Proceedings of the NACCQ 2000
Wellington NZ www.naccq.ac.nz

Testing : Comparison Between What is Taught and What is Used in Industry

Peter Henry

Central Institute of Technology
Upper Hutt
peter.henry@cit.ac.nz

ABSTRACT

There are three main testing methodologies. Firstly control flow testing, normally used for unit testing. Secondly Domain testing, normally used for user/ acceptance testing. Thirdly Cause-Effect analysis, normally used for requirements analysis. Four test analysts from Wellington were interviewed as to what they actually use. No formal evidence of unit testing was found, since this was done by the developers. A loose version of Domain testing was used. Cause-Effect analysis was not specifically used, but requirements were tested using a variety of tools. In all cases the development of automated testing, justified by the amount of regression testing, was used. Though the practitioners did not use the methods they still advised that they be included in the teaching.

Keywords
Software Testing, Degree, Diploma, Syllabus

1. INTRODUCTION

This paper attempts to find out what is actually used in software testing in business, and match that to what is taught at tertiary level. The "what is taught" is based on the syllabus of the National Diploma in Business Computing, NACCQ (1998). What is actually used is based on four interviews of test managers in a cross section of businesses in the Wellington area. For the interviews, the questions asked covered a general overview of the business in relation to Information Technology, the people and processes involved in testing, the methodologies, and examples of actual cases.

2. TESTING TODAY

The traditional model of testing is the "V-shaped model". It is based on the perception that testing means running programs to find bugs, and that therefore you can't test anything until the programs have been written. See Figure 1. Traditional V Shaped Model for Testing

For the above, testing is a Quality Control process. That is the software is run against test data (standards) to find bugs, which then can be corrected.

Boris Beiser (1992) states that "The true purpose of testing is not to find bugs but to prevent them". This requires bringing Quality Assurance into the development of software to ensure that quality gets built into the software throughout its development phases. Whichever model is used the testing stages remain the same. The four stages are, Unit testing, Integration Testing, System Testing and Acceptance testing. Quality Assurance will also bring into use code inspection as a Quality Improvement tool.

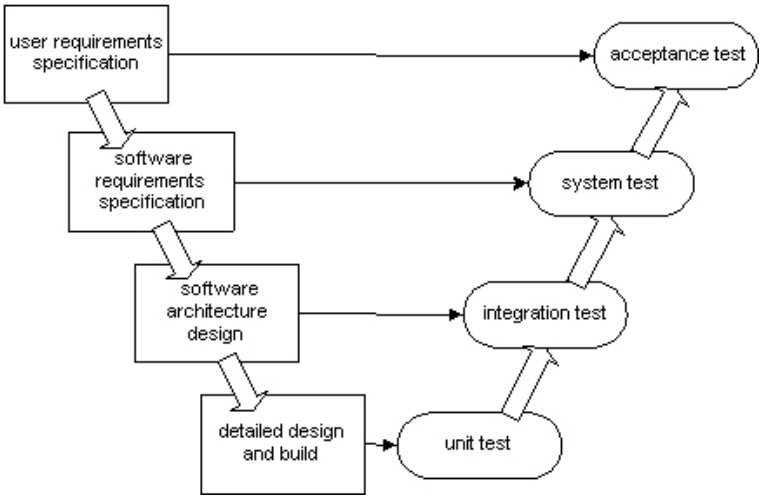

Figure 1. Traditional V Shaped Model for Testing

3. BEST PRACTICE

The following describes the best practice for unit, integration, system and acceptance testing, Beizer (1999).

Unit Testing: done by the developer as they test the code they have written. The normal technique is to use branch testing. Best practice is to test to a 100% statement (also branch and predicate) coverage. This is the minimum requirement to assure that all the code is tested at least once in its life. To assist this there are a wide range of coverage tools available.

Integration Testing: here testing is done between the interfaces of two otherwise correct components to assure that they are compatible. Best practice for integration testing is not a single event, but takes place at every level of the build process.

System Testing: this is testing the system end to end to discover common system bugs, such as resource loss, synchronisation and timing problems, and shared file conflicts.

Acceptance Testing: is done from the user's perspective, typically end-to-end, to verify the operability of every feature. Full partition boundary testing is done.

4. METHODS

For unit testing the programmer's main technique is path testing. This covers branches, predicates and loops. For unit testing and integration testing the programmer must be able to write drivers and stubs. Code inspection is the one most successful method in finding bugs in code, and so must be covered.

For system testing and acceptance testing equivalence partition testing and cause-effect testing are suitable.

To summarise the topics needed to be covered to meet the best practices for software testing are

- Quality management of software development
- Path testing
- Equivalence Partition Testing
- Cause-Effect Testing
- Code Inspection
- Drivers and stubs.

With these tools a student can cover the requirements of unit, integration, system and acceptance testing.

5. WHAT IS TAUGHT

The National Diploma in Business Computing has two papers devoted to testing.

QA200- Quality Assurance

1. Peer Review Demonstration
   Quality assurance and validation techniques and organisation

Introduction to measurement techniques and organisation.

2. Quality Assurance assignment
   Different methods of establishing quality in code using formal and informal reviews with peers as well as management
3. Module Testing
   Difference between testing to specification and testing to code
4. Application Package Test
   walk through the system development life cycle to determine each type of test conducted at the different stages
   Use decision tools to determine test cases.
   Use of completed modules and their specifications to conduct test and analyse results. Use of a print utility to print test files etc.

SI300 - Systems Implementation

1. System Testing
   use of a project planning tool for designing the system test plan.
   Consideration of a full list of factors/categories when constructing system test cases.
   Conducting function and performance tests on an integrated group of previously tested modules
2. Acceptance Testing
   Development of different approaches eg. benchmark, pilot, parallel, operation testing and acceptance testing.
   Completion of user and operations documentation, a training plan and CBT module for the acceptance test.
3. Installation Test Design
   Checks required for installation tests

QA200 is a level 2 module but is often taught in the first year with a programming language (for example Pascal, C++ or Java). SI300 is generally taught in the third year. The match between what is taught and best practice is very good. Apart from giving methods different names, all the best practice topics are covered in the two syllabi. The only area not mentioned is an "Introduction to measuring techniques and organisation".

6. WHAT INDUSTRY DOES

Comtex Ltd is a NZ owned IT consultancy group, with around 250 staff. The Integrated Quality Assurance team has grown dramatically over the last few years. A major impetus has been the Y2K problem but industry has also become more aware of quality issues in developing systems. The team offers Requirement Document Inspections, Test Strategy, Test Plan, Test Cases/Scenarios, Business Acceptance Testing Assistance, Issue Rresolution.

Unit and integration testing is done by the development team. The IQA team concentrates on requirements and acceptance testing. The team has developed expertise in the use of Rational Robot, specifically entering the requirements and developing a test plan from there. To develop test data the team develops a good set and then for the high risk parts of the system.

National Bank NZ is one of the four major NZ banks and is owned by Lloyds of London. Recently they took over CountryWide Bank. The Test team has a manager, 3 test analysts and 2 automated test analysts. The site is mainframe (IBM), but with most of their systems bought in. Most of the development is changing the current systems. There is a swing towards staying with the original system (vanilla) and changing the way the bank does things. Unit testing is one by the programmer and then checked by the Business Analyst. Testing is built into any changes. This means that from a change request, the business analyst, the analyst programmer and finally the test analyst estimate the time to do the job (in hours), and management make a decision on whether to make the change. A test is called a script and there are around 3000 scripts developed for all their systems. About 150 of those scripts are automated. Basically a set of good data is produced and then sets to test the boundaries.

Commercial Data Processing is a consultancy group, which supports Cognos products. Initially this was Powerhouse (a 4GL), but now is a set of business intelligence tools. When CDP bid for work they do a risk analysis of the project to decide on the level of testing, and thus the price. For a mission critical project, the requirements were rewritten in pseudocode and test data developed from the pseudocode using path testing and equivalence partition testing, before coding. The resulting system was highly robust and bug free. Regression testing is backed by automation.

Inland Revenue Department has developed and maintained its own systems(mostly written in Cobol). Anderson Consultants did the initial development and their methodology has been taken up by Inland Revenue (called IRD method). The functional specifications are signed off and test plans developed from them. Program specifications are developed and the coding is backed up with path testing and walk throughs. Integration and system testing are completed by the development team, and then the product is handed over for user acceptance testing. Here equivalence partition testing is used. The user acceptance tests are developed as cycles.

Cycle 1 would be straight forward good data. Each cycle would include more complex tests. SQA-ROBOT is used for automating some of the tests.

Summarising the above four cases: Unit testing is done, the thoroughness is left to each developer. Only one site did code inspection. Integration and system testing are not done formally but usually controlled by the systems analyst. User acceptance testing is done to varying degrees of thoroughness, a good set of data is the first attempt, followed by more complex tests. The decision on which tests to concentrate was normally decided on high risk. All sites were using automation tools for regression testing, but once again expediency was used in how many tests were automated. Inland Revenue was the only one using function points to estimate job size.

7. OTHER ISSUES

Other types of testing, such as

Stress Testing
Reliability Testing
Performance Testing
Useability Testing

are all specialised parts of testing. These would normally be covered in a third year paper devoted to testing. In other words there is not room for them in the syllabus. Useability testing can be included in Human-Computer Interface modules.

8. CONCLUSION

The current syllabi covers testing well, apart from bringing the nomenclature up to date. Practitioners are constrained by the willingness of customers to pay for testing, so may not practice what we teach, but all the managers I spoke to, were supportive of the existing syllabi.

One area of testing which is becoming prevalent is the automation of testing. This can be incorporated into the syllabus without any changes. The only problem is choosing the appropriate tool.

The aim of testing is to prevent faults occurring. One of the best techniques, code inspection, is only used by Inland Revenue. For the educator, teaching code inspection poses many problems when taught to first year students.

Overall, the testing modules in the NDBC have stood the passage of time well and do not need any major updating.

9. REFERENCES

Beizer (1992). Software testing techniques, Van Nostrand Reinhold, 1992.

Beizer (1999). Testing: Best and Worst Practises- A Baker's Dozen, Testing Techniques Newsletter, On-Line Edition, Nov1999

NACCQ (1998). New Zealand Polytechnic Qualifications in Business Computing, NACCQ, 1998