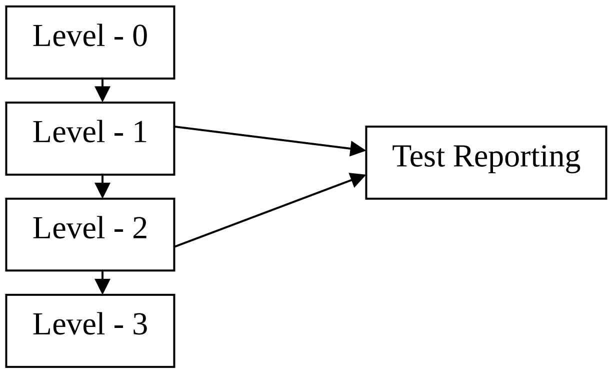
<u>Case IV:</u>

If development team released modified build due to sudden changes in project requirement then Test Engineers re-execute all P_0, P_1 & carefully selected P_2 Test Cases w.r.t that requirement modification.

h) Level – 3 (Final Regression / Pre-Acceptance testing)



7) TEST REPORTING:



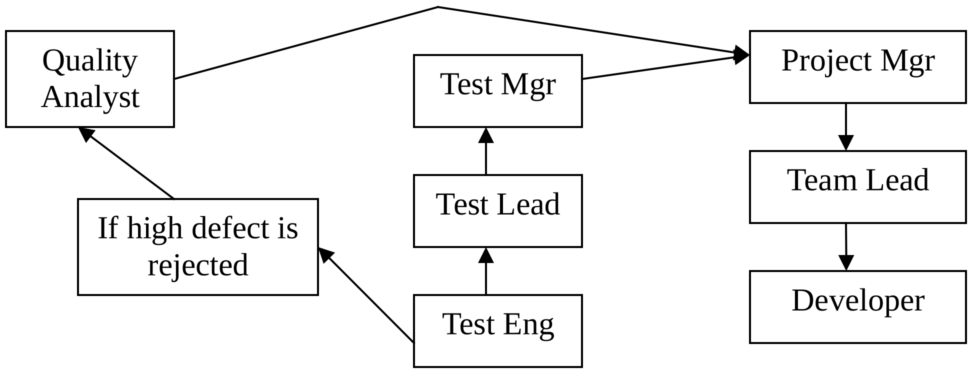
During comprehensive testing, Test Engineer are reporting mismatches as defects to developers through IEEE format.

1) Defect ID: Unique No or Name
2) Description: Summary of the defect
3) Feature: Module / Function / Service , in these module TE found the defect
4) Test Case Name: Corresponding failing test condition
5) Reproducible (Yes / No): Yes -> Every time defect appears during test execution
   No -> Rarely defect appears
6) If Yes, attach test procedure:
7) If No, attach snapshot & strong reasons:
8) Status: New / Reopen
9) Severity: Seriousness of defect w.r.t functionality (high / medium / low)

10) Priority: Importance of the defect w.r.t customers (high / medium / low)
11) Reported bug: Name of Test Engineer
12) Reported on: Date of submission
13) Assign to: Name of the responsible person in development team -> PM
14) Build Version ID: In which build, Test Engineer fount the defect
15) Suggested fix (Optional): Tester tries to produce suggestion to solve this defect
16) Fixed by: PM or Team Lead
17) Resolved by: Developer name
18) Resolved on: Date of solving → **By Developers**
19) Resolution type: check out in next page
20) Approved by: Signature of Project Manager (PM)

Defect Age: The time gap between "reported on" & "resolved on"

Defect submission process:



Defect Status Cycle:

New → Open / rejected / deferred → close reopen

Deferred => Accepted but not interested to resolve in this version

Defect Resolution:

After receiving defect report from Testers, developers review the defect & they send resolution type to Tester as a reply

1) Duplicate, rejected due to the defect is same as previously reported defect
2) Enhancement, rejected due to the defect is related future requirement of customer
3) H/w limitation, rejected due to the defect raised w.r.t limitation of H/w devices
4) S/w limitations, rejected due to the defect raised w.r.t limitation of S/w Techno
5) Not applicable, rejected due to the defect has no proper meaning
6) Functions as designed, rejected due to coding is correct w.r.t to designed doc's
7) Need more information, not (accepted / rejected) but developers requires extra information to understand the defect
8) Not reproducible, not (accepted / rejected) but developers require correct procedure to reproduce the defect
9) No plan to fix it, not (accepted / rejected) but developers want extra time to fix
10) Fixed, developers accepted to resolve
11) Fixed indirectly, accepted but not interested to resolve in this version (default)
12) User misunderstanding, need extra negotiation between testing & development team.

Types of defects:

01) User Interface bugs (low severity):
    1) Spelling mistakes (high priority)
    2) Improper alignment (low priority)

02) Boundary related bugs (medium severity)
    1) Doesn't allows valid type (high priority)
    2) Allows invalid type also (low priority)

03) Error handling bugs (medium severity)
    1) Doesn't providing error message window (high priority)
    2) Improper meaning of error message (low priority)

04) Calculations bugs (high severity)
    1) Final output is wrong (low priority)
    2) Dependent results are wrong (high priority)

05) Race condition bugs (high severity)
    1) Dead lock (high priority)
    2) Improper order of services (low priority)

06) Load conditions bugs (high severity)
    1) Doesn't allow multiple users to access / operate (high priority)
    2) Doesn't allow customers accepted load (low priority)

07) Hardware bugs (high severity)
   1) Doesn't handle device (high priority)
   2) Wrong output from device (low priority)

08) ID control bugs (medium severity)
   1) Logo missing, wrong logo, Version No mistake, Copyright window missing,
   Developers Name missing, Tester Name missing

09) Version control bugs (medium severity)
   1) Differences between two consecutive build versions

10) Source bugs (medium severity)
   1) Mistake in help documents – Manual support

8) TEST CLOSURE:

After completion of all possible test cycle execution, Test Lead conducts a review to estimate the completeness & correctness of testing. In this review, Test Lead follows below factors with Test Engineers

1) Coverage Analysis

   a) BR based coverage
   b) Use Cases based coverage
   c) Data model based coverage
   d) User Interface based coverage
   e) TRM based coverage

2) Bug density

   a) Module A has 20% percentile bugs found
   b) Module B has 20% percentile bugs found
   c) Module C has 40% percentile bugs found
   d) Module D has 20% percentile bugs found

3) Analysis of deferred bugs

Whether deferred bugs are deferrable or not

At the end of this review, Testing team concentrate on high bug density modules or all modules if time is available.

9) User Acceptance Testing (UAT)

Organization management concentrate on UAT to collect feedback, there are two approaches to conduct testing.

1. Alpha (α) test
2. Beta (β) test

10) Sign Off

After completion of User Acceptance testing & their modifications, Test Lead concentrates on final test summary report creation. It is a part of Software Release Node (S/w RN). This final test summary report consist of below documents

1) Test Strategy / Methodology (TRM)
2) System Test Plan
3) Requirement Traceability Matrix (RTM)
4) Automated Test Scripts
5) Bugs Summary Reports

The below, has to be in horizontal direction

- Bug Description
- Feature
- Found By
- Status (closed / deferred)
- Comments