01/09/2014

Test Levels

Objectives

>Apply the different test levels (component, integration, system, acceptance at the appropriate stage of the SDLC with constant reference to characteristics of good testing

Recognise and compare examples: functional/non-functional

It’s important to remember that each of testing has objectives specific to that level

Unit Testing

Aka Component/Program/Module Testing

Each Unit of code is usually created in isolation (for integration at a later stage)

Each of those units of code will relate to a certain function of area and be tested by developer to make sure it works

As more code is developed, more and more tests are created

Generally, most the system failures will be found in a small proportion of modules

Integration Testing

During integration, units are incrementally integrated and tested together based upon control flow

>Big Bang integration testing

Everything get thrown in together at the same time

>Incremental Integration Testing

-Top Down

-Bottom Up

-Hybrid

Top Down

High levels modules are created first, so a stub code is needed.

**Pro’s:**

Useful in creating software intended to be generic

This will allow for early demonstrations of the product functionality

May help identify requirements change and issues

**Con’s**

Stubs can create a lot of work

Stub definition can be difficult

Reproducing test condition may not be possible

Bottom Up

Parents are not yet developed and so Driver code used instead

**Pro’s:**

Higher accuracy at the granular level

Components are needed in a controlled manner

Test conditions are easier to create

User and business awareness used to clarify product in early stages

**Con’s:**

Driver development could increase work load, but there is software available to help with this

It is difficult to estimate top-level forecasts

Hybrid

Of course these methods can be combined

It may be obvious that some groups module may lend themselves to one method more than the other

You may look for specific criteria to dictate what method to use

System Testing

System testing looks at the behaviour of the system

Harness complete end to end scenarios in the way that the customer would use the system

A test team must endeavour to appy every input and receive the correct output to exercise the full capabilities of the system

> Test Plan contains test cases, what we know of the system, how it’s meant to work, how the users are supposed to use it and document

A wide range of possible tests

Including non functional

>Gui

>Usability

>Exception Handling

>Volume

>Scalability

>Ad-hoc

>Installation

>Recover

>Performance

>Load

>Security

>Stress

What affect the type of system testing you do?

Size of the Company

Time available for test

What resources are available?

Learning Curve – how long will it take for the tester to learn the software that the company uses?

Budget

There are many different approaches to this type of testing that can be used

ISTQB Glossary

Risk – Based Testing:

Requirements – Based Testing:

Business Processes – Based Testing: An approach to testing in which test cased are designed based o description and/or knowledge of business processes.

> Creating test is following steps. EG amazon orders. Similar to requirement, user journey processes, business flow.

Use Case Testing:

> Have a system, draw a diagram, use actors to link and describe process how they use

***Pre-Requisites:*** What needs to be implemented before testing takes place

Stories:

> Agile based

>The stories are the requirements

Acceptance Testing

Testing against the users expectation

Does the system meet all of the requirements

This is a question should be asked all the time

Typically verifies the functional fitness of the system for the business users

**Note:** Acceptance Testing can take place before System Testing starts

It is a crucial element for making sure the product will be functional in its perceived environment

Additional Aspects to think about at this point are:

* Back – Up
* Procedures for disaster recovery
* End User Training
* Maintenance Procedures
* Data Load and Migration Tasks
* Security Procedures

Contract and Regulation Acceptance Testing

Contractual – Sometimes acceptance criteria may be outline in a contract, we must make sure that we align to the contractual obligations

Regulation – You will find that some industries have a rigid regulations that must be abided by. This might be for:

* Governmental
* Legal
* Safety Reasons

Ignorance is not a defence that can be used. It is your responsibility to make sure you are working within the boundaries of the law.

Alpha Testing - Performed at the developing site, but not by the developing team.

Beta Testing – Performed by customers at their own location.

Allows for feedback from potential or existing customers.

Look Into UTest and TestBirds

Functional, non-functional and structural testing

Functional – Specification Based, Black-Box (Can be done at all levels)

Non-functional – Performance, Usability (Often used to check the readiness of the system)

Structural – Control Flow, Menu Structure, White-Box (Can be done at all levels)

Change Related – Carried out in a live system

Functional (Blackbox)

Testing of the functions of a component or system, we do activities that verify a specific action or function of the code

- Requirement Based-Testing – Requirements are prioritised and teste based on risk

- Business Process Based testing - scenario testing of the day to day business used of the system

Non Functional Testing

Efficiency

Maintainability

Load

Performance

Compatibility

Scalability

Stress

Structural (white box)

This is focusing on how the code makes the functionality under works

Confirmation, Regression and Maintenance Testing

Confirmation Testing (Retesting)

After a defect is tested and fixed, the software should be retested to confirm original defect is removed – this is confirmation testing/retesting

Regression Testing

Repeating testing of already tested program

Performed when software or environment is change

Based on risk

Preformed at all levels(functional, non-functional, structural)

Positive/Negative Testing

Change Related Maintenance Training

Testing in a live environment when the has been

>Modification

>Migration

>Retirement of software

Impact analysis (Risk) and Metrics from previous projects are very important in this area

-They help estimate the amount of re-testing and regression testing

- What are the possible consequences

Difficulties you may encounter:

Specification could be out of date

People with domain knowledge may no longer be available

Should not differ much from other test processes

Quiz Time

1. Integration Testing Level > Skeleton Code (Stubs), place holder for the code. Drivers push the functionality
2. Maintainability/Scability/Stress

Alpha

Beta

Contractual Regulation

1. Functional/Structural
2. Impact analysis and metrics from previous projects

James Back.