STATIC TESTING : TEsting QUALITY WORK PRODUCTS BY PERFORMIG REVIEWS MEETINGS, JUST REVIWING THE CODE DOCUMENT DEVELOPERS CAN FIND ERROR/

CODE EXECUTION IS DYNAMIC TESTING

GC REDDY Notes :

1.SDLC : sequential model

Incremental or Iterative

2.Test levels :

--- Unit / component / testing , developers perform unit testing. Testing

--- Integration Testing – developers perform integration testing.

---System testing --- testers perform

---Acceptance testing --- normally end users perform this type of testing.

3. Test types :

Fuctional testing (85%) :

Functional Testing is a testing technique that is used to test the features/functionality of the system or Software, should cover all the scenarios including failure paths and boundary cases.

Functional testing is mandatory.

Non functional testing (15%) : in this some sub types are available .

4: Testing Techniques : formal testing

* While box / clear box / glass box test design techniques :

Developers perform white box testing , the code is visible for testing.

Ex. Statement testing, decision testing, condition testing ETC…

* Black box technique , testers perform this , the system code is not visible.

Ex. Equivalence partition testing

Boundary Value Analysis

Decision table

state transition testing

Use case Testing…ETC

* Informal Testing technique :

Experienced testing, error guessing, exploratory testing

5. Software Test process / STLC

Test process may vary from one company to other. In formal test process four stages …

1. Test planning: Test Lead prepares test plan document.

Imp. Tasks are:

--understanding and analyzing the requirements.

--Risk analysis

--Test strategy implementation (all test levels, test types, test design techniques)

--Test estimation in terms of budget, time, resources and scope of the project.

--Team formation

--Test plan documentation

--Configuration management planning (configurable items)

--Test environment setup (test lead or team lead **defines** Environment i.e what HW or SW requires, network admins **implement** test environment setup and testers **verify** test environment setup)

Understanding and Analysing the Requirements : two sorts of testing

1.Formal Testing

2.Informal Testing

Sometimes testers perform testing without the requirements, that is ADHOC testing experience based testing(informal)

Req Spec.based : based on SRS

Used case Based : Business process based Testing : based on business scenario or user scenario in dialogue between system(AUT) and actor.

Risk Analysis : Employee issues, Team member may resign job in the middle of the project, Supplier(vendor) issues, Training issues,(Automation training or domain training, SME (subject matter experts) ,Knowledge sharing, Deadlines, Budgets, team member EGO’s. Test lead analyses all these issues and prepares some solutions also.

Test Strategy Implementation: Test strategy company level document. All test levels, test types test design tech. etc. Test lead implements test strategy based on company standards.

Team Formation: If team formation is done first, all team members can contribute in test planning. Test lead consider some factors to form Team …Scope of the project , availability of Knowledge resources, time, budget, environmental issues etc.

Test Plan Documentation: Test lead prepares it. Test plan template may vary from company to company or project to project.

Three types of concepts :

Universal concepts

Local concepts: ex. templates no universal templates depend on their feasibility

Controversial concepts: Ex. sanity and smoke synonyms

Most of the companies use word document for test plan document (more theory less tables).

1. Test Plan ID: any unique name … IBS\_TP\_001

2. Introduction: 5 to 10 lines description

3. Test Items: components and modules

4. References: (Mandatory): Requirements, project plan, test strategy

Optional: Use cases, high level design documents, low level design documents, process guide line doc, prototypes (photos without functionality)

5. Features to be tested (for a new project all field to be tested)

6. Features not to be tested ( for any existing project if we add any new module and dint made any changes to the previous existing modules , in this case we can mention not to be tested for the existing modules).

7. Entry Criteria: defines when to start testing.

8. Exit Criteria: defines when to stop testing.

9. Suspension Criteria: defines when to stop testing temporarily.

10. Roles and Responsibilities

11. Schedule:

1. Test design : primary Role is team member , test lead will guide and monitoring and controlling.

Imp. Tasks are : Generating test scenarios

* Test case documentation
* Test data collection

1. Test Execution : Imp tasks are :

- Verify the test environment setup( test lead defines the test environment setup at planning stage and testing execute the environment setup at this stage)

- create test batches (Queue)

- test execution :

Sanity / smoke / build verification testing (BVT) / BAT ( verifying whether the build is acceptable or not by executing basic functionality test )

-comprehensive testing

- defect reporting

- tracking the defect

- regression testing –(general regression and final regression)

1. Test closure : Imp Tasks are :

* Evaluating the Exit Criteria: describes when to stop testing. (lead prepares entry criteria, exit criteria and suspension criteria in test planning stage)
* Collecting all artifacts from test activities
* Test summary report (test lead prepares summary report document like how many raised how many solved etc. details)
* Sending test deliverables to the customer ( in order to perform acceptance testing need to send the test deliverable document—test plan test data etc.).
* Improvement suggestions for future projects( optional task)

Quality Standards :Testing improves Quality, need to learn about Quality Standards.

Testing point of view three organisations are important .

ISO : International Standard Organizational :

IEEE : institute of Electrical and Electronics Engineers

CMM / CMI : CAPABILITY MATURITY MAINTAINENCE IN

Domain Knowledge : banking bfsi , ERP, telecom, healthcare, ecommerce ETC.

Test level Documents :

Test policy (company level document)

Test strategy (company level document)

Test plan (project document )

Traceability matrix (Project level - lead prepares all the team members updates it throughout the project)

Test scenarios

Test case document (team memebers)

Test data

Defect report

Test summary report document ( test lead prepares)

Q: do testers need to make a copy of all these documents.

**Software test design techniques :**

Two types of test design techniques :

1.Static

2.Dynamic

Independent testers performs Black box testing and developers perform White box techniques.

In black box testing we have five imp test design techniques

1.Equivalence Partioning Testing

2.Boundary value Ananlysis (extension of EPT)

3.Decision table Testing

4.State Transiotion Testing

5.Use case Testing

**What is Technique** ? : An Efficient way of doing or achieving something is called a technique.

**What is test Design Technique ?**: A test design technique is used to select a good set of test cases from the all possible tests for given a system or an application.

By design we mean to create a plan for how to implement an idea and technique is a method or way for performing a task. So, **Test Design** is creating a set of inputs for given software that will provide a set of expected outputs.

A test design technique basically helps us to select a good set of tests from the total number of all possible tests for a given system. There are many different types of software testing technique, each with its own strengths and weaknesses. Each individual technique is good at finding particular types of defect and relatively poor at finding other types.

**Why test design technique?** Exhaustive testing is impossible. Testing with all the possible inputs is not possible, so we need to use test design techniques in order to reduce the size of inputs simultaneously covering all the test areas.

Exhaustive testing is test approach in which the test suit(series of tests) compromises all the combination of input values and pre conditions.

Exhaustive testing is not recommended due to time and budget consideration.

And all these test design techniques are for manual testing, if we use Automation testing then no need of using of any techniques because of tools we use.

Because we cant test any functionality with all the inputs that is Exaustive testing is not possible.

There are two main categories of test design techniques :

1.Static

2.Dynamic techniques

STATIC :

* Static testing is the testing of the software work products manually, or with a set of tools, but they are **not executed**.
* It starts early in the Life cycle and so it is done during the verification process.
* **It does not need computer as the testing of program is done without executing the program.** For example:  reviewing, walk through, inspection, etc.
* Most static testing techniques can be used to ‘test’ any form of document including source code, design documents and models, functional specifications and requirement specifications.

Testing the s/w documents or s/w work products manually without executiong the program. By reviewing the documents, walk through , inspections and etc we find some error and correct them.

Two types : 1.Reviews (manual examination)

2.Static Ananlysis (Automated Analysis)

Reviews : 4 types : i) Informal Review

ii)Walkthrough

iii)Technical review

iv)Inspection

2)static Ananlysis( tools are typically used by developers)

Ex. Compilors (converts high level language to machine languages).

USES of Static techniques :

* Since static testing can start early in the life cycle so early feedback on quality issues can be established.
* As the defects are getting detected at an early stage so the rework cost most often relatively low.
* Development productivity is likely to increase because of the less rework effort.
* **Types of the defects that are easier to find during the static testing** are: deviation from standards, missing requirements, design defects, non-maintainable code and inconsistent interface specifications.
* Static tests contribute to the increased awareness of quality issues.

DYNAMIC TECHNIQUES :

* This testing technique needs computer for testing.The Software is tested by executing it on computer.
* It is done during Validation process.
* The software is tested by executing it on computer.
* Example of this **Dynamic Testing Technique**: [**Unit testing**](http://istqbexamcertification.com/what-is-unit-testing/), [**integration testing**](http://istqbexamcertification.com/what-is-integration-testing/),**[system testing](http://istqbexamcertification.com/what-is-system-testing/)**.

Categories of Dynamic techniques:

i)Specification based or Black box technique (Formal techniques)

a)EQ.Par

b)BVA

c)Decision Tables

d) State transition testing

e) Use Case testing etc.

ii) Structured based or White Box techniques (Formal techniques)

* 1. Statement testing and coverage
  2. Decision testing
  3. Condition testing and multi condition ETC.

iii)Experienced based technique (this is informal techniques)

a)error guessing

b)Exploratory testing etc.

EQUIVALENCE PARTITIONING TECHNIQUE :

* Equivalence partitioning (EP) is a specification-based or black-box technique.
* It can be applied at any level of testing and is often a good technique to use first.
* The idea behind this technique is to divide (i.e. to partition) a set of test conditions into groups or sets that can be considered the same (i.e. the system should handle them equivalently), hence ‘equivalence partitioning’.**Equivalence partitions**are also known as equivalence classes – the two terms mean exactly the same thing.
* In equivalence-partitioning technique we need to test only one condition from each partition. This is because we are assuming that all the conditions in one partition will be treated in the same way by the software. If one condition in a partition works, we assume all of the conditions in that partition will work, and so there is little point in testing any of these others. Similarly, if one of the conditions in a partition does not work, then we assume that none of the conditions in that partition will work so again there is little point in testing any more in that partition.

It can be applied at any level of testing (unit, integration, system and acceptance testing )

IN EP, inputs to the software are divided into groups that are expected to execute similar behavior.

EP can be found for valid data and invalid data.

DATA FACTORS :

1. Type (Alpha bytes, numeric, Alpha Numeric, Alpha numeric special)
2. Size (10 digit phone numbers)
3. Range (20yrs to 30yrs, 5 to 7 yrs)

DECISION TABLES :

It may be applied for all situations when the actions of the software depends on logical decisions.

The techniques of equivalence partitioning and boundary value analysis are often applied to specific situations or inputs. However, if different combinations of inputs result in different actions being taken, this can be more difficult to show using equivalence partitioning and boundary value analysis, which tend to be more focused on the user interface. The other two specification-based [**software testing**](http://istqbexamcertification.com/what-is-a-software-testing/) techniques**, decision tables and state transition testing are more focused on business logic or business rules.**

 A **decision table**is a good way to deal with combinations of things (e.g. inputs). This technique is sometimes also referred to as a ’cause-effect’ table. The reason for this is that there is an associated logic diagramming technique called ’cause-effect graphing’ which was sometimes used to help derive the decision table (Myers describes this as a combinatorial logic network [Myers, 1979]). However, most people find it more useful just to use the table described in [Copeland, 2003].

* Decision tables provide a systematic way of stating complex business rules, which is useful for developers as well as for testers.
* Decision tables can be used in test design whether or not they are used in specifications, as they help testers explore the effects of combinations of different inputs and other software states that must correctly implement business rules.
* It helps the developers to do a better job can also lead to better relationships with them. Testing combinations can be a challenge, as the number of combinations can often be huge. Testing all combinations may be impractical if not impossible. We have to be satisfied with testing just a small subset of combinations but making the choice of which combinations to test and which to leave out is also important. If you do not have a systematic way of selecting combinations, an arbitrary subset will be used and this may well result in an ineffective test effort.

STATE TRANSITION TESTING :

In state transition testing , test cases are designed to execute valid or invalid state transitions.

A system (Application under test) may exhibit a different responses on current conditions or previous history.

EX. Internet banking system funds transfer functionality.

When we have the initial bal.45000 and if we transfer some amount then the balance may change..if we again do the same transaction of transefering the amount ,but there cound be insuffuecient bal and the result may come differ.

Depends on the state expectaions vary.inputs and operations are same but based on the state outputs vary.

* State transition testingis used where some aspect of the system can be described in what is called a ‘finite state machine’. This simply means that the system can be in a (finite) number of different states, and the transitions from one state to another are determined by the rules of the ‘machine’. This is the model on which the system and the tests are based.
* Any system where you get a different output for the same input, depending on what has happened before, is a finite state system.
* A finite state system is often shown as a **state diagram** (see Figure 4.2).
* One of the advantages of the state transition technique is that the model can be as detailed or as abstract as you need it to be. Where a part of the system is more important (that is, requires more testing) a greater depth of detail can be modeled. Where the system is less important (requires less testing), the model can use a single state to signify what would otherwise be a series of different states.
* A **state transition model has four basic parts:**
* *The states that the software may occupy (open/closed or funded/insufficient     funds);*
* *The transitions from one state to another (not all transitions are allowed);*
* *The events that cause a transition (closing a file or withdrawing money);*
* *The actions that result from a transition (an error message or being given your cash).*

Hence we can see that in any given state, one event can cause only one action, but that the same event – from a different state – may cause a different action and a different end state.

USE CASE TESTING : Actors and system

SOFTWARE TEST PROCESS:

Will check later

**Programming languages :** C, C++, JAVA, VB, COBOL

**Database Management Systems :** Oracle, Sybase, Access, SQL SERVER, DB2(mainframes), MYSQL (open source)

**Scripting Languages :** SHELL Script, Pearl Script, VB , Java Script, Python ,RUBY , REXX etc

**ERP Packages :** SAP, Oracle Apps, PeopleSoft, Siebel etc

**Web technologies** : Html, Dhtml, XML, CSS, IIS, SOAP, JASON(open source), JAVA Script, APACHE,weblogic web speare

**Operating systems** : Dos, UNIX, MS Windows, LINUX,

SW Development Business Process (Before SDLC)

**Software Bidding** : A proposal to develop a software. Customer specific application proposal comes from the customer. Project based or Market Specific Application Proposal comes from the organisations, Product based.

**PIN Document** : (Project Initiation Note ) : After the project is confirmed PIN document is prepared. Higher management conduct a review after confirmation of the project and they appoint a project manager for the SDLC of the project.Project Maneger prepares the PIN Document and it consists the overall estimations of Human resources, Budget, time, resources and etc.

KICK OFF MEANING : is the first **meeting** with the project team and the client of the project. This **meeting** would follow definition of the base elements for the project and other project planning activities.

**MAIN PROCESS : SDLC** : A systematic approach to develop a software.

**I)Requirements Gathering & Analyssis :**

Two Scenorios : 1) Customer specific Application (project Based)

2) Market Specific Application ( for multiple customers / product based )

In case of Customer Specific Applications, Business Analyst gathers the business requirements and the req. are documented, which is called BRD Document.(CRS / URS)

For each documents three steps :

1) Document Preparation

2) Document review and

3) Document Approval

Different kinds of reviews like PEER reviews, same level team members reviews the documents, Inspection or Technical review senior persons reviews the documents, After review they write some commentsnAuthor of the document modifies the document.

Lastly PM approves the document. Output would be the Based Lined Document.

In case of SW Product Application, Business team gathers the requirements from Existing custometers, and they analyse the competitors product .

BRS is the non-technical document and project Leader or BA converts that into a Technical Document called SRS Document.

SRS : describes required technologies ,Hardware and SW to develop the SW

FRS : required functionalities to develop.

Output : SRS Document.

For converting BRS to SRS, Considerable factors are :

Are they correct Req or not

Are they complete Req or not

Are they achievable Req. ornot

Are they testable Req or not

considerable factors for technology selection are : Organisation exp.availability of the resources , nature of the application, sometimes customers intrest also and sometimes budget aso.

**II) PLANNING** :

Input : SRS Document

Project Manager prepares the project plan and he/She sends to higher management after the approval PM executes the project plan.

Project Pan includes : human resources ,environment resources, budget, time,technologies

Output : Project Plan

**III) DESIGN:**

**Input : Project Plan**

Sr.Developer / System Architect/ Technical Manager prepares Design Document.

Two types of Software Design Documents:

1. Global Design or High Level Design (overall architecture of System)

2. Detailed Design or Low Level Design ( Detailed each module and components of the system)

**Output : Design Document**

**IV) CODING or Inplementation :**

Developers write the code.

**V) TESTING :**

**Input : SRS Document**

VI) Release & Maintenance :

PM forms the release team (one or two developer ,testers, NW admin pm executive )

Release team will go to customer place , deploy the tested code and give some training for the customers if required.

**Software Maintenance**: continuous process.

Modifications –1.corrective changes

2.Enhancements

Migrations : migrating from old technology to new Technology

Software Retirement : retiring the old system and developing the new system.

Questions : diff between fuctional and non fuctional testing

Why test design testing? Exhaustive testing is impossible. Testing with all the possible inputs is not possible.