TESTING

BY

NAGESWARA RAO

Software:- Set of executable programs in our computer

S/W application Vs S/W product:-

If a S/W developed for a specific customer w.r.t their requirements then the S/W is called as application or project.

If a S/W developed w.r.t requirements in market then the S/W is called as product.

S/W quality:-

1. Meet customer requirements.
   [Ex: Features / functionalities]
2. Meet customer expectations.
   [Ex: usability / Compatability / Performance / Security...]

Quality assurance (QA) Vs Quality Control (QC)

To release a quality S/W to customer, an organization is monitoring & measuring the strength of S/W development Process. This checkup is called as quality assurance.

While S/W development, an organization is validating each deliverable for correctness & completeness. This validation is called as testing or QA.

FISH Model:-

PIN:- (Product or project initiation Note)

[This document consists of proposal and overall plan to develop a new S/W.]

BRS :- [Business requirement specification]

This document consists of complete & correct requirements. to develop a new S/W.

SRS :- [S/W requirement specification]

This document specifies the how to develop a new S/W.

HLDS:- [High-level Design or Architectural design]

This document provides overall architecture of S/W. in diagramatic notation.

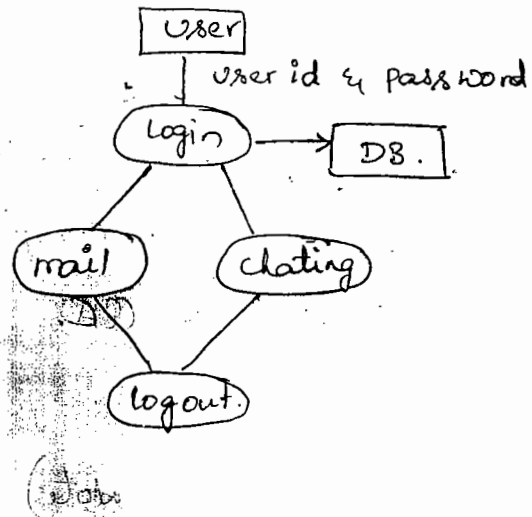
LLDS:- [Low-level designs]

These documents specifies internal logic of every module or functionality. in diagramatic notation.

→ HLD is software level & LLD are module level.

Review:- It is a testing technique to verify correctness & completeness of documents. like BRS, SRS, HLD & LLDs

LLD's



Program:- A set of executable statements.

Unit testing:- A program testing is called as unit testing. In this testing testers are following white-box testing testing techniques.

Programs Integration:- To form a s/w programmers are interconnecting programs.

Integration testing:-

To validate correctness of programs interconnections developers are conducting integration testing.

s/w build:- A set of interconnected programs is called as s/w build.

System or s/w testing:- The correctness and completeness of a s/w w.r.t customer requirements & expectations is called as s/w or system testing.

Case study:-

SDLC	STLC
1. BRS ↓ 2. SRS ↓ 3. HLD & LLD's	1. Review [Document testing or static testing]
Coding	Unit testing Integration testing
s/w build	s/w or system testing
Acceptance	Acceptance testing [α-test & β-testing]

Release

Maintance | support

Release testing or port testing or onsite testing

S/w changes testing [Re & Regression testing]

SDLC :-

SDLC specifies multiple stages of development. like shown below.

1. Proposal / Bidding [BIN] → overall plan
↓
Requirements gethering [BIS]
↓
Analysis & planning [SRS & project plan] → detailed plan
↓
Design [HLD & LLD's]
↓
Coding [Programs]
↓
System testing
↓
Acceptance
↓
Release
↓
Maintaince

STLC :-

Documents testing
↓
Unit testing
↓
Intigration testing
↓
System testing

↓

Acceptance testing

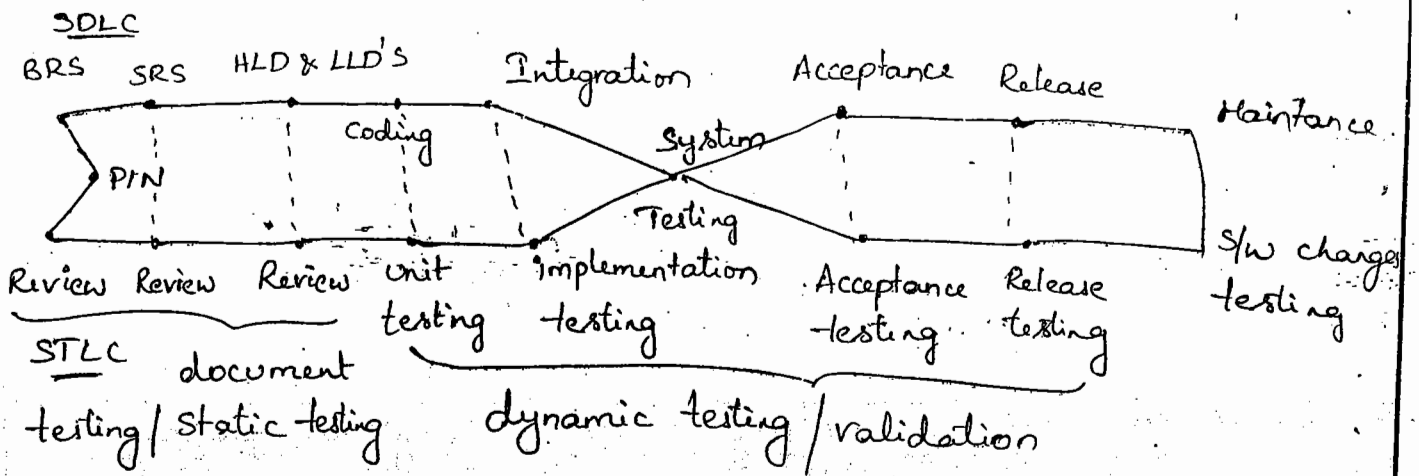
↓

Release testing

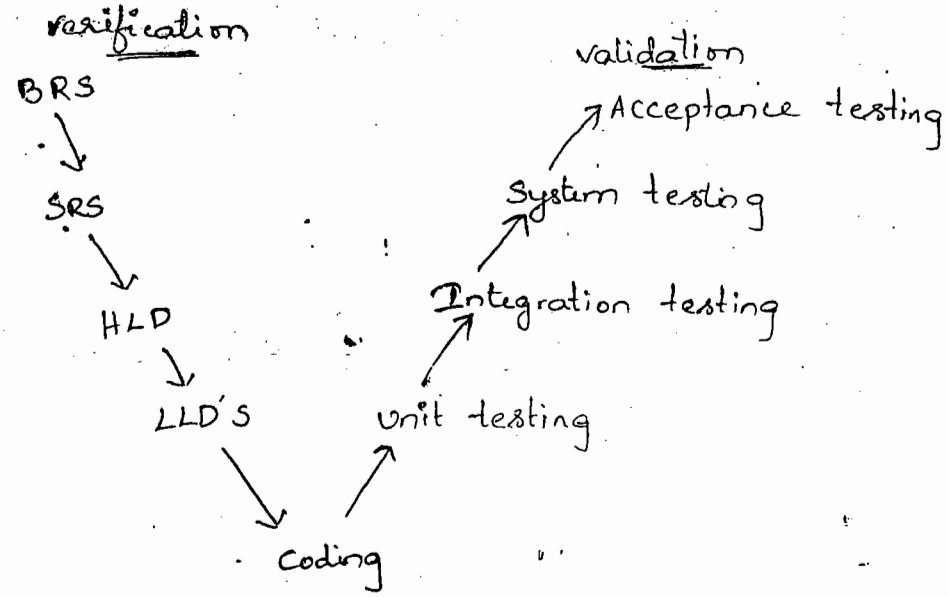
↓

S/w changes testing.

→ FISH model defines conceptual mapping inbetween SDLC & STLC.



V - model :- verification



→ V - stands for verification & validation.

→ Documents testing, is called as verification.

→ Unit, Integration, S/w, Acceptance, Release & Changes testing is called as dynamic testing or validation.

→ Documents testing, Unit testing & Integration testing are conducting by developers.

→ System testing is conducting by testing team.

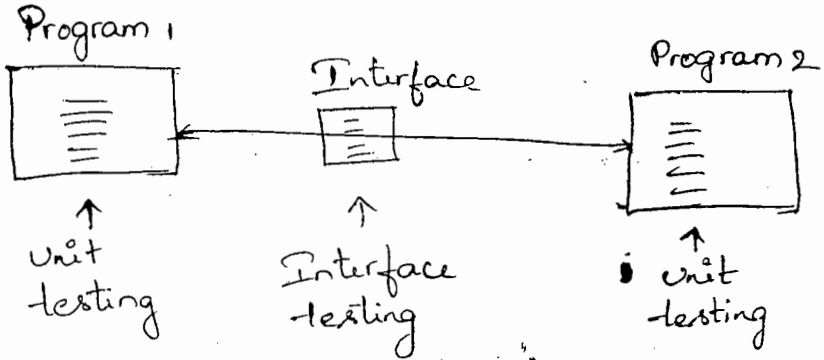
→ Acceptance testing is conducting by project management with involvement of developers, testers & customers site people.

→ Few developers & testers formed as release team. To release s/w in customer site.

→ To test s/w changes in maintanance project management can form CCB (change control board) with few developers & testers.

→ Developers are conducting unit testing on programs w.r.t LLD's.

→ Developers are conducting integration testing on Programs interconnections w.r.t HLD. due to this reason integration testing also known as interface testing.



→ System testing is conducting by testing team w.r.t SRS

I. Document Testing

After completion of project initiation, SDLC process starts with requirements gathering. In this stage business analyst is gathering requirements from real customer in application development & is gathering requirements from model customers in product development. In this stage business analyst delivers BRS or URS (user) or CRS (customer)

After completion of BRS preparation corresponding business analyst is reviewing the document for completness & correctness.

After completion of BRS review and changes, corresponding business analyst submits BRS to system analyst to prepare SRS

BRS by BA

Addition of Two numbers

What ?

⟹

SRS by SA

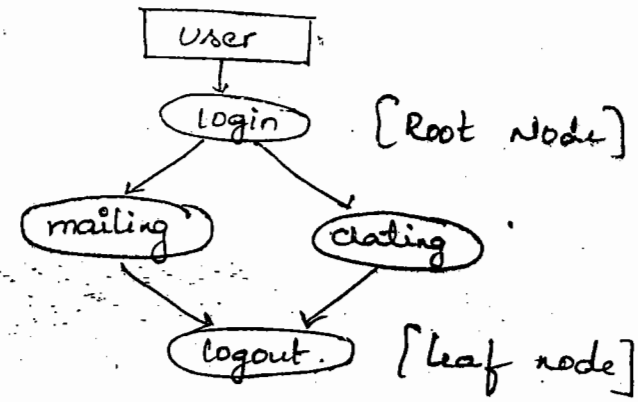
Functional requirements
2 i/p's + operator 1 o/p
Non-functional requirements
Screen is user-friendly Run on Windows & Linux

From the above diagram SRS is (How ?) classified into functional requirements & non-functional requirements.

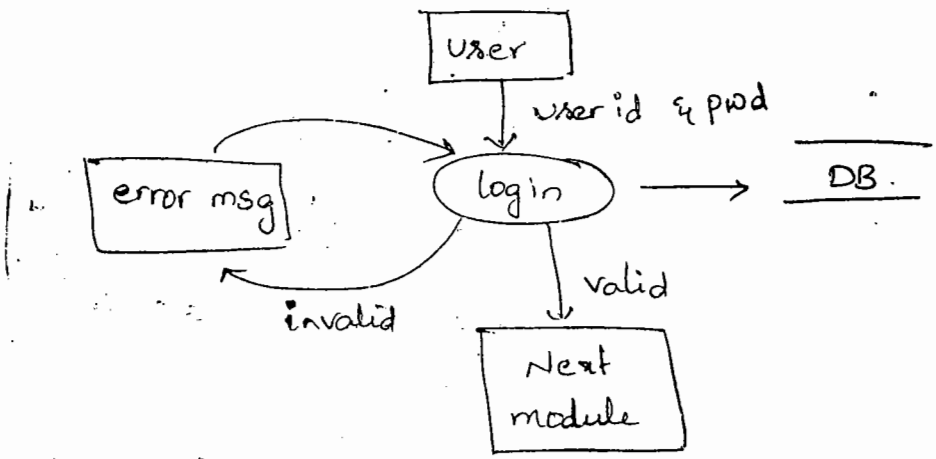
After Completion of SRS preparation corresponding system analyst is reviewing that SRS for completeness & correctness.

After Completion of SRS review & changes, System analyst submits SRS to technical architect or S/W designer to prepare high level design & low level designs. Here HLD specifies overall architecture of a S/W.

Ex: for HLD:

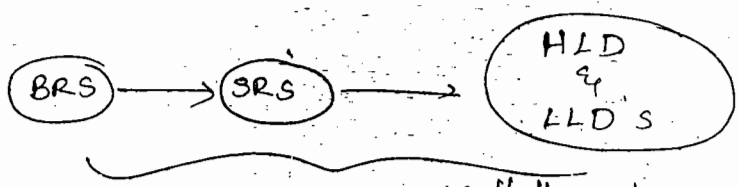


Here LLD is specifing internal logic of corresponding module or functionality.



So, one project or product consists of one HLD & multiple LLDS. After completion of design, the corresponding technical architect is reviewing design documents for Completeness & correctness.

In above reviews



Reviews → Walkthroughs
Inspection
Peer Reviews.

In above reviews BRS, SRS & HLD & LLD's + Corresponding people following below review techniques

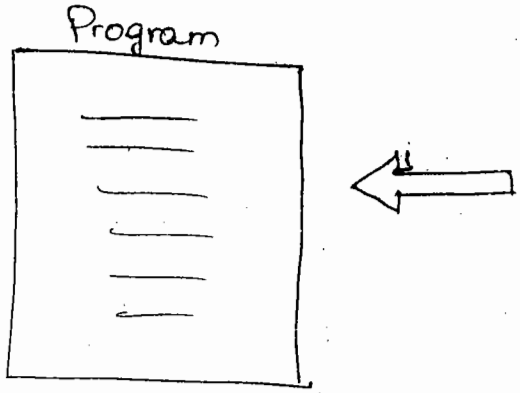
Walkthroughs:- Study a document from first to last for Completeness & correctness.

Inspections:- Searching a specific factor in document.

Peer Reviews:- The comparision of two similar documents Point-point for Completeness & Correctness.

II: Unit testing:-

After completion of documents preparation & their reviews, programmers are starting coding or implemental In this stage programmers are writing programs & testing them by using white-box testing techniques. These techniques are also known as open-box or clear box/glass box testing techniques.



→ Basic paths testing
→ Control structure testing.
→ Program technique testing.
→ Mutation testing

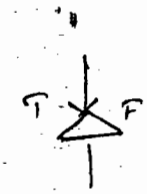
White-box testing

(a) Basic paths Coverage :- Whether a program is running or not

Ex :

```
if (a>b)
{
  ≡
}
else
{
  ≡
}
```

⟹



Flow diagram

→ No. of paths or Cyclomatic complexity = 2

↓

Run program 2 times to cover all paths.

(b) Control structure coverage :- Whether a program is correctly running or not.

various i/p's ⟶ [Program] ⟶ Returns o/p

↓

correct ?

(c) program technique Coverage :- Whether a program is running is fastly or not (performance).

To calculate performance of a program, programmers are using some testing tools.

Ex :- JUnit for java code & Nunit for .Net code

Ex : Different logic for swaping.

```
a = 10
b = 20
c = a
a = b
b = c
```

Print a, b

→ High performance but take extra memory

```
a = 10
b = 20
a = a + b
b = a - b
a = a - b
```

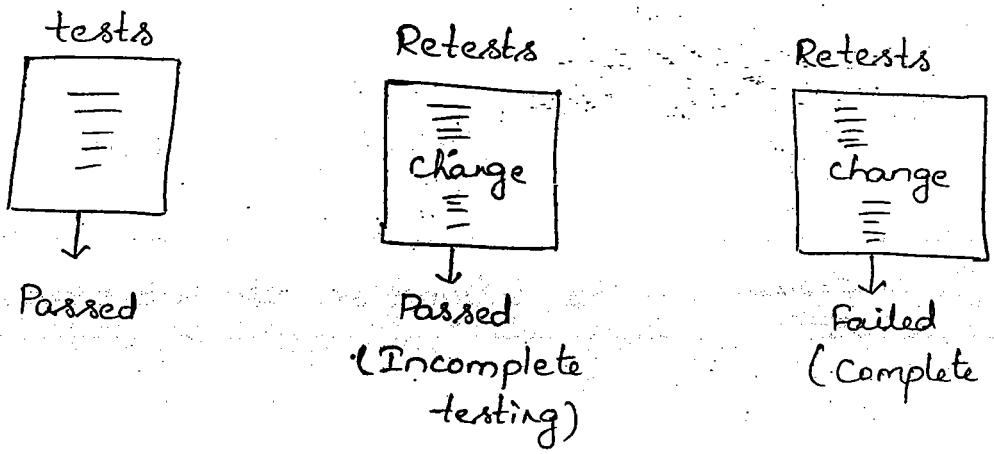
Print a, b.

→ low performance

In above ex. programmer can use first logic to achieve high performance.

(d) Mutation Coverage:-

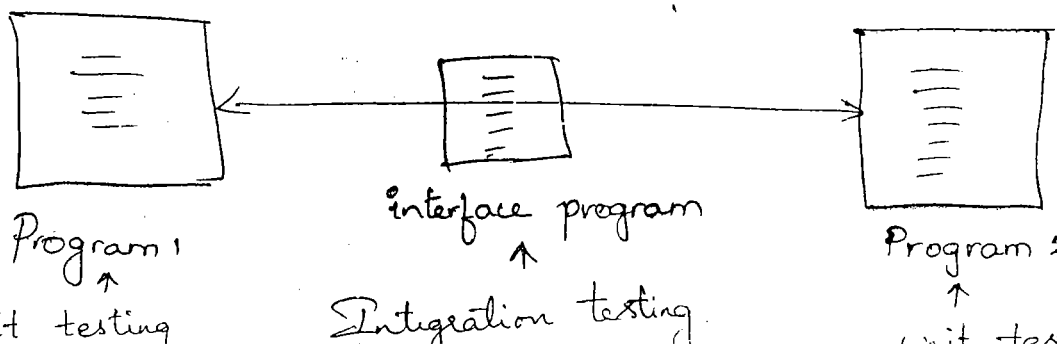
After completion of above three types of tests on a Program, programmers will change program and repeat Previously passed tests. If repeated test was failed then programmer can called testing has complete otherwise testing as incomplete.



Note:- First 3 white-box techniques can be applied on programs to test for completeness & correctness. The fourth techniques is used to estimate completeness & correctness of testing on program.

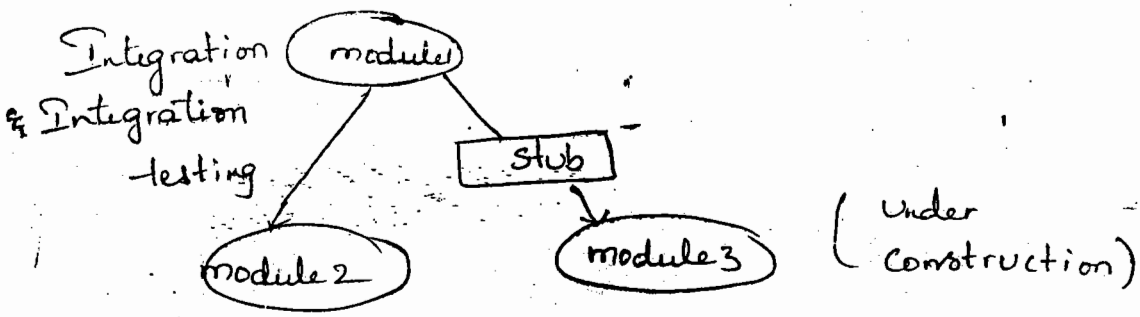
III: Integration testing:-

After completion of related programs unit testing, Programmers are concentrating on integration of those programs.

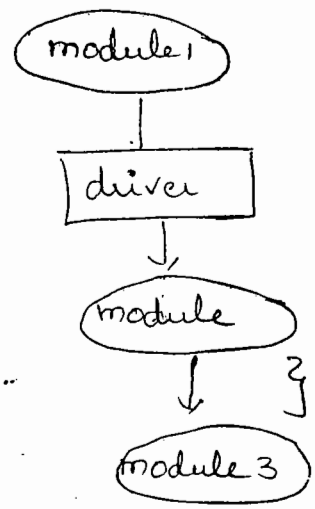


There are 4 approaches in programs integration to form a s/w build.

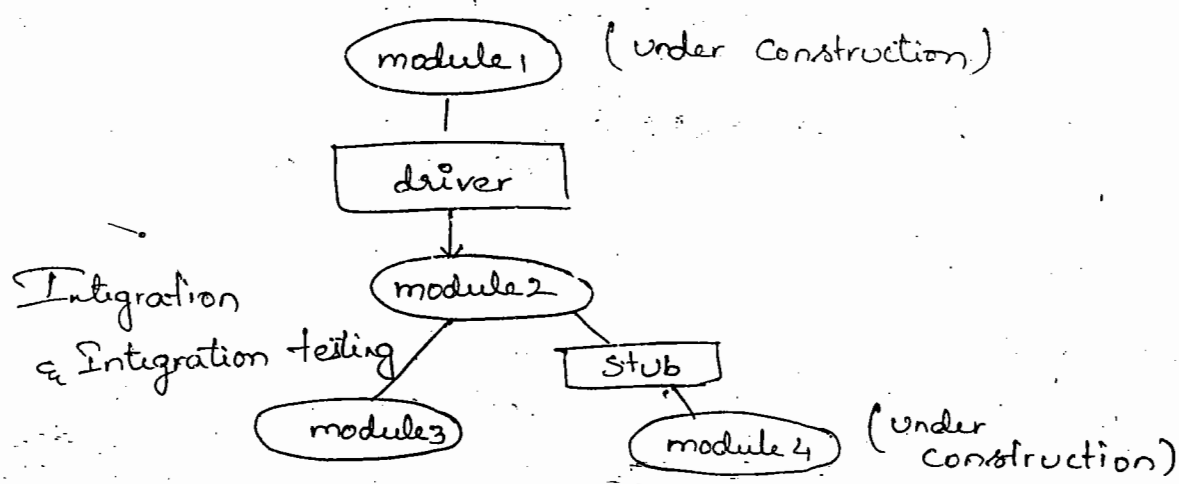
a) Top-down approach :- The integration of top module with bottom modules is called as top-down approach. Here programmers are using temporary programs instead of some under constructive sub modules. These temporary programs called as Stubs.



b) Bottom-up approach :- The integration of sub modules with-out involvement of under constructive main module is called as bottom-up approach. Here a temporary Program is replacing main module, called as driver.



(c) Hybrid Approach :- The combination of top-down & bottom-up approaches is called as hybrid approach or Sand-witch approach.



d) System approach :-

From this approach, programmers are starting integration of programs of completion of 100% of coding. Due to this reason driver & stub concept is not coming in this approach. This approach is also known as Big-Bang approach

IV System testing :- or S/W Testing :-

When S/W build is ready after integration & integration testing, testing team is starting S/W of system testing phase. This phase classified into

1. Functional testing 2. Non-Functional testing.

Functional testing is validating S/W w.r.t customer requirements. Non-functional testing is validating S/W w.r.t customer expectations like user friendly, performance, Compatability, security....

1) Functional testing:- The important part of S/w testing is functional testing. Due to this reason, a separate testing team is validating S/w build w.r.t to customer requirements. There are some sub tests in functional testing.

a) Behavioural / Control flow testing

In this stage, tester can check flow of operations in corresponding S/w under testing (SUT).

(b) Input domain testing:-

In this stage, testers can check size and type of inputs in corresponding SUT.

c) Error handling testing:-

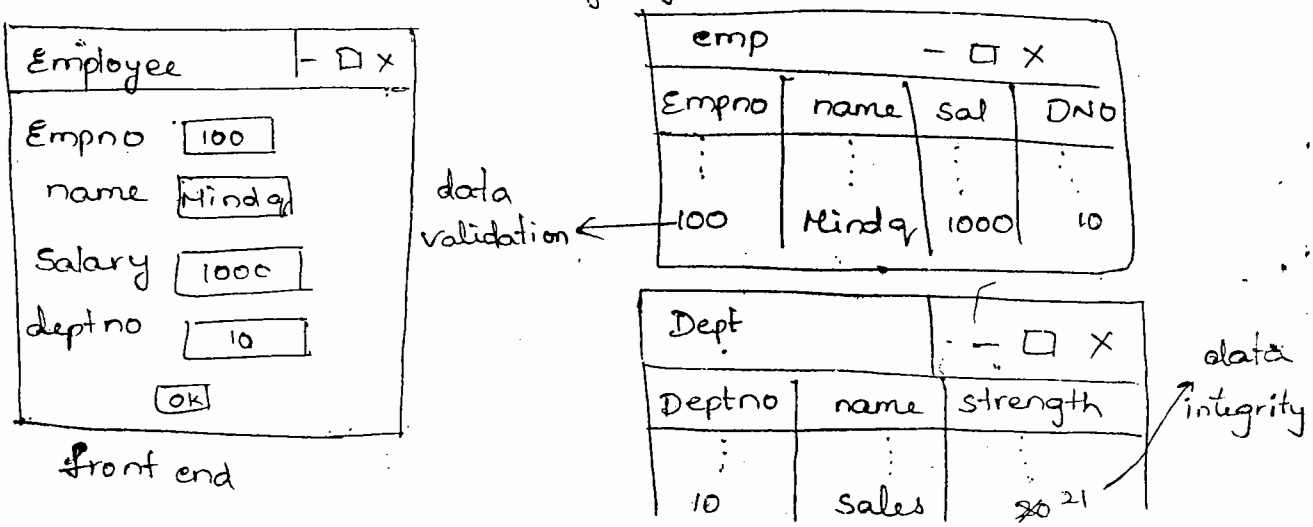
In this stage, testers can operate S/w in wrong manner to get error messages.

d) Manipulations Testing:-

In this stage, testers can validate correctness of each operation output in SUT.

e) Database Testing:-

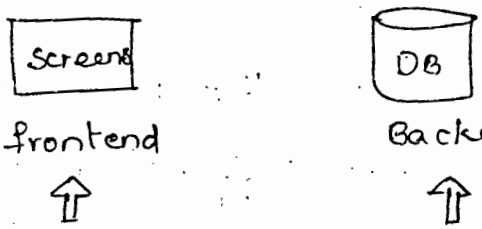
In this stage, testers can check the correctness of frontend operations impact on backend database in terms of data validation and data integrity..



→ frontend and backend names no need to be with same names. a, b, c, d testing are said to be front end testing.

case study:-

In above functional testing topics the first four topics related to front end testing and last topic related to backend testing.



- Behavioural testing
- Inputs domain testing
- Error handling testing
- Manipulations testing

- Database testing

2) Non-functional testing:-

After Completion of functional testing, the corresponding testing team is concentrating on non functional testing to validate S/W w.r.t to customer expectations.

a) Usability testing:-

In this stage, testers can check user friendlyness of SUT

- easy to use
  - Ex: understandable screens
- look & feel
  - Ex: Colour, font, style ....
- speed in navigations
  - Ex: short navigations.

} User friendlyness

b) Compatability testing :-

In this stage, testers can check that whether our SUT will run on various platforms or not?

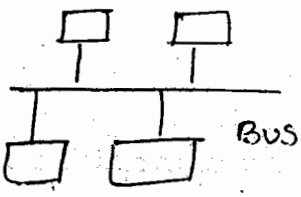
Here platform means that os, browser and other system softwares.

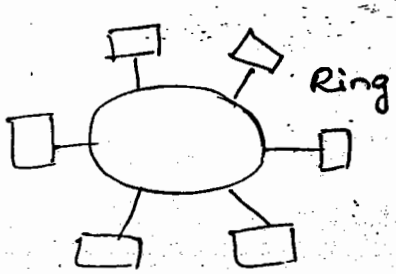
This Compatability testing is also known as portability testing.

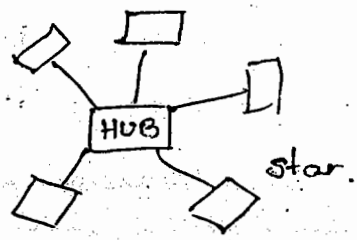
c) Configuration testing :-

In this stage, testers can check that whether our s/w is working with various types of h/w configurations or not? Ex: various types of printers, Networks (LAN's) and other h/w devices. This testing is also known as h/w Compatability testing.

LANs

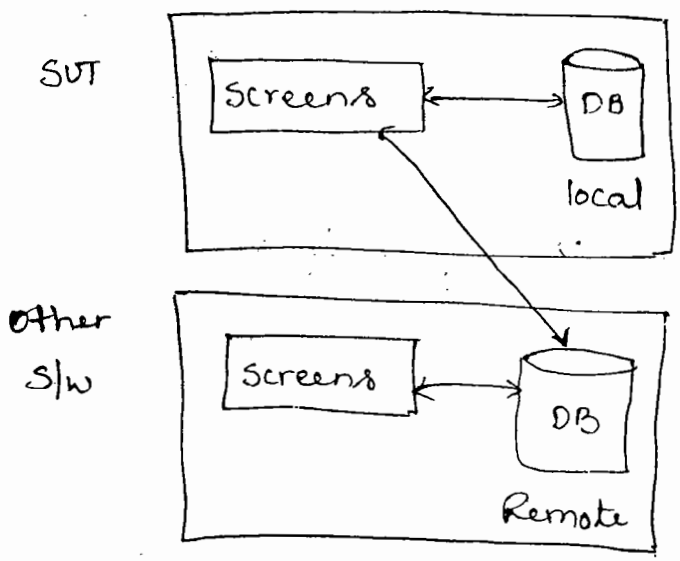






d) Intersystem testing: /SOA (Service oriented Application) Testing

In this stage, our s/w is a sharing resources of another s/w to Complete transactions. Here tester can check Correctness of each end-to-end transactions.



Here our SUT is working with our DB and also working with other DB is, said to be remote.

In above diagram SUT is connecting to remote DB of other s/w to share data. This intersystem testing is also known as interoperable testing, End-to-end testing (or) web services testing. Here SUT & other s/w are two websites.

e) Performance Testing :-

In this stage, testers can check "speed in processing" of SUT. Here testers can operate s/w in below ways.

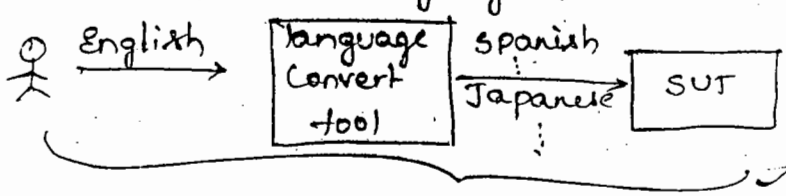
→ Run SUT under customer expected configuration and customer expected load (No. of users) to calculate "speed in processing", called as load testing.

→ Run SUT under customer expected configuration and more than customer expected configuration and "reliability", called as stress testing.

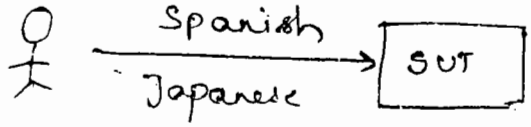
→ Run SUT under customer expected configuration and customer expected load continuesly to estimate longuity / durability called as endurance testing.

f) Multilanguity Testing :-

In this stage, testers can check SUT with various Processor supported languages (Unicode)



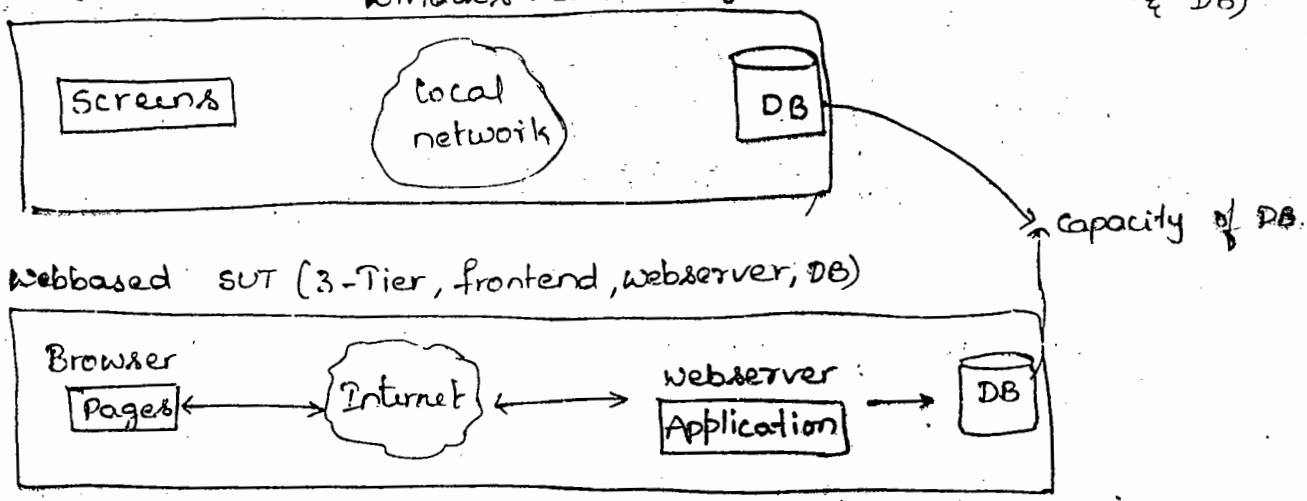
Globalization
Internationalization



Franch → localization

a) Data volume testing

In this stage, testers can check the capacity of database in SUT. This testing is also known as memory testing (or) capacity testing.



In this data volume testing, testers are inserting model data into database SUT until database violation error to calculate capacity of DB.

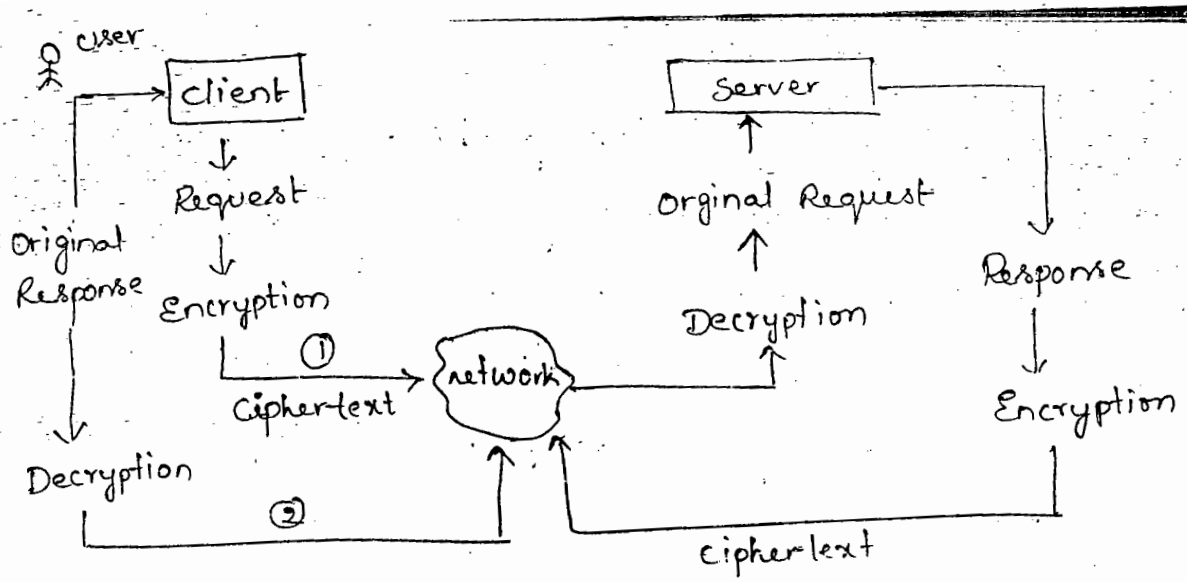
b) Security Testing

In windows-based or web-based multi user s/ws, security is a mandatory factor to present unauthorized access.

→ Authorization testing: Whether a user is valid or not? This is also known as Authentication testing.

→ Access Control testing:- Whether a valid user have permission or not to use specific functionality?

→ Encryption / Decryption testing: Whether encryption / decryption process is reliable or not.

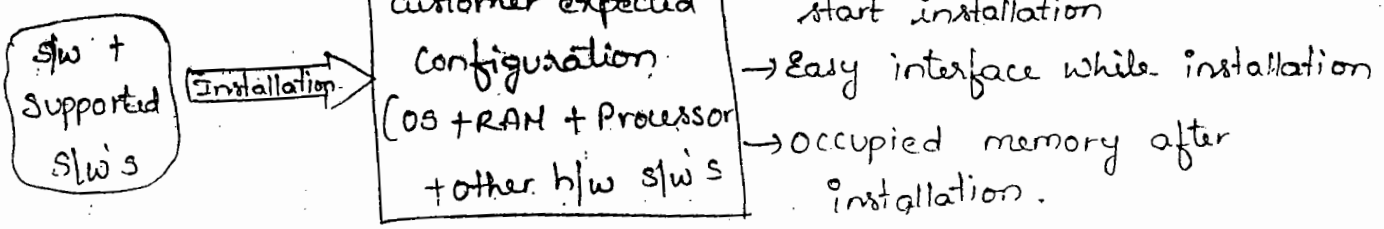


B.

Note:- In above security testing, general testing teams are covering authorization and access control testing topics, but to conduct encryption/decryption testing, organizations are maintaining separate developers and testers called as E-trust.

i) Installation Testing:-

In general windows-based s/w's are installing at client machine and web-based s/w's are installing at server machine. Due to this reason, testing team is checking installation process of any SUT.



j) Parallel Testing:-

In this stage, testing team is comparing our SUT with various versions (or) with similar s/w, to identify weekness and strengths. This testing also known as Computationtime testing or Comparision testing.

V Acceptance Testing

After Completion of system testing, the project management is Concentrating on acceptance testing to collect feedback from real customers and model customers. In this acceptance testing, developers and testers are also involving to convience customers. There are two ways in acceptance testing such as α-testing and β-testing.

α-Testing	β-testing
→ Suitable for applications	→ suitable for product
→ Conducting by Real Customers Site people in development site	→ Conducting by model customer site people in their site.
→ Direct involvement of developers & testers with real customer site people.	→ online involvement of developers and testers is possible with model customers.

The Combination of α-test & β-test is called as γ-test (Here two time feedback is needed)

VI Release Testing :-

After Completion of acceptance testing and their changes, Project management is concentrating on s/w release. Here project manager will form release team (or) onsite team, with few developers, few testers, few h/w engineers and one delivery manager has head. This team will come to customer site and start s/w installation in customer's site. Here, testers in release team are observing below issues in that customer site.

→ Complete installation
→ Overall functionality
→ Input devices handling

→ output devices handling (ex: Monitor, Printer .....)

→ Secondary storage devices handling (Ex: Pendrive, cd-drive, ...)

→ Co-existance with OS

→ Co-existance with other S/w's to share resources.

After Completion of above observations in customers site, release team will provide training sections to end-users (or) customer site people. and then release team back to organization.

VII Maintanence testing/support :-

After completion of S/w release and user training, project management is forming maintance team with few developers, few testers and project management people. This team is also known as change control Board (CCB). This team is phasing two types of support to customers. (when customer asking new requirement is said to be enhancement and money will be payed to our company from the customers. But in corrective maintance the money will not be paied to our company. Here tester has missed the defect and identified the mistake by the customer.

S/w change Request (SCR)

(Enhancement) | (Missed defect)

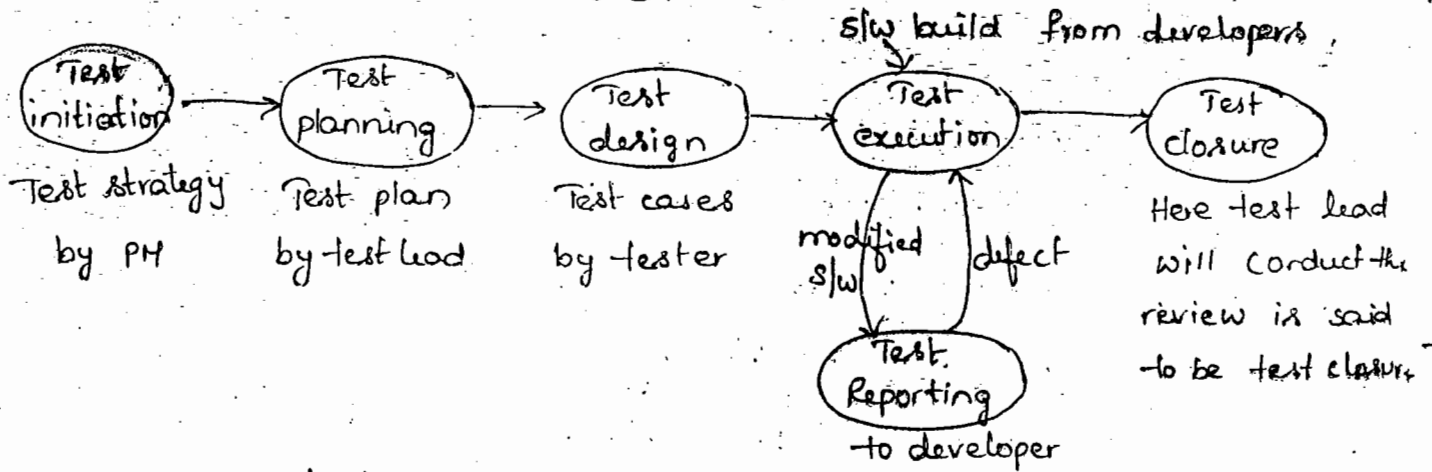
Impact analysis
↓
Perform changes in S/w
↓
Test S/w changes
↓
Release patch (to customer site)
(extra added code)

enhancement Maintanance

Impact analysis
↓
Perform changes in S/w
↓
Test S/w changes
↓
Release patch
↓
Improve testing process

Corrective maintance

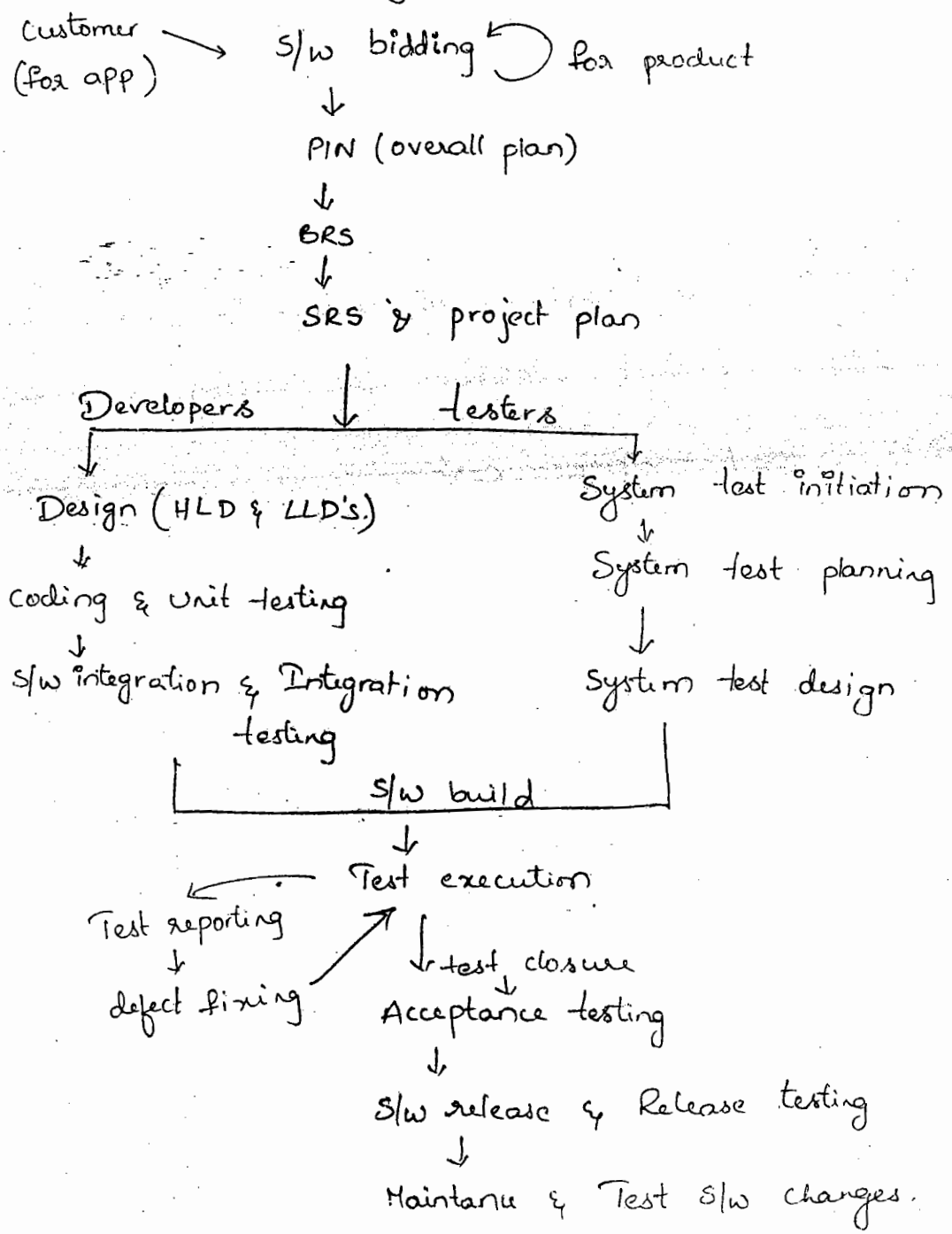
System testing process



System testing process initially starts with test initiation for that a test strategy was prepared by PM and is given to test lead and then test lead prepares a plan for testing called as test planning and is given to tester. Now the tester on the basic of SRS he will develop test cases for each module called as test design after that for the first time s/w build was released by developers & the tester will do execution if any defect is found by them then they will send a test report to developers then the developers fix the problem and release the modified s/w for execution, again the modified s/w is retested by using Re & Regression testing until the s/w build passed all the tests and then the test lead will conduct the review so that the testing is closed at the test closure.

1) System Test initiation / Test Comencement :-

In general system testing process starts with test initiation. In this stage project manager or test manager will prepare test strategy document. This document is also known as test methoddogy or test approach. While preparing test strategy, project manager or test manager is depending on SRS.



From the above process system testing process is starting when SRS was baselined [finalized] ready to use]. But system test execution will system start when s/w build was ready by developers.

The first document in system testing process is test strategy. This document defines required testing approach to be followed by testers. In general, project manager or test manager can follow IEEE 829 format to prepare test strategy. [829 - testing document format
830 - developent " "]

Test strategy document [IEEE 829]

1 Scope & objective :- The importance of testing in current project

2. Business issues :- Budget allocation for testing in current Project.

Ex :- 100% → Project cost

64% (Development & maintance) 36% (System testing)

3. Test responsibility Matrix (TRM) :- List out system testing topics to be Conducted

Ex:

System testing topics	YES/NO	comments.
Functional testing	Yes	
Usability testing	Yes	
Compatability testing	Yes.	
Configuration testing	NO	NOW H/W requirement in current project.
Performance testing	No	needed, but no resources

4) Roles & Responsibilities :- List of roles in testing team for a current project & responsibilities of each role.

Ex: Role	Responsibilities
Test lead	→ Write test plan → Co-ordinate with team → Conduct review.
Senior tester	→ Write test cases. → Co-ordinate with junior tester.
Junior tester	→ Execute test cases on fw build. → Report defects.

5) Communication and status reporting :-

The required negotiations in between every two roles in testing team.

6) Test automation and testing tools :-

The need of automation in this project & available testing tools in our company.

7) Defect reporting & tracking :-

The required negotiation in between developers & testers when defect was found in testing.

8) Testing measurments & metrics :-

The list of measurements & metrics to finish testing in right time.

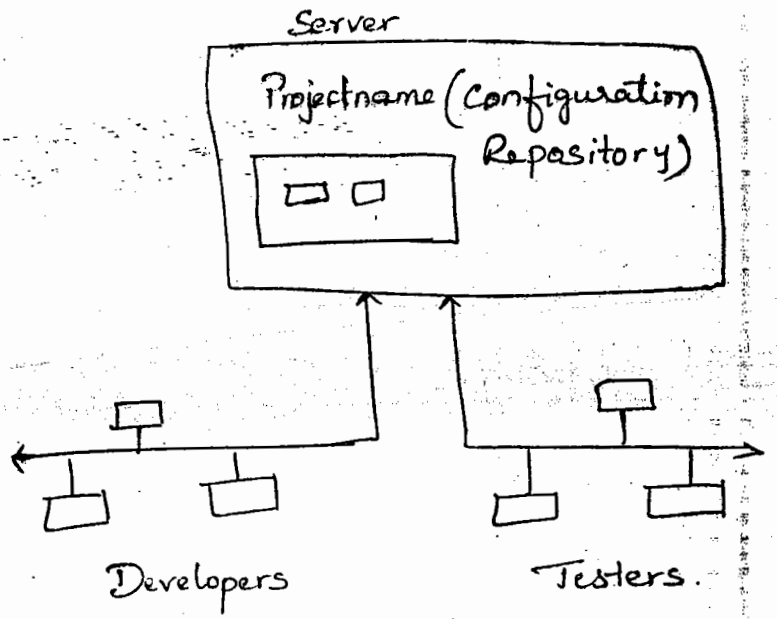
Ex: 5 defects detected per day (8 hours)
30 test cases written per day.

9) Risks and assumptions :-

Risks indicate a future problem. Here project manager can list out expected risks related to testing and assumptions to overcome them.

10) Training plan :- Schedule for training on project requirements to testing team.

11) Configuration Management :- Specifies location in server to store all testing deliverables.



II : Test planning :-

After completion of test strategy preparation by project manager, the corresponding test lead will take the test strategy & then prepare a detailed test plan. While Preparing test plan, test lead will study SRS, project plan and test strategy. After study of required document study, test lead will follow below process for test planning.

(a) Testing team formation:

In general test planning process starts with testing team formation. In this stage test lead depends on below factors.

→ Size of project. [ex. No. of functionalities]

→ Availability of testers on the bench

→ Available test duration.

→ Availability of resources [Ex. testing tools]

Depends on above four factors, test lead will form testing team.

Case Study:-

Type of project	Developers Vs testers	Expected test duration.
Web, ERP, Windows based projects	3 : 1	3 to 5 months.
System Softwares or Embedded softwares	1 : 1	7 to 9 months.
Astifical intelligence (or) Machine caitical s/w Ex: Robotics, Sattelites	1 : 7	12 to 15 months.

b) Identify tactical risks:-

After completion of testing team formation test will identify tactical risks will come in future testing.

Ex: Risk 1: Lack of domain knowledge to testers.

Risk 2: Lack of time.

Risk 3: Lack of resources (ex: testing tools)

Risk 4: Lack of documentation to understand requirements.

Risk 5: Lack of development process seriousness.

(ex: Quality gap or more defects in pending)

Risk 6: Delay in delivery.

Risk 7: Lack of communication to testers.

c) Test plan writing:-

After completion of team formation & risks analysis, lead to lead will go to prepare test plan document in IEEE 829 format.

Format:-

1) Test plan Id: [Name of the document] Unique no or name for test plan document.

2) Introduction: About current project.

3) Test items: List of all modules in current project.

4) Features to be tested: List of testable modules in current project.

5) Features not to be tested: List of modules to skip in testing, because those modules are already tested.

→ What to test?

6) Test approach: Attach test strategy given by project manager.

7) Test environment: Required H/w & S/w for testing.

8) Entry criteria: To start test execution.

→ Study SRS to understand requirements in current project.

→ Establish test environment after writing test cases.

→ Receive s/w build from developers.

9) Suspension criteria:- [To interrupt testing]

→ Major defect detected in current project [Ex: show stopper]

→ Test environment abonded. [Ex: Computer problem]

→ More defects are in pending in development side.

10) Exit criteria:- To stop testing.
→ All testable modules tested.
→ Time excedded.
→ All major defects fixed by developers.

11) Test deliverables:- List of documents to be prepared by testers in testing.
Ex: Test scenarios, Test cases, Automation test scripts, defect reports and status reports.

→ How to test?

12) Staff and training needs:-
Names of testers [who are on the bench] and training schedule for them on project requirements.
Ex: Training provided by SME [subject matter expert] or training provided by business analyst & System analyst or no need of experience because of past experiance.

13) Roles and responsibilities:-
Work allocation to selected testers in modules wise or testing topics wise.

→ Who to test?

14) Schedule:- Dates and time to be followed by testers in testing.

→ When to test?

15) Risks and assumptions:- List of expected risks & solutions to overcome them.

16) Approvals:- Signatures of project manager (PM) & test lead.

(d) Review test plan :-

After completion of test plan document preparation the corresponding test lead will discuss with project manager and select testers for current project. After estimating completeness and correctness of test plan, corresponding selected testers will receive SRS and get training if possible.

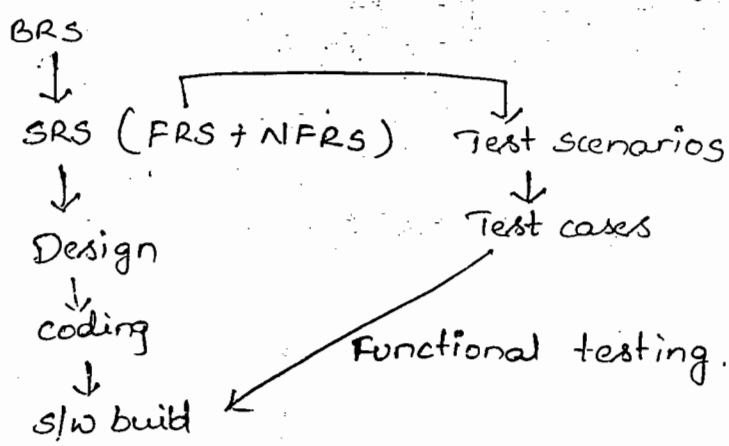
III. Test design :- After study SRS or after completion of training on SRS, testers will concentrate on test design for responsible modules in project. In general test design have two levels such as high level test design and low level test design. In high level test design each testers will prepare test scenarios for responsible modules. and then each testers can enhance each test scenario as detailed test case. in low level test design, Here test scenario specifies an unique test condition. Test case is a document with detailed step by step test procedure.

While preparing test scenarios & test cases testers can follow 4 methods.

1. Functional specifications based test design.
2. Use cases based test design.
3. Screens based test design.
4. Non-functional specifications based test design.

1. High level test design

a) Functional specification based test design :-



From the above diagram, testers will prepare test scenarios depends on SRS functional specifications in SRS.

Functional specification 1 :

In a multi-user s/w, user's are connecting through a login process. Here users are giving user Id & password as inputs. user Id is taking alpha-numerics in lower case from 4-16 characters long. Password is taking alphabets in lower case from 4 to 8 characters long. This login process is providing cancel button to close login screen at any time.

Prepare test scenarios :-

1. verify user id field

Boundary value analysis :- (Size)

min - 4 chars - passed
Max - 16 chars - passed
min-1 - 3 chars - failed
min+1 - 5 chars - passed
max-1 - 15 chars - Passed
max+1 - 17 chars - failed

Equivalence class partition (ECP) :- type

Valid	Invalid
alphanumeric in lowercase	Upper case chars Special chars Blank field.

Test Scenario 2:

Verify password :-

BVA :-

min — 4 — passed
max — 8 — Passed
min-1 — 3 — failed
min+1 = 5 — passed.
max-1 = 7 — passed
max+1 = 9 — failed.

ECP :-

Valid	Invalid
alphabets in lower case	Upper case chars Special chars. Blank field. Numericals.

Test scenario 3 :-

Verify login operation :-

Decision table

Userid	password	outcome
Valid	Valid	next window
Valid	Invalid	Error msg
Invalid	Valid	Error msg
Value	Blank field	"
Blank field	value	"

Test scenario 4:-

Verify cancel button operation

Decision table

user id	password	outcome after Click cancel.
value	value	window closed
value	Blank field	" "
Blank field	value	" "
Blank field	Blank field	" "

Black-box testing techniques:-

While preparing scenarios & cases for functional testing, testers are following black box or closed box testing techniques. because exhaustive testing is impossible. There are 6 black-box techniques to be used.

→ Boundary value analysis to check size or range of i/p field

→ Equivalence class partition to check type of i/p or o/p.

→ Decision tables - to make decisions for validating functionality

→ Orthogonal arrays - to skip repeated decisions.

Ex:

	user id	password	outcome
✓	valid	valid	next window
✓	valid	invalid	error msg
✓	invalid	valid	" "
✗	Invalid	Invalid	" "
✓	value	Blankfield	" "
✓	Blankfield	value	" "
✗	Blankfield	Blankfield	" "

→ State-transitions: to specify flow of operations.

→ Error guessing: Depends upon past experience.

Functional specification-2

In an insurance application the users are Selecting different types of insurance policies. If you select type A insurance then our S/w asks age. Here age field value should be greater than 18 years & should be less than 60 years.

Perepare test scenario.

Test scenario 1: verify Type-A insurance policy selection.

2: verify focus to age. when type-A insurance was selected

3: verify age field → BVA (Range)

min — 19 — valid Passed
max — 59 — valid Passed
min-1 — 18 — inv. failed
min+1 — 20 — Passed
max-1 — 58 — Passed
max+1 — 60 — failed

ECP (type)

valid	invalid
Numerics	alphabets in upper & lower case Special chars. Blank field.

Functional Specification-3

In an embedded S/w a door is opened when Person comes to infront of the door and the door is closed when person went to inside.

Test scenario 1 :-

Verify door open operation :-

Desicion table

Person	door	criteria
Present	opened	passed
Present	closed	failed
abscent	opened	failed
abscent	closed	Passed

Test scenario 2 :-

verify door close operation :-

Decision table :-

	Person	door	criteria
✓	Inside	closed	passed
✓	Inside	opened	failed.
✗	outside	opened	passed
✗	outside	closed	failed.

→ Removed decisions due to repetition [By using orthogonal arrays]

Verify door open and close operations when person is standing at middle of the door :-

Functional specification - 4

In an online banking S/W, users are connecting to bank server through a login process. In this login user can fill below fields.

1. Account no. prefix. [3 digits no.] but does not start with 0 & 1.

2) Account no. Suffix - 6 digits no

3) Password - 4 digits no.

4) Area code - 3 digits no, but it is optional.

5) Purpose - cheque deposit, money transfer, mini statement and bills pay.

Prepare test scenarios :-

Test scenario 1 :-

Verify account no. prefix :- field :.

BVA :- (Size)

min - 3 digits - Passed

max - 3 digits -

min-1 - 2 digits - failed

min+1 - 4 digits. - failed

ECP (type)

Valid	Invalid
[Numarics, but first person does not allow 0 & 1] or [2-9] [0-9] [0-9] (regular expression) or {2}	Alphabets in both cases special chars Blank field (or) [a-z A-Z]

Test scenario-2 :-

Verify account no. Suffix :-

BVA (size)

min = max - 6 digits. Number - Passed

min-1 = 5 digits = failed

min+1 = 7 digits = failed.

ECP : (type)

Valid	Invalid
[0-9]{6}	alphabets special chars empty field

Test scenario-3

verify password field :-

BVA (size) :-

min = max = 4 digits = passed

min-1 = 3 digits = failed

min+1 = 5 digits = failed.

Ecp (type)

Valid	Invalid
[0-9] {4}	alphabets in both cases
	special chars
	Blank fields.

Test scenario-4 :-

verify area code field :-

BVA (size) :-

min = max = 3 digits - passed

min-1 = 2 digits - failed

min+1 = 4 digits - failed.

Ecp (type]

Valid	Invalid
[0-9] {3}	alphabets
Blank field	special chars.

Test Scenario-5 :-

Verify purpose selection like cheque deposit or money transfer or mini statement or bills pay.

Test scenario-6

verify login operation to connect to bank server.

Decision table:

Mandatory fields	Area code (optional)	Expected outcome
All are valid	Valid	Connect to server.
All are valid	Blank field	Connect to server
Any one invalid	Valid/ Blank field	Error msg.
All are valid	Invalid	Error msg
Any one blankfield	Valid/Blankfield	Error msg

Note:- While functional test design, testers are using regular expressions as optional. In general regular expressions are used to match with a type of values. Due to this reason, testers are using the regular expressions to specify expected type of fields in ECP.

[ ] - one position

[0-9] - one digit

[a-z] - one character in lower case.

[A-Z] - one character in upper case.

[a-z A-Z] - one character in lower or upper case.

[a-z][A-Z] - First character as lower case and second character as upper case

{ } - Specifies no. of positions.

[0-9]{4} - Numerics with 4 digits only.

[0-9]{2,4} - Two digits to four digits number.

[0-9]{,4} - No digits to " " "

[0-9][4,} - Min. 4 digits to infinite digits

[0-9]+ - one digit to infinite number.

[0-9]* → No digit to infinite digit number.

[0-9]? → No digit or one digit number.

[A-Z][a-z]* – First char upper char & 2nd char is no or more lower case.

[A-Z][a-z]+ – First char upper case & 2nd char is one or more lowe case.

[RAM][a-z]* – start with R (or) A (or) M and than no or more lowe case.

Functional Specification-5 :-

In a shopping application users are applying for different type items purchase orders. Every purchase order is allowing users to enter item number and quantity. Here item number is a ~~plo~~ four digits number with "I". For ex. I4026 is an item number. Quantity is allowing upto 10. Prepare test scenarios for above purchase order creation.

Test Scenario 1 :-

Verify item number field :-

BVA (Size) Positions

Min = Max = 5 = passed

Min - 1 = 4 = failed

Min + 1 = 6 = failed.

Ecp : (type)

Valid	Invalid
[I][0-9]{4}	Alphates except I Blank fields Special chars.

Test Scenario - 2

Verify quantity field [Range] :-

BVA (range)

min - 1 - Passed
max - 10 - Passed
min - 1 = 0 - failed
min + 1 = 2 → Passed
max + 1 = 9 - Passed
max + 1 = 11 - Failed.

ECP :-

Valid	Invalid
[1-9]/([1][0])	Alphabets Special chars Blank field

Test scenario - 3

Verify purchage order operation

Decision table :-

Fields	outcome
All are valid	purchase order created
Any one invalid	Error msg.
Any one blank field	Error msg

Note :- To alternate more than one regular expression, we can use pipe [|].

Ex : (Regular ex_1) | (Regular exp_2)

Functional Specification 6:-

In a library management s/w readers are applying for registration to get reader Id. In this registration Process, readers can fill below fields.

1) Reader name - alphabets with initcap as sigle word.
2) City name - alphabets in any case as one or more words.
3) Telephone no if available [valid in India].
4) State - Two or more characters in upper case.

After filling above fields user can click "register" button and then our s/w returns, reader Id in below format.

- mm-yy-xxxx
  - mm: month
  - yy: year
  - xxxx: 4 digits number.

Test scenario 1:-

Verify reader name:-

BVA (Size):-

min = 1 char - Passed
max = 256 char - Passed
min-1 = 0 (Blank field) - Failed
min+1 = 2 - Passed
max-1 = 255 - Passed
max+1 = 257 - failed.

Ecp (Type)

Valid	invalid.
[A-Z][a-z]*/ [A-Z][a-z]{,255}	Numarials. Special chars Blank field.

Test Scenario 2

Verify city Name :-

BVA :- min = 1 - Passed
max = 256 - Passed
min - 1 = 0 - Blank field - failed
min + 1 = 2 - Passed
max - 1 = 255 - Passed
max + 1 = 257 - failed.

ECP (type) :-

Valid	Invalid
([A-Z a-z] + [ ] ? [a-z] [A-Z] * [ ] ?) {1, }	Numarics
con	Special chars except blank space.
[A-Z a-z] ([a-z a-z] + [ ] ? {1, }	Blank field.

Test Scenario 3

Verify telephone no. field :-

BVA (Size) :-
min - 10 digits - Passed
max - 12 digits - Passed
min - 1 - 9 digits - failed
min + 1 - 11 digits - Passed
max - 1 - 11 digits - "
max + 1 . 13 digits . failed.

Ecp (type)

Valid	Invalid
[0-9] {10-12}	alphabets
Blank field	Special characters

Test scenario-4

Verify state: field:

BVA (size) :-

Min = 2 characters = Passed

Max = 256 " = Passed

Min-1 = 1 " = failed

Min+1 = 3 " = Passed

Max-1 = 255 " = "

Max+1 = 257 " = failed

ECP (Type) :-

Valid

[A-Z] {2,256}

Invalid.

Numerics

Alphabets in lower case

Blank field

Special chars.

Test scenario-5:-

After Verify registration operation after click register button.

Decision table :-

Manditory fields	phone number (Optional)	Expected outcome.
All are valid	valid	"Reader Id" Returns.
All are valid	Blank field	"
Any one invalid	valid/Blank field	Error msg.
All are valid	Invalid	Error msg.
Any one blankfield	valid/Blankfield	Error msg

Test scenario-6:-

Verify format of reader Id after successful registration.

BVA (size):-

min = max = 10 positions = passed

min-1 = 9 = failed

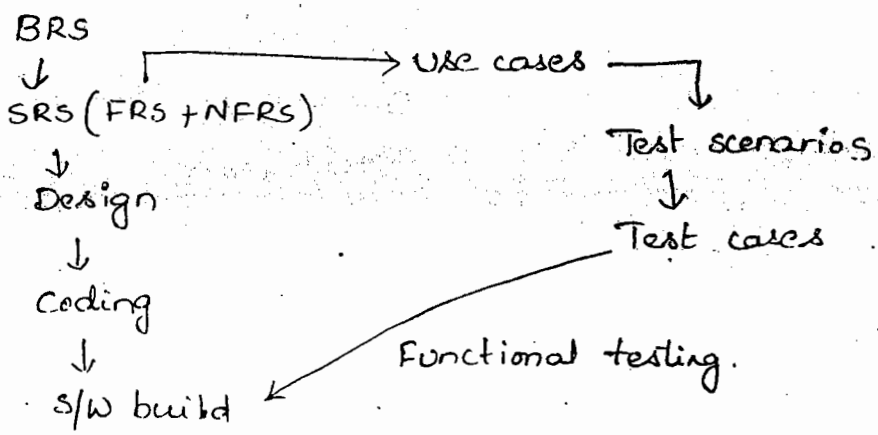
min+1 = 11 = failed.

ECP (type) :-

Valid	Invalid
(([0][1-9])/([1][0-2])) [-] [0-9]{2} [-] [0-9]{4}	Alphabets, Special char except - Blank field.

b) Use-cases based test design :-

In some organizations project management is providing usecases insted of functional specifications in SRS. In general use-cases are more elaborative than function specifications. to understand customer requirements. Generally test management requires usecases when project requirements are complex. or project was outsourced (developed by another company)



Usecase 1 :-

1. Usecase id : UC_LOGIN
2. Description : our s/w allows users to access after successful login.
3. Actors : A valid user consists of user id & password, Here User id allows alphanumerics in lowercase from 4-16 chars long. Password allows alphabets in lowercase from 4-8 chars long.
4. Active flow :

Events	Expected outcome
Activate login window →	For valid data, next window appear.
Enter user id → enter	For invalid data, error msg appears
Password - click "ok"	

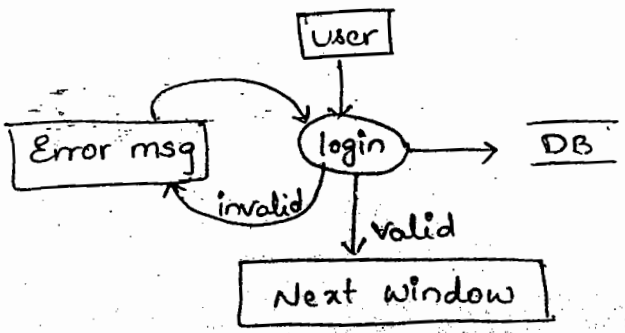
5. Pre-conditions:-

Before login new users are registering for user id & pwd

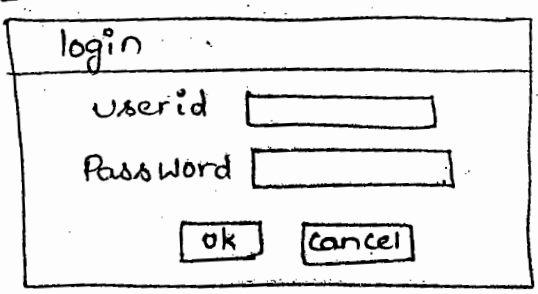
6. post conditions:-

After login, valid user can logout at end

7. Use inf Activity flow diagram:-



8. Prototype (Sreen shot):-



9. Alternative Events:- None.

10. Related use cases:- UC_REGISTER, UC_LOGOUT

Prepare test scenarios.

Test scenario 1:-

Verify user id field

BVA (Size)

Min - 4 chars - Passed
Max - 16 chars - Passed
min-1 - 3 chars - failed
min+1 - 5 chars - passed
Max-1 - 15 chars - passed
Max+1 - 17 chars - failed.

ECP (type)

Valid	Invalid
[a-z] {4,16}	Upper case chars, special chars, Blank field.

Test scenario 2

verify password field :-

BVA :-

min - 4 - Passed
max - 8 - passed
min-1 - 3 - failed
min+1 - 9 - failed

max-1 = 7 - passed
max+1 = 9 - failed.

ECP :-

Valid	Invalid
[a-z] {4, 8}	Upper case chars, special chars blank field.

Test scenario 3 :-

Verify login operation after click "ok" button

Decision table :-

User id	password	outcome
valid	valid	next window
valid	invalid	Error msg.
Invalid	valid	Error msg
value	blank field	"
Blank field	value	"

Test scenario 4 :-

Verify cancel button operation to close window at any time.

Decision table :-

Fields	expected outcome.
All are filled	window closed
Some fields only filled	"
All fields are empty	"

Use case 2:-

1. Usecase id : UC-BOOK-ISSUE

2. Description : This use case is describing book issue operation in a liabrary management s/w.

3. Actors : Valid reader is eligible for book issue when that book is available in liabrary. Here reader id is in "MH-YY-XXXX" format and book id is in "Book-XXXX" format.

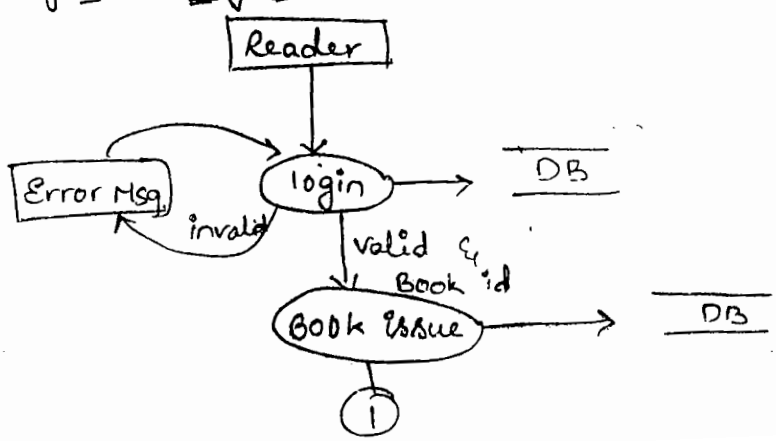
4. Pre conditions: The reader is already registered and corresponding book information is already feeded. -

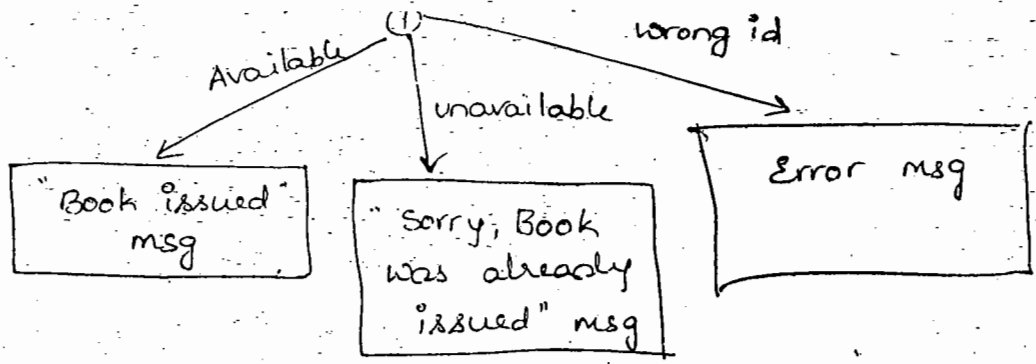
5. Activity flow:-

Events	Expected outcome
Enter reader id → click 'go' button. Enter book id → click "go" button	For valid reader, s/w focus book id field. For invalid reader, Error msg appears. For available book, "Book issued" msg appears. For unavailable book "Sorry" Book was already issued", appears. For wrong book id 'error m appears'

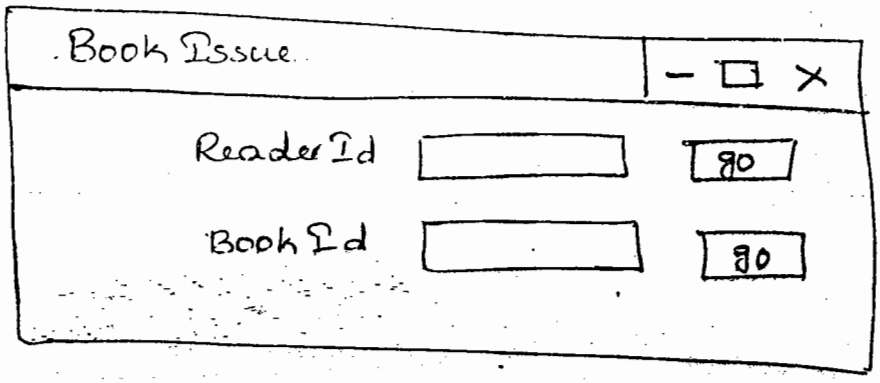
6. Post Conditions:- If reader id & book id are valid and book is available then the corresponding reader will get a link to download that book with licence (Part time).

7. Activity flow diagram:-





8. Prototype :- (Screen Shot)



9. Alternative Events :- None

10: Related usecases :- UC_REGISTER, UC_BOOK_FEED

Prepare test scenarios.

Test Seenario 1 :-

Verify Reader id field :-

Min = Max = 10 positions - Passed.

Min-1 = 9 - failed

Min+1 = 11 - failed

ECP (type)

Valid	Invalid
(([0][1-9])/([1][0-2]))[-] [0-9]{2}[-][0-9]{4}	Alphabets; Special chars except -, Blank field.

Test Scenario 2:-

Verify reader Id validation after click 'go' button.

Decision table

Reader Id	Expected outcome
Valid / Registered	focus to Book id field
Invalid	Error msg
Blank field	Error msg.

Test - Scenario 3:-

Verify book id field

BVA:-

Max · min = 9 positions = Passed

Min - 1 = 8 " - failed

min + 1 = 10 " - failed

ECP:-

Valid	Invalid
[B][O][2][K][-][0-9]{4}	Special char except - Blank field. Alphabets except B, O, K

Test scenario 4:-

Verify book issue operation after click go button.

Decision table:-

Reader Id	Book id	Excepted outcome
Valid	valid & available	"Book Issued" msg
valid	valid & unavailable	"Sorry", Book was already issued msg
valid	invalid	Error msg.
Valid	Blank field	Error msg.

Test scenario - 5

Verify minimize icon at any time.

Decision table.

fields	expected outcome.
1. All are filled	window was min.
2. Some fields filled	" " "
3. All are empty	" " "

Test scenario - 6

Verify maximize icon at any time.

Decision table.

fields	expected outcome.
1. All are filled	window was maximized.
2. Some fields filled	" " "
3. All are empty	" " "

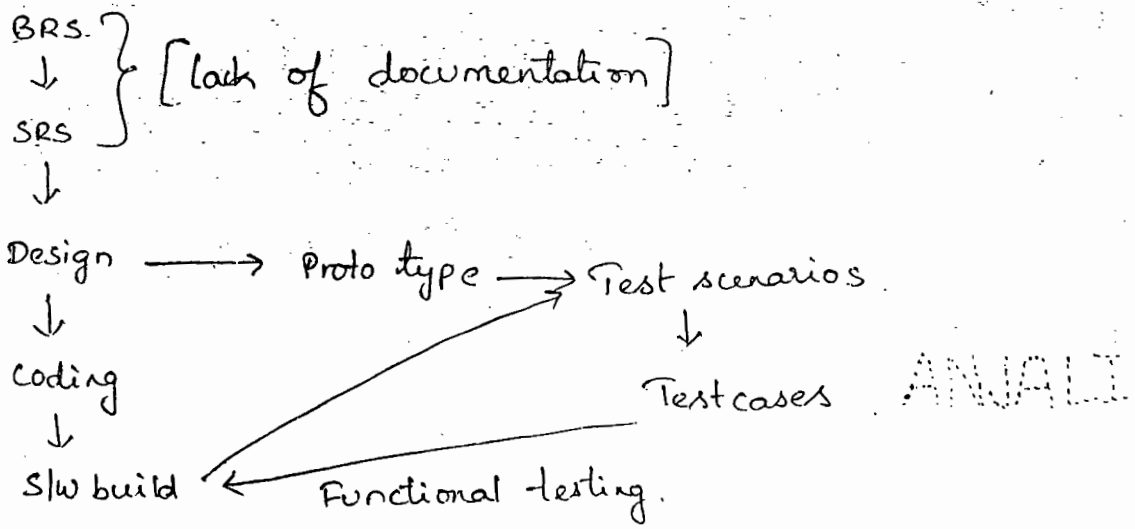
Test scenario - 7

Verify close operation at any time

Same as above

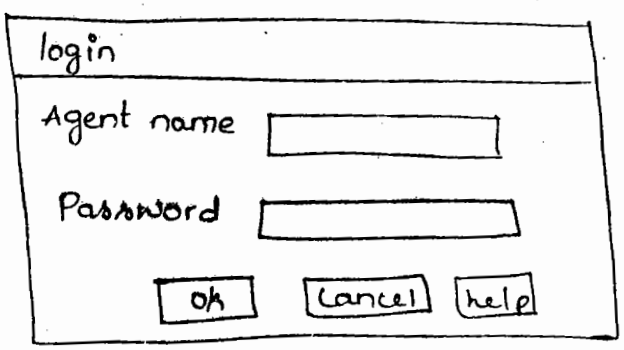
(c) Screens based test design:-

In some organizations documentation work is optional. Due to this reason, testing team directly can receive prototype or s/w build. Instead of functional spects & usecases. ~~Due to Due to this~~ In this situation, testing team can understand requirements & do test design depends on that prototype & s/w build.



From the above diagram, testing team is preparing test scenarios & test cases depending upon prototype or Orginal S/w or both. due to lack of documentation. In this model testing team is receiving one more extra document from developers like "field validations"

Ex:1 Screen :-



Field validations followed by developers :-

Agent name → 4 to 10 chars as alphanumerics with –

Password → upto to 8 chars as alphanumerics.

Prepare test scenarios :-

Verify agent name field :-

BVA :- min = 4 chars . Passed
max = 10 chars . "
min - 1 = 3 " . Failed
min + 1 = 5 " . "

max - 1 = 9 chars = passed
max + 1 = 11 " . failed.

ECP (Type):

Valid	Invalid.
[A-Z a-z 0-9 _] {4, 10}	Special char except _, Blank field

Test Scenario-2

Verify password field:-

BVA (size)

min : 1 char = Passed max-1 = 7 = Passed
max = 8 " = " max+1 = 9 = failed.
min-1 : 0 (Blank field) char = failed
min+1 = 2 " = passed

Ecp (Type)

Valid	Invalid
[A-Z a-z 0-9] {1, 8}	Special chars, Blank field.

Test Scenario 3:-

Verify login operation after clicking ok button.

fields	Expected outcome
All fields are valid	Next window
Any one invalid	Error msg
Any one blank	" "

Test Scenario 4:-

Verify close (cancel) operation after click cancel button.

fields	outcome.
All fields are (filled) valid	Window closed
Fill some fields	" "
All fields empty	" "

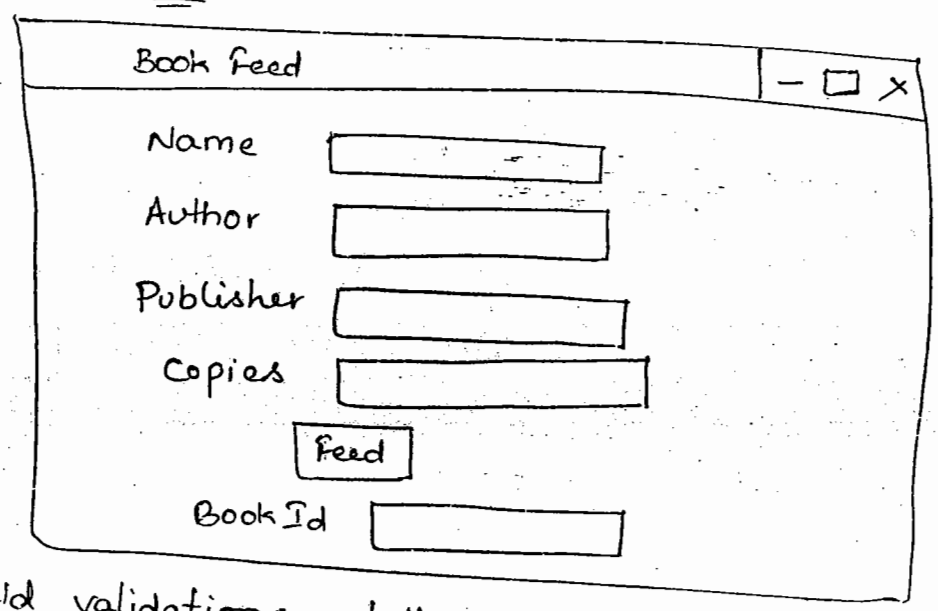
Test scenario-5

Verify "Help" window opens after click help button.

Decision table

Fields filled outcome	expected outcome
All fields filled	Help window opened
Some fields filled	" " "
All fields empty	" " "

Ex:2:- Screen:-



Field validations:- followed by developers.

Name:- Alphabets in uppercase upto 15 chars as single word.

Author:- Alphabets upto 30 chars as one or more words.

Publisher:- " " " " "

Copies :- upto 10

Book Id :- It will come after successfull book feed as "Book_xxx"

Prepare test scenarios:-

1) Verify name field:-

BVA (Size):- min = 1 char . Passed
max = 15 " . Passed
min-1 : 0 = Blank field : failed
min+1 = 2 = passed
max-1 = 14 = passed

ECP (type)

Valid

[A-Z] {1,15}

Invalid.

Numerics, blank field, special char : Alphabets in lower case.

Test scenario 2:-

Verify book author field

BVA (size): min = 1 char = passed
max = 30 chars = "
min-1 = Blank field = failed
min+1 = 2 chars = Passed
max-1 = 29 " = "
max+1 = 31 " = failed.

ECP (type):

Valid

([a-z]A-Z]+[\s]?) {1,}

Invalid.

Numerics, special chars except Blank space, Blank field or [\s]+

Test scenario 3:-

Verify publisher field

BVA (size): min = 1 chars
max = 30 chars.

→ Same as above

Test scenario 4:-

Verify, No. of copies field:-

BVA (size):-
Range

min = 1 = Passed
max = 10 = "
min-1 = 0 Blank field = failed
min+1 = 2 = Passed
max-1 = 9 = "
max+1 = 11 = failed

ECP (type):-

Valid

[1-9]|([1][0])

Invalid

Alphabets, Special chars, Blank field.

Test Scenario 5:-

Verify Book feeding operation after click feed button.

Decision table

Fields	expected outcome
All are valid	Book id returned
Any one is invalid	Error msg.
Any one blank field	Error msg.

Test Scenario-6:-

Verify Book Id field after successful Book feeding operation

BVA (size): Min = max = 9 positions - Passed
min - 1 = 08 " - failed
min + 1 = 10 " : failed

ECP (Type):

Valid	Invalid
[B][0]{2}[k][-][0-9]{4}	Alphabets except B, 0, k, Special chars (except -), Blank fields

Test scenario 7:-

Verify minimize icon in book field window.

Decision table

Fields	outcome
All are filled	window was minimized
Some fields "	" " "
All are blank fields	" " "

Test scenario 8:-

Verify maximize icon in book field window

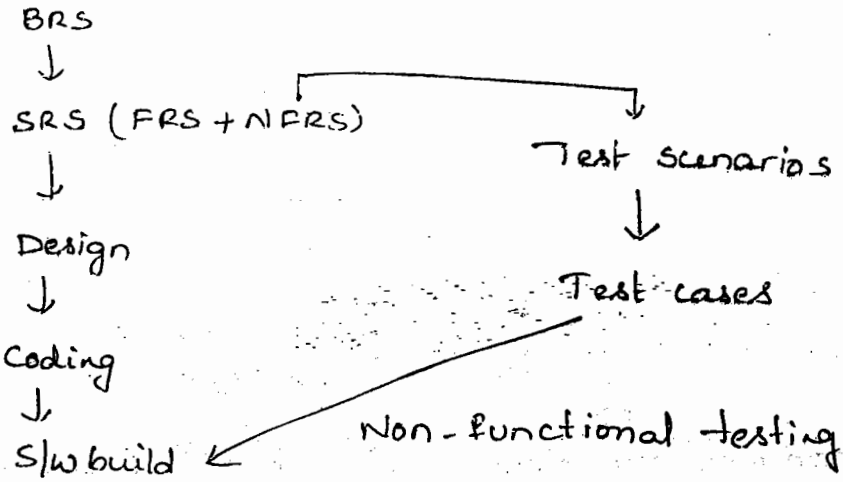
Same as above

Test scenario 9

Verify close icon in book field window

a) Non-functional specification based test design:-

After completion of functional test design w.r.t functional specifications or usecases or screens, testing team is concentrating on non-functional test design w.r.t non-functional specifications in SRS. There is no alternative for non-functional test design.



Ex:- Non functional specifications

"Library management" is a LAN based S/W. It will run on all windows flavours & LINUX versions. This S/W consists of 4 modules such as reader registration, book feeding, book issue & book returns. But this S/W is not sharing resources of other S/W. This S/W database will maintain 1 lakh readers, 10 lakh books & 1 crore transactions, like issue or return. This S/W is easy to install & user friendly S/W. In existing LAN, this S/W will be accessed by 10 library employees at a time.

Prepare test scenarios for above non-functional requirements.

a) Test scenarios for usability testing:

For every project, the organizations are maintaing common check list (test scenario) for usability testing. This check list validates user friendliness of screens

Test Scenario 1:-

Verify spelling of control labels in each screen.

Test Scenario 2:

Verify control labels initcap.

TS 3: Verify Control labels font style uniqueness.

TS 4: Verify control labels font-size uniqueness.

TS 5: Verify line spacing uniqueness in between labels and their controls.

TS 6: Verify line spacing uniqueness in between controls in a screen.

TS 7: Verify uniformity (grouping) of functionally related controls in a screen

TS 8: Verify alignment of controls in screen.

Ex: left alignment, right, top, bottom alignment.

TS 9: Verify icon symbols in throughout screens of s/w

TS 10: Verify correctness of tool tips in throughout screens of s/w.

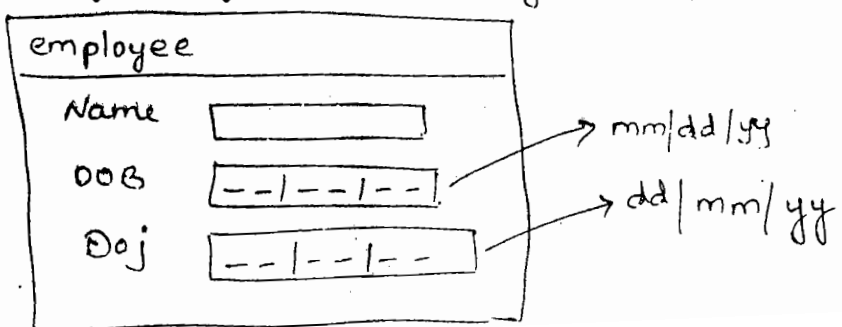
TS 11: Verify meaning of error masseges.

TS 12: Verify correctness of help documents (user manuals testing)

TS 13: Verify fullforms of shortcuts in throughout screens of s/w.

TS 14: Verify format of datafields in throughout screens of s/w.

TS 15:



TS15:- Verify keyboard access in every screen of s/w.

TS16:- Verify meaning of labels of Controls & screens.

TS17: Verify frames in screens. Here each frames consists of functionally related controls.

TS18: Verify scroll bars when screen size more than desktop.

TS19: Verify colours uniquess in throughout screens of s/w.

TS20: Verify short navigations availabilty in throughout screens of s/w.

TS21: Verify availabity of system menu for each screen of the s/w [system menu consists of restore, move, Size, minimize, max, close options]

TS22: Verify the availability of OK & CANCEL like buttons. in each screen of software. [OK like buttons can countinue navigations in s/w, but cancel like buttons can stop navigations].

Note:- TS21 & TS22 are proposed Microsoft.

b) Test scenarios for compatability testing:-

From the non-functional specification, library management SW will run on Windows & Linux versions as operating system.

TS1:- Verify reader registration operation in a system, which consists of below platform.

Platform table/Environment table

(OS)	Reader registration	criteria
Windows xp	working	Passed
" Vista	"	"
" 7	"	"
Linux Redhat	"	"
Linux Fedora	"	"

TS2:- Verify book feeding operation in a system, which consists of below platform.

platform table / Environment table

platform (OS)	Book feeding Registration table	criteria
Window xp	working	Passed
Vista	"	"
7	"	"
Linux Red hat	"	"
Linux fedora	"	"

TS3: Verify book issue operation in a system, which consists of below platform.

Same as above.

TS4: Verify book returns operation in a system, which consists of below platform.

same as above.

c) Test scenarios for configuration testing :-

From the non-functional specification, our library management s/w run on various types of LANs. [BUS topology, Ring topology, hub topology].

TS1:- Verify reader registration operation in a LAN, which have below hardware configurations.

H/W configuration table (matrix) :-

H/W configurations	Reader registration	criteria
Bus topology LAN Ex: Ether Net	working	Passed
Ring topology LAN	"	"
HUB topology LAN	"	"

TS2:- Verify book feeding operation in a LAN, which have below hardware configurations.

H/w Configuration	Book feeding	Criteria
Bus topology Ex: Ethernet	working	passed.
Ring topology	"	"
Hub topology	"	"

TS3: Verify book issue operation in a LAN, which have below hardware configurations.

Same as above

TS4: Verify book returns operation in a LAN, which consists of below hardware configurations.

Same as above.

d) Test scenarios for performance testing

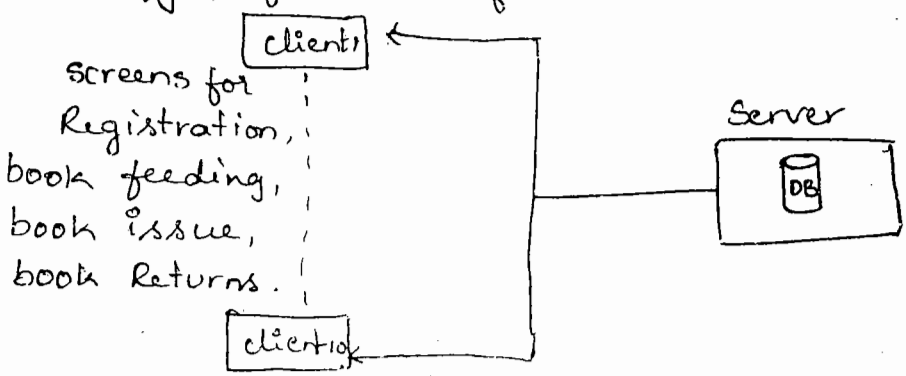
From the non-functional specification our library management S/w is allowing 10 concurrent users.

TS1: Verify Performance of reader registration operation under load 10 users.

TS2:- Verify performance of book feeding operation under load 10 users.

TS3:- Verify performance of book issue operation under load 10 users

TS4: Verify performance of book return operation under load 10 users



TS5 :- Verify reader registration operation users limit and more than 10 users.

TS6 :- Verify book feeding operation users limit under more than 10 users.

TS7 : Verify book issue operation users limit under more than 10 users.

TS8 : Verify book returns operation users limit under more than 10 users. load.

TS9 : Verify reader registration operation reliability while increasing load suddenly.

TS10 : Verify book feeding operation reliability while increasing load suddenly.

TS11 : Verify book issue operation reliability while increasing load suddenly.

TS12 : Verify book return operation reliability while increasing load suddenly.

TS13 : Verify reader registration operation durability under load 10 users.

TS14 : Verify book feeding operation durability users limit under load 10 users.

TS15 : Verify book issue " " " " "

TS16 : Verify book returns " " " " "

TS1 to TS4 → load testing

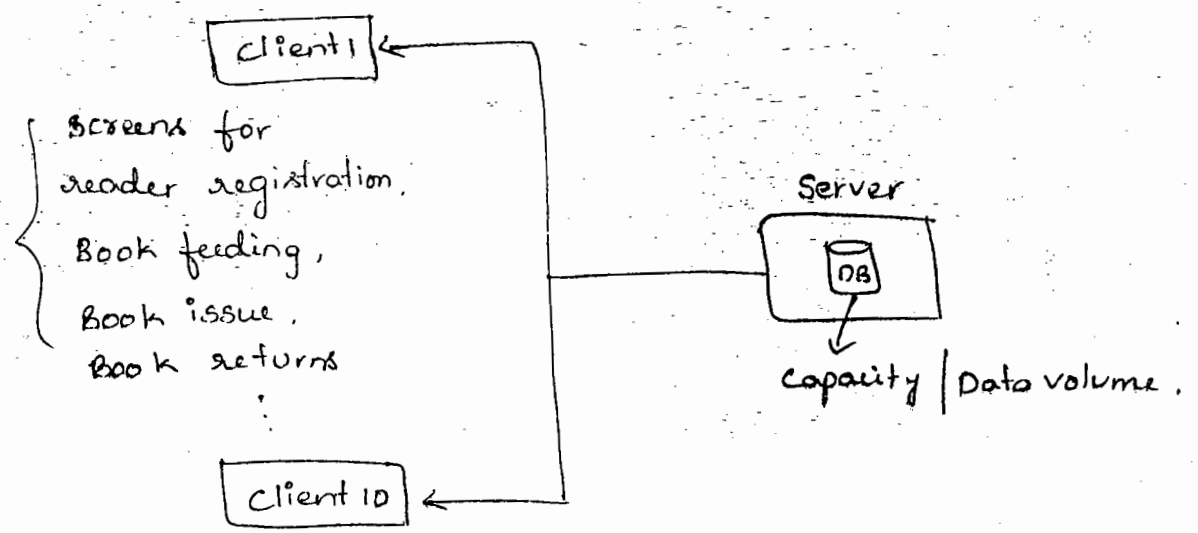
TS5 to TS8 → Stress "

TS9 to TS12 → Spike "

TS13 to TS16 → Endurance or durability testing.

⑨ Test scenarios for data volume testing

From the non-functional specification library management S/W database capacity was 1 lakh readers, 10 lakh reader book & 1 crore transactions it is customer expectation



TS1:- Verify data base capacity on no.of readers as one lakh by using reader registration operation.

TS2:- Verify data base capacity on no.of books as 10 lakhs by using book feeding operation.

TS3:- Verify data base capacity on no.of transactions as 1 crore by using book issue & book return operations.

f) Test scenarios for installation testing:-

In general testing team is using a common check list for instalbation testing.

TS1:- Verify setup program execution to start installation.

TS2:- Verify screens understandability while installation.

TS3:- Verify occupied disk space after installation.

Case Study:-

Testing topic	Test design deliverables.
Functional testing	Test scenarios → Test cases,
Usability testing	Check list
Compatability testing	platform matrix or environmentmatrix
Configuration testing	H/w Configurations matrix

Performance testing	User scenarios
Data volume testing	Data base capacity matrix / data volume matrix
Installation testing	check list
Inter system testing	Test scenarios → Test cases

II. Low level test design :-

After completion of test scenarios selection, testing team is implementing test scenarios as test cases. Each test case is a document and its specifies a step by step procedure to run a test on a SUT (S/w under testing).

Test case format (IEEE 829) :-

1. Test case Id : Unique no. or name for future reference.
2. Test case name : Corresponding test scenario.
3. Feature to be tested : Corresponding module ~~name~~ in SUT.
4. Test suite Id : Name of a batch (This case is a member in that batch)
5. priority : Importance of test case w.r.t customer requirements and expectations.

   Ex :- high (P_0) → Functional test cases.
   Medium (P_1) - Non-functional test cases except usability
   low (P_2) → usability test cases
6. Test environment : Required H/w & H/w to run this test case on SUT.
7. Test duration : Expected time to run this test case on SUT.

   Ex : 20 min. is avg. time to run one test case on SUT
8. Test setup / pre condition :- The status of SUT before start this case execution.
9. Test procedure :

Step no	Description	Test data	Expected outcome	Actual outcome	Step Result	Defect Id

Test design | Test execution

10. Test result :- The final result of test case after execution.

Functional specification :-

In a S/w users are connecting by using login. This login process allows user id from 4-16 chars long as alphanumerics in lower case. It allows password from 4-8 chars long as alphabets in lower case. This login provides cancel button to close login window at any time.

Prepare test scenarios & test cases.

Test scenarios :-

TS 1 :- Verify user id field.

TS2 : Verify password field.

TS3 : verify login operation.

TS4 : Verify cancel button to close window.

Note : In general testing team is using MS-word or open-office word. to write test scenarios. They can use MS-Excel or open office excel for test cases writing.

Test cases :-

Test case document 1 :-

1. Test case id : TCD_LOGIN_FT_PO_ANJALI_1st March_1
2. Test case name : Verify user id field.
3. Test suite id : TS_LOGIN_1

4. Test setup : Login window opened. and we have valid & invalid data.

5. Test procedure :

step No	step description	Test data	Expected outcome
1	Activate login window & then fill userid field	4 chars	Allows
		16 chars	Allows
		3 chars	Not Allow
		5 "	Allow
		15 "	Allow
		17 "	Not Allow
		[a-z 0-9] +	Allow
		[A-Z] +	Not Allow
		special chars	"
		Blank field	"

Testcase document 2 :-

1. Test case id : TCD - LOGIN - FT - P$_0$ - ANJALI - 1st March - 2

2. Testcase name: Verify password field.

3. Test suite id : TS - LOGIN - 2

4. Test setup : Login window opened & we have valid & invalid dat

5.

step No	step description	Test data	Expected outcome
1.	Activate login window and then fill password field.	4 chars	Allows
		8 chars	"
		3 chars.	Not allows
		5 chars	allows
		7 chars	"
		9 chars.	Not allows
		[a-z] +	Allow
		[A-Z] +	Not allow
		special chars	"
		Blank field	"
		[0-9] +	"

Test case document 3

1) Test case id : TCD_LOGIN_FT_P_0_ANJALI_1st March_3

2) Test case name : Verify login operation.

3) Test suite id : TS_LOGIN_3

4) Test setup : Login window opened & we have valid & invalid data

5) Test procedure :

Setp no	step description	Test data	Expected outcome
1.	Activate login window → fill user id & Pwd → "ok"	All are valid Any one invalid Any one blankfield	next window Err msg. Err msg.

Test case document 4

1. Test case id = TCD_LOGIN_FT_P_0_ANJALI_1st March_4

2. Testcase Name : Verify cancel button to close window.

3. Test suite id : TS_LOGIN_4

4. Test setup : Login window opened & we have valid & invalid data to fill

5. Test procedure.

step no	step description	Test data	Expected outcome
1.	Active login window → Fill user id & Pwd → click "cancel"	All fields filled Some fields fill All fields empty	window closed " "

Functional specification 2:

After successful login, corresponding bank employee will select required options in next window. If an employee selected "fixed deposit" option, then employee can get fixed deposit window. This window consists of below fields.

→ Depositor name → Alphabets as one or more words.
→ Amount → 1500 to 1 lakh.
→ Time → upto to 12 months.
→ Interest → Numeric with 2 decimal points.

After filling above fields, employee can click deposit butto to create fixed deposit transaction. If time is greater than 10 months then our fixed deposit operation allows interest greater than 10%..

Prepare test scenarios and test cases.

Test scenarios :-

TS5: Verify the selection of "fixed deposit" option.
TS6: Verify depositor name field.
TS7: Verify amount field.
TS8: Verify time field.
TS9: Verify interest field.
TS10: Verify fixed deposit operation.
TS11: Verify bank rule on fixed deposit operation.

Test cases :-

Test case document 5 :-

1) Testcase id : TCD_FD_FT_P0_ANALI_1st march_5
2) Testcase name : Verify the selection of "fixed deposit" option.
3) Test suite id : TS_FD_01.
4) Test setup : Login was successful & options window opened.
5) Test procedure.

step no	step description	test data	expected outcome.
1.	Do login with valid data → select fixed deposit option in next window.	None.	fixed deposit window opened.

Test case document 6

1) Test case Id : TCD - FD - FT - P_0 - ANJALI - 1st March - 6

2) Test case Name : Verify depositor name field

3) Test suite id - TS - FD - 2

4) Test setup : options → fixed deposit window opened.

5) Test procedure :

stepNO	setp description	Test data	expected outcome.
1.	Do login with valid data → select "FD" option in next window → Fill "depositor name" field in fixed deposit window.	1 char	Allows
		256 char	"
		Blank field.	Not allow
		2 char	allows
		255 char	"
		257 "	Not allow.
		([a-z A-Z] + [ ]?) {1,}	Allows.
		special chars ex. [0-9] + blank space	Not allows
			Not allows

Test case document 7:

1) Test case Id : TCD - FD - FT - P_0 - ANJALI - 1st March - 7

2) Test case name : Verify amount field

3) Test suit id : TS - FD - 3

4) Test setup : "Fixed deposit" window opened.

5) Test procedure

step No	step description	Test data	expected outcome
1.	Do login for valid data → select "fixed deposit" option → Fill Amount field in "fixed deposit" window.	1500	allows
		1 lakh	"
		1499	Not allow
		1501	Allows
		99,999	Allows
		100001	Not allows
		[a-z A-Z]+	"
		Special chars	"
		Blank field	"

Test case documentg:

1. Test case id: TCD - FD - FT - P0 - ANJALI - 1st March - 8
2. Test case Name: Verify time field
3. Test suite Id: TS - FD - 4
4. Test setup: "Fixed deposit" window opened.
5. Test procedure:

Step No	step description	test data	Expected outcome
1.	Do login with valid data → select "FD" option → Fill time field in "FD" window	1 month	Allow
		12 months	Allow
		0	Not allow.
		2 months	Allow
		11 months	"
		13 months	not allow.
		Alphabets	"
		Special chars	"
		Blank field	"

Test case document 9:

1. Test Cased id : TCD - FD - FT - P0 - ANJALI - 1st March - 9
2. Test case Name: Verify interest field
3. Test suite Id : TS - FD - 5
4. Test setup: "Fixed deposit" window opened

5. Test procedure :-

StepNo	step description	Test data	Expected outcome
1.	Do login for valid data → select "FD" option → Fill "interest field in "fixed deposit" window	min 0.01	allows
		max 100	allows.
		0.009	Not allow
		0.02	Allow
		99.99	Allow
		101.01	Not allow
		[a-z A-Z] +	"
		Special chars except .	"
		Blank field	"
		[0-9] + [.] [0-9] {2}	Allows.

Test case document 10 :-

1. Test case id = TCD - FD - FT - P0 - ANJALI - 1st March - 10
2. Test case name : Verify fixed deposit operation.
3. Test suit Id : TS - FD - 6
4. Test setup : " Fixed deposit " window opened.
5. Test procedure :

Note : In regular expressions "\s" indicates blank space char. Here "." indicates any char. so, we can use "\." for decimal point.

StepNo	step description	Test data	Expected outcome
1.	Do login operation for valid data → Select "FD" option → Fill fields in "FD" Window → click "deposit" button.	All are valid	Successful deposit
		Any one invalid	Error msg.
		Any one blank field	" "

Test case document 11:

1. Test case id : TCD - FD - FT - P_0 - ANJALI - 1st March - 11
2. Test case Name : Verify bank rule on fixed deposit operation
3. Test suit id : TS - FD - 7
4. Test setup : "Fixed deposit" window opened
5. Test procedure

StepNo	Step description	Test data	expected outcome
1.	Do login operation → for valid data → Select "FD" option →	valid depositor name, valid amount, time> 10 months, & Interes > 10%	"Successful deposit"
	fill fields in "FD" window → click "deposit" button.	→ valid depositor name, valid amount, time> 10 months and interest ≤ 10%	Error msg

CASE STUDY:- [Tips in test design]

While preparing test scenarios & test cases for responsible modules in SUT, test engineers can follow below conventions.

→ A test case name must be start with Verify or validate or check

→ A test case must be specify what tester has to perform & what response coming from SUT. Ex: Test procedure in test case format.

→ A test case must be cover a field or an operation completely

→ A test case must be divided into positive & Negative

→ A test case must be reusable (Allows editing w.r.t changes in requirements of customer in future).

→ A test case must be give consistent results to independent of testers. Ex: NO Confusions/ambiguities in test cases.

→ A test case must be divided into different test cases, when the test case consists of more than 10 to 15 steps

→ A test case must be reviewed by test lead.

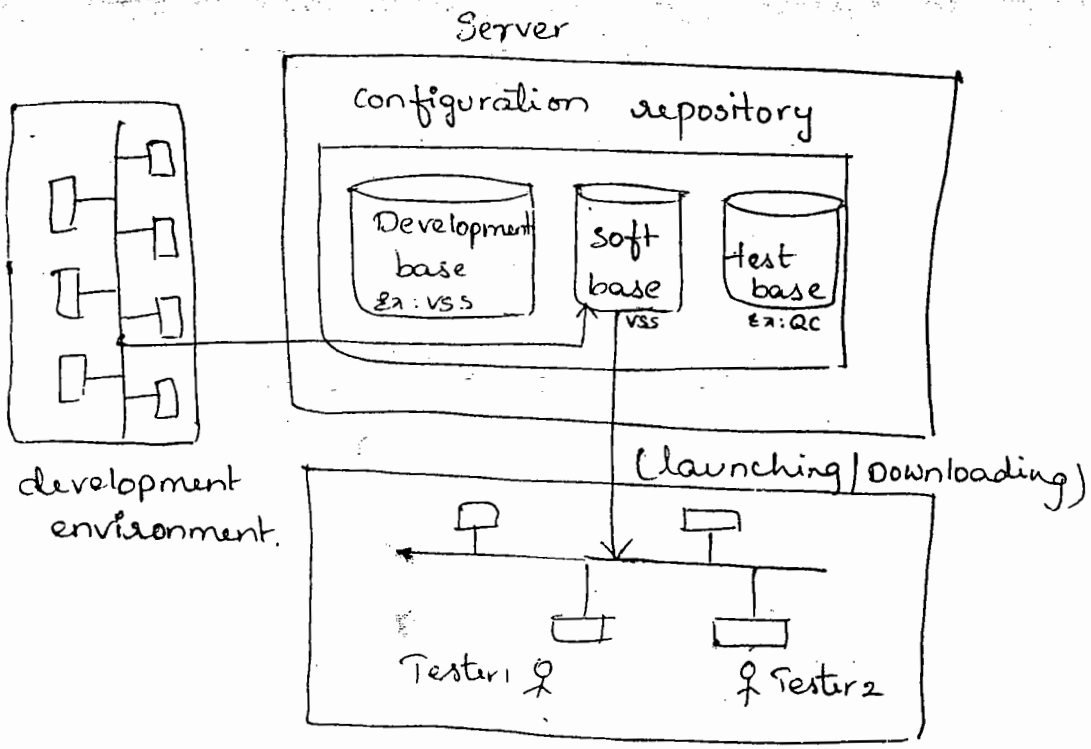
→ A test case must be approved by customers site, when we are in application development.

→ A test case must be hold a defect id when the test case was failed on SUT.

IV: Test execution

After Completion of test design & review, Project management is concentration on test, execution to detect defects in SUT. In this stage project management Conducts a formal meeting in between development & testing teams.

a) Formal meeting:- In this meeting developers & testers are Concentrating on SUT version Control & defect tracking process. In SUT version Control process, developers are releasing modified S/W with understandable version numbers.



Test environment.

From the above diagram, Project management maintaing sharable location or folder in server, called as Configuration Repository (CR This folder divided into 3 sub folders. Such as development base, Softbase & test base. Here development base consists of documents related to development like BRS, SRS, HLD, LLDs and other development documents. This development base is maintaing by developers & accessing by testers.

Soft base consists of versions of SUT. This soft base is maintaing by developers and accessing by testers.

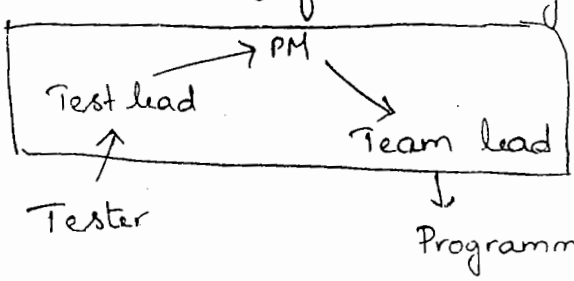
Test base consists of documents related to testing such as test strategy, test plan, test scenarios, test cases, automatic scripts, defect reports & other testing related documents. This test base is maintaing by testers & accessble by developers.

To provide security and to maintain information related to development base and soft base, developers are using visual source safe (VSS).

To maintain security and to get more information related to test base, testing team is using test management tools.

Ex: Quality Center (QC) of HP.

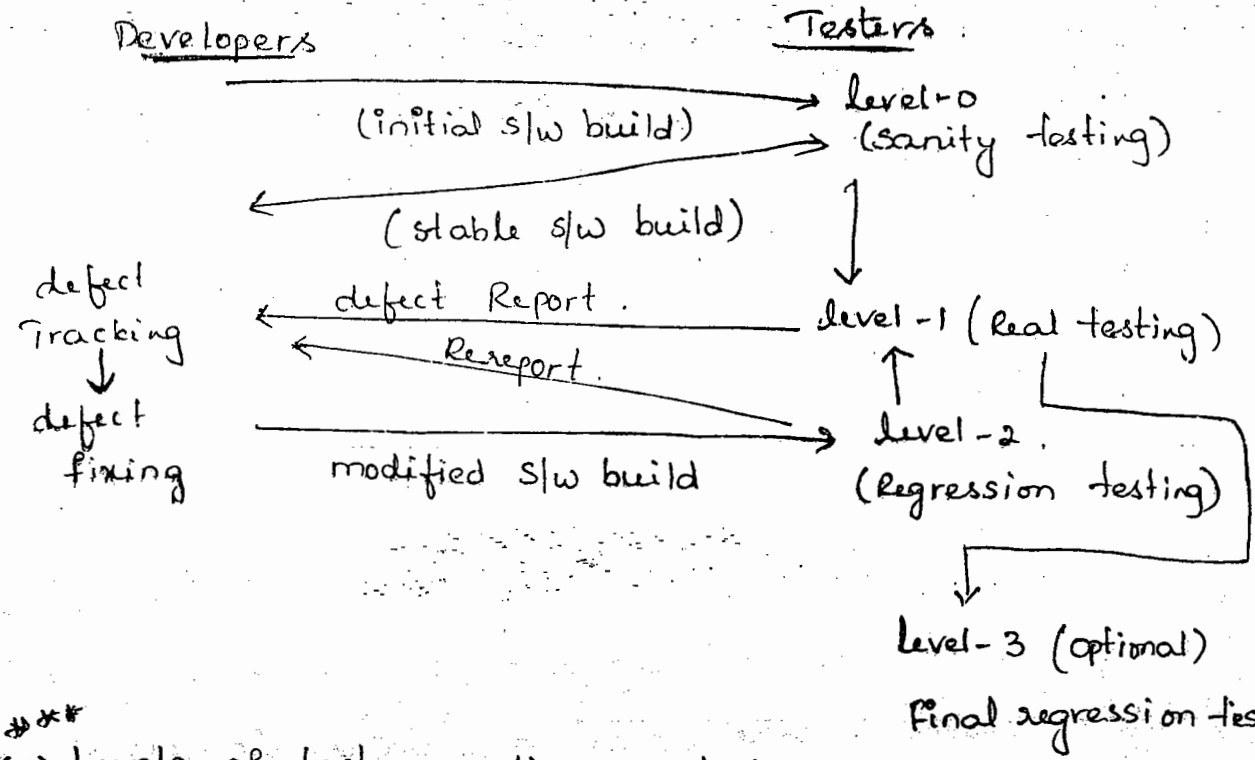
While testing, testing team is downloading or launching SUT from Soft base while testing. If they got any defect in SUT, then testers can prepare defect report and store it in test base and then defect tracking process is starting.



defect tracking team.

b) Test execution levels :-

After completion of formal meeting with developers, testing team is defining levels of test execution.



***

(c) Levels of test execution Vs test cases :-

level-0 (Sanity testing) :- Run test cases related to basic functionality of SUT

level-1 (Real testing) :- Run all test cases on SUT versions.

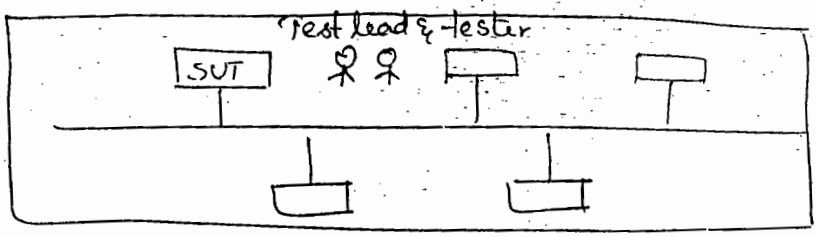
level-2 (Regression testing) :- Run all related test cases w.r.t modifications in SUT

level-3 (final regression testing) :- Run test cases related to high defect density modules in SUT

d) Level-0 (Sanity testing) :-

After Completion of s/w integration & integration testing developers can store first version of SUT in soft base. The testing team is downloading or launching SUT in test environment. And then testing team will Conduct

Sanity testing → to estimate testability of SUT.



Test Environment.

→ Understandable: (SUT is understandable to tester)
→ operatable : operatable SUT without runtime errors.
→ observable : SUT was observable by testers while doing internal process.
→ Controllable : Tester is able to do and undo operations in SUT.
→ Consistency : SUT have consistent navigation while operating repeatedly
→ Simplicity : No complexity in SUT ex: short navigation.
→ Maintainable :- No need for reinstallation.
→ Automatable :- Able to automate test cases on SUT by using available testing tools.

The above 8 factors combination is called as testability. By estimating above 8 factors, testing team is concentrating on real or comprehensive testing. If SUT is not testable, testing team will wait for testable build from developers. This 8 factors checking on SUT is called as sanity testing or testability testing or tester acceptance testing or build verification testing or octangle testing.

e) Level-1 (Real or comprehensive testing) :-

Step 1: Receive stable or testable SUT from developers after sanity testing.

Step 2: Take previously prepared and reviewed test cases.

Step 3: Make related test cases as batches (Test batches are

Step 4 :- Take one test batch.

Step 5 :- Take one test case in that batch.

Step 6 :- Take one step in that case.

Step 7 :- Compare expected value in test case with actual value of SUT. If both are same, go to step 6. until All steps in the test case was passed. If expected is not equal to actual, then tester can stop execution and Prepare defect report, send to defect tracking team

Step 8 :- Goto step 5 until all steps in all test cases was Passed.

Step 9 : Goto step 4 until all steps in all cases in all batches was Passed.

Testing team can follows above process in real or Comprehensive testing. In this process each tester can prepare test log document in daily bases.

Ex:

Tester : Anjali
Date : 2nd March-2011
SUT version : 1.0

Test batch id	Test case id	Result	Comments
1	TCD-1	Passed	—
	TCD-2	Passed	—
	TCD-3	Failed	Send defect report
	TCD-4	Blocked	TCD-4 depends on TCD-3
⋮	⋮	⋮	⋮

Passed - Executed and all steps expected values equal to actual values of SUT

failed - Executed and any step expected value is not equal to actual value

Blocked :- Yet to execute, because related test cases was failed.

(f) Defect report format :- (IEEE 829)

If any expected value of test case is not equals to actual value of SUT, then corresponding test engineer will Prepare defect report in IEEE 829 format.

1. Defect id :- Unique no. or name
2. Defect description : summery of detected defect.
3. SUT version : Version no. of SUT. [tester detected defect in this version of SUT]
4. Feature|module : Name of module [Tester detected defect in that module]
5. Testcase id : Id of failed test case [Tester detected defect while executing this test case].
6. Reproducable : Yes / No → Defect not appear every time
   ↳ Defect appears every time when teste repeated test case.
7. If Yes, attach test procedure :-
8. If No, attach test procedure and screen shots.
9. Severity :- The seriousness of defect w.r.t functionality of SUT.

   High (showstopper) → Not able to perform further testing until fixing this defect.

   Medium → able to countinue further testing, but mandatory to fix this defect.

   Low → able to countinue further testing, but may or may not to fix this defect.

10. Priority :- Importance of defect w.r.t customer.

    High ; medium, low

11. Test Environment : Used H/w & S/w while detecting this defect.

12: Status :- New or Reopen → Reopening.

↳ Sending defect report first time.

13. Reporting By : Name of tester.

14. Reporting on : Date

15. Send to : Defect tracking team.

16. Reviewed by : Test lead name.

17. Suggested fix (optional) : The suggestion to fix this defect After accepting by tracking team.

g) Error, defect and bug :-

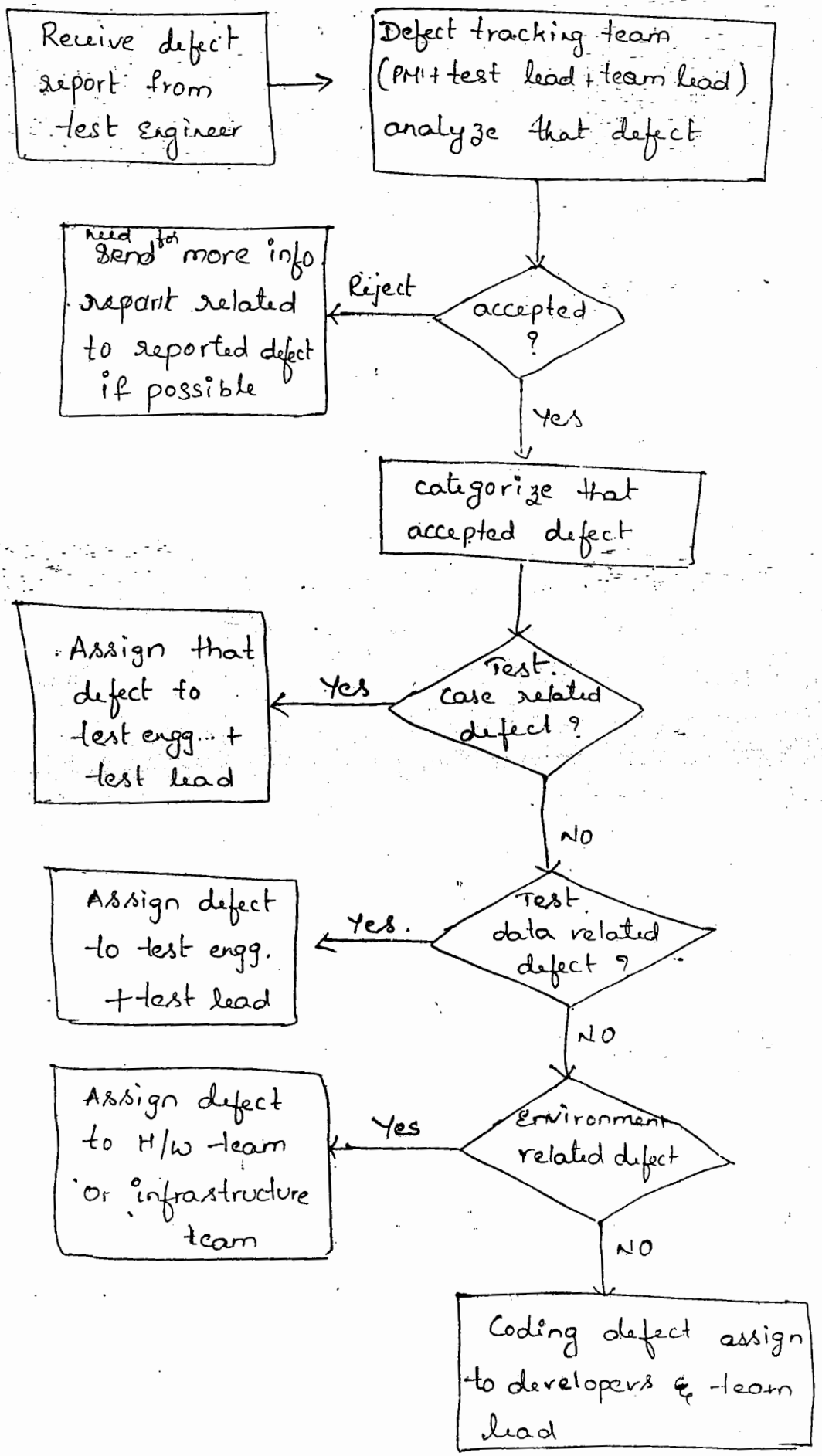
This 3 are synonyms. An error is a mistake in coding found by developers.

A defect is a mismatch in between expected value and actual value of a software found by testers. Some times defect is also known as issue or flaw.

A bug is a problem in S/w during utilization, found by customers or users.

h) Defect tracking process :-

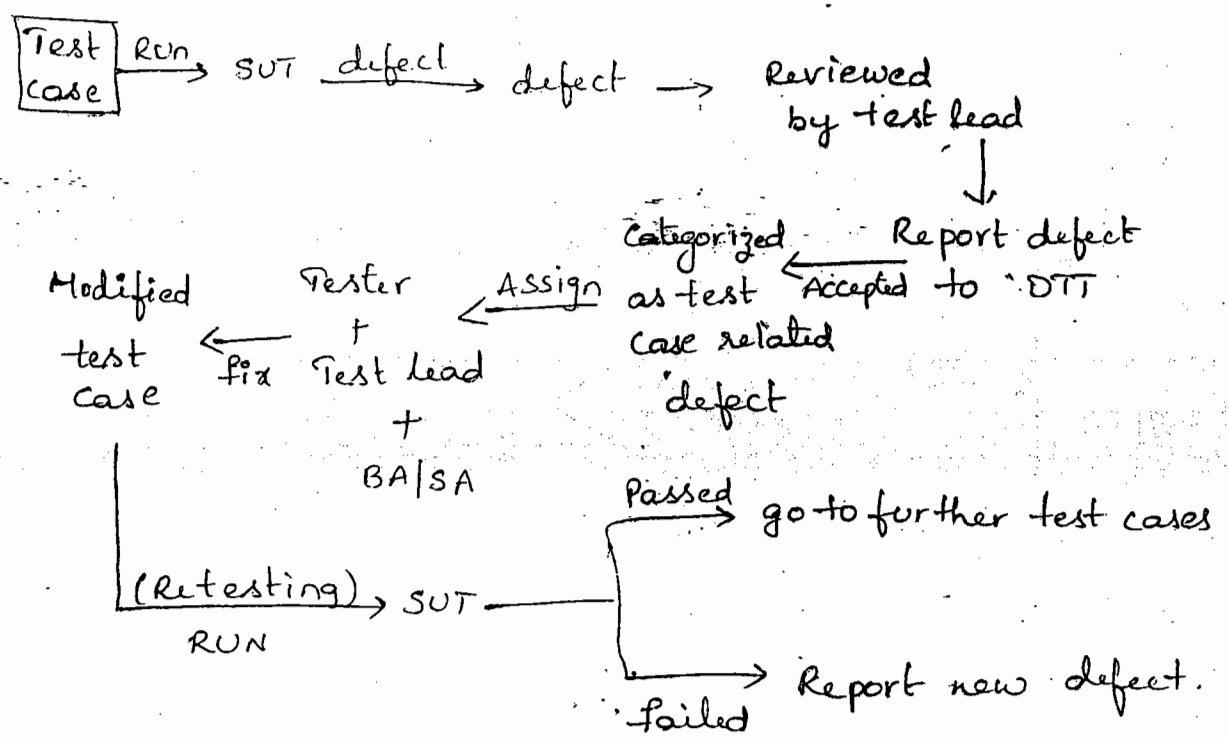
After preparing defect report, corresponding test engineer can get approval from test lead. to forward to defect tracking team. After receiving defect report from test engineer corresponding defect tracking team will follow below process.


Receive defect report from test engineer
Defect tracking team (PM1 + test lead + team lead) analyze that defect
accepted ?
Reject
Need Send for more info report related to reported defect if possible
Yes
categorize that accepted defect
Test case related defect ?
Yes
Assign that defect to test engg... + test lead
NO
Test data related defect ?
Yes.
Assign defect to test engg. + test lead
NO
Environment related defect
Yes
Assign defect to H/w team or infrastructure team
NO
Coding defect assign to developers & team lead

From the above process an accepted defect categorized into any one of four types such as test case related, test data related and environment related & S/w coding related defects.

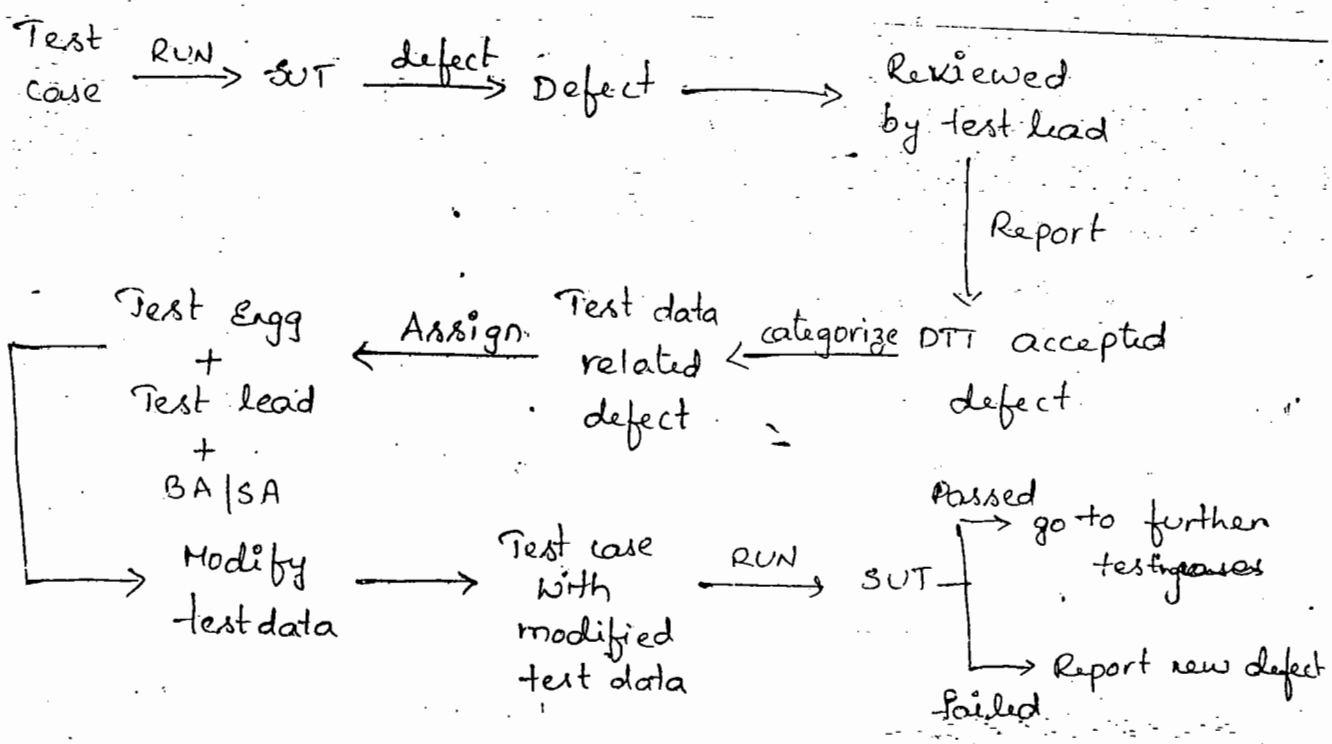
1) Test case related defects fixing

If a reported defect was accepted & categorized as test case related defect, then corresponding test engineer can take the help of test lead & business analyst & system analyst to rewrite correct test case.



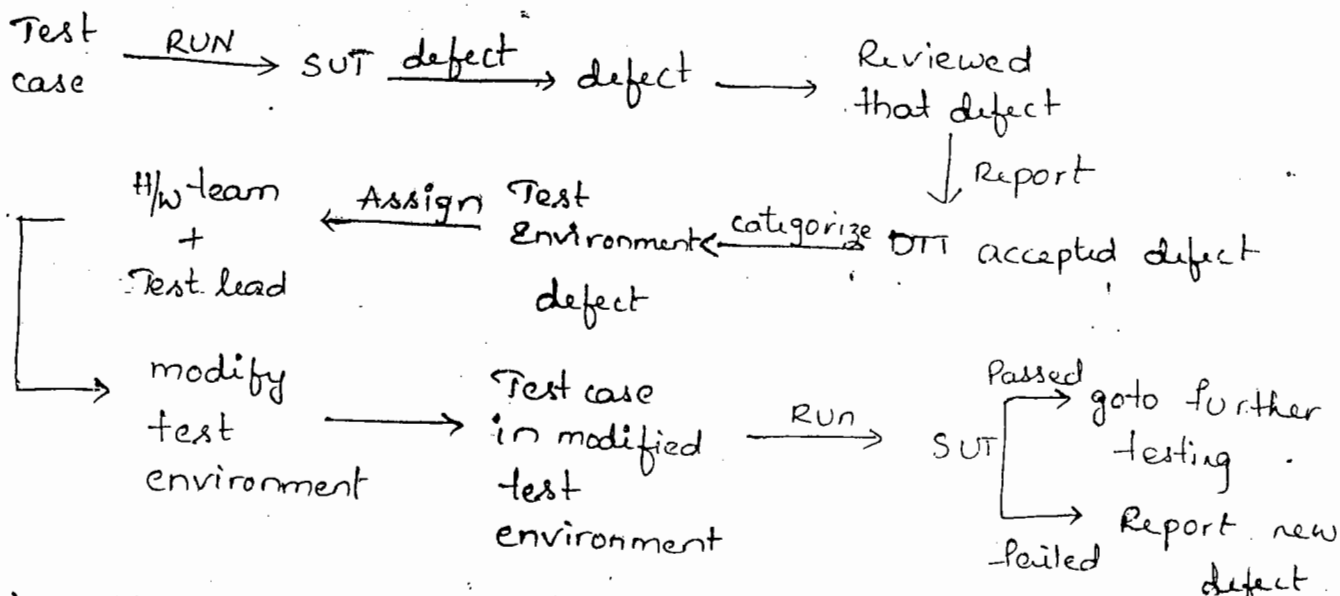
2) Test data related defect fixing

If an accepted defect was categorized as test data related defect, then the corresponding test engineer can take the help of test lead & BA & SA to get correct test data.



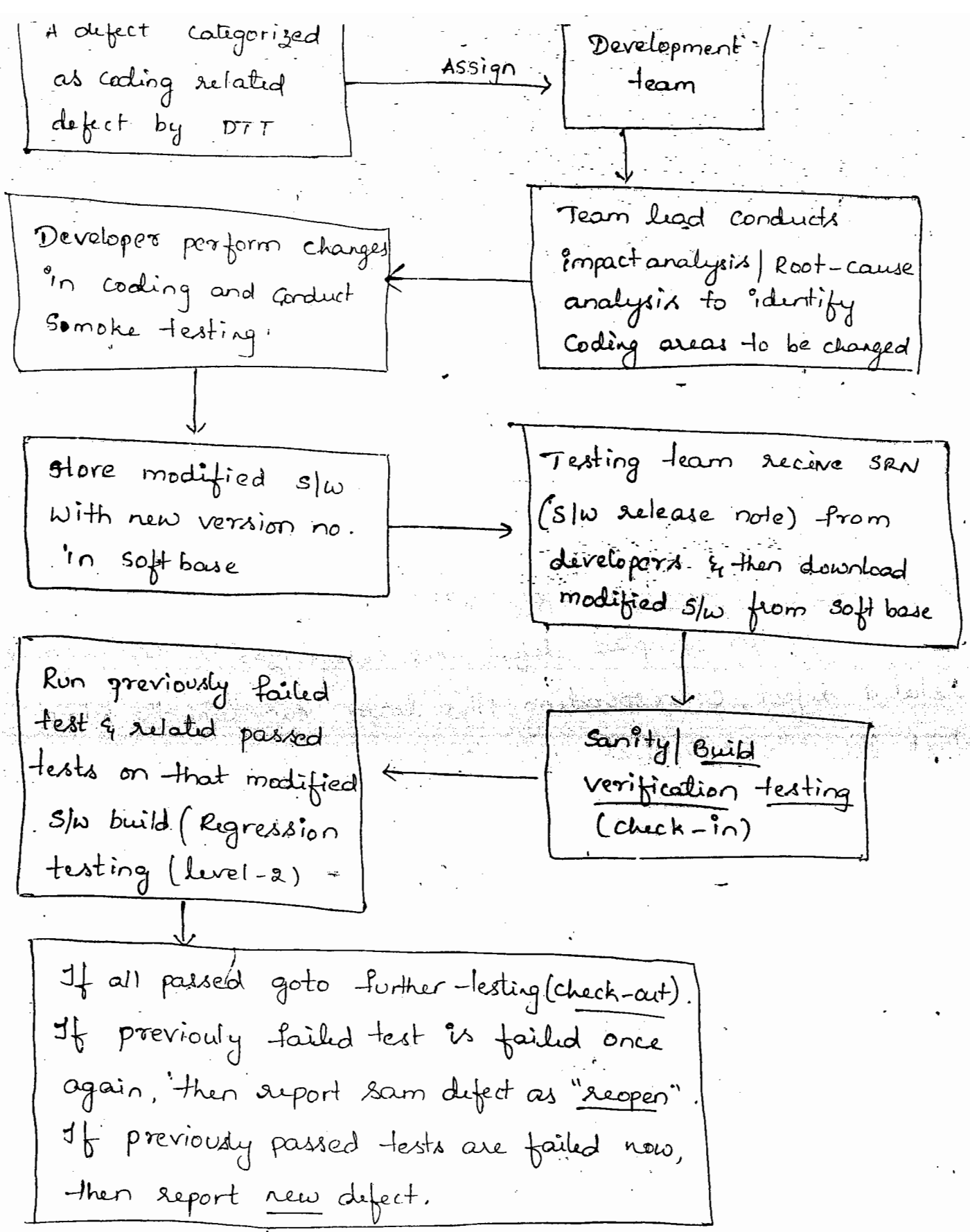
k) Environment related defect fixing

If an accepted defect was categorized as environment related defect, corresponding H/w team can take the help of test lead to establish correct test environment.



l) Coding related defect fixing:-

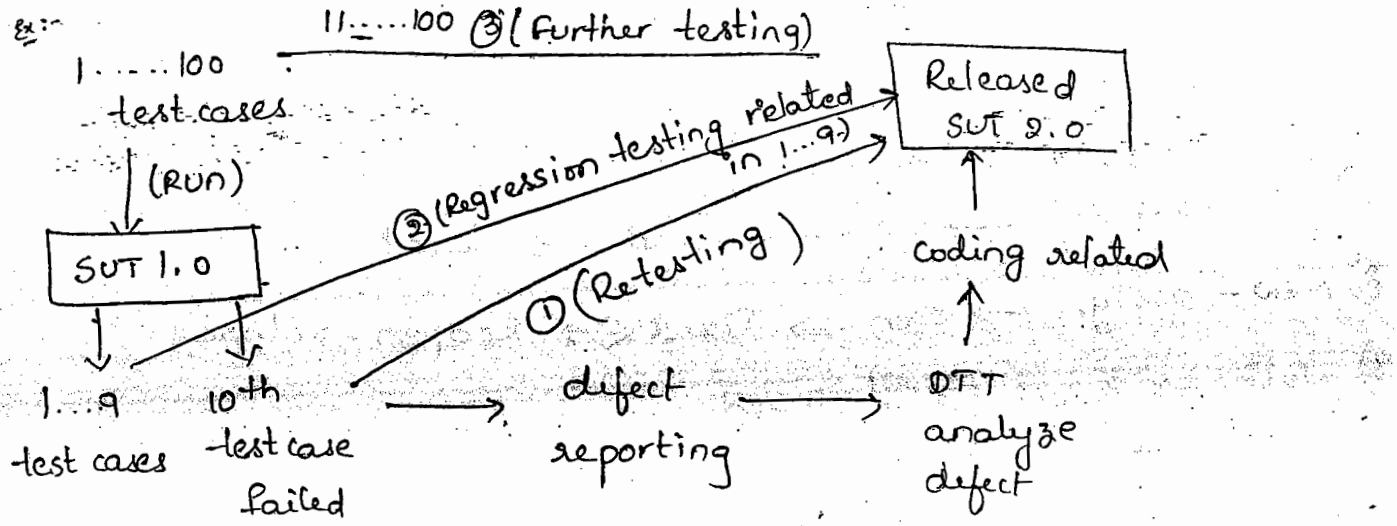
If an accepted defect categorized as coding related defect then corresponding development team will concentrate on changes in coding.


A defect categorized as coding related defect by DTT
Assign
Development team
Team lead conducts impact analysis / Root-cause analysis to identify coding areas to be changed
Developer perform changes in coding and Conduct Smoke testing
Store modified s/w with new version no. in soft base
Testing team receive SRN (S/W release note) from developers & then download modified S/W from soft base
Sanity / Build verification testing (check-in)
Run previously failed test & related passed tests on that modified S/W build (Regression testing (level-2)
If all passed goto further-testing (check-out).
If previouly failed test is failed once again, then report Sam defect as "reopen".
If previously passed tests are failed now, then report new defect.

M) Level-2 (Regression testing)

After fixing a coding related defect, testing team will receive SRN from developers. This SRN documents specifies detailes regarding changes in coding performed by developers. After study SRN, testing team can download or launch modified SUT in test environment. And then correspondi testers will conduct build verification test to estimate testability of that modified SUT [check-in].

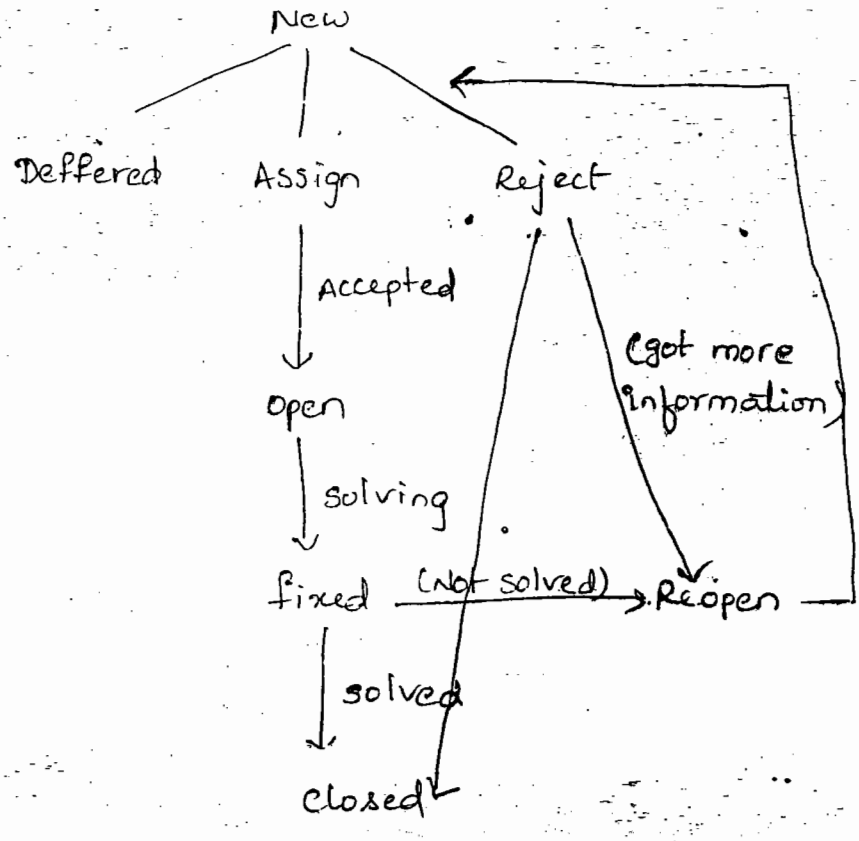
Ex:-



After check-in, corresponding tester can re-execute Previously failed test case on modified SUT to ensure Correctness of defect fixing. And then testers can re-execute all or related passed test cases once again on modified SUT to ensure disappearers of side effects. after fixing defect in SUT.

After completion of above all test cycles, testing team will Concentrate on "test closure" meeting.

N) Defect life cycle:-



1) New → assign → open → fixed → closed

2) New – assign → open → Fixed ↔ Reopen → closed

3) New → Reject —Agreed→ closed.

4) New → Reject —(got more information)→ Reopen → fixed → closed

5) New → deferred. (due-to low severity & low priority)

V) Test closure:-

While test execution, test engineers are detecting defects & developers are fixing those defects. Testers are receiving modified S/W and then conducts retesting and regression testing to close defects. The process of defects detection, defects tracking, defect fixing and defect closing in modified S/W is called a test cycle.

In general developers are releasing modified SUT immedia
When testers reported defect severity was high. If testers
reported defect severity was low or medium then develope
are releasing modified build on weekend in regular basies.
So one test cycle can indicates one or more defects fixing & closing.

Type of defect	Fixed by	closed by
Test case related defect	Corresponding test engg + test lead can modify test case	Run modified test case on SUT called as retesting
Test data related defect	Test engg + test lead can get correct test data	Run test case on SUT with modified test data, called as retesting.
Test environment related defect	H/w + test lead can provide Correct test environment	can provide Tester to run test case on SUT in modified test environment, called as retesting
coding related defect	By developers + team lead can modify Coding of SUT	Run previously failed test on modified SUT to ensure correctness of defect fixing, called as Retesting. Run previously Passed & related tests on modified SUT to estimate disappearens of side effects, called as regression testing

From the above table testing team is responsible to Confirm every defect fixing correctness and completeness. Due to this reason, re & regression testing also known as Confirmation testing or defects closing testing.

Note: In general testing teams will finish testing on 3rd in 8 cycles as minimum. Here each cycle duration is one week minimum (8 hr by 5 days) i.e 8×5)

After completion of all test cases execution and major defects fixing and closing [some defects are deferred], testing team will Conduct a review meeting to stop testing. This review is called as test closure. In this review meeting test lead will talk with test engineers to estimate Correctness & Completeness of testing process. In this review test lead is concentrating on below factors.

→ Coverage analysis:-

- → Modules / functionalities based coverage
- → Testing topics based coverage.

→ Defect density:-

Ex: Modules	No. of defects %	
A	30 %	
B	30 %	
C	40 %	↗ Need for final regression.
	100 %	

→ Analysis of deferred defects.

↳ Whether deferred defects are postponable or not?

After Completion of test closure review meeting, testing team is Concentrating on level-3 final regression testing. if time is ~~do~~ available in this final regression testing team follows below process.

→ Identify high defect density modules in SUT

↓

plan final regression testing

↓

Run test cases on identified modules in SUT

↓

Report golden defect after analysis if detected

↓

Developers fix golden defect and then testers can close defect if time is available.

Note :- If there is no time to fix golden defect then project management intimates to customer site people regarding those defects and then plan those defects fixing later by sending patc

(VI) User acceptance testing

After completion of final regression testing or postmortam testing or pre acceptance testing, project management is collecti feed back from customer site people on final SUT in any one of two ways. such as α-test and β-test.

If feed back is not good, then developers can perform changes in final SUT and testers can confirm those changes correctness and completeness by conducting regression testing.

VII : Sign-off :-

After completion of acceptance testing and their changes, test lead role-off all the testers in testing team. In this team, few testers selected to release team to release S/W and selected to CCB (Maint. for future maintance of S/W. The remaining testers will wait on the bench for next project testing.

After testers role-off, test lead can prepare Traceability matrix (RTM). This matrix defines mapping in between modules and defects via test cases.

Requirement module	Test cases	Results	defect id	closed/ deferred	Comment
Login	TCD-1	Passed	-	—	—
	TCD-2	Passed	-	—	—
	TCD-3	Failed	D_1	closed	—
	TCD-4	Passed	-	—	—

Test lead can submit RTH to PM to role-off from current project.

Case Study-1 [Exhaustive or optimal or ad-hoc testing]

From testing principles exhaustive testing is impossible. Due to this reason all testing teams are planning optimal or formal testing. [Previously discussed testing process in note book]. But in rare cases testing teams are following ad-hoc testing or informal testing due to risks.

There are different types in ad-hoc testing.

1. Monkey testing: Due to lack of time, testing team is executing test cases on main functionalities of SUT only. This type of test execution is called monkey testing.
2. Buddy testing:- Due to lack of time the project management is defining groups with developers & testers. They are conducting developement & testing parallely. This type of testing called as buddy testing.
3. Exploratory testing:- Due to lack of documentation on requirements of project, testing team is depending on

available documents, discussions with developers, discussion with other people in project, video conferencing with customer site people, browsing similar websites in internet & operating old versions of current project if available to get complete information related to project requirement And then testing team will conduct testing on SUT. This type of testing called as exploratory testing/artistic testing.

4) Pair testing:- To share knowledge on testing, senior testers are grouping with junior testers in testing. This type of testing is called as pair testing.

5) Agile testing:- Due to sudden changes in customer requirements while testing, testing team is changing test plan and test cases w.r.t changes in requirements. This type of customer involved testing is called as agile testing.

6) Bebugging:- To estimate effectiveness & efficiency of testers, development team is releasing SUT with known defects. This type of testing is called as bebugging | defect seeding.

case study-2 (Common responsibilities of test engineer)

→ Study SRS to understand customer requirements.
→ Assist with test lead to understand test strategy prepared by PM.
→ Assist with test lead in test plan preparation.
→ Develop high level test scenarios.
→ Implement test scenarios as test cases with detailed test procedure

→ Understand and gather required test data.
→ Execute responsible test cases on SUT versions.
→ Report defects to defect tracking team.
→ Involved in tracking process of defects.
→ Involved in re & regression testing to close defects after fixing.
→ Assist the test lead with his duties.
→ Automated test cases by using available testing tools (optional)
→ Knowledge on sql. to conduct database testing.

CASE-STUDY-3

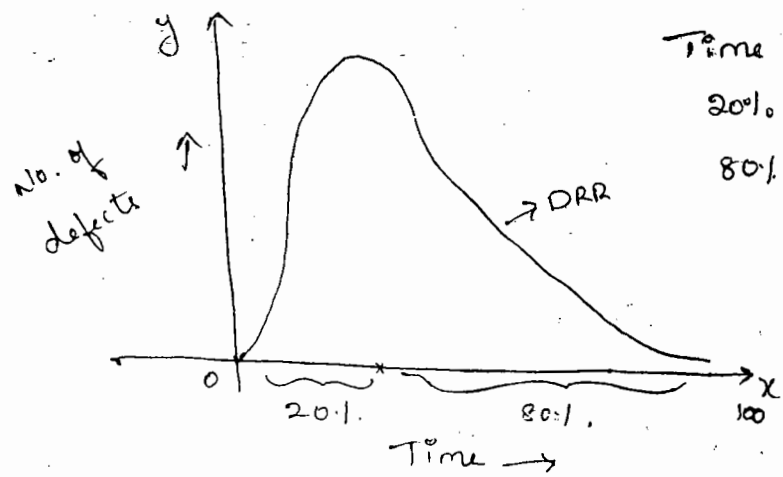
Testing measurements & metrics

To finish testing process in right time & to improve testing process project to project, project management is defining a set of measurments & metrics [Quality assurance]. These measurements classified into 3 categories...

1. Quality assessment measurements.

These measurements used by project manager or test manager during testing in every review meeting.

a) Defect removal rate:-



Time	No. of defects
20%	80%
80%	20%

b) Sufficiency:- The available remaining time from test plan is sufficient to finish complete testing.

c) Defect priorities allocation:-

While testing, test engineers are estimating priority of a defect w.r.t importance of customer and our organization trend in market.

Note:- Quality assessment reviews are going on twice in a month & conducting by PM or TM

2) Test management measurments:-

These measurements are used by test lead during testing. [weekly once].

a) Test status:-

- No. of test cases executed
- No. of test cases in execution
- No. of test cases yet to execute.

b) Quality gap:- or pending defects:-

Pending defects = [No. of defects detected and reported - No. of defects fixed)

c) Tester efficiency:-

- No. of test cases prepared per day.
- No. of test cases executed per day.
- No. of test cases defects detected & reported per day.

3) Testing process capability improvement measurements

These measurements used by project management to improve existing testing team capabilities depends on previous

4) Defect removal efficiency: - (DRE)

$$DRE = \frac{A}{A+B}$$

Here A = No. of defects detected by testers during testing.

B = No. of defects faced by end users in customer site.

→ (≥0.8 → Good testing team)

(0.8 to 0.7 → average efficiency)

(<0.7 → bad testing team)

5) Test effectiveness: -

- → No. of test cases prepared per day in previous project and No. of test cases prepared per day in current project.
- → No. of test cases executed per day in previous project and No. of test cases executed per day in current project
- → No of defects detected and reported per day in previous project & No. of defects detected and reported per day in current project.

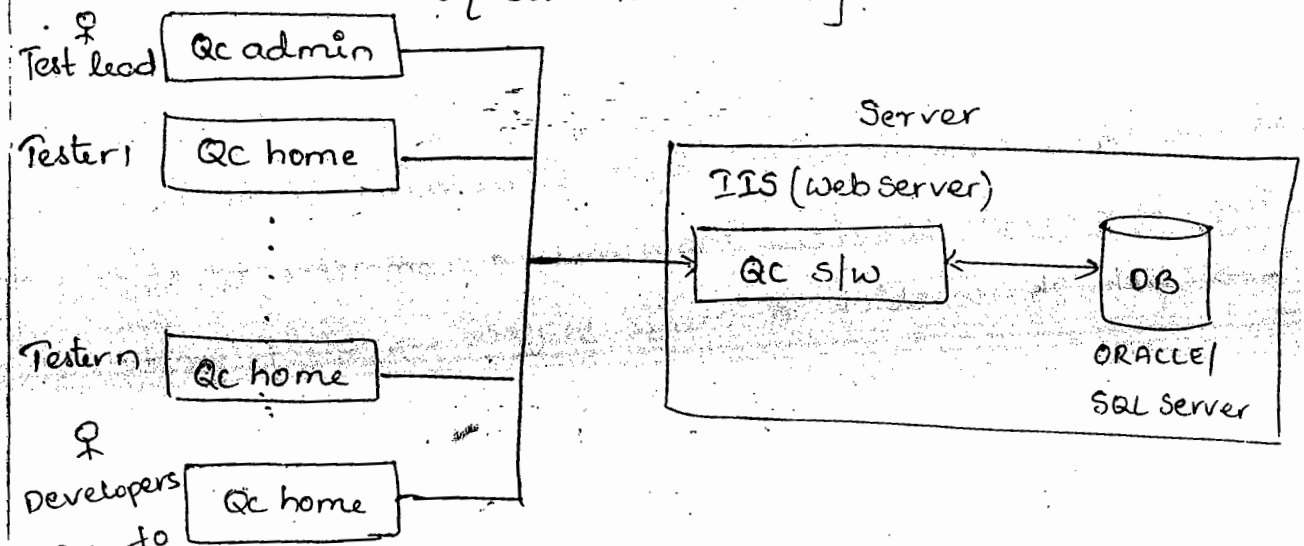
Quality Center 10.0

→ Developed by Mercury Intractive and take over by HP.

→ Released in 2009 January.

→ Test management tool to create and maintain test base of a project. To create and maintain development base & Soft base, developers are using VSS like management tool.

→ Quality center tool is working as website in a company network of testers [Test environment].



From the above architecture Quality center s/w is accessble to PM, test lead, testers & dev responsible developers. Here test lead is working as Quality center administrator. Due to this reason, Qaality center is providing 2 interfaces such as Qc admin & Qc home. Here test lead is only responsible Using Qc admin & others are using Qc home.

After installing Qc in server computer, the corresponding People will access Qc in client computers by using Qc Explorer browser. or other browsers like IE.

The URL for Qcadmin page is http://localhost:8080/qcbin

→ The URL of Qchome page is http://localhost:8080/qcbin

1) Test lead responsibilities in QC:

a) Creating users

Test lead open admin page → login → site users → click add user → enter new user details → click ok → click Password → enter & retype password → click ok → follow above navigation to add more new users.

b) Creating test base for new project testing

Open Qcadmin page → login → site projects → click create domain → enter current project type as domain name (ex: banking, financial services, insurance, e-commerce, telecommuni....) → click ok → click on create project → select empty Project option → click next → enter project name → click next → Select database technology (oracle, SQL server) → click next → Select administrators for current project (By default test lead is administrator) → click next → click create after confirmation.

c) Mapping users and projects:-

Open qchome → Enter user name & pwd → click authenticate → Select domain & project name → click login → click close in welcome screen → Tools menu → customize → project users → add users → add user from list → click ok → select user name → click ok → select responsibility as tester or developer or PM or admin → follow above navigation to add all employees to current project

2bir

2) Tester responsibilities in QC

After Completion of empty test base creation and mapping of users to that project with different roles, corresponding test lead directs testers, developers & project manager to access that test base. To access test base, testers and developers including PM can follow below navigation.

Open QcExplorer → browse Qcbin address → click go → Enter user name and password → click authentiicate → Select domain name and responsible project name → click login → click close in welcome screen →

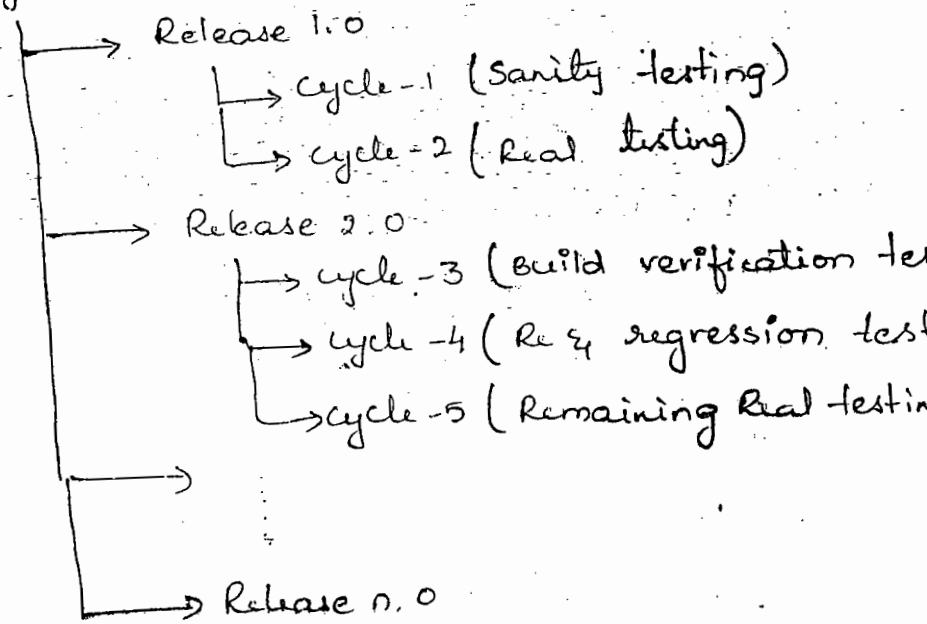
After login to specific project, corresponding user will work with below components in Qc.

a) Management :- Tester can use this Component to store information related to SUT versions and test execution cycles by following below navigation.

Select management in Qc → click releases menu → New release folder → enter project name as folder name → releases menu → new release → enter version number → click ok → releases menu. → new cycle → enter cycle name → click ok → follow above navigation to create required cycles.

Ex:

Flight reservation



b) Requirements :- This component is also used by testers with help of test lead to store responsible modules information by following below navigation.

Select requirements in QC → requirements menu → new folder → Enter project name as folder name → click ok → requirements menu → new requirement → enter responsible module name → click ok → follow above navigation to create more requirements.

c) Test plan :- This component is also used by testers to Prepare test scenarios and test cases. by following below navigation

Select test plan → tests menu → new folder → enter testing topic as folder name [Functional, usability, compatability, Performance, configuration testing....] → click ok → tests menu → new test → select test type as manual → Enter test scenario or test case name → click ok → select details tab →

Write required detailes for a test case [Test batch id, Test setup, test environment.... etc] → select design steps → click new step ico → Enter step description with expected result → click new step icon once icon to create more steps if needed → click attachments to attach related files if available → click requirements coverage → Select Corresponding module → follow above navigation to write more test cases for all responsible modules and testing topics.

d) Test lab:- This Component is used by testers to execute test cases on SUT.

Select test lab → Test sets menu → new folder → Enter Conducting testing topic as folder name → click ok → test sets menu → new test set → Enter batch name [module name] → click ok → Select test icon → Select related tests one by one into batch [doubl click] → follow this navigation to create more batches → Select a batch → Select a case in that batch → click Run icon → click begin Run → Select one step in case → Operate SUT and then compar Step expected value with project actual value → Specify step result Passed or failed → ~~Specify~~ follow above navigation until all steps are passed in a test case → click end Run icon → following above navigation for all test cases in all test batches.

Note: If step was failed, then tester can stop execution to report defect.

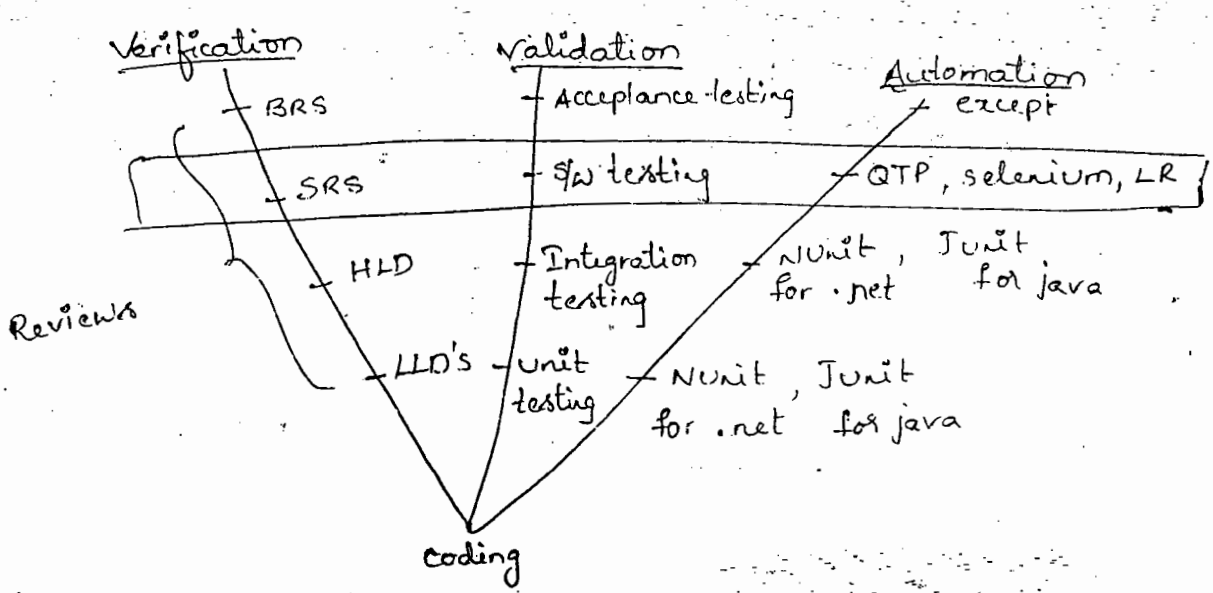
e) Defects:-

Defects menu → new defect → fill defect report → click submit → click close.

Note: In general tester can assign a new defect to the Lead after detection. Test lead will assign that defect to PM after review. PM can assign the defect to defelopers after accepting. Developer can change status to open while fixing the defect. They will change status to fixed after fixing. Tester can change status to "closed" after successful re & regression testing. If re & regression was failed then tester can change status to "reopen".

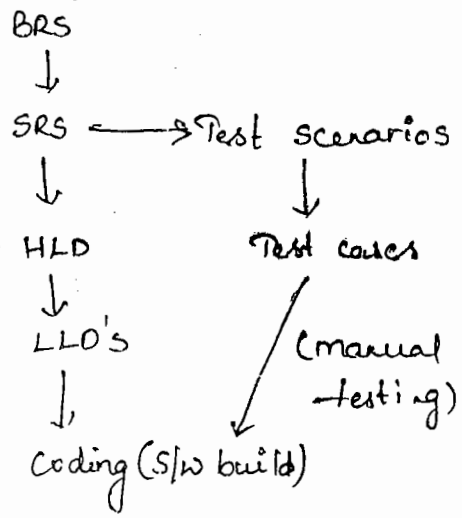
S/W testing or System testing automation

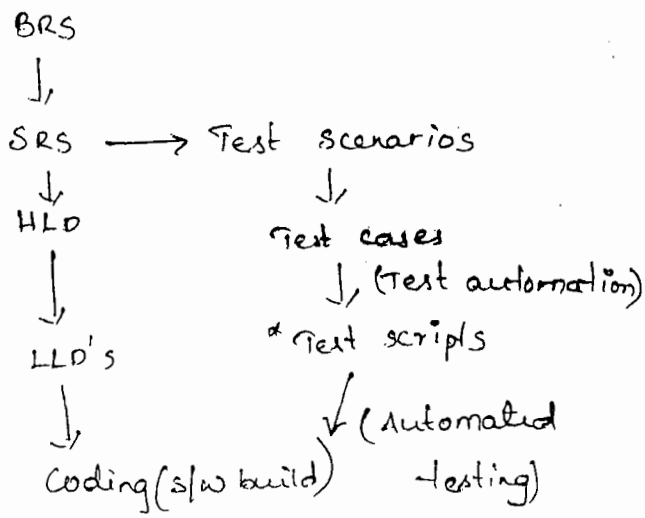
V - model



Manual testing Vs Automation testing

In general testing teams are involving in system testing or s/w testing stage. In this stage testing team is conducting functional testing on a s/w to check customer requirement nothingbut functionalities or features. They are conducting non-functional testing to check a s/w w.r.t customer exceptations. like usability, performance, compatability, configurations, inter system... etc While testing on a s/w, testing team follows two styles.





Need for test automation in system or S/W testing stage

System testing topics	Level-0 (Sanity/BVT)	Level-1 (Real)	Level-2 Regression	Level-3 (Final regression)
Functional testing	manually	manually	automation	Automation
Performance testing	×	Automation	Automation	Automation
other non-functional testing	×	manually	manually	manually

From the above table, testing team will go for automation in regression levels of functional testing. They will go to automation in all levels of performance testing. But other non-functional tests like usability testing, Compatability testing, configuration testing ... etc are testable manually only, because they are easy to conduct, not expensive and take less time. One more reason is that there is no well known tools for those testing topics.

QTP (QuickTest Professional) 10.0 :-

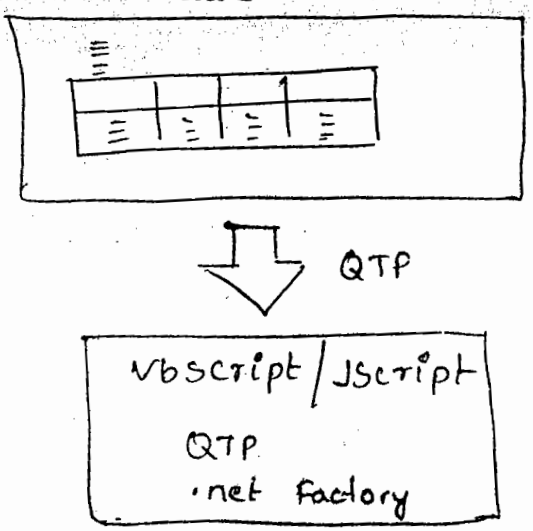
→ Developed by Mercury Interactive take over by HP.

→ Used for functional test automation at regression levels

→ Run on windows operating system. client & sever versions.

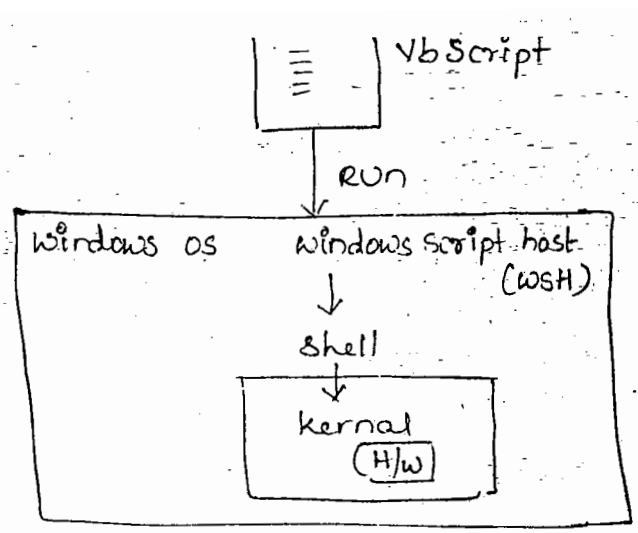
→ QTP supports functional test automation on a s/w, which screen developed in VB, .net, Java, SAP, sibel, Peoplesoft, Oraclefinacials, Delphi, HTML (web), web services, stingrey, Powerbuild standard windows, ActiveX, VisualAge and terminal Emulators (to test C, C++, VC++ and Ibm mainframes).

→ To convert manual test cases into automated test scripts, we can use below technologies in QTP.



Vbscript :-

It is a light-weight language. This language is in-built in windows operating system.



Like as programming languages vbscript also consists of variables and constants. A variable or constant is used to hold a value. but variable is changing that value. To declare a variable in vbScript we can follow below syntax.

```
option explicit
dim x, y, z
```

ex

From the above syntax x, y & z variables can allow any type of value. This is called as "varient" type.

Note: → VbScript is not case sensitive language.

→ There is no delimeter at end of any statement in vbscript.

→ In general vbScript programs saved with filename.vbs

→ vbscript is interpreted language [compiling & running at a time]

Ex1:

```
option explicit
dim x,y,z
x=10
y=20
z=x*y
msgbox z
```

To declare constants, we can follow below syntax

```
option x, y, z
x=10
y=20
Const z=30
```

Note:→ In general QTP testers are using lowercase names for variables and uppercase names for constants.

→ To write comments, we can use singlequote at start of comment.

→ If you lanch QTP Tool without selecting any add-in then QTP will work as editor for vbscript

operators in vbscript:-

a) Arithmetic operators:-

Exponent (^), *, /, mod, +, -,

b) Assignment operator:-

=

c) Comparision operators:-

<, <=, >, >=, = (equal to), <> not equal

d) Logical operators

and, or, not

Conditional statements

a) If condition.

```
if condition then
    ≡
end if
```

b) If...else statement.

```
if condition then
    ≡
else
    ≡
end if
```

c) If .... elseif statement:-

Multi-Conditions checking

ex:

```
option explicit
dim x=10, y=20
if x>y then
  msgbox x
Else
  msgbox y
end if
```

```
If Condi 1 then
    ≡
elseif condi 2 then
    ≡
else
    ≡
end if
```

Ex: Write Vbscript code to display grade of a student w.r.t total marks. If total marks ≥ 800 then display grade as A. If total marks < 800 & ≥ 700 then display grade as B. If total marks < 700 & ≥ 600 then display grade as C. If total marks < 600 then display grade as D.

```
option explicit
dim total.
total = inputbox ("enter student total marks").
If total >= 800 then
    msgbox " Grade is A"
elseif total < 800 and >= 700
    msgbox " Grade is B "
elseif total < 700 and >= 600
    msgbox " Grade is C."
else
    msgbox " Grade is D"
Endif
```

Note: QTP testers are using inputbox function to read input from keyboard and msgbox function to display o/p while running programs.

Ex 2: Write Vbscript program to display gross salary of an employee w.r.t basic salary. If basic salary ≥ 15,000, then gross salary is basic salary + 10% basic salary. If basic salary < 15000 and ≥ 8000, then gross salary is basic salary + 5% of basic salary. If basic salary < 8000 then gross salary is basic salary + 200.

```
option explicit
dim gross, basic
basic = inputbox ("Enter employees basic salary")
if basic >= 15000 then
    gross = basic + 10/100 * basic
    msgbox &gross.
elseif basic < 15000 and >= 8000 then
    gross = basic + 5/100 * basic
    msgbox &gross.
    else
    gross = basic + 200
    msg box &gross
    elseif
```

Ex 3: Write vbscript to display commission or interest on fixed deposits in a bank w.r.t deposited amount and time. If time ≥ 12 months, then commission is 10% of amount for each month. If time < 12 months and ≥ 6 months, then commission is 8% of amount for each month. calculate total commission.

```
Option explicit
dim amount, t, tc, mc
amount = inputbox ("Enter amount");
t = inputbox ("Enter time in months")
if t >= 12 then
    mc = amount * 10/100
elseif t < 12 and >= 16 then
    mc = amount * 8/100
else
    mc = 0
End if
```

tc = mc * t

msgbox (" Total Commission is " &tc)

d) Select case statement :- It is alternative to if-elseif statement. But this statement is not depending on condition. It depends on direct value.

Ex 1 :-

```
option explicit
dim day-number
day-number = inputbox (" Enter day number");
Select case day-number
   Case 1
      msgbox " Monday"
   Case 2
      msgbox " Tuesday"
   Case 3
      msgbox " Wednesday"
   case 4
      msgbox " thursday"
   case 5
      msgbox " friday"
  case 6
      msgbox " saturday"
   case else
      msgbox "invalid day number"
end select
```

Ex 2:

```
option explicit
dim credit_card
credit_card = inputbox (" Enter your credit card type");
 Select case credit_card
   Case "visa"
      msgbox "Accepted"
   case " Master"
```

```
    msgbox "Accepted"
  case "American Express"
    msgbox "accepted"
  case else
    msgbox "Invalid card type"
  end select
```

Note: VbScript allows you to enclose characters or strings by using double quotes

Loop statements:- To execute a block of statements more than one time we can use loop statements such as while, do-while, do-until, for & for-each.

a) While loop:- We can use this loop to run specified block of code iteratively as long as condition was true.

```
While Condition
    ≡
Wend.
```

b) do-while:- we can use this loop to run specified block of statements iteratively as long as Condition was true.

```
do
  ≡
loop while Condition.
```

But block of statements will be executed atleast one time when Condition was false.

c) do-until loop:- we can use this loop to run specified block of co iteratively as long as condition was false.

```
do                  → this statement is reverse concep
  ≡                   to do-while statement.
loop until Condition
```

d) For loop:- If we [illegible] of
we can use for loop.

Ex1: for i=1 to 10 step 1 → increment i by 1

≡

next

Ex2: for i=1 to 10 step 2 → increment i by 2

≡

next

Ex 3: for i=10 to 1 step -1

≡

next

e) For-each loop:- we can use this loop to run specified block of statements for all values in a list or array.

for each i in array

≡

next.

ex: Write vbscript code to reverse a given number.

```
option explicit
dim num, r, rnum
num = inputbox("Enter number")
rnum = 0
While num > 0
   r = num mod 10
   rnum = rnum * 10 + r
   num = int(num / 10)
wend
msgbox "reverse number is " & rnum
```

10)123 (12
120
3

n%10

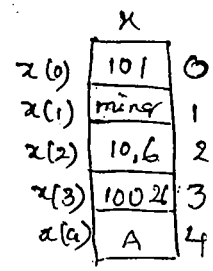
Note:- int() function returns intger part of given value.

Ex: int(10.7) = 10

Arrays:- Arrays are used to store more than one dissimil values.

Ex: option explicit

dim x(4)



In above diagram arrays index started with 0. An array size is last index + 1.

To store values in an array, we can follow below ways.

```
option explicit
 dim x
  x = array (101, "mindq", 10.6, "A")
```

(or)

```
option explicit
dim x(3), i
for i = 0 to 3 step 1
   x(i) = inputbox ("enter value")
next
```

Ex:1: Write vbscript program to read 5 subjects marks of a student and then display total marks.

```
option explit.
dim x(4), i, total
for i = 0 to 4 step 1
   x(i) = inputbox ("enter no subjet" &i& "marks")
next
```

```
total = 0
for i = 0 to 4 step 1
  total = total + x(i)
next
msgbox "total marks are :" & total
```

Ex 2:- Write vbscript program to read 5 subjects marks and then display highest marks in that subject.

By default vbscript can treat array values as strings due to this reason QTP testers are using type casting functions.

- cint() → convert to integer
- clng() → Convert to long integer
- cdbl() → Convert to double (float)
- cdate() → Convert to date
- cboolean() → Convert to boolean (true or false)

Ex:3 Write vbscript program to display count of even numbers and add numbers in given array.

```
option explicit
dim x(4), i, ecount, ocount
```

```
for i=0 to 4 step 1
  x(i) = inputbox ("enter number")
next
  ecount = 0
  ocount = 0
for i = 0 to 4 step 1
    if x(i) mod 2 = 0 then
      ecount = ecount + 1
    else
      ocount = ocount + 1
    End if
next
msgbox "even numbers count" & ecount
msgbox "odd numbers count" & ocount
```

Ex:4 Write vbScript program to find largest & smallest no. in an array

```
option explicit
dim x(4), i, max, min
for i = 0 to 4 step 1
  x(i) = inputbox ("enter numbers")
next
  min = max = x(0)
for i=0 to 4 step 1
    if max <= cint(x(i)) then
      max = x(i)
    end if
    if min >= cint(x(i)) then
      min = x(i)
    end if
  next
  msgbox "max. number is" & max
  msgbox "min. number is" & min
```

Note → In VBScript ∞ ... of ...

→ Vbscript allows you for dynamic allocation of arrays.

```
dim x()  'array declaration
≡
Redim x(10)  'Dynamic allocation
```

→ Vbscript allows you to memory deallocation for arrays.

```
erase x  ' memory deallocation
```

→ Vbscript allows you to allocate memory dynamically without disturbing existing values in an array.

```
dim x(4)  ' 5 elements array
≡
Redim preserve x(5)  ' 6 elements array without
                       disturbing previous 5 elements
```

→ Like as programming language, Vbscript also allows you to declare multi-dimensional arrays.

```
dim x(4)(4)  ' 5 rows & 5 columns
```

Dictionary Objects in Vbscript

Like in perl (Practical extraction & reporting language) Vbscript also allows you to create dictionary object. Each dictionary object allows you to store pairs of data. [keys & items]

keys

English	50
Telugu	60
Physics	35
maths	40
Chemistry	70

items.

In a dictionary keys must be unique. To add key-item pairs to a dictionary, we can follow below code.

```
option explicit
 Dim d
 Set d = createobject("Scripting.dictionary")
 d.Add "English", 66
 d.Add "Maths", 70
```

In above code we used Add() method to insert key/Item pair.

To access existing key/item pair from dictionary, we can follow below code.

```
x = d.keys ' get keys
y = d.Items ' get Items
```

Ex:1 Write Vbscript program to create a dictionary with subject names & marks as pairs & then calculate total marks for display

```
option explicit
Dim d, x, sum, i
Set d = createobject("Scripting.dictionary")
d.add "Maths", 67
d.Add "Maths", 67
d.Add "physics", 77
d.Add "chemistry", 97
d.Add "Telugu", 66
d.Add "English", 55
x = d.Items
Sum = 0
for each i in x
   Sum = Sum + i
next
msgbox "sum of marks is" &Sum
```

(OR)

```
option explicit
Dim d, n, s, m, x, sum, i
Set d = createobject ("scripting.Dictionary")
  n = inputbox ("enter no. of subjects")
  for i = 1 to n step 1
    s = inputbox ("Enter Subject name")
    m = inputbox ("Enter marks")
     d.Add s, m
  Next
  x = d.Items
   Sum = 0
  for each i in x
     sum = sum + i
  Next
   msgbox " sum of marks is " &sum
```

Ex 2: Write Vbscript program to create a dictionary with employee names & their salaries & then display highest salary employee name.

```
option explicit
Dim d, n, s, m, x, sum, i
Set d = createobject ("scripting.Dictionary")
   n = inputbox ("Enter no. of emp subjects")
 for i = 1 to n step 1
  s = inputbox ("enter emp subject name")
  m = inputbox ("enter salary marks")
   d.add s, m
Next
```

```
max = 0
x = d.keys
for each i in x
    if max < cint(d.Item(i)) then
        max = cint(d.Item(i))
        temp = i
    End if
Next
    msgbox "max. salary employee is" & temp
```

Note:→ To change existing key with new key we can use below code. d.key("old key") = "New key"

→ To change existing item in a dictionary, we can use below code

d.item("key") = "New Item"

→ To find no. of key/item pairs in a dictionary, we can use below code. n = d.count

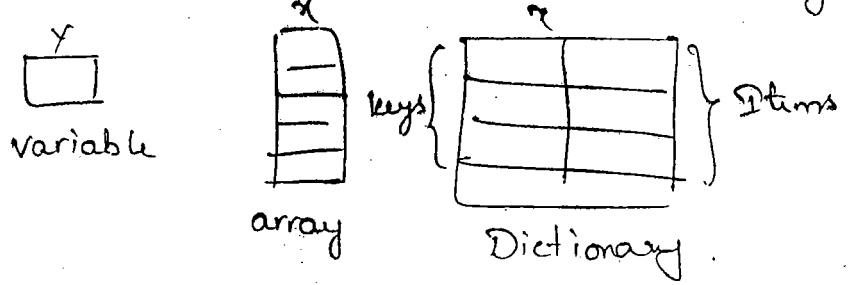
→ To remove key/item pairs from dictionary, we can use below code

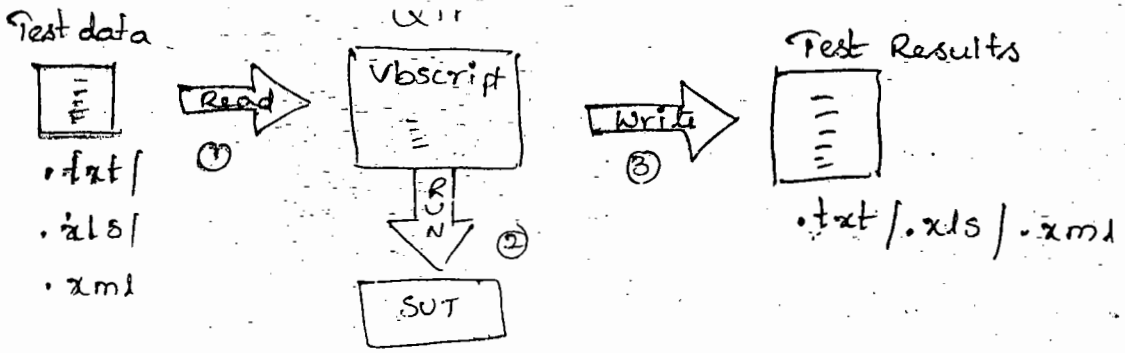
d.remove("key") 'specific pair removed

d.removeall 'All pairs removed but d alive

set d = nothing 'All pairs removed and d was dead

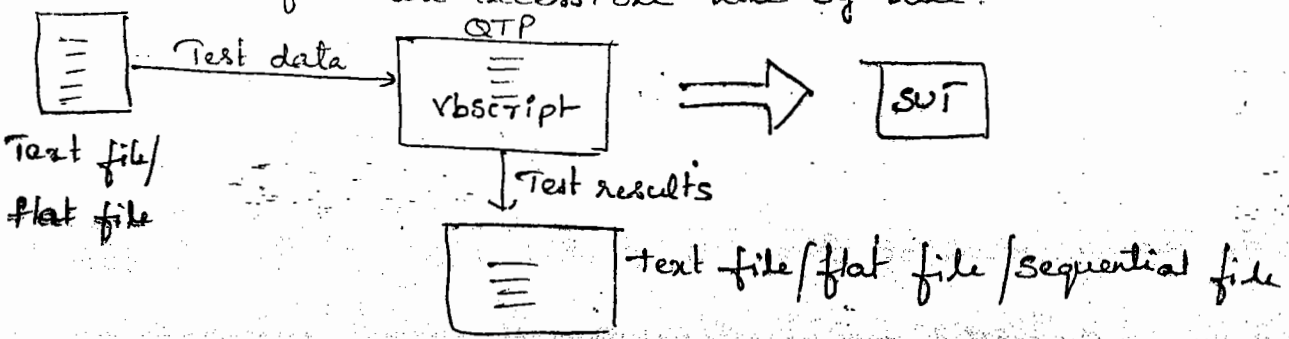
→ previously discussed variables, arrays & dictionaries are allowing you to store data related to testing but their life time is going end when our program execution was finished. Due to this reason testers can go to work with files





Working With Text files:-

To maintain test data & test results, test automators are using text files. Text files are also known as flat files/sequential files because these files are accessible line by line.



To work with a file in QTP, we can use below code in vbscript language.

```
option explicit
dim fso, fo
Set fso = createobject ("Scripting.filesystemobject")
Set fo = fso.OpenTextFile("Path of file", 1/2/8, true/false)
```

Read write append

true = create new file when file does not exist
false = No file creation when file does not exist

If you open a file in write mode then the existing data in a file before open will be replaced by new data. If you open a file in append mode, then the new content will be added to at the end of existing data in a file.

open text

OpenTextFile() :- We can use this file system object method to open a specified file in specified mode (1 - read, 2 - write, 8 - append)

AtEndOfStream :- we can use this file object property to know whether controller at end of file or not.

ReadLine() :- we can use this file object method to get current line of text from a file. This method moves controller to next line after getting current line.

SkipLine() :- we can use this file object method to go to next line without reading current line.

Read() :- we can use this file object method to get specified no. of characters in current line.

fo.read(5) 'To get 5 characters from the current line.

Skip() :- we can use this file object method to skip specified no. of characters in current line.

Ex: #my name is khan

fo.skip(1)

x = fo.ReadLine

msgbox x 'my name is khan

Ex1: Write vbscript program in QTP to display existing lines in a file.

C:\testdata.txt

Check de India
My name is Anjali
I am a tester

```
option explicit
dim fso, fo, x
Set fso = createobject("scripting.filesystemObject")
Set fo = fso.opentextfile("c:\testdata.txt", 1, false)
While fo.AtEndOfStream <>true
    x = fo.ReadLine
```

```
if
  x = fo.Read(1)
if x <> #
  x = fo.ReadLine
    msgbox x
  else
fo.skipline()
```

```
fo.close
Set fo = nothing    'destroy file object
Set fso = nothing   'destroy file system object.
```

Ex:2 Write vbscript program in QTP to display lines in a file, which does not start with #.

```
option explicit
dim fso, fo, x
set fso = CreatiObject("scripting.fileSystemobject")
Set fo = fso.Opentextfile("c:\testdata.txt", 1, false)
While fo.AtendOfStream <> true
    x = fo.Read(1)
    If x = "#" then
     fo.Skipline
    else
      msgbox x & fo.Readline
    EndIf
Wend
fo.close
Set fo = nothing
Set fso = nothing
```

Ex:3 Write vbscript program in QTP to display lines of a file, which consists 'mindq' string.

```
I am mindq statement
My name is anjali
Mindq is an institute
```

```
option explicit
dim fso, fo, x
set fso = createobject ("scripting. filesystemobject")
set fo = fso. opentextfile ("c:\testdata. txt", 1, false)
 while fo. AtEndofStream <> true
    x = fo. ReadLine
    If instr (x, "mindq") then
    msgbox x
   End If
 wend
fo.close
 set fo = nothing
 set fso = nothing.
```

Instr():- we can use this vbscript function to check the existance of substring in main string.

Instr (mainstring, substring)

Ex:4:- Write vbscript program in QTP to display lines of a file, which lines have numerics.

```
I am 100th mindq student
My name is khan
Mindq is an institute
check de India 165 times.
```

```
option explicit
dim fso, fo, x, r
set fso = createobject ("scripting. filesystemobject")
set fo = fso. opentextfile ("c:\testdata. txt", 1, false)
 set r = new regexg 'create regexp object
  r. pattern = "[0-9]+" specify regexp
 while fo. AtEndofStream <> true
```

```
x = fo.readline
If r.test(x) then
    msgbox x
End if
Wend
fo.close
Set r = nothing
Set fo = nothing
Set fso = nothing
```

Ex:5: Write vbscript program in QTP to read line by line from one file and then write into another file.

```
Option explicit
dim fso, fo1, fo2, x
set fso = createobject ("scripting.filesystemobject")
set fo1 = fso.opentextfile ("c:\testdata.txt",1,false)
set fo2 = fso.opentextfile ("c:\testdata1.txt",2,true)
While fo1.AtEndOfStream <> true
    x = fo1.readLine
    fo2.writeLine(x)  fo2.WriteBlankLines(2)
wend
    fo1.close
    fo2.close
    set fo1 = nothing
    set fo2 = nothing
    set fso = nothing.
```

Write():- we can use this file object method to write a specified text in a current line of file.

Writeline():- We can use this file object method to write specified text in a current line & then move controller to next new line.

WriteBlankLines():- We can use this file object method to write specified no.of blank lines and then move controller to next new line.

AtendOfLine:- we can use this file object property to know wheather controller at end of current line.

Vbscript allows you to work with folders also. In general every folder consists of subfolders & files. To work with folders, test automaters are using below code.

```
option explicit
dim fso, foo
set fso = createobject("Scripting.filesystemobject")
Set foo = fso.Getfolder("Path of the folder")
```

Ex:1:- Write vbscript program in QTP to display list of subfolders and files in a specified folder (C:\flightreservation)

```
option explicit
dim fso, foo, fs, sfos, i
set fso = Createobject("scripting.filesystemobject")
Set foo = fso.Getfolder("c:\FlightReservation")
Set fs = foo.Files
Print "Files are:"
Print "....."
for each i in fs
  Print i.name
Next
Set sfos = foo.SubFolders
Print "Subfolders are"
Print "....."
for each i in sfos
  Print i.name
next
```

→ write vbscript program in QTP to display .txt files in a folder.

Print(): It is a built-in function of QTP, not related to vbscript. We can use this function to display output in Quicktest print log window.

Ex:

```
option explicit
Dim fso, foo, fs, i
Set fso = createobject("scripting.filesystemobject")
set foo = fso.getfolder("C:\flightreservation")
set fs = foo.Files 'get all files in that folder.
for each i in fs
 if instr(i.name,".txt") then
  Print i.name
 End if
Next
Set fs = nothing
Set foo = nothing
Set fso = nothing.
```

getfolder(): - we can use this file system object method to create a folder object for specified folder.

```
fso.getfolder("path of folder")
```

To work with drives, vbscript allows you to create drive objects.

Ex: Display c drive free space in GB.

```
Option explicit
dim fso, dro, x, y
set fso = createobject("scripting.filesystemobject")
```

```
set dro = fso.Getdrive("c")
  x = dro.freespace
  y = x/(1024*1024*1024)
  Print y & " GB"
```

Ex 2: Display used space of c-drive

```
option explicit
Dim fso, dro, x, y, z
set fso = createobject("scripting.filesystemobject")
set dro = fso.GetDrive("c")
  x = dro.totalsize
  y = dro.freespace
usedspace - z = (x-y)/(1024*1024*1024)
  Print y & "GB"
```

GetDrive():- we can use this file system object method to create a drive object for specified drive.

```
fso.getdrive("drivename")
```

createfolder():- we can use this file system object method to create folder with specified path.

```
fso.Createfolder("Path of new folder")
```

Ex 3:- Create a new txt file in a new folder.

```
option explicit
dim fso, foo
set fso = createobject("scripting.filesystemobject")
  fso.createfolder("c:\grealibatch")
  set foo = fso.getfolder("c:\greatebath")
  foo.createTextfile("test.txt")
```

Working with excel files [spread sheet]

Vbscript in QTP will support Excel-sheet accessing of MS-office. In general, most of the testing teams are using MS-Excel for test data & test results instead of text files.

a) creating excel object.

```
option explicit
dim exo
Set exo = creatiobject ("Excel.Application")
  exo.visible = true
  wait (10)
  exo.quit
```

b) creating work book object

```
option explicit
dim exo, wbo
Set exo = creatiobject ("Excel.Application")
  exo.visible = true
Set wbo = exo.workbooks.Add 'New workbook
        (Or)
Set wbo = exo.workbooks.Open ("path of excel file")
  . . .                          'Existing work book.
  exo.Quit.
```

c) creating worksheet object.

```
option explicit
dim exo, wbo, wso
Set exo = creatiobject ("Excel.Application")
  exo.visible = true
Set wbo = exo.workbooks.Add open ("path of excel file")
```

```
Set wso = wbo.Worksheets("sheetname")
.....
exo.Quit
```

Note: In general one workbook of Excel S/w Consists of 3 sheets by default. Each consists of 256 columns and 65,536 rows. But test automaters are using specific no. of rows & columns for testing. This used area of a sheet is called as "Usedrange"

Ex1: Prepare Vbscript in QTP to get existing values in an Excel sheet. C:\Book1.xls.

xx		1	
xx			
xx			
xx			

Sheet1

```
option explicit
dim exo, wbo, wso, x
set exo = createobject("Excel.Application")
exo.visible = true
Set wbo = exo.Workbooks.open("c:\Book1.xls")
set wso = wbo.Worksheets("Sheet1")
 r = wso.Usedrange.rows.count
for i=1 to r step 1
    x=wso.cells(i,1)
     Print x
Next
 exo.Quit
Set wso = nothing
set wbo = nothing.
set exo = nothing.
```

Ex2: Write a vbscript program to display all values in excel sheet.

```
option explicit
dim exo, wbo, wso, x, r, i, c, j
Set exo = createobject ("Excel.Application")
Exo.visible = true
Set wbo = exo.workbooks.Open ("c:\Book1.xls")
Set wso = wbo.worksheets ("sheet1").
  r = wso.usedrange.rows.count
  c = wso.usedrange.columns.count
   for i = 1 to r step 1.
     for j = 1 to c step 1
        x = wso.cells(i, j)
      Print x
    Next
   Next
   exo.Quit
    Set wso = nothing
    Set wbo = nothing
    Set exo = nothing.
```

Ex3: Write vbscript program in QTP to display all rows & columns values of multiple sheets.

```
option explicit
dim exo, wbo, wso, x, r, i, c, j, t
Set exo = createObject ("Excel.Application")
exo.visible = true
Set wbo = exo.workbooks.Open ("c:\Book1.xls")
for t = 1 to 3 step 1
```

cel
et

```
Print " sheet" &t
Print "-----"
Set wso = wbo.Worksheets ("sheet" &t)
r = wso.usedrange.rows.count
c = wso.usedrange.columns.count
for i = 1 to r step 1
    Print "Row" &i
    Print "....."
  for j = 1 to c step 1
      x = wso.cells(i, j)
        Print x
  Next
  Next
  Next
  exo.Quit
  Set wso = nothing
  Set wbo = nothing
  set who = nothing.
```

Ex4: Write VbScript program in QTP to evaluate given expression in first column, compare with 2nd column value and then display result in third column.

Expression	expected	Result
24 - 10	14	Passed
23 + 23	66	failed
45 * 10	450	Passed
44/11	4	Passed

Sheet1

option explicit

```
dim exo, wbo, wso, r, e, ex, i
Set exo = Createobject ("Excel.Application")
exo.visible =true
set wbo = exo.workbooks.open ("C:\Book2.xls")
Set- wso = wbo.worksheets ("sheet1")
  r = wso.usedrange.rows.count
   for i = 2 to r Step 1
      e = wso.cells (i,1)
      ex = wso.cells (i,2)
    if eval(e) = ex then
        wso.cells (i,3).font.Colorindex = 4 → green
        wso.cells (i,3) = "passed"
    else
        wso.cells (i,3).font.Colorindex = 3
        wso.cells (i,3) = "failed"
    End if
  Next
  wbo.save
  exo.Quit
 Set wso = nothing
 Set wbo = nothing
 Set exo = nothing.
```

Vbscript in QTP allows you for formatting cells in an excel sheet

```
if eval(e) = ex then
  wso.cells (i,3).font.bold =true
  wso.cells (i,3).font.size = 20
  wso.cells (i,3).font.colorindex = 4
```

```
wso.cells(i,3),interior.colorindex = 1 → black
wso.cells(i,3) = "passed"
else
wso.cells(i,3).font.bold = true
wso.cells(i,3).font.size = 20
wso.cells(i,3).font.colorindex = 3 → red
wso.cells(i,3).interior.colorindex = 1
wso.cells(i,3) = "failed"
End if
```

Note :- Vbscript does not allow color names. Due to this reason test automators are specifing colorindexes from 1 to 56.

Ex Write Vbscript in QTP to know 56 colors.

```
option explicit
Dim exo, wbo, wso, i
Set exo = createobject("Excel.Application")
exo.Visible = true
Set wbo = exo.workbooks.Add
Set wso = wbo.worksheets("sheet1")
for i = 1 to 56 Step1
wso.cells(i,1).font.colorindex = i
wso.cells(i,1) = i
Next
wbo.SaveAs("c:\colors.xls")
exo.Quit
Set wso = nothing
Set wbo = nothing
Set exo = nothing.
```

VBScript in QTP is allowing you to generate chart or graph in an excel sheet. w.r.t existing data. This concept is useful to test automaters to generates test reports.

Ex: Write vbscript in QTP to generate a graph or chart depends on existing data in below excel sheet.

c:\Book3.xls

Overs	Runs
1	5
2	6
3	2
4	4
5	12
6	5
7	9
8	6
9	8
10	17

```
option explicit
dim exo, wbo, wso, rngo, cho
Set exo = createobject ("excel.application")
 exo.visible = true
Set wbo = exo.Workbooks.Open ("c:\book3.xls")
Set wso = wbo.Worksheets ("Sheet1")
set rngo = wso.range ("A2", "B11") 'create range object
set cho = wbo.charts.add 'create chart object
 cho.chartType = 67 'specify chart type
 cho.setsourcedata rngo, 2 'specify data range as source
 cho.location 2, "Sheet1" 'specify location to display
  wbo.Save
  exo.Quit
```

```
set cho = nothing
set rngo = nothing
set wso = nothing
set wbo = nothing
set exo = nothing
```

Creating chart object:-

We can use below code in VbScript to generate a chart or graph object.

```
Set cho = wbo.charts.add
```

This chart object consists of two main properties such as "charttype" and "setsourcedata". we can use first property to specify type of the chart which vary from 61 to 70.

```
cho.charttype = 68
```

we can use second property to specify data to be used in chart preparation.

```
cho.setsourcedata range, 2/3
```

(range → specify data.)

2 - 2-Dimensional

3 - 3-Dimensional

After completion of chart preparation we can use location() method to specify place of chart in sheet.

```
cho.location 2, "sheetname"
```

(2 → constant)

Note:→ To get no. of sheets in a work book; we can use below code

```
set wbo = exo.workbooks.open("C:\Book.xls")
n = wbo.worksheets.count
```

→ some organizations are not having licence of MS-office (MS-excel). In their companies testautomaters are using in-built "data-table". for every test in QTP data table consists of 2 sheets, such as global & action (local). This data table is easy to access than MS-excel

For every test, test automater can use data-table object to read and write data. This built-in object consists of below properties and methods.

a) addsheet(): We can use this method to add new sheet to data table.

datatable.Addsheet("sheetname")

b) Deletesheet(): - This method removes existing sheet in data-table except global and action

datatable.deletesheet(sheetid)

1 - for global sheet

2 - for action1 sheet

3 - for added sheet .... etc.

c) Value :- We can use this property to read or write data from data table.

Ex:- Write QTP script to add two column values and then display o/p in 3rd column of data table.

Data table

input1	input2	output

Global Sheet.

```
option explicit
dim x, y, z
x = datatable.value("input1", 1)
y = datatable.value("input2", 1)
z = cint(x) + cint(y)
datatable.value("output", 1) = z
```

Note:-

→ While using data table data, our program will run more than one time automatically with out having explicit loop. Generally out program runs for all rows in data table.

→ by default. If you want change settings, we can follow below navigation.

File menu → Settings → Run → Select "Run one iteration only" (OR "Run all rows" (OR) "Run from __ to __ rows". → click apply → ok.

→ After execution of test script w.r.t data table, o/p will be visible in run-time data table. This run-time data table is locating in results windows. To get results window for every test, we can follow below navigation

Tools menu → option → run → select view results "when run session ends" → click apply → click ok.

Ex2:- Write QTP script to display addition of two numbers, which located in different sheets of data table.

Data table

input1	output
x x x x	

Global sheet

Global sheet

input 2	
x x x x	

Action1 sheet

```
option explicit
dim x,y,z
x = datatable.value("input1", 1)
y = datatable.value("input2", 2)
z = cint(x) + cint(y)
datatable.value("output", 1) = z
```

Note :→ To get the visibility of data-table for current test we can use view → data table option.

→ In general QTP will save every test as a folder. In this folder, Corresponding data table will be saved as default.xls

→ Some in our test script, we want to run a part one time and remaining part more than one time. But in QTP, test script will be executed more than one time for all rows in data-table. Due to this reason we can follow below navigation & script.

File menu → Settings → Run → Select Run one iteration only

→ Click apply → click ok → follow below format for script

```
n = datatable.getsheet(1).getrowcount.
for i = 1 to n step 1
  datatable.setcurrentrow(i)
  ==
  ==
  ==
next
```

Ex 1: Write QTP script to display "Good morning" msg and then display all existing names in data table.

Data-table (global sheet)

Name
Anjali
Sirisha
Pradhi

Change run settings to one iteration only

```
Print "Good morning"
n = datatable.GetSheet(1).GetRowCount
for i = 1 to n step 1
  datatable.SetCurrentRow(i)
```

```
Print datatable.Value("Name", 1)
Next.
```

d) GetRowCount :- We can use this method property of data table to get existing no. of rows in specified sheet.

```
datatable.GetSheet(SheetId).GetCurrentRowCount
```

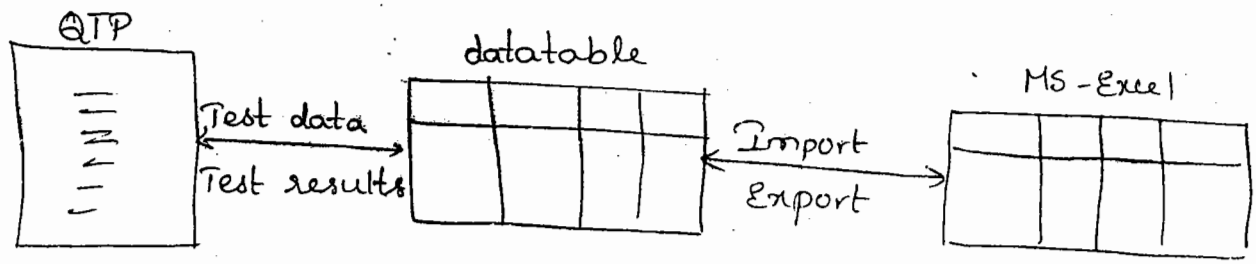
e) SetCurrentRow():- We can use this data table method to point specified row in current sheet of data table.

```
datatable.SetCurrentRow(rownumber)
```

f) GetSheetCount():- We can use this method to get no. of sheets in data table. By default data table consists of 2 sheets such as Global & action1.

```
datatable.GetSheetCount
```

Note:- In some organizations test automaters are using datatabl of test and external MS-Excel sheets both.



g) Import():- We can use this datatable method to import an external excel file into data table

```
datatable.Import("Path of external excel file")
```

While importing, first sheet of excel file data stored into global sheet and then second sheet of excel file stored into action1 sheet of data table. Similarly further sheets are stored into added sheets of data table.

h) Importsheet():- We can use this method to import specied sheet of external excel file into data table.

datatable.Importsheet("Path of excel file", "sheet name", datatablesheetId)

Ex: datatable.Importsheet "c:\Book.xls","sheet1",1

Note:- Instead of above importing methods, we can follow a navigation.

Place mouse pointer on data table → right click → sheet → import → from file → browse path of excel file → select sheet name → click ok.

i) Export():- we can use this data table method to export data-table to specified excel file.

datatable.Export("Path of excel file")

j) Exportsheet():- we can use this method to export specified sheet of data table to excel file.

datatable.Exportsheet("Path of excel file", datatablesheetId)

Note:- Instead of exporting methods, we can follow below navigation

Right click on data table → sheet → export → browse path of excel file → click save

k) Addparameter():- We can use this method to create a new column in specified sheet of data table.

datatable.Getsheet(sheetid).Addparameter("columnname", value)

l) deleteparameter():- we can use this method to delete previously added column in specified sheet of data table.

datatable.Getsheet(sheetid).deleteparameter columnid

Working With XML files

XML stands for extensive markup language. It is marking data in universal format.

Ex:-
```
<users>  ---> Root element
   <entry> --> child element
      <userid> anjali </userid>
      <Password> ANJALI </Password>
   </entry>
   <entry>
      <userid> kranthi </userid>
      <password> Vamsi </password>
   </entry>
</users>
```

Subchild elements

Value

To work with above like xml files in QTP, we can use an utility object of QTP like XMLUtil. This object Consists of a set properties & methods to work with xml files.

a) Creating xml file object:-

```
option explicit
dim xmlo
set xmlo = XMLUtil.CreateXML
xmlo.LoadFile("c:\login.xml")
        (OR)
option explicit
dim xmlo
set xmlo = XMLUtil.CreateXMLFromFile("c:/login.xml")
```

b) Creating Root element object:-

```
option explicit
dim rooto
```

Set xmlo = xmlutil.CreateXMLfromFile("C:\login.xml")

Set rooto = xmlo.GetRootElement

c) Creating objects for child elements

```
Option Explicit
dim xmlo, root, childs
Set xmlo = xmlutil.createXMLfromFile("Path of xml file")
set root = xmlo.GetRootElement
Set childs = root.childElements
```

To create objects for specified childs, we can use below statement..

```
Set childs = root.childElementsByPath(".XPath")
```

In above syntax XPath specifies hierarchy of elements in XML file.

Ex1: Write QTP script to display values of userids in below xml file.

C:\logindata.xml

```
<users>
    <entry>
        <userid> Anjali </userid>
        <Password> testing </Password>
    </entry>
    <entry>
        <userid> Mindq </userid>
        <Password> Systems </Password>
    </entry>
    <entry>
        <userid> Admin </userid>
        <Password> myself </Password>
    </entry>
</users>
```

```
Option explicit.
dim xmlo, root, childs, i, x
set xmlo = xmlutil.createXMLFromfile("c:/logindata.xml")
Set root = xmlo.GetRootElement
set childs = root.ChildElementsBypath("entry/Userid")
for i = 0 to childs.count - 1 step 1
    x = childs.Item(i).Value
    Print x
Next
Set childs = nothing
set root = nothing
set xmlo = nothing.
```

ex2: Write QTP script to display below type message depends on above xml file.

"xxxx (Userid) password is *xxx (Password)."

```
option explicit
dim xmlo, root, childs, i, x, y
set xmlo = xmlutil.createXMLfromfile("c:/logindata.xml")
Set root = xmlo.GetRootElement
Set childs = root.ChildElements
for i = 1 to childs.count step 1
    x = childs.Item(i).ChildElements.Item(1).value
    y = childs.Item(i).ChildElements.Item(2).value
    Print x & " Password is " & y
Next
Set childs = nothing
Set root = nothing
Set xmlo = nothing
```

Ex:

in below xml file.

c:\Profile.xml.

```
<players>
    <player>
        <Name> Sania </Name>
        <Age> 24 </Age>
        <Address> India </Address>
    </player>
    <player>
        <Name> Sachin </Name>
        <Age> 37 </Age>
        <Address> Mumbai </Address>
    </player>
    <player>
        <Name> Lakshman </Name>
        <Age> 35 </Age>
        <Address> Hyderabad </Address>
    </player>
</players>
```

```
option explicit
dim xmlo, root, childs, i, x, y
Set xmlo = xmlutil.CreatexmlfromFile("c:\Profile.xml")
Set root = xmlo.GetRootElement
Set childs = root.childElements.
for i=1 to childs.count step 1
x = childs.Item(i).childElements.item(1).value
y = childs.Item(i).childElements.item(3).value
Print x&" Address is "&y
Next.
```

```
set childs = nothing
set roots = nothing
set xmlo = nothing
```

ex: Write QTP script to display value and attribute value of each subchild in xml file.

```
<Temperature>
  <Converttype>
    <Temp type="Foreignheet">100</Temp>
  </Converttype>
    <Temp type="celcious">34</Temp>
  </Converttype>
</Temperature>
```

Attribute value

value

:- C:\temp.xml

```
option explicit
dim xmlo, root, childs, x, y
Set xmlo = xmlutil.CreatexmlfromFile("C:\temp.xml")
Set root = xmlo.GetRootElement
Set childs = root.ChildElementsbypath("converttype/Temp")
for i = 1 to childs.count step 1
   x = childs.Item(i).value
   y = childs.Item(i).Attributes.Item(1).value
   Print x &" in " &y
Next
   Set childs = nothing
   Set root = nothing
   Set xmlo = nothing.
```

ex:- write qtp script to change value of a text box

temp.xml

```
option explicit
dim xmlo, root, childs, i, x
set xmlo = xmlutil.CreateXMLfromfile ("c:\temp.xml")
set root = xmlo.GetRootElement
Set childs = root.childElementsBypath ("ConvertType/Temp")
for i = 1 to childs.count step 1
  x = inputbox ("enter temp value")
  childs.Item(i).setvalue(x)
Next
xmlo.SaveFile ("c:\temp.xml")
set childs = nothing
set root = nothing
set xmlo = nothing.
```

a) createXML() :- We can use this method to create an xml object.

```
Set xmlo = xmlUtil.CreateXML
```

b) Loadfile() :- We can use this method to open an existing xml file.

```
Set xmlo = xmlutil.Createxml
  xmlo.loadfile ("Path of the file")
```

c) createxmlfromfile() :- We can use this method to create an xml object and to open xml file.

```
Set xmlo = xmlUtil.Createxmlfromfile ("Path of file")
```

d) getrootelement() :- We can use this method to create an Object for root element of xml file.

```
xmlo.getrootelement
```

e) childelements():- We can use this method to create object for all childs of root element or child element.

<Parent object>.childelements

root.childelements 'childs

child.childelements 'subchilds

f) childelementsbypath():- We can use this method to get specified childs by using Xpath

<Parentobject>.childelementsbypath("Xpath")

g) count():- We can use this method to find no.of childs in collection object.

set childs = <parentobject>.childelements

childs.count ' no.of childs.

h) Item():- We can use this method to work with specified child in collection of childs.

set childs = <Parentobject>.childelements

childs.item(index)

↳ starts with 1

i) Value():- We can use this method to capture value of specified element.

childs.item(i).value.

j) setvalue():- We can use this method to change existing value of specified element.

childs.item(i).setvalue("value to change")

k) Attributes():- We can use this method to access attributes of elements in xml file.

childs.item(i).Attributes.Item(j).value

↑ index of element ↑ index of attributes of element

Creating xml files using QTP

"XMLUtil" object of QTP allows you to create new xml files. In general test automators are using this concept to generate test results as xml files.

Ex1: Write QTP script to develop below xml file. code.

```
<Environment>
   <platform>
      <os> Linux </os>
      <version> Redhat </version>
   </platform>
   <platform>
      <os> windows </os>
      <version> xp </version>
   </platform>
</Environment>
```

```
option explicit
dim xmlo, root, child, x, y, i, n
set xmlo = XMLUtil.CreateXML
xmlo.CreateDocument ("Environment") 'Root element creation
set root = xmlo.GetRootElement 'Goto root element.
n = inputbox ("Enter number of childs")
for i = 1 to n step 1
   root.AddChildElementByName "platform","" 'child creation)
   set child = root.elements 'Goto child
   x = inputbox ("enter os")
   y = inputbox ("enter version")
```

```
child.Item(child.count).AddchildElementByName "OS", x
child.Item(child.count).AddchildElementByName "version", y
Set child = nothing.
Next
xmlo.saveFile("c:\greatbatch.xml")
Set root = nothing.
Set xmlo = nothing.
```

a) Createdocument():- We can use this method to create a new xml document with a root element.

```
Set xmlo = xmlutil.createxml
xmlo.createdocument("Name for root")
```

b) AddchildelementbyName():- We can use this method to add child element to specified parent.

```
<Parent object>.Addchildelementbyname "name of child", "value
```

c) SaveFile():- We can use this method to save an xml file by using absolute path.

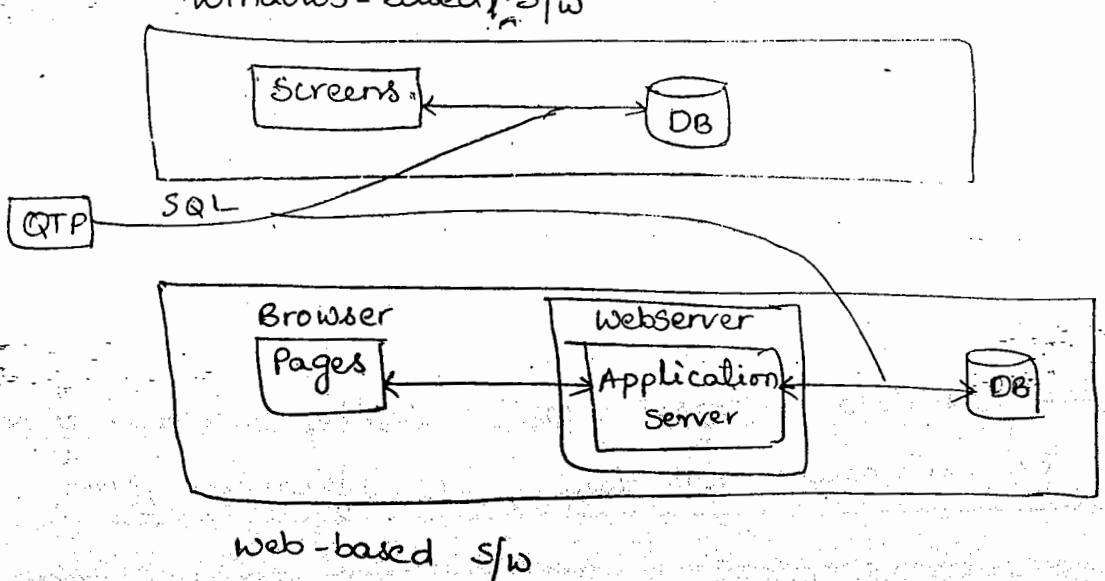
```
xmlo.SaveFile("Path of file")
```

Ex2:- Write QTP script to generate below xml file.

```
<e-banking>
  <cashdeposit>
    <testcase testtype="Fun">cashdeposit</testcase>
    <testresuts>pass</test-1results>
  </cashdeposit>
  <cashdeposit>
    <testcase testtype="Nonfunc">cashdeposit</testcase>
    <testresults>pass</testresults>
  </cashdeposit>
</e-banking>
```

Working with databases:-

In general an SUT consists of front end and back end levels. Back end levels is called as data base. In functional test automation using QTP test automaters can test database of SUT also.

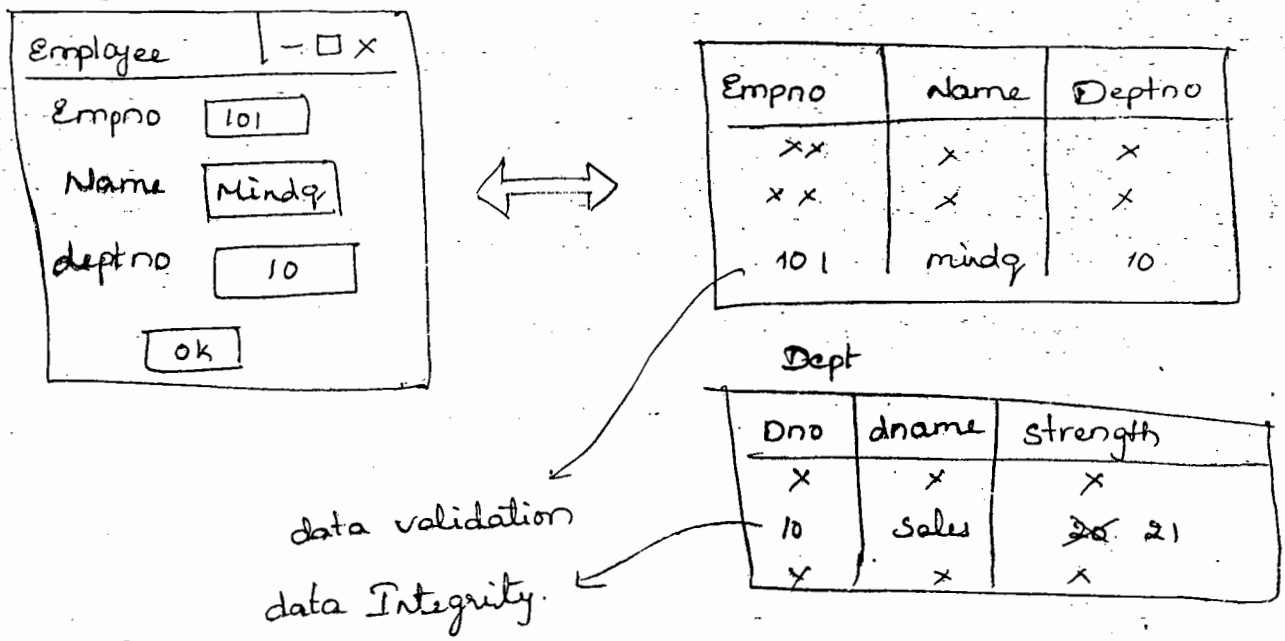


Web-based s/w

From the above diagram windows based & web-based s/w consists of data base to store and maintain data of users. To validate this data base, we can use QTP like functional testing tool. This tool provides 3 ways to validate database of SUT.

1) Data-base check point.
2) Data-base output value.
3) Vbscript programming.

While validating database of SUT, corresponding test automater is concentrating on data validation and data integrity.



From the above ex, the Correctness of new data insertion is called as data validation. The Correctness of changes in existing data is called as data integrity. While testing database of SUT, Corresponding test automaters will take below information from developers.

- Names database tables & Columns of each table.
- Connectivity in between front end and database of SUT.
- Mapping in between database tables and front end screens.
- data base design document (DDD)

1) Data base checkpoint: -

It is a ready made concept in QTP to test data base of SUT. This check point will Compare data base before operation and after operation on SUT.

Ex1: "Flight reservation" is a windows based S/W. This S/W DB is developed in MS-Access. This database Consists of Orders table. This table Consists of Columns like "Orders_number", customer_name, tickets_order, flight_number.... etc

To Connect to this data base we can use a DSN-(Data Source name) such as "QT_Flight 32". If we perform any operation on flight reservation screen, then this operation will impact on orders table.

Navigation

Launch QTP → Insert menu → check point → database check Point → Select "specify SQL Statement manually option" → click Next → click create to select DSN → Select developer given DSN in avaliable list (QT_Flight 32) → click ok → write "select" statement → (ex: select * from ~~table~~ orders) → click finish → click ok after getting current data → open SUT front end (ex: flight reservation screen) → Perform operation on that screen (ex: update or delete or insert) → Run check point in QTP → Analyze results → to estimate Correctness of front end operation impact on backend database table.

2) Database output value:-

We can use this concept to retrive data from data base into data table (Run-time) and then test automaters can check that data table.

Navigation:-

Launch QTP → Insert menu → output value → Data base output value → Select "specify SQL statement manually option" → click Next → click create to select DSN (QT_Flight 32) → Click ok → write "select" statement (Select customer_name from orders where order_number = 6) → click finish → Remember Column name in data table → extend script using if conditions in vbscript.

Ex: option explicit

```
Dim x
DbTable("DbTable").output checkpoint("DbTable") 'navigation
x = datatable.Value("DbTable_Row-1_Col_1_out", 1)
If x = "Anjali" Then
  Print "passed"
else
  Print "Failed"
End if
```

3) By using Vbscript objects

Vbscript is providing some built-in objects to connect to data base, to retrive data from data base and to disconnect from data base.

a) Connectionobject :-

We can use this object to connect to specified database of SUT.

```
option explicit
dim Con
Set Con = Createobject("ADODB.Connection")
  con.open "DSN=xxxx"
  - - - -
  - - - -
  con.close
    set con = nothing
```

b) Retriving data by using record set object

We can use this object to retrive data from Connected data base.

```
option explicit
 dim con, rs
set con = createobject ("ADODB . Connection")
  con.open "DSN = xxxx"
 Set rs = createobject ("ADODB . Recordset")
  rs. open " Select statement", con
   - - - - - -
   - - - - -
  rs . close
  Con. close
  Set rs = nothing
  Set Con = nothing.
```

Ex1: Write Vbscript in QTP to check customer_name, Column value of order_number = 6. of orders table in data base of flight reservation software.

```
option explicit
 dim con, rs, x
set con = createobject ("ADODB.connection")
  con . open " @ DSN = QT_Flight32"
  set rs = Createobject ("ADODB. Recordset")
rs. open" Select customer_name from orders where
                    Order_number = 6", con
  x = rs. Fields ("customer_name"). value
 if x = "Anjali " then
  Print " Passed"
  else
   Print " failed"
 End if
  rs . close
  Con. close
  set rs = nothing
```

Ex2:- Write VbScript in QTP to display the existing values in order_number column of orders table in data base of flight reservation.

```
option explicit
dim con, rs, q, x
Set con = Createobject("ADODB.Connection")
con.Open = "DSN = QT_Flight32"
set rs = createobject("ADODB.Recordset")
q = "select order_number from orders"
rs.open q, con
rs.MoveFirst
While rs.EOF <>true
x = rs.Fields("order_number").Value
Print x
rs.MoveNext
Wend.
rs.close
Con.Close
Set rs = nothing
Set con = nothing
```

Ex3:- Write VbScript in QTP to display customer names and their reserved tickets in orders table of DB flight reservation.

```
option explicit
dim con, rs, q, x
Set Con = Createobject("ADODB.Connection")
Con.open "DSN = QT_Flight32"
```

```
Set rs = createobject ("ADODB.Recordset")
q = "Select customer_name, tickets_ordered from orders"
rs.open q, con
rs.MoveFirst
while rs.EOF <>true
x = rs.Fields ("customer_name").Value
y = rs.Fields ("tickets_ordered").Value
Print x& " reserved " &y &" tickets"
wend
    rs.close
    con.close
```

ADODB : ActiveX data object data base

Note:- In above ex. scripts we connected to data base by using DSN. Some times developers and hard ware team is not Providing DSN. In this situation, test automaters can create their own DSN or directly use Connection String. If we used Connection string, then our program will run in any syster in network. If we used DSN, then our program will run in current machine only.

Creating DSN:-

DSN stands for data source name. It is a machine level variable to store all data base connection details.

start → Control panel → administrative tools → data sources → Click add button → select driver name for Corresponding data base [ex: Microsoft access driver [*.mdb], Microsoft ODBC for oracle, SQL server .... etc] → click finish → Specify name for DSN → give some description for understandability → select data base Path → Click ok → click ok.

Some times ... to create DSN because they are interested to use Connection strings instead of DSN. But connection strings are changing technology to technology.

MS-Access Connection string

"Provider = Microsoft.Jet.OLEDB.4.0 ; DataSource = Path-of-file.mdb ; Userid = value ; Password = value ;"

SQL-Server Connection string

For user authentication

"Provider = SQLOLEDB ; Server = Servername ; Database = databasename ; Userid = username ; Password = passwordvalue ;"

For windows os authentication

"Provider = SQLOLEDB ; server = Servername ; Database = databasename ; Trusted_connection = yes ;"

Oracle Connection string

"Provider = ORADEDB.oracle ; server = Servername ; HostString = TNSname ; Userid = username ; Password = passwordvalue ;"

In above Connection string, TNS stands for transparent network substrate

Mysql Connection string

"Provider = MSDASQL ; DRIVER = {Mysql odbc 3.5.1 driver} ; Database = Databasename ; Uid = username ; Password = password ; option = 3".

Ex 4:- Write Vbscript in QTP to get data from below specified data base.

Technology : Sql Server.

Servername : .\SYS

Authentication : Windows authentication.

Database name : master.

Table name : Dbo.spt_values

Column names: name, number, type, low, high, status

```
Option Explicit
Dim Con, rs, x, y
Set Con = CreateObject("ADODB.Connection")
Con.Open "Provider=SQLOLEDB; Server=.\SYS; Database=
          master; Trusted_Connection=yes;"
Set rs = CreateObject("ADODB.Recordset")
rs.Open "Select * from dbo.spt_values", Con
rs.MoveFirst
While rs.eof <> true
    x = rs.Fields("name").Value
    y = rs.Fields("number").Value
    Print x & " have " & y
    rs.MoveNext
Wend
    rs.Close
    Con.Close
Set rs = nothing
Set Con = nothing.
```

Ex5: Write VB script in QTP to access data from sql server.

Technology : SQL server.

Servername : SYS

Database name : Master.

Authentication : Windows authentication.

Table name : Dbo. Spt_monitor.

Column names : lastrun, cpu_busy, IO_busy, idea! ..... etc

```
option explicit
dim con, rs, x, y
Set con = CreatiObject("ADODB.Connection")
Con.open "provider = SQLOLEDB; Server = SYS; Database = master;
                Trusted_Connection = Yes;"
Set rs = Creatiobject ("ADODB: recordset")
rs.open "select * from dbo.spt_monitor", con
  rs. MoveFirst
While rs. EOF <> true
  x = rs. Fields ("cpu_busy"). value
  y = rs. Fields ("io_busy"). value
  Print x & "and" &y
  rs. MoveNext
wend
  rs. close
  Con. close
  Set rs = nothing
  Set con = nothing
```

Connection object methods and properties

We can use this object to Connect to specified data base. This object consists of below properties & methods.

a) Open():- We can use this method to open a connection for specified data base.

```
Con.open " dsn = xxxx"
    OR
Con.open " Connection String"
```

b) Errors:- We can use this property to know whether the occurance of errors while opening Connection.

```
option explicit
Dim Con
Set Con = CreateObject ("ADODB.Connection")
Con.open " DSN = QT_Flight32"
If Con.Errors.Count<>0 then
  Print " Error occured"
  exittest.
end if
```

c) Execute():- We can use this method to run insert, update or delete like queries on specified data base

```
Con.execute " Query "
```

Recordset object methods and properties:-

We can use recordset object to run a select statement on data base to retrive data.

a) Open():- We can use this method to run a select statement on an opened data base.

```
rs.open " select query ", con
```

b) MoveFirst() :- We can use this method to point first record in recordset. Here recordset consists of selected data from database.

rs.MoveFirst

c) MoveNext() :- We can use this method to point next record in record set.

rs.Movenext

d) Move() :- We can use this method to move to specific record in record set.

rs.move (recordnumber)

e) Fields :- We can use this property to specify a column in recordset.

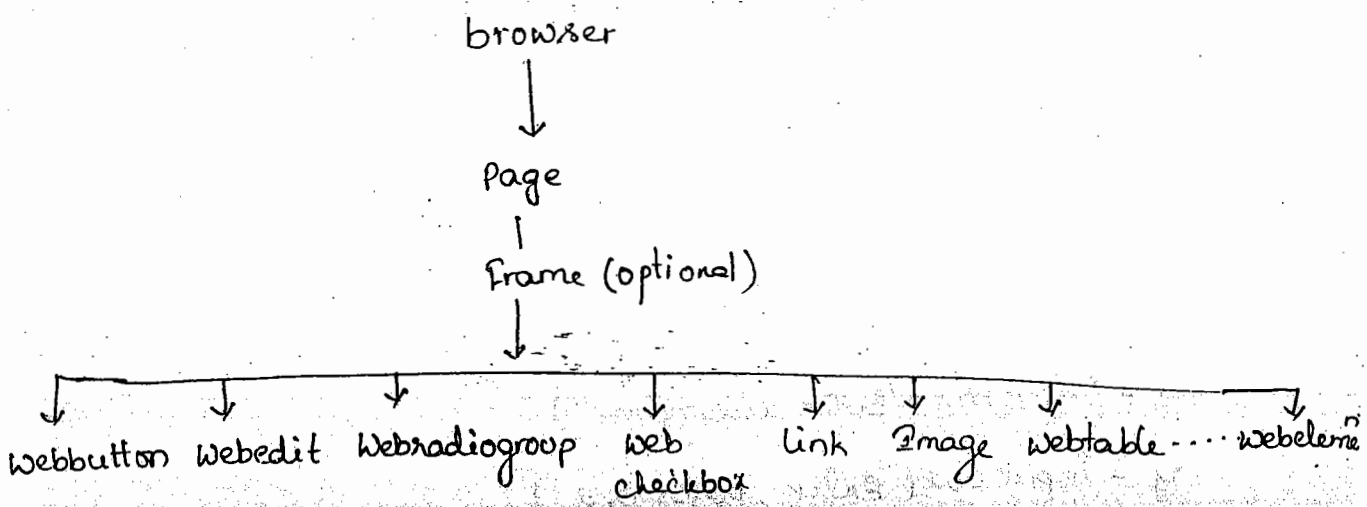
rs.fields("columnname")

f) Value :- We can use this property to get value from specified Column.

rs.fields("columnname").value

Web test automation

QTP is a powerful tool to automate web-based s/w functional testing. In this stage we can launch QTP with web add-in. while working with web-based applications testers can follow below hierarchy for objects.



While conducting testing on websites, testautomaters are using identification properties for above categorized objects in web pages.

Web object	Identification
Browser	Title.
Page	Title.
Frame	Name
Webbutton	Name
Webedit	Name
Webradiogroup	Name
Webcheckbox	Name
Link	Name
Image	Name
Webtable	Name
Webelement	Name

Note:- If any more than one object name is same in web page, then we can provide more properties values to distinguish those objects in web page.

To get properties values of web objects, we can use tools menu → objects spy

This spy s/w providing list of identification properties & native properties of specified object. While writing script test automaters are depending on identification properties

Ex1:- Write QTP Script to do login in mercury tools website

```
option explicit
dim x,y
x = inputbox("Enter username")
y = inputbox("Enter password")
invokeapplication "c:\Programfiles\InternetExplorer\iexplore.exe"
browser("title:=about:blank").Navigate "http://localhost:8080
                /mtours/servlet/com.mercurytours.servlet.
                WelocomeServlet"
browser("title:=Welocome: Mercury Tours").Page("title:=
                Welcome: Mercury Tours").Webedit("name:=
                username").Set x
browser("title:=Welcome: Mercury Tours").Page("title:=
                Welcome: Mercury Tours").webedit("name:=password".sety
browser("title:=Welcome: Mercury Tours").Page("title:=
                Welcome: Mercury Tours").image("name:=login").click
```

Ex 2:- Write QTP script to launch google site and search a wo...

```
option explicit
dim x
x = inputbox("Enter a word to search")
invokeapplication "c:\Programfiles\InternetExplorer\iexplore.exe
Browser("title:=about:blank").Navigate "http://www.google.co.in"
Browser("title:=Google").page("title:=Google").Webedit("name:=
                                                    .Set x
Browser("title:=Google").Page("title:=Google").WebButton(
                                    name:=Google Search).click
```

Note:- To decrease size of statements in QTP, we can use "With" statement concept.

```
With Browser("title:=Google").Page("title:=Google")
    .Webedit("name:=q").Set x
    .webbutton("name:=Google Search").click
End With
```

Invokeapplication():- We can use this vbscript function to open an executable s/w on the desktop.

```
invokeapplication "path of .exe"
```

Ex 3: Write QTP script to launch google site, search a word and then validate title of next page. Here next page title must consists of searched word.

```
option explicit
dim x
x= inputbox("enter a word to search")
invokeapplication ""c:\Programfiles\InternetExplorer\iexplore.exe"
```

```
Browser ("title:=about:blank").Navigate "http://www.google.co.in"
With Browser("title:=Google").Page("title:=Google")
    .Webedit("name:=q").Set x
    .Webbutton("name:=Google Search").click
End with.
if Browser("title:=" &x& " - Google Search").exist Then
    Print "passed"
  else
    Print "failed"
End if.
```

Note:- To get results in test results window we can use reporter object.

Reporter.reportevent micpass/micfail/micdone/micwarning, "Test name", "Description"

Ex 4:- Write QTP script to launch mercury tours site, do login, and then check availability of next page. Here, next page is "Find a Flight : Mercury Tours" page appears for valid user.

```
option explicit
dim x,y
x = inputbox("Enter valid userid")
y = inputbox("enter valid password")
invokeapplication "c:\Program Files\Internet Explorer\iexplore.exe"
Browser("title:=about:blank").Navigate "http://localhost:8080/
    mtours/servlet/com.mercurytours..servlet.welcomeservlet"
With Browser("title:=Welcome : Mercury Tours").Page("title:=Welcome : Mercury Tours")
```

```
    .Webedit ("name:=UserName").set x
    .Webedit ("name:=password").set y
    .Image ("name:=login").click
  End with
End with.
If Browser ("title:=Find a Flight: Mercury Tours:").exist then
   Reporter.ReportEvent micpass, "Login testing", "successful"
else
   Reporter.ReportEvent micfail, "Login test", "unsuccessful"
end if.
```

ex5:- Write QTP

Stepno	step description	Test data	Expected outcome
1.	Launch mercury tools website	URL	Home page opened.
2.	Do login operation	valid & invalid User ids & Pwds available in an Excel sheet.	For valid user, next Page is "find a flight: Mercury Tours:". For invalid user the next Page is "Sign-on: Mercury Tours"

Userid	password.	criteria.
mindq123	mindq123	Valid
Mindq123	testing	invalid.
mindq987	mindq123	invalid
mindq987	mindq987	invalid

C:\Testdata.xls.

Testscript

```
option explicit
dit exo, wbo, wso, r, i, x, y, z
Set exo = Createobject ("Excel. application")
Set wbo = exo. Workbooks. open ("c:\testdata. xls")
Set wso = wbo. Worksheets ("sheet1")
  exo. visible = true.
  r = wso. usedrange. rows. count.
  for i = 2 to r step 1
    x = wso. cells (i, 1)
    y = wso, cells (i, 2)
    z = wso. cells (i, 3)
invokeapplication "c:\Program Files\Internet Explorer\iexplore.exe"
Brower ("title: = about: blank"). Navigate "http://localhost:8080
    /mtours"
With browser ("title: = welcome: Mercury Tours")
  With . Page ("title: = welcome: Mercury Tours")
    . webedit ("name: = UserName"). Set x
    . webedit ("name: = password"). set y
    . Image ("name: = login"). click
  End with
End with.
If I = "valid" and Browser ("title: = Find a Flight: Mercury Tours").
                                    exist then
Reporter. ReportEvent micpass, "Login testing", "Successful login"
```

```
elseif z="invalid" and browser("title:=Sign-on: Mercury Tours
                                    exist then
 Reporter.ReportEvent micpass, "Login test", "Unsuccessful login"
 else
  Reporter.ReportEvent micfail, "login test", "Login not working"
 end if
   Browser("title:=.*").close
Next
   exo.Quit
   Set wso = nothing
   set wbo = nothing
   Set exo = nothing.
```

Note:- To launch a web site directly while testing we can follow syntax also

```
invokeapplication "IEpath URL"
```

Description object:-

We can use description object to store information of an object in SUT.

```
set object = description.create
object("Property").value = "Property value"
```

childobjects():- we can use this method to find specified type of childs in specified parent.

```
object("micclass").value = "Brower"
desktop.childobjects(object) ' Finding no. of browsers on desktop.
```

Ex:6 Write QTP Script to display Mercury home page on desktop.

```
option explicit
dim odesc, list, i, c
'create description for browser object
set odesc = Description.Create()
odesc("micclass").value = "Browser"
' Find opened browsers on desktop
set list = Desktop.childobjects(odesc)
'Check each browser title
c = 0
for i=0 to list.count-1 step 1
If list(i).getroproperty("title") = "Welcome: Mercury Tours" Then
c = c+1
End if
Next
Print "Mercury Tours web site opened " &c&" times"
```

Ex:- Write QTP script to close already opened mercury Tours site home page, on desktop.

```
option explicit
dim odesc, list, i
Set odesc = Description.create()
odesc("micclass").value = "Browser"
Set list = Desktop.childobjects(odesc)
for i=0 to list.count-1 step 1
If list(i).getroproperty("title") = "welcome: Mercury Tours" then
list(i).close
end if
```

Ex 8:- Write QTP script to open mercury. Tours homepage. But if already opened then no need to open home page. If more than one time opened then we can close all times except one.

```
option explicit
dim odesc, list, i, c
set odesc = Description.create()
odesc("micclass").value = "Browser"
set list = Desktop.childobjects(odesc)
c = 0
for i = 0 to list.count - 1 step 1
If list(i).getroproperty("title") <> "welcome: Mercury Tours" then
   list(i).close
else
   c = c+1
   If c>1 then
   list(i).close
   end if
end if
Next.
'If Mercury Tours not found, open it
if c = 0 then
invokeapplication "http:\ c:\ Programfiles\Internet Explorer\Iexplore
   exe URL"
End if
Set odesc = nothing
set list = nothing
```

Getproperty() :- We can use this method to get the property value of run-time object in SUT.

Object.getroproperty ("Property name")

Ex 9: Write QTP script to launch mercury Tours web site and then display count of links.

```
option explicit
dim odesc, list, i, c
Set odesc = Create.Object("description")
odesc("micclass").value = "link"
invokeapplication "c:\Programfiles\Internetexplorer\iexplore.exe URL"
Set list = Browser("title:=Welcome: Mercury Tours").
Page("title:=Welcome. Mercury Tours").childobjects(odesc)
Print list.count
Browser("title:=.*").close.
Set list = nothing.
Set odesc = nothing.
```

Ex 10:- Write QTP script to display no. of links, no. of images and no. of webedit boxes in mercury tools home page.

```
option explicit
Dim odesc1, odesc2, odesc3, list1, list2, list3.
Set odesc1 = description.create
odesc1("micclass").value = "link"
Set odesc2 = description.create
odesc2("micclass").value = "image"
```

```
Set odesc3 = description.create
odesc3("micclass").value = "webedit"
Invokeapplication "c:\Programfile\Internet explorer\Iexplore.exe URL"
With Browser("title:=welcome: Mercury Tours")
With page("title:=welcome: Mercury Tours")
Set list1 = .childobjects(odesc1)
Set list2 = .Child objects(odesc2)
Set list3 = .Child objects(odesc3)
End with
End with.
Print "No. links " & List1.count
Print "No. Images " & list2.count
Print "No of webedits " & list3.count
Browser("title:=welcome: Mercury Tours").close.
Set list1 = nothing
set list2 = nothing
Set list3 = nothing
Set odesc1 = nothing
Set odesc2 = nothing
Set odesc3 = nothing.
```

Ex11:- Write QTP script to display names and their URLs of all links in mercury tools website home page.

```
option explicit
Dim odesc, list, i, x, y
Set odesc = description.Create
odesc("micclass").value = link
```

```
invokeapplication  C:\Program Files\Internet Explorer\iexplore.exe ...
Set list = Browser("title:= Welcome : Mercury Tours").Page("title:=
                    Mercury Tours").ChildObjects(odesc)
For i=0 to list.Count-1  Setp 1
  x = list(i).getroproperty("name")
  y = list(i).getroproperty("url")
  Print x & "link address is" & y
Next
  Browser("title:= Welcome : Mercury Tours").close
  Set list = nothing
  Set odesc = nothing
```

Ex 12:- Test case

Setp no	Setp description	Test data	Expected outcome
1.	Launch mercury tools website	URL	Home page opened.
2.	Find all links in that Page to match with given URLs	C:\URLs.xls (Get from developers)	Specified links was found and URLs was matched.

Test data

Link name	URL
Anjali	Http:\\.....
⋮	⋮

C:\ urls.xls.

```
option explicit
Dim exo, wbo, wso, x, y, i, odesc, list, r, j
Set exo = createobject("excel.Application")
Set wbo = exo.workbooks.open("c:\uds.xls")
set wso = wbo.worksheets("sheet1")
Set odesc = createobject description.creati
odesc("micclass").value = "link"
invokeapplication "c:\Program files\Internet Explorer\iexplore.exe URL
set list = Browser("title:= Welcome : Mercury Tours").Page(title:=
                    welcome: Mercury Tours").childobjects(odesc)
r = wso.usedrange.rows.count.
for i=2 to r step 1
    x = wso.cells(i,1)
    y = wso.cells(i,2)
      flg = 0
    for j=0 to list.count-1 step 1
            if x=list(j).getroproperty("name") Then
              flag=1
            if x=list(j).getroproperty("url") then
reporter.reportEvent micpass, x& "link test", "Passed"
            else
reporter.reportEvent micfail, x& "link test", "link with wrong
            end if.                                          address"
        end if
Next
  If flag=0 then
reporter.reportEvent micfail, x& "link test", "link was not
```

```
Browser("title:=Welcome : Mercury Tours").Close
exo.Quit
Set wbo = nothing
Set wso = nothing
Set exo = nothing
Set list = nothing
set odesc = nothing
```

Ex:13 Write QTP script to find no. of images in google home page.

```
option explicit
Dim odesc, list, n
Set odesc = Description.create
odesc("micclass").value = "image"
invokeapplication "c:\Programfiles\Internet explorer\iexplore.exe http://www.google.co.in"
set list = Browser("title:=Google").Page("title:=Google").childobjects(odesc)
n = list.count
Print "No: of images is" &n
Browser("title:=Google").close
```

Ex:14 Write QTP script to display image names (src - source of each image)

```
option explicit
Dim odesc, list, x, y
Set odesc = Description.create
odesc("micclass").value = "image"
```

```
invokeapplication "c:\Programfiles\Internet Explorer\iexplore.exe
                                    Http://www.google.co.in"
set list = Browser("title:=Google").page("title:=Google").childobjects
                                                    (odesc)
  for i=0 to list.count-1  Step 1
      x=list(i).getropsroperty("name")
      y=list(i).getroproperty["src"]
    Print x& " image source is " &y
  Next
  Browser("title:=Google").close.
```

Ex:15 Write QTP script to check complete loading of images in google page.

```
option explicit
Dim odesc, list, bro, uri, i, obj
Set odesc = Description.create
odesc("micclass"), value = "image"
bro = "C:\Programfiles\Internet Explorer\iexplore.exe"
Url = "http://www.google.co.in"
invokeapplication bra &" " & url
Set list = Browser("title:=Google").Page("title:=Google").childobject
                                                    (odesc)
  for i=0 to list.count-1  step 1
    set obj = list(i).object
  If obj.complete =true then
  Reporter.ReportEvent micPass, "Image Test", "Completely loaded"
  else
  Reporter.ReportEvent micfail, "Image Test", "Incomplete loading"
```

```
end if
set obj = nothing
Next
Browser("title:= Google").close
set list = nothing
set odesc = nothing
```

Ex:16 Write QTP script to check each image visibility and Complete loading in mercury Tours home page.

```
option explicit
Dim odesc, list, bro, url, i, obj
Set odesc = description.create
odesc("micclass").value = "image"
bro = "c:\Programfiles\Internetexplorer\iexplore.exe"
Url = "http://newtours.demoaut.com/"
invokeapplication bro & " " url
with browser("title:= Welcome: Mercury Tours")
    with .Page("title:= welcome: Mercury Tours")
set list = .childobj(odesc)
for i = 0 to list.count-1 step 1
    x = list(i).getroproperty("visible")
    Set obj = list(i).object
if x=true and obj.complete=true then
reporter.reporevent micpass, "image test", "visible & Complete"
    else
reporter.reporlevent micfail, "image test", "invisible & incomplete"
end if
```

```
    Set obj = nothing.
Next
    Browser("title:=.*").close
    set list = nothing
    set odesc = nothing
```

Note:- While testing webobjects in a website, test automaters are using different ways to capture values of those objects properties, because any object properties classified into identification and native properties. To captured value of identification property we can use getroproperty() method. To capture values of native properties, we can create corresponding web object as test object by using object() method.

Ex:
```
with browser("title:= welcome: Mercury Tours")
    with .page("title:= welcome: Mercury Tours")
        with .image("name:= login")
            x = .getroproperty("visible") ' Identification property
        End with
    End with
End with
```

→
```
with browser(". . . .")
    With .page(" . . . . ")
        With .image(" . . . ")
            x = .object.Complete " Native property
    End with
End with
```

Ex#17 Write QTP script for launching mercury tours

```
option explicit
Dim bro,url
bro = "c:\program files\Internet explorer\Iexplore.exe"
url = "http://newtours.demoaut.com/"
Invokeapplication bro &" "& url
With browser("title:=Welcome: Mercury Tours")
  With .page("title:=Welcome: Mercury Tours")
    .Webedit("name:=userName").set "admin"
    .Webedit("name:=Password"), set "admin"
    .Image("name:=login").click
  End with
End with
```

Find a flight

```
With browser("title:=Welcome: Mercury Tours:")
  With .Page("title:=Find a Flight: Mercury Tours")
    .Webradiogroup("name:=tripType").Select "#1" 'first-
                                          radio will be selected
    .Weblist("name:=passCount").Select "1"
    .Weblist("name:=fromPort").Select "Paris"
    .Weblist("name:=FromMonth").Select "May"
    .Weblist("name:=FromDay").Select "31"
    .Weblist("name:=ToPort").Select "India"
    .Weblist("name:=ToMonth").Select "June"
    .Weblist("name:=ToDay").Select "10"
    .Webradiogroup("name:=ServClass").Select "#1"
```

```
    .WebList("name:=airline").Select "nopreference"
    .Image("name:=findFlights").click
End with
End with
```

Ex18 :- write QTP script to automate launching, login, filling flight details, selecting flight and then calculate total cost: (upfare + downfare) * no. of passengers + tax.

CSTAF (customized S/w Test automation framework)

Framework is process of automation. While automating functional testcases using QTP testautomaters are following this framework. It is a combination of four standard frameworks. such as

→ Linear sequential framework.

→ Modular driven framework

→ keyword driven framework

→ Data driven framework

① Linear Sequential framework

From this framework test automaters are preparing testscripts with line by line statements. Here testscripts are not sharing sharable resources.

Note:- Previously prepared programs in notebook are related to linear sequential model

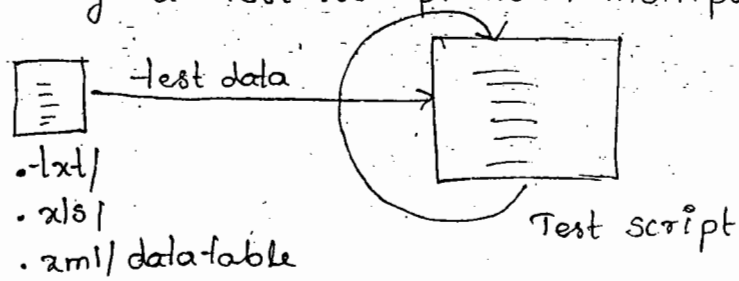
Modular driven framework

From this framework test automaters are preparing test scripts by calling user defined functions. Here each function specifies a reusable code.

keyword driven framework:-

From this framework test automaters are preparing testscripts by calling keywords. Here each keyword specifies description of objects in SUT

d) Data driven framework:- From this framework testautomaters are running a test script with multiple test data.



Note:- In above 4 standard frameworks first 3 frameworks are used to create test scripts. Fourth script framework is used to run test script.

Our CSTAF is a combination of above 4 standard frame works. From CSTAF, each test automater follows below steps in automation.

Step1:- Test lead will create a folder structure in our server Computer, which is sharable to all testers.

ex:- <Project name> / <Project_automation>

- Documents (SRS like requirements documents available)
- Test cases (Test cases to be automated)
- Test data (Test data related to Test cases)
- Test scripts (Automation programs)
- Function library (reusable function)
- object repository (Descriptions of objects in SUT)
- Recovery scenarios (Recoveries to recover from abnormal situations)
- Utilities (other suporting Programs or s/w)
- Test results (Results after run test scripts)

Note:- Above folder structure is no need when there is Quality center s/w for test management.

Step 2:- Collect manual test cases to automate, and then save those test cases in test cases folder.

Step 3:- Gather related test data for each test case and then save that test data in test data folder. as .txt / .xls / .xml files

Step 4:- Prepare keywords for descriptions of each testable object in AUT.

Ex:-
```
Set home = description.Create
home("title").value = "welcome: Mercury Tours"
Set un = description.create
un("name").value = "userName"
Set Pwd = description.create
Pwd("name").value = "Password"
```

In above ex. code, keywords are home, un & Pwd. This type of keywords saved into object repository folder.

Step 5: Collect previously saved test cases to identify repeatable steps. and then prepare automation code for those repeatable steps. as user defined functions.

Ex:
```
function launch()
bro = "c:\Program files | Internet Explorer | iexplore.exe"
url = "http://www.google.com"
invokeapplication bro & " " & url
end function.
```

After preparing user defined functions, test automators will save them in function library. folder

Step 6:- prepare test scripts by using keywords and user-defined functions, and then save those test scripts in test scripts folder.

Step 7:- Enhance each test script with multiple test data (data driven testing)

Step 8:- Define recoveries to recover from abnormal states while running test scripts. and then save those recoveries in recovery scenarios folder.

Step 9:- Make related test scripts as test batches by writing driver scripts. and then save those driver scripts in "utilities" folder.

Step 10:- Run those drivers on modified SUT for re & regression testing.

case study:-

Test cases:-

Test case 1:-

Step no	description	Test data	expected outcome
1.	Launch mercury Tours web site	URL	Welcome: Mercury Tours Page opened.

Test case 2:-

Step no	description	Test data	expected outcome
1.	Launch mercury Tours web site	URL	Welcome: Mercury Tours Page opened
2.	Do login operation	D:/MercuryTours/Testdata/login.xls	For valid data, "Find a flight: Mercury

Test case 3

Stepno	Description	Test data	expected outcome.
1.	Refer step1 in test case 1		
2.	Refer step2 in test case 2.		
3.	Fill "find a flight : Mercury Tours : "page	D:/Mercury-Tours/ Testdata/flight-details.txt	For valid details, Select a flight : Mercury Tours " Page opened.

Test case 4

Stepno	Description	Test data	expected outcome.
1.	Refer step1, step2 & step 3 of test case 3.		
2.	Fill "select a flight: Mercury Tours" page	D:/Mercury_Tours/ Testdata/flights.text	Summary page opened; and total = (upfare + down fare) * no of passengers + tax.

Automate above test cases in CSTAF.

Prepare keywords:-

After collecting Test cases & Test data Test automater start keywords creations for objects in SUT. Here Testers can take help of objects spy to get descriptions of objects in SUT.

Navigation

open note pad → write key words → save filename.vbs

keywords:

```
Set home : description.create
home("title") = "welcome : Mercury Tours"
set un = description.create
un("name") = "userName"
```

```
Set pwd = description.create
pwd("name").value = "Password"
Set signin = description.create
signin("name").value = "login"
Set fdetails = description.create
fdetails("title").value = "Find a Flight : Mercury Tours:"
Set trip = description.create
trip("name").value = "tripType"
set nop = description.create
nop("name").value = "passCount"
set fport = description.create
fport("name").value = "fromPort"
set fm = description.create
fm("name").value = "fromMonth"
Set fd = description.create
fd("name").value = "fromDay"
set tport = description.create
tport("name").value = "toPort"
Set tm = description.create
tm("name").value = "toMonth"
set td = description.create
td("name").value = "toDay"
set serv = description.create
serv("name").value = "servClass"
set airline = description.create
airline("name").value = "airline"
```

```
Set find = description.create
find("name").value = "findFlights"
Set fselect = description.create
fselect("title").value = "select a Flight : Mercury Tours"
Set upflight = description.create
upflight("name").value = "outFlight"
Set downflight = description.create
downflight("name").value = "inFlight"
Set reserve = description.create
reserve("name").value = "reserveflights"
Set booking = description.create
booking("title").value = "Book a flight : Mercury Tours"
Set summary = description.create
Summary("height").value = 190
Summary("width").value = 484
Set logout = description.create
logout("innertext").value = "SIGN-OFF"
Set relogin = description.create
relogin("title").value = "Sign-on: Mercury Tours"
```

Note: After writing keywords for responsible objects in SUT, Corresponding test automator will save those keywords in .vbs file.

Ex: D:\Mercury_Tours\objectrepository\keywords.vbs.

Creating user defined functions

In our case study test cases, some steps are repeat more than one time. For those steps test automaters will prepare user-defined functions to reuse in future.

Navigation

Launch QTP → File menu → new → Function Library → write functions one by one → save library with filename.vbs/txt/.qf (QTP function library).

ex: functions

Functions follow below syntax

```
function name (Byval/Byref arguments .....)
    ========
End function.
```

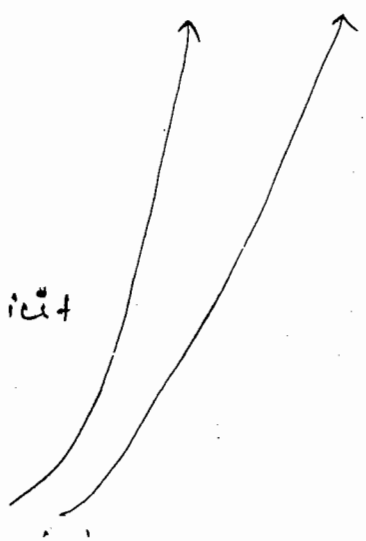
In above syntax arguments are two types. Such as Byval and Byref. If an argument is byval then that argument will receive the data. If an argument is byref, then that argument will receive the data and will return the data

Ex:-

```
function add (Byval x, Byval y)
    x = x + 10
    y = y + 20
End function
```

Test script.

```
option explicit
dim a, b
a = 10
b = 20
```



msgbox a 10

msgbox b 20

Eg#: Function add (Byref x, Byref y)

x = x + 10

y = x + 20

End function

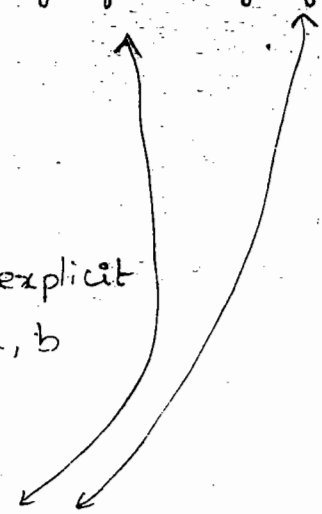
Test script option explicit

dim a, b

a = 10

b = 20

add (a, b)



msgbox a 20

msgbox b 40

Like in other programming languages, Vbscript functions also returns values to calling test. Here, returned value assigned to function name.

function add (Byval x, Byval y)

Dim z

z = x + y

add = z

End function

Test script

option explicit

dim a, b, c

a = 10

b = 20

c = add (a, b)

msgbox c

In general test automaters are using user-defined functio Concept to mark reusable code.

Launch function :-

```
function launch(Byval b, Byval u)
    invokeapplication b&" "&u
End function
```

Login function

```
function login(Byval uid, Byval pwd)
  With browser(home)
    With .page(home)
        .webedit(un).set uid
        .webedit(pwd).set p
        .image(signin).click
    End with
  End with.
```

While writing above like functions using keywords, we can associate keywords file to functions first.

executefile "Path of keywords file"

After associating keywords and functions we can save functions in function library folder with filename.qfl

Creating test scripts

TestScript for testcase 1

```
executefile "d:\Mercury_Tours\object repository\keywords.xls"
executefile "d:\Mercury_Tours\Functionlibrary\functions.qfl"
launch("c:\Program files\Internet Explorer\iexplore.exe
        http://newtours.demoaut.com"
```

```
if browser(home).exist then
reporter.reportevent micpass, "launching test", "successful"
else
reporter.reportevent micfail, "launching test", "unsuccessful"
end if
```

Testscript for testcase 2

```
option explicit
Dim exo, wbo, wso, r, i, x, y, z
Set exo = createobject("Excel.application")
exo.visible = True
Set wbo = exo.workbooks.open("D:\Mercury_Tours\
                              testdata\login.xls")
Set wso = wbo.worksheets("sheet1")
executefile "D:\Mercury_Tours\objectrepository\keywords.xbs"
executefile "D:\Mercury_Tours\functiolibrary\functions.qfl"
r = wso.usedrange.rows.count
for i = 2 to r step 1
    x = wso.cells(i,1)
    y = wso.cells(i,2)
    z = wso.cells(i,3)
    login(x,y)
launch "c:\Prgromfiles\--", "URL"
if z = valid and Browser(fdetails).exist then
reporter.reportevent micpass, "logintest", "correct"
elseif z = invalid and Browser(relogin).exist then
reporter.reportevent micpass, "logintest", "corret"
else
```

```
reporter.reportevent.micfail,"logintest","incorrect"
end if
Browser("title:=:*").close
Next
set exo.Quit
Set wso = nothing
Set wbo = nothing
set exo = nothing
```

Userid	password	criteria
xx	xx	xx
xx	xx	xx

Testcase Script for testcase 3

flight_details.txt

```
option explicit
admin, admin, #0, 2, London, November, 1, Paris, December, 31, #2,
Blue Skis Airlines
```

(y(0) y(1) y(6))

```
option explicit
Dim fso, fo, x
set fso = createobject("scripting.filesystemobject")
set fo = fso.opentextfile("D:\Mercury_Tours\Testdata\
                flight_details.txt", 1, false)
executefile "D:\Mercury_Tours\objectrepository\keywords.vbs"
executefile "D:\Mercury_Tours\Functionlibrary\functions.qfl"
while fo.AtEndofStream <> true.
x=fo.readline
y= split(x,",").
launch "c\Programfiles ..... ","URL"
login y(0), y(1)
```

```
With browser(fdetails)
    With , page(fdetails)
        . webradiogroup(trip). select y(2)
        . weblist(nop). select y(3)
        . weblist(fport). select y(4)
        . weblist(fm). select y(5)
        . weblist(fd), select y(6)
        . weblist(Tport). select y(7)
        . weblist(tm). select y(8)
        . weblist(td). select y(9)
        . webradiogroup(serv): select y(10)
        . weblist(line). select y(11)
        . image(find). click
    end with
End with
if browser(fselect). exist then
Reporter. reportevent micpass, "flight details testing", "correct"
else
Reporter. reportevent micfail, "flight details test", "incorrect"
end if
    browser("title:=.*"). close
Wend
    fo. close
Set fo = nothing
Set fso = nothing.
```

Executefile():- We can use this QTP function to associate required resources to test.

Syntax: Executefile "path of file"

split():- We can use this VbScript function to divide a line into peaces.

Split ("line of text", "separator")

```
x = "check De India"
y = split(x, " ")
```

x

check	y(0)
De	y(1)
India	y(2)

Testscript for testcase 4

```
option explicit
Dim fso, fo, x, y, upf, df, t, tax, tot
Set fso = createobject ("Scripting.FileSystemObject")
Set fo = fso.opentextfile ("D:\Mercury_Tours\testdata\flights.txt",
                                                    1, "false")
executefile "D:\Mercury_Tours\objectrepositry\keywords.vbs"
executefile "D:\Mercury_Tours\objectrepository\functions.qfl"
While fo.AtEndOfStream <> true
    x = fo.readLine
    y = split (x, ",")
launch "c:\Programfiles\Intent Explorer\Iexplore.exe",
                "http://www.newtours.demoaut.com"
Login y(0), y(1)
```

```
With browser(fdetails).Page(fdetails)
    .Webradiogroup(trip).Select y(2)
    .Weblist(nop).Select y(3)
    .Weblist(fport).Select y(4)
    .Weblist(fm).Select y(5)
    .Weblist(fd).Select y(6)
    .Weblist(tport).Select y(7)
    .Weblist(tm).Select y(8)
    .Weblist(td).Select y(9)
    .Webradiogroup(serv).Select y(10)
    .Weblist(aline).Select y(11)
    .Image(find).Click
End With
  With browser(fselect).Page(fselect)
    .Webradiogroup(upflight).Select y(12)
    .Webradiogroup(downflight).Select y(13)
    .Image(reserve).Click
  End With
  With browser(Booking).Page(Booking)
  upf = .Webtable(summary).getcelldata(3,3)
  df = .Webtable(summary).getcelldata(6,3)
  t = .Webtable(summary).getcelldata(7,2)
  tax = .Webtable(summary).getcelldata(8,2)
  to2 = .Webtable(summary).getcelldata(9,2)
  End With
```

```
upf = cint(upf)
df = cint(df)
t = cint(t)
tax = mid(tax, 2, len(tax)-1
tax = cint(tax)
tot = mid(tot, 2, len(tot)-1)
tot = cint(tot)
if tot = (upf + df) * t + tax then
Reporter.ReportEvent micpass, "calculation test", "correct"
else
Reporter.ReportEvent micfail, "calculation test", "incorrect"
end if
browser(booking).page(booking).link(logout).click
browser(relogin).page(relogi close
wend
fo.close
set fo = nothing
set fso = nothing
```

mid() :- we can use this vbscript function to get substring from mainstring.

```
mid("mainstring", starting position, length)
mid("Check de India", 3, 4)    'ack
mid("RS:286/-", 4, 3)          '286
x = inputbox("Enter a value")  'RS:4867/-"
mid(x, 4, len(x)-5)
```

Remove RS:& /-

Left() :- we can use this function to get specified no. of characters from left side of string.

```
left("My love is gone", 6)   'My love
```

right():- We can use this function to get specified no. of characters from right side of the string.

Ex: right("My love is gone") ' gone

join():- We can use this vbscript function to join more than one string as one string.

join("Comparator", values....)

join(",", 10, 20, 30, 40) ' 10, 20, 30, 40

join(" ", "My", "love", "is", "gone")

' My love is gone.

Case study 2:-

Google is a search website in this website we will apply below test cases to validate searching functionality.

Test case 1:-

step no.	Description	Test data	expected outcome.
1.	Launch Google website	URL	Google page opened

Test case 2:-

step no.	Description	Test data	expected outcome
1.	Launch google site.	URL	Google page opened
2.	Enter a word to search	D:\Google\Test data\words.xml	Next page opened with searched word as title

Test case 3

step no	Description	Test data	Expected outcome
1.	Launch google site.	URL	Google page opened.
2	Enter a word to Search	D:\Google\Test data\words.xml	next page opened with search results. Here

of

about "xxxx" results message appear

```
' Creating keywords
  Set home = Description . creati
   home("title").value = "www Google"
   set search = Description . create
   home ("name"). value = "q"
   Set gsearch = Description . creati
   gsearch ("name") .value = "Google Search"
```

Note:- Save above keywords in "D:|google_automation|object repository|keywords.vbs"

Creating functions

```
executefile "D:|google_automation|object repository|keywords.vbs"
function launch(Byval b, Byval u)
  invokeapplication b& " "&u
end function
function find (Byval x)
    with browser(home) . page (home)
        .webedit (search) . set x
        . webbutton (gsearch) . click
    end with
  end function
```

Creating test scripts

Testscript for testcase:-

```
executefile "D:|google_automation\objectrepository|keywords .vbs"
executefile "D:|google_automation\function|ibrary\functions. Qtf"
```

```
Launch "C:\Program Files\Internet Explorer\iexplore.exe", "http://
                                    www.google.co.in"
If browser(home).exist then
Reporter.Reportevent micpass, "Launching test", "opened"
else
Reporter.Reportevent micfail, "Launching test", "Not opend
End if
  Browser("title:=.*").close
```

Test Script for Test case2

```
<Words>                                          words.xml
    <Word>
        <Value> Hindq </Value>
    </Word>                                  D:\Google_automation\
    <Word>                                   Testdata\words.xml
        <Value>Testing </Value>
    </Word>
    <Word>
        <value> QTP </value>
    </Word>
    <Word>
        <Value> ICC World cup 2011 </Value>
    </Word>
</Words>
```

executefile ",

```
Option explicit
Dim xmlo, root, child, i, x
Set xmlo = XMLUtil.CreateXMLfromfile("D:\Google_automation\
                                    testdata\words.xml")
```

```
executefile " D:\Google_automation\objectrepository\keywords.vbs"
executefile " D:\Google_automation\functionlibrary\functions.Qfl"
  set root = xmlo.getrootelement
  set child = root.childelementsbypath ("word/value")
    for i=1 to child.count step 1
      x = child.item(i).value
      launch ("C:\Program files\Internet Explorer\IExplore.exe", "URL"
      find x
  if browser("title"):="&xx".*").exist.then
reporter.ReportEvent micpass, "searching test", "successful"
  else
reporter.ReportEvent micfail, "searching test", "unsuccesful"
  End if
   Browser ("title :=.*).close.
  Next
   set child = nothing
   set root = nothing
  set xmlo = nothing.
```

Testscript for testcases

```
    option explicit
    Dim xmlo, root, child, i, x, y, r
  Set xmlo = XHLutil.creatXMLfromfile ("D:\google_automation\
                                        testdata\words.xml")
executefile " D:\Google_automation\objectrepository\keywords.vbs"
executefile " D:\Google_automation\functionlibrary\functions.Qfl"
```

```
Set root = xmlo.GetRootElement :
Set child = root. ChildElements bypath ("word /value")
 for i = 1  child.count  step 1
   x = child . item(i). value
Launch " c:\Programfiles\Internet Explorer\. IExplore.exe", " http://www.
                                                    google. co. in"
   find x
With  browser ("title :=" & x & " . * "). exist
 With. Page ("title : = " & x & " . * ")
  y = . Weblable ("html id : = mn"). Getcelldata (2, 1)
End with
End with
  Set r = new regexp        ([0-9]+[,]?){1,}
                              (OR)
r. Pattern = "[A][b][o][u][t] [is]([0-9]+[,]?[0-9]+){1,}[is]
                                  [r][e][s][u][l][t][s]"
if r.test(y) then
Reporter. ReportEvent micpass "Content testing", "Correct message"
else
Reporter. ReportEvent micfail "Content testing", "incorrect message
end if
  browser ("title : = . * "). Close .
  Set r = nothing
Next
  Set child = nothing
  Set root = nothing . Set xmlo = nothig
```

Note 1:→ While working with web pages test automaters are working with web tables. To get web table architecture they can follow below way. In first way, test automater can open source of page and then parse required web table rows & columns (<tr> & <td>). In second way test automaters can try to insert checkpoint on a web table. Here we can get architecture of web table on the desktop.

Insert menu → check point → standard check point → show web table → See architecture to know rows and columns → Click cancel → stop recording.

Challanges in web testing using QTP

Challenge1
→ Working with hidden objects : Some times web pages consists of hidden objects. To automate these objects in testing testautomats are using windows shell object in QTP.

```
Set obj = createObject("WScript.Shell")
```

This windows shell object consists of SendKeys() method to operate any hidden object.

```
obj.SendKeys("required keys")
```

From the above syntax required keys can specify in below format.

key	Argument
BACKSPACE	{BACKSPACE}, {BS}, {BKSP}
BREAK	{BREAK}
CAPSLOCK	{CAPSLOCK}
DEL or DELETE	{DELETE} or {DEL}
DOWN ARROW	{DOWN}

END	{END}
ENTER	{ENTER} or ~
ESC	{ESC}
HELP	{HELP}
HOME	{HOME}
INS or INSERT	{INSERT} or {INS}
LEFT ARROW	{LEFT}
NUM LOCK	{NUMLOCK}
PAGE DOWN	{PGDN}
PAGEUP	{PGUP}
PRINT SCREEN	{PRTSC}
RIGHT ARROW	{RIGHT}
SCROLL LOCK	{SCROLLLOCK}
TAB	{TAB}
UP ARROW	{UP}
F1	{F1}
F2	{F2}
F3	{F3}
⋮	⋮
F16	{F16}
SHIFT	+
CTRL	^
ALT	%

Note:- From the above table some keys are used for other keys to represent. For ex. + used for shift key. If you want '+' key, then we can write

obj. Sendkeys ("{+}")

→ Ex1: Prepare QTP script to launch google site, fill edit box to search a word, select 5th item in google cache and then click google search button.

```
option explicit
dim i, obj
Set obj = Createobject("wscript.shell")
invokeapplication " c:\Program files\Internet Explorer\iexplore.exe
" http://www.google.com"
obj.sendkeys("Mindq")
for i = 1 to 5 step 1
obj.sendkeys ("{DOWN}")
Next
obj.sendkeys ("~")
```

Challenge-2:- Working with similar objects:-

Some times, our webpages consists of similar objects, which Properties values are same. To distinguish these objects in script, test automaters are using index property. Similar objects index starts with zero.

Ex:- Browser("title:=mindq").Page("title:=mindq")
.Link("name:=register", "index:=0").Click

In above ex. statiment QTP can click first linked named as "Register".

Ex-1: Prepare QTP script to launch yahoo website, login, click inbox link to open mail box & then check availability of "welcome" mail (subject)

```
option explicit
dim flag,i,r
invokeapplication "c:\program files\Internet Explorer\iexplore.exe
                   http:\www.yahoomail.com"
wait 10
with browser("title:=.*")
   with .page("title:=.*")
  .webedit("name:=login").set "mail id"
  .webedit("name:=passwd").set "your password"
  .webbutton("name:=sign in").click
end with
end with
wait 10
   with browser("title:=yahoo.*")
    with .page("title:=yahoo.*")
    with .webtable("name:toggleAll",with:=561",height=199")
    r=.Rowcount
    flag=0
    for i=1 to r step 1
   if .getcelldata(i,6)="welcome to yahoo!" then
reporter.reportevent micpass,"yahoomail box test","mail was found"
    flag=1
    End if
    Next
    if flag=0 then
reporter.reportevent micfail,"yahoomail box test","mail was not
                                                    found"
  End if
End with
```

```
.webelement("innertext:=Sign out","index:=0").click
End with
End with
wait 30
browser("title:=.*").close
```

Challenge-3:- Working with multi browser.

Sometimes test automaters are working with multiple browsers which have same title. In this situation test automatr can use a Property "creationtime" to distinguish recently opened browser and already existing browser.

```
Eg: invokeapplication "c:\Program files\Internet Explorer\iexplore.exe
                       http://www.google.co.in"
with browser("creationtime:=0").page("creationtime:=0").
    webedit("name:=q").set "mindq"
End with.
```

Challenge-4:- Using regular expression for type testing:-

In general, test automaters are using regular expressions in descriptions of dynamic objects.

```
ex: Browser("title:=.*").Page("title:=.*").webedit("name:=uname").
                                                Set "mindq"
```

In above ex. statement we used ".*" to specify title for browser and page when those browser and page titles are charging regularly or dynamically.

But test automaters are using regular expressions for another purpose also in testing. To validate any type of value, we can use 'regexp' object. This object consists of below

a) Pattern Property :- We can use this property to specify [illegible] regular expression.

```
Set r =new regexp
r.pattern = "[A-Z][a-z]+"
```

b) Ignore case :- We can use this property to specify the case sensitive. By default this property value is false.

```
ex:  set r= new regexp
     - r. pattern = "[A-Z][a-z]+"
       r. ignorecase = true  'case in-sensitive
       r. ignorecase = false 'case sensitive.
```

c) Global :- We can use this property to apply pattern on total target string. By default this property value is true.

```
Ex:  set r = new regexp
     r. Pattern = "[0-9]+"
     r.global = true
if r.test("Jamesbond 007") then ' math from starting of string
end if
```

d) test() :- We can use this method to match given expression with given string. It returns true or false depending upon matching.

```
Set r= new regexp
  r.pattern = "[0-9]+"
if r.test("007") then
   msgbox "Number"
else
   msgbox "Not a number"
end if
```

e) Execute() :- We can use this method to search matched string in given string. This function returns array of matched values.

```
option explicit
dim r, i, matches
Set r= new regexp
  r.global =true
  r.pattern = "[0-9]+"
set matches = r.Execute["Sweety 45 anjali 04"]
for each i in matches
   msgbox i.value &" was matched at" &i.firstindex
Next
```

(f) Replace():- We can use this method to replace matched string with given string.

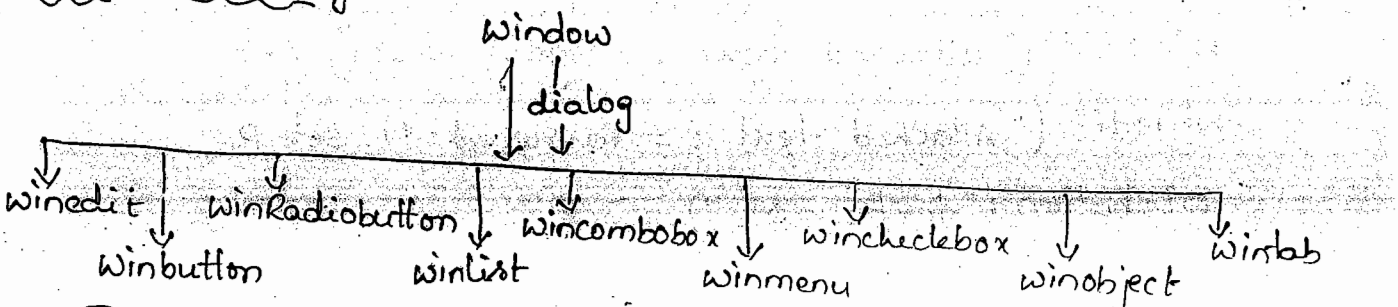
```
option explicit
dim r
set r = new regexp
r.global = true
r.pattern = "[0-9]+"
r.replace "sweety.45 Anjali 04", " "
```

Note:- From the above discussion, Regular expressions used for objects identification, data mining, object value type testing.

Working with standard window add-in

It is default add-in in QTP. If you launch QTP without selecting any add-in, then QTP will support automation on standard windows based s/w.

Object hierarchy:-



To work with standard windows s/w, test automaters can use below properties in descriptions.

objects	Identification Property
Window	Text
Dialog	Text
Winedit	Attached Text
Winbutton	text
WinRadiobutton	text
Winlist	attached text
Wincombobox	attached text
Winmenu	text

Eg1:- Write QTP script for login windows operation in flight Reservation s/w

```
With Dialog("text:=Login")
    .Winedit("attached text:=Agent Name:").set "Hindq"
    .Winedit("attached text:=password:").set "Mercury"
    .Winbutton("text:=Ok").click
End With
```

Eg: Write QTP script to automate login operation. If your login was success, then we can get "FlightReservation" window. If you gave invalid data then we can get "FlightReservations"

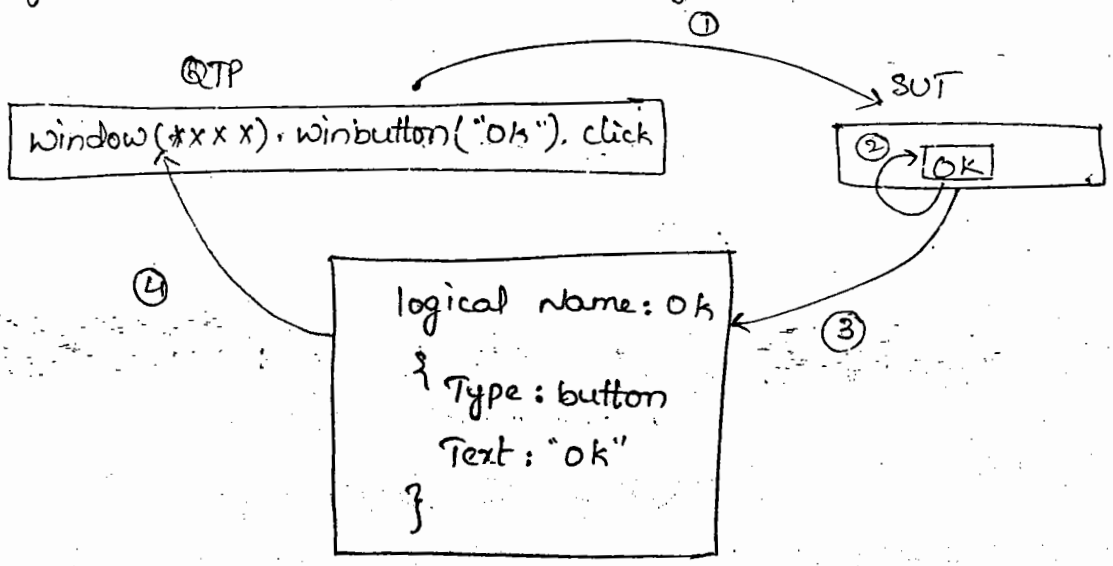
```
option explicit
Dim u,p
invokeapplication "c:\Programfiles\HP\QuickTest Professional\
                   Samples\flight\app\flight4a.exe"
u = inputbox ("enter a value")
p = inputbox ("Enter a password")
with Dialog ("text:=Login")
   .winedit ("attached text:=Agent Name").set u
   .winedit ("Attached text:=Password:").set p
   .winbutton ("text:=ok").click
End with
if window ("text:=Flight Reservation").exist then
Reporter.ReportEvent micPass,'Login Test","Successful"
else
Reporter.ReportEvent micfail,"Login test","unsuccessful"
End if
```

Recording :-

In previous scripts we followed descriptive programming but it will take more time to generate automation scripts. Due to this reason test automations are following one

in more way such as recording. If you generate script in recording then the script will be complex to maintain long time.

In recording mode QTP gathers operated objects of SUT information in object repository.



Object Repository.

step1: Start recording in QTP by tester.

step2: operate objects in SUT by tester.

step3: Gather details of objects by QTP

step4: Generate statement to automate that object operation by QTP.

To open object repository, we can follow below navigation

open a test in QTP (recorded) → resources menu → object repository

CSTF :- in Record/playblack methodology

step1: collect manual test cases to automate

step2:- Launch QTP with suitable add-in

step3:- Launch or open SUT

step4:- Create folder structure.

Step5: Study manual test cases and identify repeatable steps to make them as reusable actions.

Step6: Call those actions and record operations on SUT to generate test scripts.

Step7: Enhance each test script with multiple test data in .txt/.xls/.xml/QTP data table.

Step8: Make test scripts as batches to conduct regression testing on each modified SUT

CSTAF with descriptive Programming	CSTAF with Record/playback Methodology
1) Collect manual test cases	1) Collect manual test cases.
2) QTP & SUT launching	2) QTP & SUT launching.
3) Take the help of object spy and prepare keywords for those object descriptions	3) NONE.
4) Creating functions for reusable steps. Ex: launching, login, logout..... etc	4) Create actions for reusable steps. Ex: launching, login, logout.... etc.
5) Create test scripts by using keywords, functions typing required steps	5) Create test scripts by calling actions and recording
6) Make test scripts as batches for regression testing	6) Make test scripts as batches for regression testing.

Case study

Flight Reservation is a standard windows based s/w. This s/w database developed in MS-access. "QT_Flight32" is D.S.N. please automate below test cases in CSTAF with recording.

Test case 1:

Test step No	Description	Test data	Expected outcome
1.	Launch Flight Reservation S/W & do login.	D:\Flight-reserve \Testdata\Test login.xls	Flight Reservation window appears for valid data and Flight Reservations window appear for invalid data

Test case 2

step No.	Description	Test data	Expected outcome.
1.	Launch FR S/W & do login	• valid agent name and password.	Flight Reservation window opened
2.	open an existing order	D:\Flight-reserve\ \Test data\orders.txt	In that opened order, Name field value in alphabets with initcap as one or more words.

Test case 3

step No	Description	Test data	Expected outcome
1	Launch FR & do login	valid agent name and password	Flight Reservation window opened

2. Open existing group — None group window appears which was used in login time

Testcase 4

Step No.	Description	Test data	Expected outcome.
1.	Launch FR S/w and Do login	valid agent name & password	"Flight Reservation" window opened.
2.	Open an existing order.	All existing order numbers in database	Total = tickets * Price

Creating Reusable Actions:-

→ Launch QTP → File menu → New → test → insert menu → call to new action → Enter action name → write description for future understandbility → Select reusable action checkbox as "on" → Click ok → record corresponding operation on SUT → follow above navigation to create more reusable actions.

Parameterizing actions

After creating reusable actions, test automaters Parameterizing those actions for any input to run.

Open action → Edit menu → action → action properties → Parameters tab → click add(+) icon to create parameter → enter parameter name along with type, description, default value → Click add(+) icon to create more parameters if needed → Click ok in parameters window → Replace those Parameters in action code → follow above navigation to Parameterize more actions.

Ex1: Launch

```
' No parameters
invokeapplication "c:\Programfiles\HP\QuickTest Professional
                   Sameples\flight\app\flight4a.exe"
```

Ex2: Login

```
' 2 parameters
With Dialog("Login")
    .Activate
    .winEdit("Agent Name:").Set Parameter("un")
    .winEdit("Password:").Set Parameter("pwd")
    .winButton("ok").Click
End With
```

Ex3: order_open

```
' open parameter for order Number
With Window("Flight Reservation")
    .Activate
    .WinMenu("Menu").Select "File;open order....."
    With .Dialog("open order")
        .withcheckBox("order No.").Set "ON"
        .winEdit("Edit").Set Parameter("num")
        .winButton("ok").Click
    End With
End With.
```

Note:- Actions are 3 types, Local action, reusable action and external action. By default one test is one local action. Reusable actions invokable in any test. A called

reusable action in that test is called as external action.

→ To call a reusable action in a test as external, we can follow below navigation.

insert menu → call to existing action → browse actions file path → Specify action name → click ok → come back to test action → specify values to parameters → follow above navigation to call more actions if needed.

Runaction "Action name [filename]", oneiteration, values to Parameters

Note:- If you call any action by using "call to existing action" option, then that action is read-only.

Test script for test case:

userid	password	criteria
admin	mercury	valid
adm	mercury	invalid
admin	mindq	invalid
adm	mindq	invalid

D:\Flight_Reserv\Testdata\login.xls.

```
option explicit
Dim exo, wbo, wso, x, y, z, i, r
Set exo = createobject("Excel.Application")
Set wbo = exo.Workbooks.open("D:\Flight_Reserv\Testdata\
                              login.xls")
Set wso = wbo.Worksheets("Sheet1")
```

```
r = wso.Usedrange.rows.count
for i = 2 to r step 1
    x = wso.cells(i,1)  ' for Agent name
    y = wso.cells(i,2)  ' for Password
    z = wso.cells(i,3)  ' for criteria
  RunAction "launch [actions]", oneIteration  'By Navigation
  RunAction "login [actions]", oneIteration, x, y  'By Navigation.
  if window("Flight Reservation").Exist and z="valid" then
 Reporter.ReportEvent micPass, "Login test", "successful"
   window("Flight Reservation").close
elseif Dialog("login").Dialog("Flight Reservations").Exist and z="invalid"
                                                          then
 Reporter.ReportEvent micpass, "logintest", "unsuccessful"error"
  Dialog("login").Dialog("Flight Reservations").close
  Dialog("login").close
else
 Reporter.ReportEvent micfail, "login test", "unsuccessful"
End if
Next
   exo.Quit
 Set wso = nothing
 Set wbo = nothing
Set exo = nothing.
```

Test Script for testcase 2 :-

Order number is 6

Order 9 is a number

number 5 is also a order

an order 8

4 is an order number.

flight order 1 in flight reservation

D:\flight_reserv\Testdata\orders.txt

```
Option Explicit
Dim fso, fo, x, y
Set fso = Createobject ("scripting.filesystemobject")
Set fo = fso.opentextfile ("D:\flight_reserv\Testdata\orders.txt", 1, false)
RunAction "launch [actions]", oneIteration 'By navigation
RunAction "login [actions]", oneIteration, "admin", "mercury" 'By navigation
While fo.AtEndOfStream <> true
    x = fo.ReadLine
  set r = new regexp
  r.pattern = "[0-9]+" 'To match numbers.
  r.global = true
  set m = r.execute (x) 'Number array
  for each n in m
RunAction "orderopen [actions]", oneIteration, n.value
Next
```

```
r.pattern = "([A-Z][a-z]+[\s]?){1,}" 'To match one or more words
y = window("Flight Reservation").WinEdit("Name:").GetVisibleText()
if r.test(y) then
Reporter.ReportEvent micpass, "Name value type testing", "correct type
Else
Reporter.ReportEvent micfail, "Name value type testing", "wrong type
End if
set r = nothing
wend.
fo.close.
window("Flight Reservation").close.
set fo = nothing.
set fso = nothing.
```

Note :-

In recorded/playback methodology test script generated by calling actions, by generating statements via step generator; by recording. By using step generator, test automaters are creating vbscript function calls, QTP utility object access & out SUT test objects access.

Category in step generator	Purpose	Examples
Functions	To use vbscript builtin functions	invokeapplication(), split(), join(), cint(), cstr(), executefile(), exittest(), cdbl(), mid(), array(), left(), right() ....
Testobjects	To access objects in SUT depends add-in selection.	window, dialog, winedit, winbutton ..... webedit, browser, page, link, image ...

utility objects	QTP special objects	Datatable, xmlutil, Reporter, Repositories Collections, Environment, Encrypt.....

Test Script for test case 3 :-

```
RunAction "launch [actions]", one iteration
RunAction "Login [actions]", one iteration, "admin", "mercury"
  With Window("Flight Reservation")
     .Activate
     .WinMenu("Menu").Select "Analysis ; Graphs....."
   With .Dialog("Graph")
      .Maximize
     x = .WinObject("GS_Drawing").GetVisibleText()
  if instr(x,"admin") then
reporter.ReportEvent micpass, "Graph Content testing", "correct"
  else
reporter.ReportEvent micFail, "Graph Content testing", "incorrect"
 End if
   .close
End With
   .close
End with.
```

Test Script for test case 4

```
option explicit
dim con, rs, q, x, t, p, tot
```

```
Set Con = createobject ("ADODB.Connection")
    Con.open " DSN = QT_Flight 32"
Set rs = createobject ("ADODB.Recordset)
    q = "Select order_number from orders"
RunAction "launch [Actions]", one iteration → rs.Open, q.Con
RunAction "login [actions]", one iteration, "admin", "mercury"
While rs.EOF <> true
x = rs.Fields ("order_number").value
  RunAction "orderopen [actions]", oneIteration, x
t = window ("Flight Reservation").WinEdit ("Tickets:").GetvisibleText()
P = Window ("Flight Reservation").WinEdit ("Price:").GetVisibleText()
tot = window ("Flight Reservation").WinEdit ("Total:").GetVisibleText().
  P = mid (P, 2, len(P)-1)  ' to cut $
tot = mid (tot, 2, len(tot)-1)
if cdbl (tot) = Cint (t) * cdbl (P) then
   reporter.reportevent micpass, "calculation test", "correct"
else
   reportir.reportevent micfail, "calculation test", "incorrect"
End if
   rs.MoveNext
Wend
Window ("Flight Reservation").close
   rs.close
Con.close
Set rs = nothing
Set con = nothing
```

Note:- While using type casting functions all the decimal values to floar or ceil. Due to this reason some calculation test will go to fail with correct values also.

Test case 5 :-

Step no	Description	Test data	Expected outcome.
1.	Launch FR s/w and Do login	Valid agent name & password	Flight Reservation window opened.
2.	open existing order	valid order number	order opened
3.	open fax order, fill fields and click send	valid fax number and signature	"Fax Sent successfully" message appears.

1
4 → D:\flight_Reserve\Testdata\numbers.txt
5
8
9
10

Testscript

```
Option explicit
Dim fso, fo, x
Set fso, fo, x = Createobject ("scripting.filesystemobject")
Set fo = fso.opentextfile ("D:\flight_reserv\testdata\numbers.txt",
                           1, false)
Runaction "launch [actions]", oneIteration
Runaction "login [actions]", oneIteration, "admin", "mercury"
While fo.AtEndOfStream <>true
    x = fo.ReadLine
```

```
Runaction "orderopen[actions]", oneIteration, x
With window("Flight Reservation")
 .winMenu("Menu").Select "File;Fax Order...."
 .Dialog("Fax order No.1").winObject("Fax Number:").Type "111111:
 .window("Fax order No.1").RunAnalog "Track1"
 .Dialog("Fax order No.1").winButton("send").Click
wait 10
If .winobject("Fax Sent successfully").Exist then
 Reporter.ReportEvent micPass,"Fax testing","successful"
else
 Reporter.ReportEvent micfail,"fax testing","unsuccessful"
End if
End with
wend
 Window("Flight Reservation").close
fo.close
 set fo=nothing
 set fso=nothing.
```

Case Study1: [Working with signature objects]

In descriptive or in recording methodologies test automaters are using analog recording to record "signature" like operation.

Ex: Digital signatures, graph drawing, paint .....

To go to this recording mode we can follow below navigation Click start recording → automation menu → select analog recording → Select screen level or window level → show window if you

selected window → click start analog record → move mouse for signature or graph drawing or paint → automation menu → analog recording to stop → click stop recording.

→ desktop.RunAnalog "Trackname" 'screen level recording
Window("Paint").RunAnalog "Track1" 'window level recording.

Case study 2: Update object repository to identify dynamic objects.

In recording methodology some objects are windows, titles are changing dynamically. To identify these objects and windows successfully at runtime, we can change corresponding reference in object repository.

Ex:- Recording :- logical name : Fax order No.1

{ Text : Fax order no.1
Type : window
}

Runtime :- logical name : Fax order no.1

{ Text : Fax order no.*
Type : window
}

↘ Regexpression

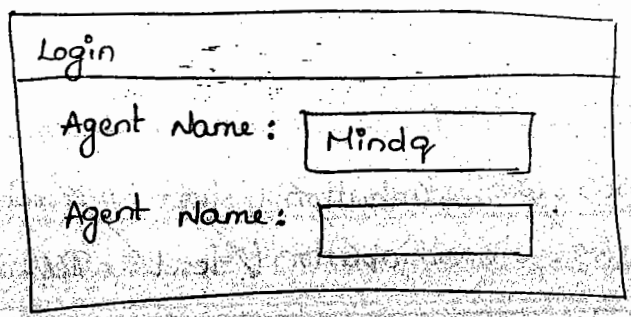
Resources menu → object repository → Select dynamic object reference → Select options icon for text property → select Regular Expression → write text with expression → click ok → close repository

Case Study 3 :- Distinguish similar objects

In most of the times test automaters are struggling with similar objects in same screen. In descriptive methodology test automaters can use "index" property in corresponding object statements.

```
Dialog("text:=login").winedit("attached text:=Agent Name:",
                              "index:=0").set "mindq
```

Above code can operate first agent name edit box



If we are in recording methodology QTP can try to distinguish similar objects by following default property lists. To identify an edit box, by default QTP can use "attached text", and "nativeclass" properties as mandatary "windowid" as assistive property. in recording time. For ex. more than one edit box "attached text, nativeclass and windowid" is same, then QTP struggled to identify specific edit box. To distinguish above like similar objects while running recorded code then test automaters can follows below types.

Way 1 :- Tools menu → object identification → Select SUT technology as environment → select similar objects type in available test → Add distinguishble properties to mandatory & assistive Properties list → click ok

Ex:

Mandatory	assistive
Attached text	windowid
nativeclass	

Mandatory	assistive
Attached text	windowid
native class	x
	y

Way 2: To distinguish similar objects we can depends on ordinal identifier concept. In this concept we can distinguish similar objects by using index or location. Similar objects index starts with 0. For ex.

0 ○ Mindq
1 ⊙ Mindq
2 ○ Mindq
3 ⊙ Testing
4 ○ Testing

<Parent>.winradiobutton("text:=Mindq").set "ON"
<Parent>.winradiobutton("text:=Testing").set

<Parent>.winradiobutton{"text:=Mindq","index:=1"}.set "ON"
<Parent>.winradiobutton{"text:=Testing","index:=0"}.set "ON"

Location: In general index is depending on name or text of similar objects, but location depends on type of similar objects

0 ○ Mindq
1 ⊙ Mindq
2 ○ Mindq
3 ⊙ Testing
4 ○ Testing

<Parent>.winradiobutton ("text:=Mindq", "location:=1").set "ON

<Parent>.winradiobutton ("text:=Testing", "location:=3").set "ON

Here location also starts with 0. This location numbering follows top to buttom and left to right of the screen.

Way 3:- To distinguish similar objects test automaters are using smart identification concept. From the concept tester is directing tool to identify an object depends on tester specified properties only instead of default properties.

Tools menu → object identification → select SUT technology as environment → select similar objects type → select enable smart identification check box → click configure → select base and optional properties → click ok → click ok.

Ex: For winedit.

Default Identification.

Mandatory	Assistive
Attached text Nativeclass	Windowid

Smart identification

Base	optional
Height	x
Width	y

Note:- Recording people are using way1 and way3 to distinguish similar objects but descriptive people can follow way2. to distinguish similar objects in same screen.

Case Study 4:- Identifying unknown objects

In discriptive programming first automaters are using sendkeys() method of window shell object to operate an unknown object or hidden object. But recording people are using virtual object concept.

Tools menu → virtual objects → new virtual object → next → select expected type [button, list, check box, table, radiobutton ....] → click next → Mark object → select unknown object area → click next → click next after confirmation of selected area → enter a name to virtual object → say yes or no to create more virtual objects → click finish.

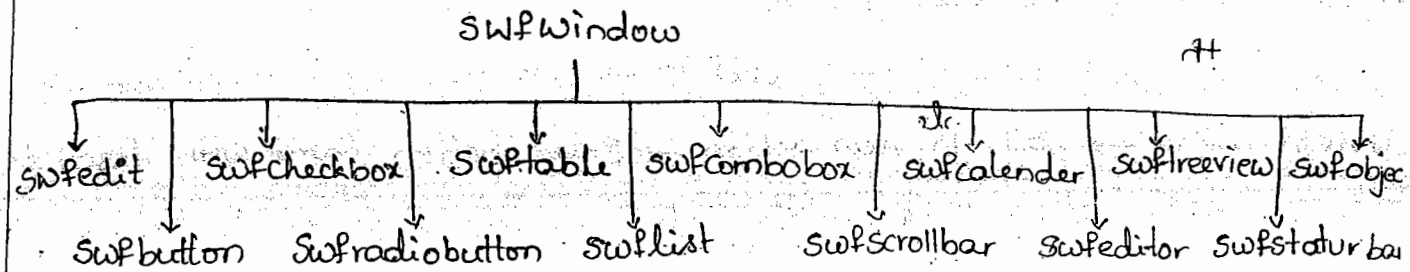
Note:- All virtual objects information is available in tools menu → virtual objects → virtual object manager.

Working with .Net Add in

.Net is a famous technology to develop Web-based & Windows based projects. If a s/w screens developed in .Net only, then we can launch QTP with .Net add-in. If a web site page developed in HTML with .Net then we can launch QTP with Web add-in and .Net add-in.

I) .Net windows based s/w testing:-

We can launch QTP with .Net add-in only to work with windows developed in .Net. While working with .Net add-in, QTP follows below objects hierarchy.



SWF stands for standard window form.

Case Study :-

Test case 1

Step No	Description	test data	expected outcome
1.	Launch class s/w	.exe path	frmclass window opened

Test case 2.

Step No	Description	test data	expected outcome
1.	Launch class s/w	.exe path	frmclass window opened
2.	click add, fill fields and click save button	valid class id, class name & strength	saved data inserted into front end table as a new record

Test case 5

step No.	Description	Test data	Expected outcome
1.	Launch class S/w	.exe path	frmclass window opened.
2.	front end table consists of records.	None	equal to data in data base.

Note:- This .Net project data base developed in sql server. Here Server name is SYS and followed windows authentication. SampleDB is database name and dbo.class is table name. Here table consists of 3 columns such as classid, className and Strength.

CSTAF with recording

Step1: Creating reusable actions.

Launch QTP with .Net add-in → file menu → new → Test → insert menu → call to new action → Enter action name with description → click ok → Record reusable code → follow above navigation to create more actions → Save actions in a file.

Note:- To run actions for different data in test we can Parameterize those actions.

Note:- After Completion of actions creation and parameterization, then test automaters will save those actions in a separate file.

Launch action:-

' No parameters

invokeapplication "E:\Program\dotnetapp\dotnetapp\bin\debug\dotnet app.exe"

adding action:-

```
' three parameters for classid, classname and strength
    With SwfWindow("frmclass")
        .Activate
        .SwfButton("Add").Click
        .SwfEdit("txtid").SetParameter("a")
        .SwfEdit("txtname").SetParameter("b")
        .SwfEdit("txtsize").SetParameter("c")
        .SwfButton("save").Click
    End With.
```

Testscript for Testcase1

```
Runaction "launch [actions]", oneIteration
if SwfWindow("frmclass").Exist then
Reporter.ReporterEvent micpass, "window existance testing", "Existi
    SwfWindow("frmclass").Close
else
Reporter.ReporterEvent micfail, "window existance testing", "Not existi
End if.
```

Testscript for Testcase2.

Classid	cname	Size
100	MCA	200
101	B.Tech	440
102	M.Tech	542

D:\School\Testdata\Book1.xls.

```
option explicit
Dim exo, wbo, wso, r, x, y, z, i, j, ci, cn, cs, n, flag
Set exo = Createobject("Excel.Application")
Set wbo = exo.workbooks.open("D:\School\Testdata\Book1.xls")
Set wso = wbo.worksheets("Sheet1")
r = wso.usedrange.rows.count
Runaction "launch[actions]", oneIteration
for i = 2 to r step 1
x = wso.cells(i, 1)
y = wso.cells(i,2)
z = wso.cells(i,3)
Runaction "adding[actions]", oneiteration, x, y, z
n = swfwindow("frmclass").swftable("grdclass").Rowcount
flag = 0
for j=0 to n-1 step 1
ci = swfwindow("frmclass").swftable["grdclass"].Getcelldata(j,0)
cn = swfwindow("frmclass").swftable("grdclass").Getcelldata(j,1)
cs = swfwindow("frmclass").swftable("grdclass").Getcelldata(j,2)
If cint(x) = cint(ci) and strcomp(cstr(y), cstr(cn)) and
cint(z) = cint(cs) then
Reporter.ReportEvent micpass, 'Dotnet table test", "added"
flag = 1
End If
Next
If flag=0 then
Reporter.ReportEvent micfail, "DotNetTableTest", "Not added"
End if
```

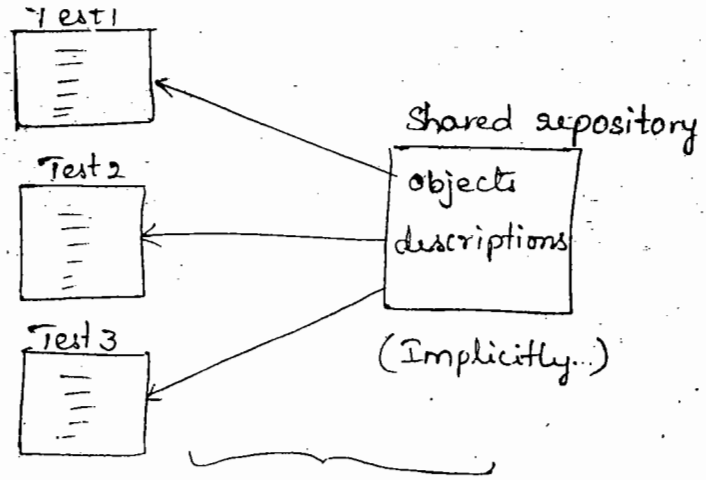
```
Next
Swfwindow("frmclass").close
exo.Quit
set wso = nothing
Set wbo = nothing
set exo = nothing.
```

TestScript for Testcase 3

```
option explicit
Dim con, rs, x, y, z, i, Ci, cn, cs, flag, r.
set con = createobject("ADODB.Connection")
con.open "Provider = SQLOLEDB; server = SYS; Database = SampleDB;
                    Trusted_Connection = yes;"
set rs = createobject("ADODB.Recordset")
rs.open select * from dbo.class, con
RunAction "launch [actions]", oneIteration
While rs.Eof <> true
x = rs.fields("classid").value
y = rs.fields("classname").value
z = rs.fields("strength").value
r = swfwindow("frmclass").swftable("grdclass").Rowcount
for i = 0 to r-1 step 1
Ci = swfwindow("frmclass").swftable("grdclass").Getcelldata(i,0)
Cn = swfwindow("frmclass").swftable("grdclass").Getcelldata(i,1)
Cs = swfwindow("frmclass").swftable("grdclass").Getcelldata(i,2)
If cint(x) = cint(Ci) and strcomp(y, Cn) and cint(z) = cint(Cs) the
reporter.reportevent micpass, "data base test", "matched"
flag = 1
End if
```

```
Next
If flag = 0 then
  reporter.reportevent micfail, "Database testing", "mismatch"
End if
  rs.movenext
Wend
    rs.close
    Con.close
  set rs = nothing
  Set con = nothing
   SwfWindow("frmclass").close.
```

CSTAF with discriptive programming



adding objects methodology

To create a shared repository test automators can follow below navigation.

"launch QTP with Suitable add-in & launch corresponding SUT → Resources menu → object repository manager (ORM) → object menu → add objects (Take the hand icon show the screen) → Show testable screen in SUT → click 'ok' after Confirmation → Select All object types (Radio button) → click ok → continue this adding for all testable Screens in SUT

→ Save finally with finalky filename.tsr (test shared repository)

→ Shared repository is team level.

To associate a shared repository to a specific test we can follow below navigation.

File menu → New Test → Resources menu → Associate Repositories → click addition to browse shared repository file path → select it click open → (Eg: Action1) Associate ↓ test action → By default it takes action1 → click "ok"

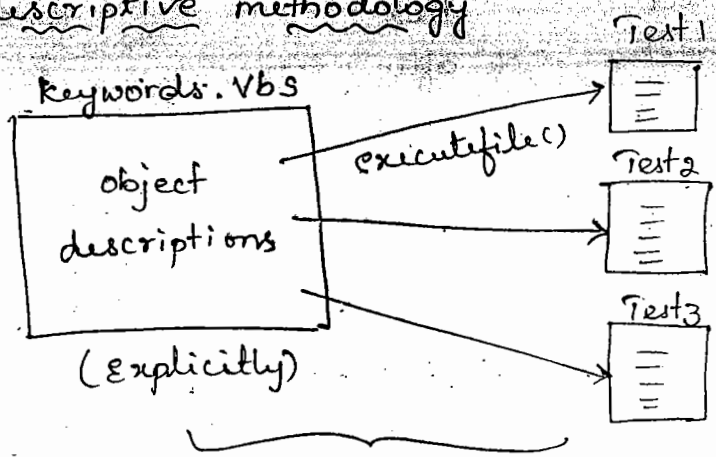
CSTAF with adding objects methodology

It is a new methodology in functional test automation. This methodology is only supporting by HP QTP. In descriptive methodology test automaters are getting descriptions of objects in SUT by using objects spy (or) other spy s/w. And then test automaters are creating keywords explicitly.

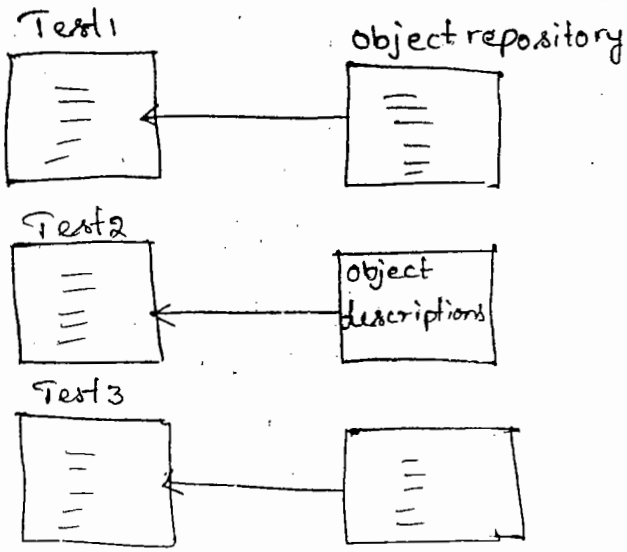
In recording methodology QTP tool can gather description of objects implicitly. And maintain them in object repository for every test.

In adding objects methodology Test automations are creating a shared repository implicitly. This repository consists of descriptions of all testable objects in SUT.

In descriptive methodology



In descriptive methodology

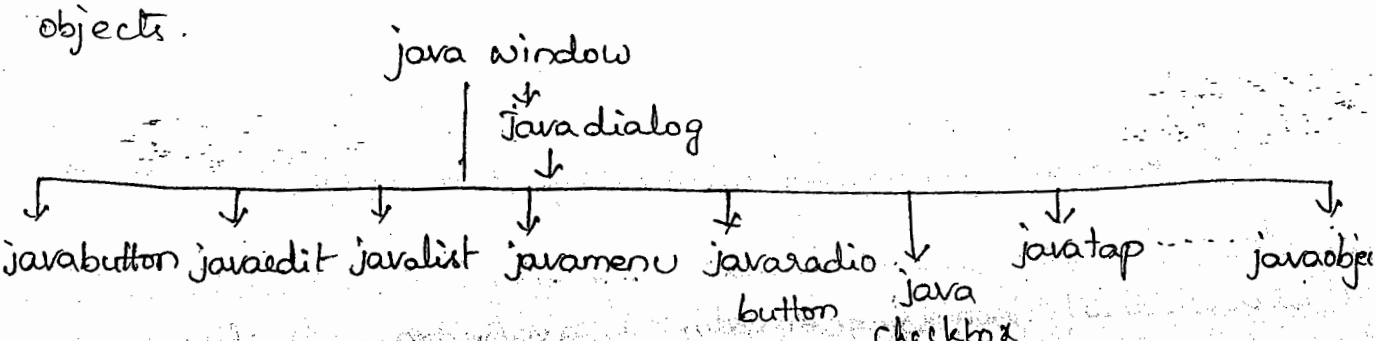


Note ⇒ After associating a shared repository to a test, test automati can open that repository using

Resources menu → object repository (ctrl+R)

→ To generate script, test automators can drag & drop objects from object repository to test pane. (QTP tool test page)

Case study :- A calculator developed in java as one-tier s/w, t test this java s/w we can launch QTP with java s/w. Whi working with java projects, we can get below hierarchy of objects.



→ If any java object not able to recognized, that time my tool is taken java object.

→ In web, if any web object is not able to recognized that time my tool is taken webelement.

In java calculator we can get addition, subtraction & multiplic We can automate below test cases.

Step No	Description	Test data	Expected outcome
1.	Open calculator	None	Calculation window opened
2.	Enter data and Perform addition	D:\Calculator\ Testdata\data.txt	Correct addition

→ Parameterization is two types

→ Data type objects.

→ name of data type object

Shared repository created

'Associate repository to test

option explicit

Dim fso, fo, x, y, i, j, z, l1, l2, d1, d2

Set fso = createobject("scripting.filesystemobject")

Set fo = fso.opentextfile("D:\calculator\testdata\data.txt", 1, false)

While fo.AtEndOfStream <> true

x = fo.readline

y = split(x, " ")

l1 = len(y(0))

for i = 1 to l1 Step 1

d1 = mid(y(0), i, 1)

Java window("java swing calculator").JavaButton(d1).click

Next

javawindow("javaswingcalculator").javaButton("+").click

l2 = len(y(1))

for j = 1 to l2 step 1

d2 = mid(y(1), j, 1)

javawindow("javaswingcalculator").javabutton(d2).click

Next

javawindow("javaswingcalculator").javabutton("=").click

z = javawindow("javaswingcalculator").javastatictext("o(st)").

Getroproperty("attached text")

if cdbl(z) = cdbl(y(2)) then

Reporter.ReportEvent micpass, "calculation test", "correct"

else

Reporter.ReportEvent micfail, "calculation test", "incorrect"

End if

wend

fo.close

set fo = nothing

set fso = nothing

56	42	198
17	52	69

Notepad

Note:- In above calculator s/w o/p object lable is initially '0' But o/p will be changinging depends on i/ps and operations. Due to this reason we charged o/p object lable from 0 to ".*" in shared repository.

Testcase 2:-

step No	Description	Test data	expected outcome
1.	Launch java calculator Perform arithmetic operations" such as add, sub, mul & division	D:\javacalc\ Testdata\ expressions.txt	valid o/p for add, sub, mul and division.

Testscript

```
                                        y(0)    y(1)
    Shared repository associated        (23+45) = 68
    option explicit                     45-44   = 1
Dim fso, fo, x, y, i, l, d, z           44/2    = 22
                                        23*2    = 46
set fso = createobject("scripting.filesystemobject")
set fo = fso.opentextfile("D:\javacalc\testdata\expressions.txt", 1,
    While fo.atendofstream <> true.                    false)
        x = fo.readline
        y = split(x, "=")
        l = len(y(0))
    for i = 1 to l step 1
        d = mid(y(0), i, 1)
javawindow("javaswingcalculator").javabutton(d).click
    Next
Javawindow("javaswingcalculator"), javabutton("=").click
z = javawindow("java swingcalculator").javastatictext("o(st").
                                    Getropeprty("attachedtext")
    if cdbl(z) = cdbl(y(1)) then
```

```
Reporter.ReportEvent micPass, Arithmetic operations Testing, Correct
Else
Reporter.ReportEvent micFail, "Arithmetic operations testing", "incorrec
End if
Wend
fo.close
set fo = nothing
set fso = nothing.
```

Working with web add-in and java add-in

→ java code is running on server is called as java servelets.

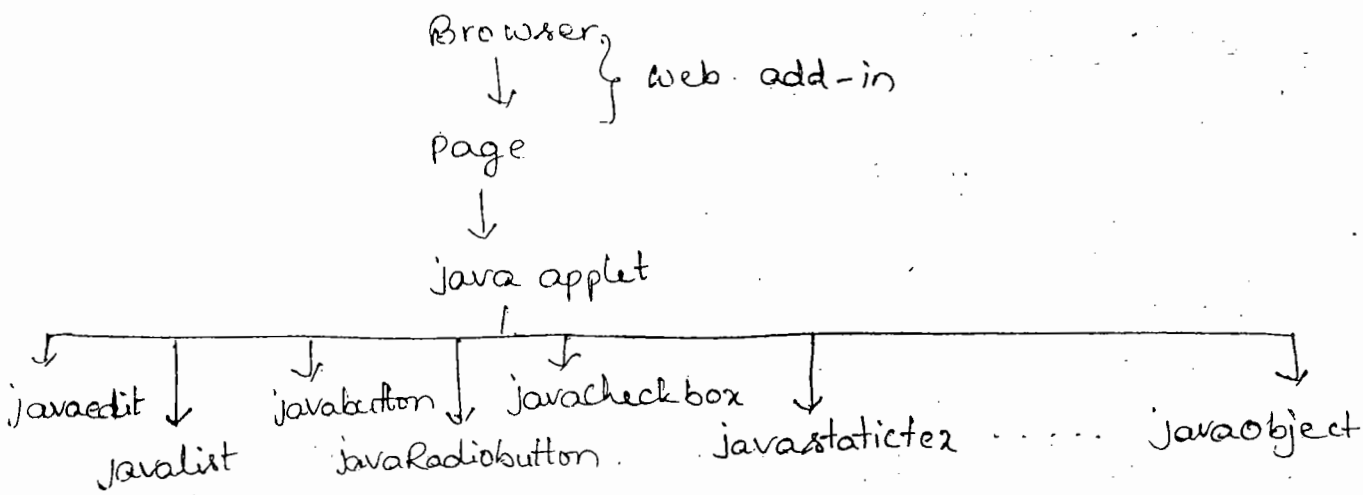
→ java code is running on client is called as java applet.

→ any url coming with .aspx, may that webpage some .Net controls are there

Some times developers are designing web pages by using html and java controls. To test this type of web pages, we can launch QTP with web add-in and java add-in. Here QTP statements are forming like as felow.

```
Browser("title").Page("title").Javaapplet("xxxx").Javabutton("xxx").
                                                          Click
```

→ If any screen of java is opened in web page then it is called as javaapplet.



Continuation at last pages.

b) User-defined environment variables:-

To store reusable data like file paths, user ids, Passwords..... etc, we can use user defined environment variab To create user-defined environment variables, we can follow below navigation.

File menu → Settings → Environment → Select variable type as user defined → click add(+) icon → Enter variable name → specify value → click ok → click add(+) icon to create more environment variables → click export after creating required environment variables → Specify xml file path

After saving environment variables in xml file, test automaters are associating that xml file with every test to use those variables.

environment.loadfromfile ("xmlfilepath")

After associating xml file with test, test automaters can use environment variables like as below

environment.value ("Variable name")

Driverscripts writing

In general manual testers or automation testers are applying tests on SUT as batches. To make test scripts as batches in QTP, we can follow any one of two ways.

a) Test batch runner:-

Start → Programs → QTP → Tools → Test Batch Runner → click add icon → browse required test path → click open

→ click add icon → to browse more test paths → click Run icon finally

Note:- To use this concept we can get permission by setting tools menu → options → select test batch runner check box → click apply → click ok in QTP.

b) AOM (Automation object model)

This is an alternative way to run tests as batches. In this AOM concept, test automaters are automating QTP launching, required test opening, running, results saving and QTP closing. To automate QTP launching we can use below code

```
option explicit
Dim qtApp
Set qtApp = Createobject ("QuickTest.application")
qtApp.Launch
qtApp.visible = True.
```

We can write above like code in notepad & save with driver.vbs.

After launching QTP we can use below code to open a test and run.

```
qtApp.open "D:\ path of test"
qtApp.Test.Run.
```

After running a test, test automaters are automating results generation, by using below code.

```
option explicit
Dim qtApp, qtrs
Set qtApp = Createobject("QuickTest.Application")
qtApp.Launch
qtApp.Visible = True
Set qtrs = Createobject("QuickTest.RunResultsOptions")
qtrs.ResultsLocation = "Path of the file"
qtApp.open "Path of the file"
qtApp.Test.Run qtrs, true
qtApp.Quit
Set qtrs = Nothing
Set qtApp = Nothing
```

To prepare driverscripts for multiple tests or batch, we can extend above code like as below.

ex: Testnames

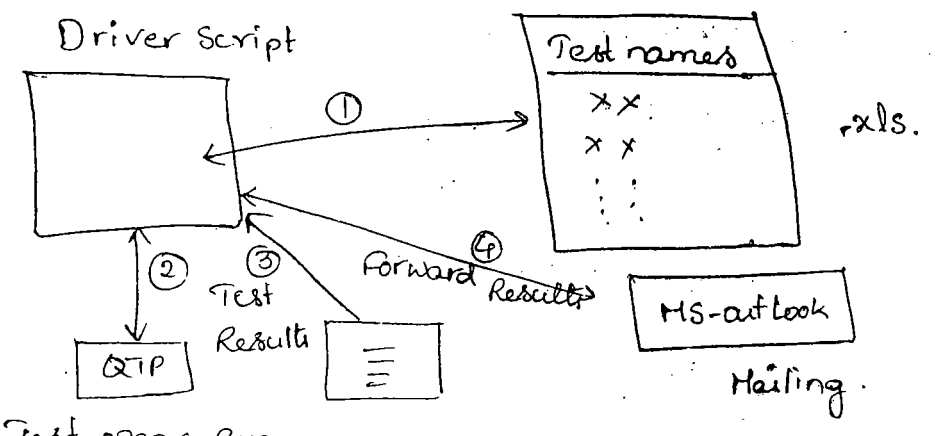
```
Login_test
mail_open        c:\Tests.xls
mail_reply
mail_forward
Logout_Test
```

To run this prepare test scripts. Store results in c:\Testresults folder.

```
option explicit
Dim qtApp, qtrs, wbo, wso, exo, r
Set exo = createobject("excel.application")
Set wbo = exo.Workbooks.open("c:\Tests.xls")
Set wso = wbo.Worksheets("Sheet1")
Set r = wso.usedrange.Rows.count
```

```
Set qtApp = CreateObject("QuickTest.Application")
    qtApp.Launch
    qtApp.Visible = True
Set qtrs = CreateObject("QuickTest.RunResultsOptions")
qtrs.ResultsLocation = "c:\Testresults"
    for i = 2 to 2 step 1
        x = wso.Cells(i,1)
        qtApp.Open "c:\" &x
        qtApp.Test.Run qtrs, True
    next
    qtApp.Quit
    Set wso = nothing
    Set wbo = nothing
    Set exo = nothing
    Set qtrs = nothing
    Set qtApp = nothing
```

After writing driver script for multiple test specified in Excel sheet, test automaters are extending those scripts with local mailing S/w (MS outlook) to forward test results as a mail.



```
option explicit
Dim qtApp, qtrs, exo, wbo, wso, r, x, i, msol, m
set qtApp = createobject ("QuickTest.Application")
   qtApp.Launch
   qtApp.visible = True
set qtrs = Creatiobject ("QuickTest.RunResultsoptions")
   qtrs.ResultsLocation = "C:\TestResults"
set exo = Createobject ("Excel.Application")
   exo.visible = True
set wbo = exo.workbooks.open ("C:\Hests.xls")
set wso = wbo.worksheets ("sheet1")
 r = wso.usedrange.rows.count
set msol = createobject ("outlook.Application")
for i = 2 to r step 1
      x = wso.cells (i, 1)
      qtApp.open "C:\" &x
      qtApp.Test.Run qtrs, true
      set m = msol.creatitem (0)
      m.to = "testlead @ mindq.com"
      m.subject = "Test result of " &x
      m.cc = "pm@mindq.com"
    m.body = "Hai test lead, Plase check my test results"
    m.attachments.add "C:\Testresults\report" &i-1
     m.send
     Set m = nothing
Next
  qtApp.Quit
   exo.quit
   mslo.quit
```

```
Set msIo = nothing
Set wSo = nothing
Set wbo = nothing
Set exo = nothing
```

Recovery Scenarios

While running test scripts as batches we are getting run-time errors rarely. To recover these run-time errors automatically, we can use recovery scenarios concept in QTP. Recovery scenarios are handling 4 types of Problems while testing.

a) Test-Run Error

Ex: Errors related to vbscript.

b) POP-up window

Ex: Window related errors in SUT.

c) object state.

Ex: A button is disable before click.

d) Application crash

To handle above 4 types of problems, QTP can apply any one of 4 types of solutions.

a) Mouse click or keyboard click.

b) Function call Ex: a set of code to recover from error.

c) Restart windows.

d) close SUT and reopen.

After recover from errors, QTP can continue remaining testing in any one of below ways

(Post recovery)

a) Repeat current step and countinue

b) *** Goto next step. [on error resume next]

c) Goto next test iteration.

d) Goto next test or action.

e) Restart current test

f) Stop test run.

Navigation

Identify problem in SUT while testing → file menu → new → function library → write a function to recover problem in SUT → Resources menu → Recovery Scenario manager → click new scenario icon → click next in welcome screen → select type of problem in SUT (Ex: POP-UP) → Click Next → Show problem to tool → click Next → click Next → select recovery type (Ex: function call) → Provide recovery details. → deselect another recovery operation check box → click Next → select Post recovery option (Ex: proceed to next step) → click Next → enter scenario name and description → click next → add scenario to current test → click finish → Save scenario with filename.qrs. (QTP recovery scenario) → click close

Note -- After completion of recovery scenarios creation and saving, test automaters are associating recovery scenarios with required test.

Open corresponding test → file menu → Settings → Recovery → add (+) icon → browse previously saved recovery scenario file.qrs → click add scenario → click apply → click ok.

Ex 1 :- Problem type : POP-UP window

Recovery type : function call.

Post recovery : proceed to next step.

Error window name. Flight Reservations

Recovery function:

```
Function recovery(byval win)
  with window("text := Flight Reservation")
  with . Dialog("Text := open order")
    . Dialog("text := Flight Reservations"). winbutton["text:=
                                                    Ok"). click
    . winedit("window id := 1016"). set "1"
    . winbutton(" text := Ok"):click
  End with
  End with
End function.
```

Ex 2: Problem type : POP-UP window

Recovery type : function call

Post recovery: Proceed to next step

Error window : Flight reservations (wrong password)

Recovery function:

```
function recovery1(byval win)
  with dialog("text:= Login")
. dialog("text := Flight Reservations"). winbutton["text:=ok").
                                                    click
```

```
    .Winedit("attached text:=password:").set "mercury"
    .Winbutton("text:=OK").click
  End with
End function.
```

Test Results Conversion

Unlike to other tools QTP will provide a facility to convert test results into .html, .doc, .pdf

Run test to get test results → file menu → Export file → select type as html, doc, Pdf → browse file name to save → click ok.

Screen recorder

QTP provides desktop recorder while running tests. To activate this option we can follow below navigation

Tools menu → options → Run → Screen capture → select save movie to results always → click apply → click ok.

To see recorded movie we can follow below navigation in test results window

view menu → Screen recorder.

Active Screen:-

While recording a test QTP capture status of SUT every time. To see SUT snapshots after recording, we can use view menu → Active Screen.

Working with passwords:-

To read passwords in vbscript we can use inputbox() function. But this function shows typed password on the desktop. There is no other function in vbscript to hide

Password

To hide passwords in testscript we can use password encoder. It converts orginal password into encrypted passwords.

start → programs → QTP → Tools → Password encoder → Type orginal password → click generate → copy encrypted passwords to use in tests.

(OR)

```
x=Crypt.Encrypt("Mercury")
msgbox x
```

In above navigation or in code we can type orginal password atleast one time. To prevent this orginal typing the best solution is .Net factory.

```
option explicit
dim myform, mytext, mybutton, pos, a, b, x, y
Set myform = Dotnetfactory.createInstance("System.windows.
                forms.form", "system.windows.forms")
Set mytext = Dotnetfactory.createInstance("system.windows.forms.
                textbox", "system.windows.forms")
Set mybutton = Dotnetfactory.createInstance("system.windows.forms.
                Button", "system.windows.forms")
Set pos = Dotnetfactory.CreateInstance("system.Drawing.Point",
                "system.drawing", x, y)
Pos.x = 90
Pos.y = 100
mybutton.location = pos
→ Mytext.Location = Pos
→ Mytext.width = 200
Mytext.UseSystempasswordchar =
                true
```

```
Pos.x = 100
Pos.y = 130
mybutton.location = Pos
mybutton.text = "Submit"
myform.Controls.add mytext, my
myform.Controls.add mybutton.
myform.cancelbutton = mybutton.
myform.showdialog
b = mytext.text
a = inputbox("Enter agent name")
Dialog("Login").Activate
Dialog("Login").winEdit("Agent Name:").set a
Dialog("Login").winEdit("Password:").set b
Dialog("Login").winButton("OK").Click
```

Debugging:-

Like as in programming languages QTP allows you to run script line by line by using F11 & F10 functional keys.

Transactions:-

To calculate execution time of QTP code in your test script we can use transaction point Concept.

open test → select position → insert menu → start transaction → enter transaction name → click ok → select another position in script → insert menu → end transaction

```
- - - - - -
- - - - - -
Services.startTransaction "name"
- - - - -
- - - - - -
Services.EndTransaction "name"
```

(OR)

```
x = now
- - - - - - - - -
- - - - - - - - -
y = now
msgbox datediff ("s", x, y)
```

(OR)

```
mercurytimers.Timer.Start
- - - - - - -
- - - - - - -
mercurytimers.Timer.Stop
msgbox mercurytimers.Timer.ElapsedTime
```

Ways to launch SUT :-

a) Launching windows-based SUT :-

→ invokeapplication ("Path of .exe")

→ SystemUtil.Run "Path of .exe"

b) Launching web-based SUT :-

→ invokeapplication ("Path of browser")

Browser ("title:=about:blank").navigate "URL"

→ invokeapplication "path of browser URL"

→ SystemUtil.Run "Path of browser", "URL"

→ Set ie = CreatiObject ("InternetExplorer.Application")

ie.navigate "URL"

ie.visible = true.

API testing / Component Testing

Sometimes organizations are developing internally working S/W. These S/w's are not having user interfaces (front end). To connect to these S/ws we can use an application programming interface called as Extern object.

Ex: Write QTP script to test ~~Find~~ Window FindWindow(), which is available in user32.dll. This function will take two inputs and returns one o/p. Here i/p s are required window name and parent name. Here o/p is window id.

```
option explicit
    Dim x
Extern.Declare micHwnd,"FindWindow","user32.dll","FindWindowA",
                                        micString, micString.
x = Extern.FindWindow("Notepad", vbNullString)
if x = 0 then
MSgbox "Notepad window not found"
else
MSgbox "Notepad window found"
end if
```

Ex2:- Write QTP script to test setWindowText() function in user32.dll library. Here function will take two inputs and return one output. Inputs are windowid and name to Set.. o/p is long integer.

```
option explicit
  dim x
Extern.Declare micHwnd, "FindWindow", "user32.dll", "FindWindowA
                                        micString, micString.
```

Extern.Declare micLong, "SetWindowText", "User32.dll", "SetWindowTe
micHWnd, micString.

x = Extern.FindWindow("Notepad", vbNullString)

Extern.SetWindowText{x, "Hindq"}

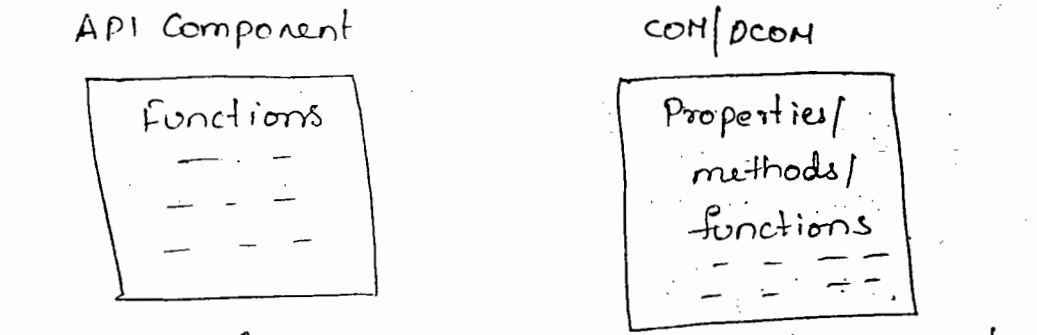
Declare() :- We can use this method to declare testable Component with inputs and output.

Extern.Declare "Returntype", "function name", "Libraryname", "aliasname", "Input types"

To specify inputtypes and return type, we can use below Constants.

For integer → micInt
long integer → micLong
character → micchar
String → micString
windowid → micHWnd

COM/DCOM testing [Componet object model/Dynamic Component object model]



From the above diagram COM and DCOM Consists of a set of properties and methods. Here properties are specifying status of objects and methods are specifying

actions of an object.

Ex1: To work with MS-Excel with out depending on interface, we can use xd excel.Application.class.

Ex2: To work with MS-Word with out depending on interface, we can use "word.Application" class.

Ex3: To work with internet explorer browser with out depending on interface, we can use "internetexplorer.application" class.

Like in above examples QTP have many com objects to work with internally.

Note:- To work with PDF like non Microsoft s/ws, we need user defined COM in QTP.

for ex "LearnQuickTest.ManipulatePDF" is a user-defined Component.

Navigation to download:-

Open www.learnqtp.com

↓

type "manipulati PDF" in search button

↓

we can get "Learn QuickTest API: manipulating PDFs in QTP"

↓

Click download link.

↓

Save downloaded file in "C:\learnQTP"

↓

Extract here on downloaded zip file

↓

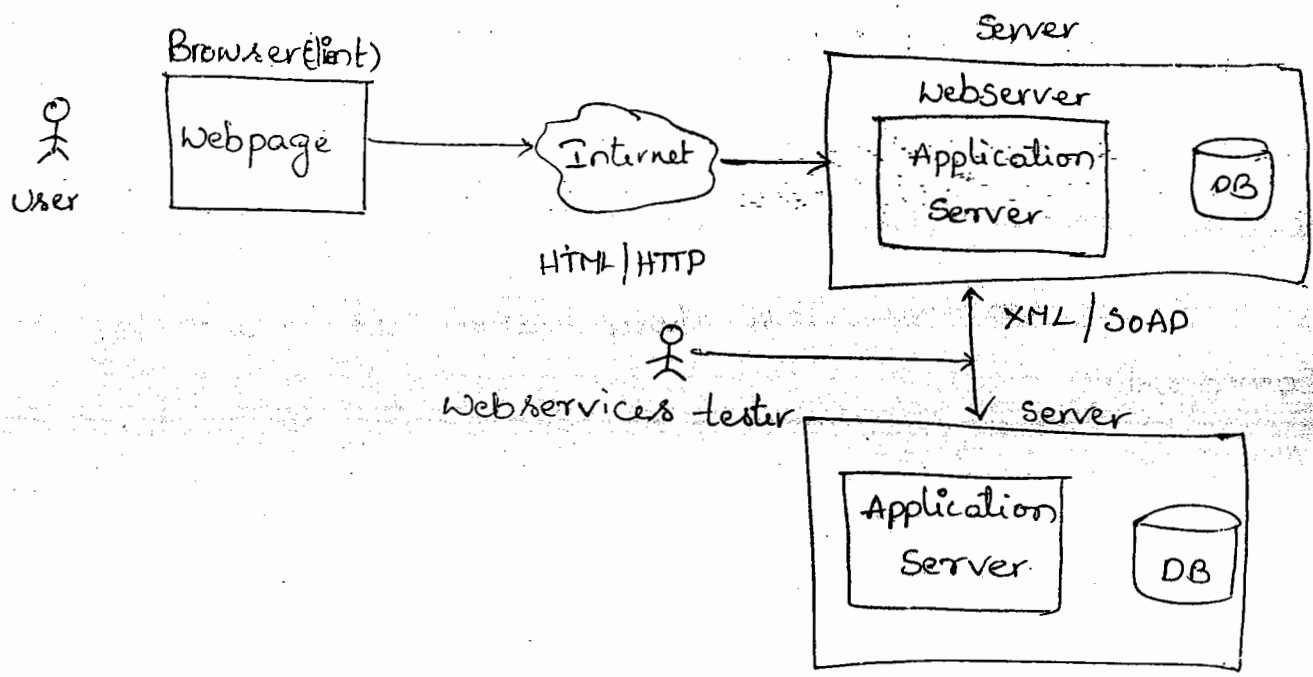
Open folder and click install.

Ex: Write QTP script to find no. of pages in given PDF file.

```
Option explicit
Dim Pdfo, n
Set Pdfo = CreateObject("LearnQuickTest.ManipulatePDF")
n = Pdfo.GetNumberOfpages("Path of PDF file")
msgbox "No. of Pages" &n
```

Middleware in b/w QTP and PDF

Webservices Testing or SOA testing (Service oriented Application)



Navigation to test web services

launch QTP with Web services add-in → Insert menu → Automation → Web service testing wizard → Next → browse path of Service oriented application WSDL library (Web services definition language → Select required service in list → Provide inputs if required → click finish → Run generated script in QTP → Analyze results to confirm

the correctness of corresponding service output.

ex: WSDL Library.

http://www.w3schools.com/webservices/tempConvert.asmx?wsdl"

The above WSDL library Consists of two services such as celcisus to foreignheit & Foreigheit to Celcisus.

QTP with QC

QTP allows you to launch from quality center when testers are using quality center for test management. To launch QTP from quality center we can follow below navigation

Launch Qcbin → Login to current project by using userid & password → click close in welcome screen → Goto testplan → select a testing (functional testing) topics or subject → Select an existing test case → change test type as automation, QTP → Test script → Launch QTP → Create test script as descriptive, recording or adding objects → Save test → Close QTP → Logout in Qc.

QC with load Runner

Load runner tool is allowing you to launch from Qc for Performance testing.

Launch Qc bin → Login to by tester by using uid & pwd → click close in welcome screen → Test plan → select testing subject as performance testing → select a test case and change test type as LR scenario → Test script → click launch → Record Vuser script for one user → Save Script → close Vugen in load Runner → Refresh Qc to see recorded script → logout

Note :- Qc is integrating with HP QTP & HP load runner. Because Qc is a management tool of HP.

sc

Selenium :-

→ Web application functional testing tool.

→ Released by Selenium Hq,

→ Consists of 3 components such as Selenium IDE, Selenium RC and Selenium Grid.

→ Allowing scripts in core java, Html, Python, Ruby, Perl & Rexx, C#

→ Selenium components run on windows, Linux, Mac

→ Selenium IDE is used for recording methodology.

→ Selenium RC (Remote control) is used for descriptive methodology.

→ Selenium Grid is used for making test scripts as batches to run on multiple browsers.

→ Core java is prefarable lagwange for selenium scripting because Selenium Components are developed in java.

→ open source testing tool. and free download from www.seleniumhq.org.

→ Selenium Components are Customizable because we are getting selenium source code from seleniumhq website.

Selenium IDE :

IDE stands for integrated development environment. This Component is used to record a web site operations in Mozilla Firefox browser, because selenium IDE is a add-on in mozilla firefox browser. Selenium IDE recent version for useful is 1.0.10. After recording, this script will be visible in table format and script format like in QTP keyword view

Recorded script of Selenium IDE is converted to html, java, python, Ruby, Rexx & perl.

Selenium RC :-

RC stands for remote control. We can use this component to write descriptive programs in core java. Most of the selenium testers are using this component because recorded scripts are not maintainable long time. It can support multiple browsers like mozilla firefox, Internet explorer, Google crome, opera & Safari.

→ Descriptive programs in core java are enhancable and modifiable in core future.

→ Selenium RC scripts are in core java & selenium objects.

→ Some times web pages are returning windows or alert dialogs or pop-ups. To handle these windows, selenium RC is Providing special objects and methods.

→ Selenium RC allows you to get data from excel & write data to excel.

→ Selenium RC will access data base of website undder testing by using JDBC driver in core java.

→ Selenium RC work with xml files also

Selenium Grid

After Completion of scripts generation in RC, test automaters are making them as batches. QTP can support one batch execution at a time but selenium grid can

support multiple batches execution on multiple browsers at a time.

Selenium IDE installation

open mozilla firefox browser → Launch http://seleniumhq.org/ → Download → click 1.0.10 → click install now → Restart firefox after installation.

Create Test using Selenium IDE

Launch testable website → tools menu in firefox → Selenium IDE → new test case → click record → operate website → click record icon once again to stop recording.

After completion of recording, tester can use file menu → Save Test case to save that recorded code.

After saving recorded code, we can use "play current test case" icon. While running recorded code we can use fast mode or slow mode.

To run all opened tests as a batch we can use "play entire test suite" icon.

After recording, recorded code will be visible in two views such table and source. Source view is showing recorded code in html by default. To change to other languages options → format → select required language.

Selenium RC: Installation & Configuration

a) Installing JDK 1.5.0.11

b) → Create a new folder with some folder name

c) → Copy jdk s/w, in that & save it in folder

d) Create new folder in that folder with folder name as lib.

e) open lib folder and paste four jar (java archive) files
Junit - 4.5.jar, Jxl.jar, Selenium-java-client-driver.jar,
Selenium-Server.jar.

f) Install eclipse

g) Goto workbench → file menu → new → java project → Enter a project name → click Configure JRE's → click add → Select standard VM → click Next → browse JRE files like JDK → click Finish → Select JDK check box → click ok → click finish

h) Configure build path

Select created project → Select JRE System Library → Right click → build path → Configure build path → add external JAR's → browse lib folder → copy existing Jar's → click open → click ok

i) Run menu → external tools → External tools configurations → Select program → Specify name → Browse file system → browse java.exe in existing JDK → click open → browse file system for working directory → Select lib folder → click ok → Specify arguments like as below.

-jar Selenium-Server.jar - Port 4444

click apply → click Run

(After starting server we can change port number)

Creating Test Scripts

We can write test scripts by using core java & selenium objects in eclipse.

```
import com.thoughtworks.selenium.DefaultSelenium;
Public class mail-sign {
Public static DefaultSelenium selenium = new Defaultselenium(
                    "localhost", 1234, "*iehta", "http://");
Public static string openURL()
      {  selenium.start()
         selenium.open("http://www.mail.in.com");
         Selenium.windowmaximize();
         return "Pass"
      }
Public static string login()
    {
      Selenium.type("f_id", "seleniumforum_anjali");
      Selenium.type("f_Pwd", "anjali");
      Selenium.click("//[@value='' and @type='submit']");
      return "Pass";
    }
Public static void main(String [] args)
      {openURL();
        login();
      }
  }
```

Note:- To get object descriptions QTP people are using object spy. But Selenium testers are using mozilla firebug for mozilla firefox browser and IE developer tool bar for IE browser.

Load Runner 9.5

→ Developed by mercury Interactive and take over by HP.

→ Performance testing tool.

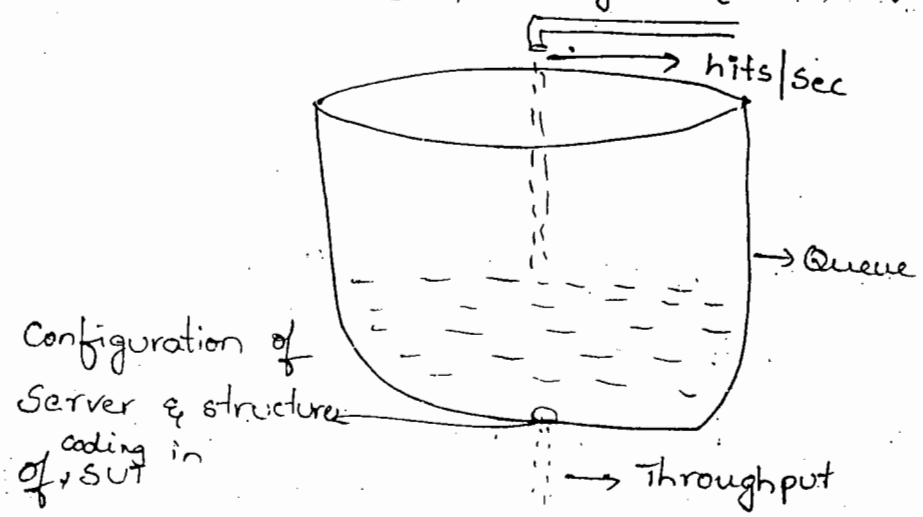
→ Running on windows platforms.

→ Creating virtual users to operate our SUT concurrently.

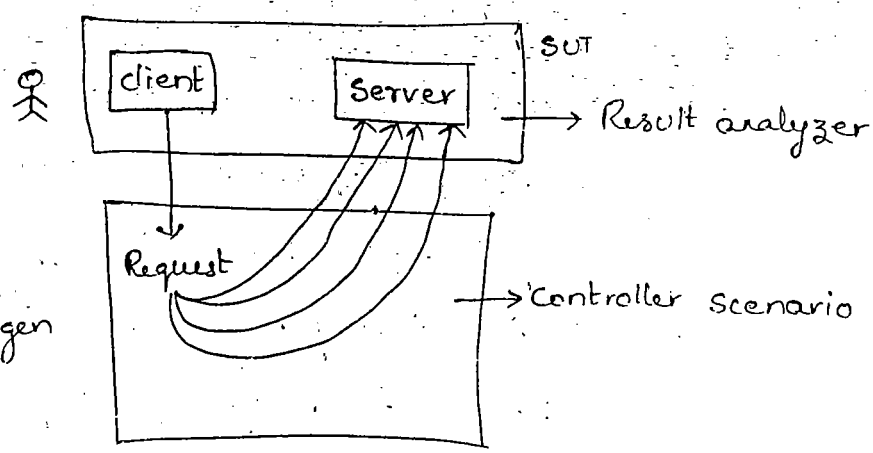
→ It supports performance testing on windows based projects, web based projects and other N-tier softwares.

Performance testing levels

While conducting performance testing, testing team is Concentrating on 4-levels such as load testing, stress testing, spike testing, endurance testing. The execution of S/w under customer expected Configuration & customer expected load to estimate speed in processing is called as load testing or scalability testing. To calculate speed in Processing, we can use elapsed time [Request transmission, Process in server, Response transmission] Response time [Time to get first response], Hits/sec [no. of requests received by server in one sec] & Throughput [Speed of server in one sec]



Load Testing by using Load Runner.



customer expected Configured Computer / Network

From the above diagram, load runner tool consists of 3 components such as vugen, controller scenario & results analyzer. Vugen stands for virtual user generator. It is allowing you to create vuser script in c/c++/javascript on your SUT for load testing. Controller scenario is used to create required no. of virtual users to run on SUT. Results analyzer returns statistics of performance. in terms of elapsed time, response time, Hits/sec and throughput.

Navigation:-

Take customer expected configured Computer → Install loadrunner and SUT → Start → programs → Load Runner → Load Runner → click Create / Edit Scripts → File menu → new → Select protocol depends on SUT (Web (HTTP/HTML)) → click create → Select SUT-type [Web based or windows based] → specify browser name when SUT is web-based → Specify URL or path of SUT → Specify working directory → Specify recording to start (Vuser-init, Action, Vuser-end) → click ok → Operate our SUT as one user →

Click stop recording after completion of required operation in SUT. → Save Vuser script → Tools menu → Create Controller scenario → Specify no of Vusers to define load → Specify results directly → click ok → click start Scenario icon → Results menu after completion of execution → analyze results → Identify test summary in terms of hits/sec, throughput, elapsed time and average response time → Report to project management.

Creating Transactions:-

To calculate performance of individual operations, we can make related code as transaction.

```
Vuser_init()
    ≡
Action()
    lr_startTransaction("name1")
    ≡
    lr_endTransaction("name1", Auto)
    ≡
    lr_startTransaction("name2")
    ≡
    lr_endTransaction("name2", Auto)
    ≡
Vuser_end()
    ≡
```

To insert transactions we can follow below navigation

Select position in action → insert menu → start transaction → enter transaction name → click ok → Select position in script → insert menu → end transaction → click ok.

Parameterization :- After recording script for one user and then making required transactions, performance test autometer can Parameterize one user data for multiple users.

Select one user data in script → Right click → Replace with param → enter parameter name → Select parameter type as file of table → click properties → click create table → click add row → enter value → click add row to enter more values → follow above navigation to parameterize all required values in script.

Rendezvous point :-

We can use this point in script to do load balancing. While running vuser scripts, fastly running users are waiting at this point until all the users comes to that point.

```
vuser_init()
    ==
    ==
Action()
  - - -
    lr_rendezvous point
    lr_transaction
      - - -
      - - -
    lr_transaction
vuser_end()
```

Navigation

Select position on top of transaction → insert menu → Rendezvous → enter name for this point → click ok.

Bench mark Testing :-

After Completion of successful load testing under customer expected configuration & load, testing team reports results to Project management in below format.

User Scenario	Load	Hits/sec	Throughput	Avg response time
login	5	1 Hit/sec	256 bytes/sec	3 sec
⋮	⋮	⋮	⋮	⋮

After receiving above like results, project manager can validate those results w.r.t:

→ Performance results of previous version.

→ Performance results of competative product in market

→ For websites, W3C rules.

Ex: 3 sec for url open / links open

12 sec for forms submission.

2) Stress Testing :-

The execution of s/w under customer expected configured environment & more than customer expected load. To estimate peak load is called as stress testing.

Navigation

Click Vusers button in controller scenario → Specify quantity to add after click add vusers button → click ok → click start scenario → follow above navigation to add users until users was failed (called as peak load).

3) Spike testing or Soak Testing

The execution of a s/w testing under customer expected Configured environment & sudden load incrementation to estimate reliability is called as spike testing or soak testing.

Navigation

Click vusers in controller scenario → click add vusers → specify quantity to add(sudden increment with high load) → click ok → click close → click start scenario

4) Endurance testing:-

The execution of s/w under customer expected configured environment and customer expected load continuously to estimate longvity is called as endurance testing or durability testing or longvity testing.

Create Vuser script for one user → Insert transaction point → parameterize script → Insert rendezvous point → Save vuser script → vuser menu → Run time settings → Runlogic → specify no.of iterations for continuous load → Select Pacing → Select any one of below 3 options.

- (•) 1) As soon as previous iteration ends.
- ( ) 2) Delay as fixed / Random after every iteration ends.
- ( ) 3) Delay as fixed / Random after interval of iterations.

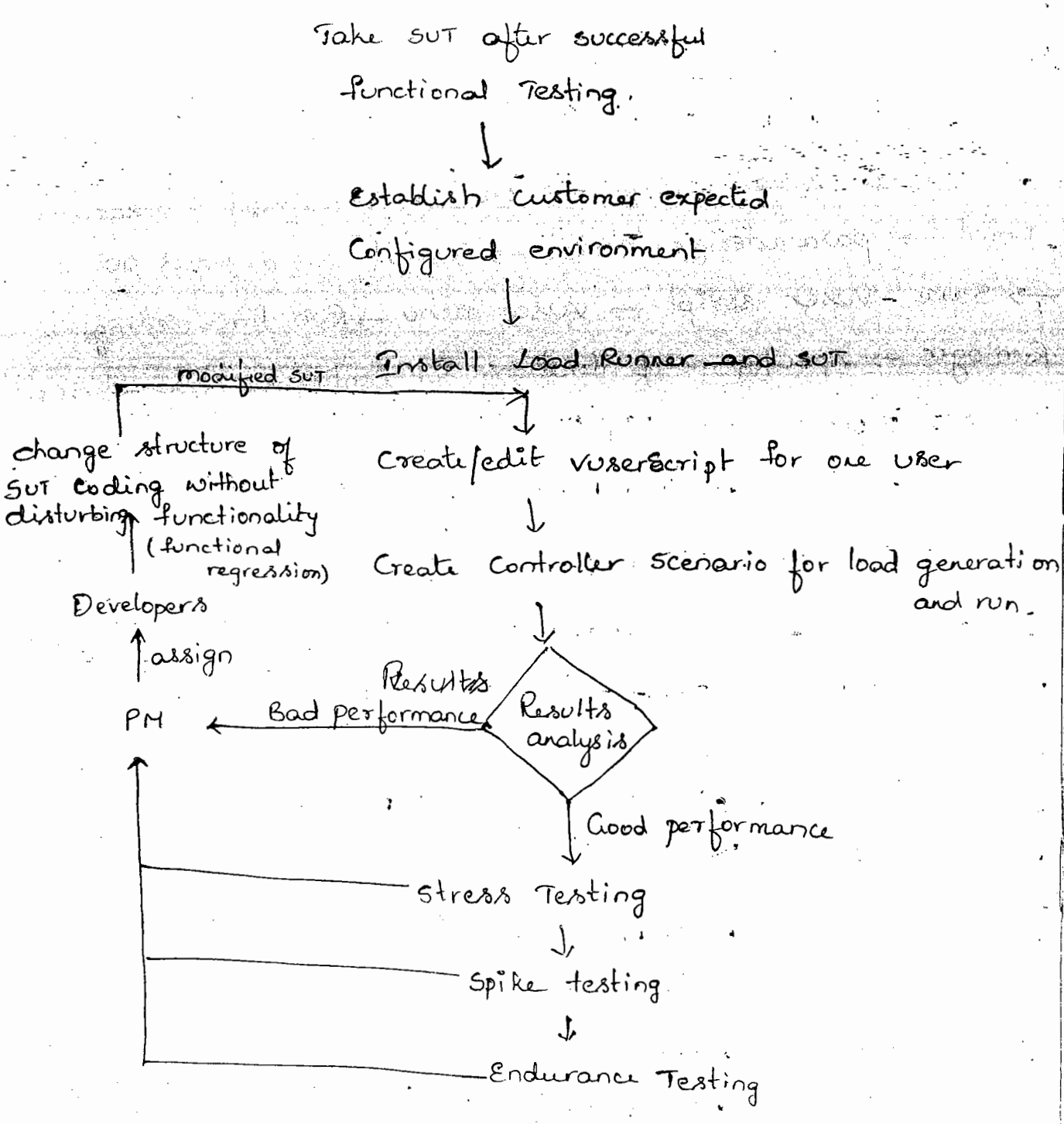
→ click ok. → Tools menu → create controller scenario → Specify customer expected load → click ok → start scenario.

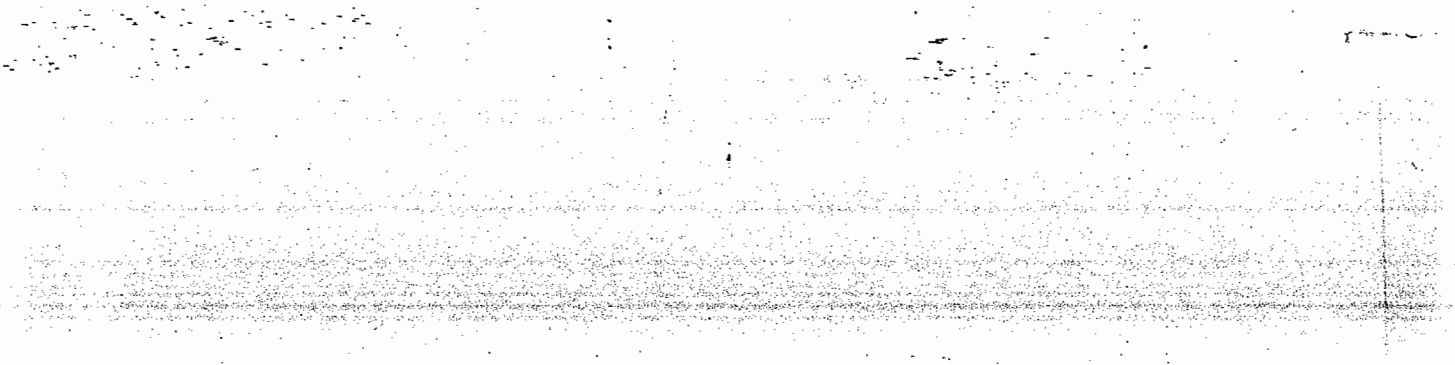
Case study:-

Performance testing topic	Style of load creation	Purpose.
Load Testing	No.of users equal to Customer expected	Speed in processing
Stress testing	Increase load in interval manner from	Find peak load

	Peak load	
3) Spike testing	Increase load suddenly from customer expected to huge load	Reliability / Recovery from abnormal state to normal state.
4) Endurance testing	countinuously customer expected load.	Memory leakages (Improper memory management)

Performance testing process:-





Conducting performance testing on SUT in network