03/09/2014

Testing Techniques

Objectives

Explain the characteristics and differences between specification based, structure - based and experience based testing

Apply equivalence partitioning, boundary value analysis

Functional – Specification Based, Black Box (

Non-Functional – Performance/Load/Structures/Stress/Usability

Structural – Control Flow, Menu structure, White Box(can be done at all levels)

Change Related – Migration of System, Maintenance Testing. (Impact analysis and Metrics)

Carrired out on a live system

Black Box aka Specification Based aka Functional

BB

Examines the functionaliy of an aopplication without knowledge of uts internal structures

It is based entirely on the software requirements and specification

Techniques include:

* Sanity/Smoke Check
* Equivalence Partitioning
* Boundary Value analysis
* Decision Table Testing
* State Transition Testing
* Use Case Testing

Sanity and Smoke Test

Smoke Test – The aim is not to find defects but to check system health.

Sanity Check – Makes sure the bugs reported in previous builds are fixed for this please before doing a full regression

Acceptance in the Integration/System/ and Acceptance Testing levels

Decision Table Testing

A black box test design technique in which test cases are designed to execute the combinations of input and/or stimuli (causes) shown for

A compact and precise ways of modelling complicated logic

Lists all the condition that can occur and their consequences

The Four Quadrants

Conditions Set of Conditions

Actions Action entires

**Set of Conditions** are Boolean values(True/False) that make up the different possible combinations of conditions

**Action Entries** are check-marks, representing which of the actions in a given column to be performed

Infeasibility

Ie Certain combinations of conditions that are logically impossible

Don’t forget – Sometimes, we may want to test an infeasible test condition to make sure it can’t happen

State Transition Testing

A black box test design technique in which test cases are designed to execute valid and invalid state transitions – ISTQB Glossary

When a system is represented as being in one state, and transitioning from that state to another

The transformations are determined by the rules of the system

This means we can follow these rules to create a diagram that represents the change of transitions and thus a test to see if it works.

What is the Goal?

Finding situations where the wrong action or the wrong new state occurs in response to a particular event

As Testers we report these problems all the time as bugs

Where a system should only be finite number of states, we can use transition testing techniques, to create our tests

States – How something exitst at that time (Open/Closed)

Transitions - The change from one state to another (Door is opened/closed). Transitions are triggered by…

Inputs or Events – (Turn handle and pull/push door)

Actions - The actions that can result from a transition(ability to walk through the door)

Example

A simple electronic clock has four modes

- Display Time

- Change Time

- Display Date

- Change Date

The change mode button switches between display time and display date

The reset button switches from display time to change time or display date to change date

The set button returns from adjust time to display time or adjust date to display date

If every valid transition n is tested 0-switch coverage

We have proven that a finite number of test can be defined from a possible infinite number of test scenarios, and an appropriate level of coverage can be achieved.

Equivalence Partioning / Boundary Value Analysis

We strategically sample information to make “infinite possibilities” into a more manageable set of tests

Equivalence Partitioning (Equivalence Classes)

**ISTQB**: A black box test design technique in which test cases are designed to execture representatives from equivalence partitions. In principle test casrs are designed to cover each partition at least once

Rather than trying to test infinite amount of inputs

Boundary Value Analysis

**ISTQB**: A black box test design technique in which test cases are designed based on boundary value

Boundary Value test: An input value or output value which is on the edge of equivalence partitioning or at the smallest incremental distance on either side of an edge, for example, the minimum or maximum value of a range.

Invalid Valid Invalid

---------------------🡪🡨----------------------------🡪🡨-------------------------------------

<14 15 25 26>

**Scenario –** “Lets imagine we’re testing from a requirement:

“The convertIntToRoman(int) function will return valid Roman numeral strings for all positive integers up to MaxInt

Use Case Testing

ISTQB – Use Case: A sequence of transatctions in a dialogue between an actor and a

A use case describes interactions between “actors” and the system

Uses can play many roles with respect to a system

If we do not understand all the roles a user plays, then we cannot understand the system

It can also be diagrammatical or as a scenario

Very useful when testing process flows and busicness rules

Very effective in defining acceptance tests as the use case usually represents hwo the system will be used

**What is a Use Case vs. a Test Case**

Remember the difference

Can also fine Use Cases as Steps

Experience Based Techniques

ISTQB – Error Guessing: A test design technique where the experience of the tester is used to anticipate what defect might be present in the component or system under test as a result of errors made, and to design tests specifically to expose them

Defect and Failure lists can help identify area that are susceptible to problems

**Fault Attack** – This approach is to lost possible errors and design tests around that list.

Exploratory Testing

This is one of the most powerful forms of testing. It has the bare minimum of a plan, if it has anything at all it is probably a relaxed scenario. Important tool in the hands of SMEs. It is especially suited to web app development in agile methods (because of the intuitive nature, but can be applied to everything)

Pro: It does require much preparation

Testers report a large proportion of bugs via this method

Dis:

There is no review of test planning, an experienced user of the system may not be an experienced tester

Testers have to remember the exact steps they took to create a defect – otherwise reproduction may be difficult

White Box aka Structure Based

Also known as Clear/Glass/Open/Transparent, Structural Testing

Tests internal structures as opposed to functional Testing

Usually done at the Unit level but can be applied to integration and system

Techniques include:

Decision Coverage

Statement Coverage

Focuses primarily on strengthening security, the flow of input and outputs through the application, and improving design and usability

It is crucial that if you are take part in the White Box Testing that you already have some knowledge of the code language used to create the software

Code / Statement Coverage Analysis

Branch/Decision Coverage

Check if each possible branch from a decision point has been executed at least once

Decision Condition Testing

A white box test design technique in which test cases are designed to execute condition outcomes and decision outcomes

Statement Coverage

Check if each statement is code has been executed at least once

Statement Testing

A white box test design technique in which test cases are designed to execute statements

Edmodo

How many test cases were gonna need to statement

No point doing other statements if the don’t’ have anything to cover

STATEMENT COVERAGE

Statement Coverage

Decision Coverage

Path Coverage

Pseudo-Code

G8 b8 m8, I r8 8/8 consum8

If means that a deicion must e taken

Works in Boolean – True/False Statement

To cover Statement here, it is enough to simply exercise the true decision. To cover Decision, you need to exercise True and False.

**Cover all statement 1 - test cases**

**Cover all decisions 2 - test cases**

If Else

**Cover all statements 2 - test cases**

**Cover all decisions 2 - test cases**

Switch

**Cover all statements 4**

**Cover all decisions 4**

While

While is a loop whih is to be executed as long as condition is the loop is True. To leave a loop, you need to have a false condition in the loop. If it has been build correctly, you should oly need obe test case to cover all decisions and statements

**Cover all statements 1**

**Cover all decisions 1**

(Do) While or Until

**Cover all statements 1**

**Cover all decisions 1**

For Statement

**Cover all statements 1**

**Cover all decisions 1**

Decision Coverage

This is related to branch testing

It is the assessment of the percentatge of decisions outcomes,eg true and false options of an IF statement. That’s has been exercided by a test case suite

It derives Test Cases to execture sp[ecific decision ou