



Testing Whitepaper

**Version 1.0
1st January 2006**

1. Document Review & Version Control Status

Name of Module	Testing
Name of Document	Testing Whitepaper
Version of Document	1.0
Creation Date	1 st January 2006
Document Owner	Anthony Cairns, Test Management Consultant ISEB Software Testing Certified
Owner's Contact Direct Telephone Number	07973 138998

Table of Contents

1. Document Review & Version Control Status	2
2. Introduction	4
2.1 Background	4
2.2 Document Scope	4
2.3 Document Structure	4
2.4 Document Purpose	4
3. Testing In Context	5
3.1 Introduction	5
4. Ten Fundamental points of testing	8
4.1 Testing Overview	8
4.2 Testing Principles	9
4.3 Testing Techniques	9
4.4 Testing Stages	10
4.5 Test Tracking & Control	12
4.6 Configuration Management	12
4.7 Test Process Improvement	12
4.8 Test Metrics	13
4.9 Test Documentation	14
4.10 Manual Testing versus Automated Testing	16
5. Test Considerations	17
5.1 Case Studies	17
5.1.1 Case Study 1 – A Strategic Project	17
5.1.2 Case Study 2 – A Time Constrained Project	17
5.1.3 Case Study Conclusion	18

2. Introduction

2.1 Background

This document outlines details relating to testing needs within any project.

2.2 Document Scope

The scope of this document is to present the reader with

2.3 Document Structure

This document is structured in the following manner:

- Section 2 covers the introduction to this whitepaper
- Section 3 puts the testing in to context
- Section 4 outlines the ‘ten fundamental points of testing’
- Section 5 addresses the various test considerations

2.4 Document Purpose

The purpose of this document is to present to the reader:

- Where testing fits into an overall project
- How testing must be addressed
- What considerations must be taken prior to, and during, testing

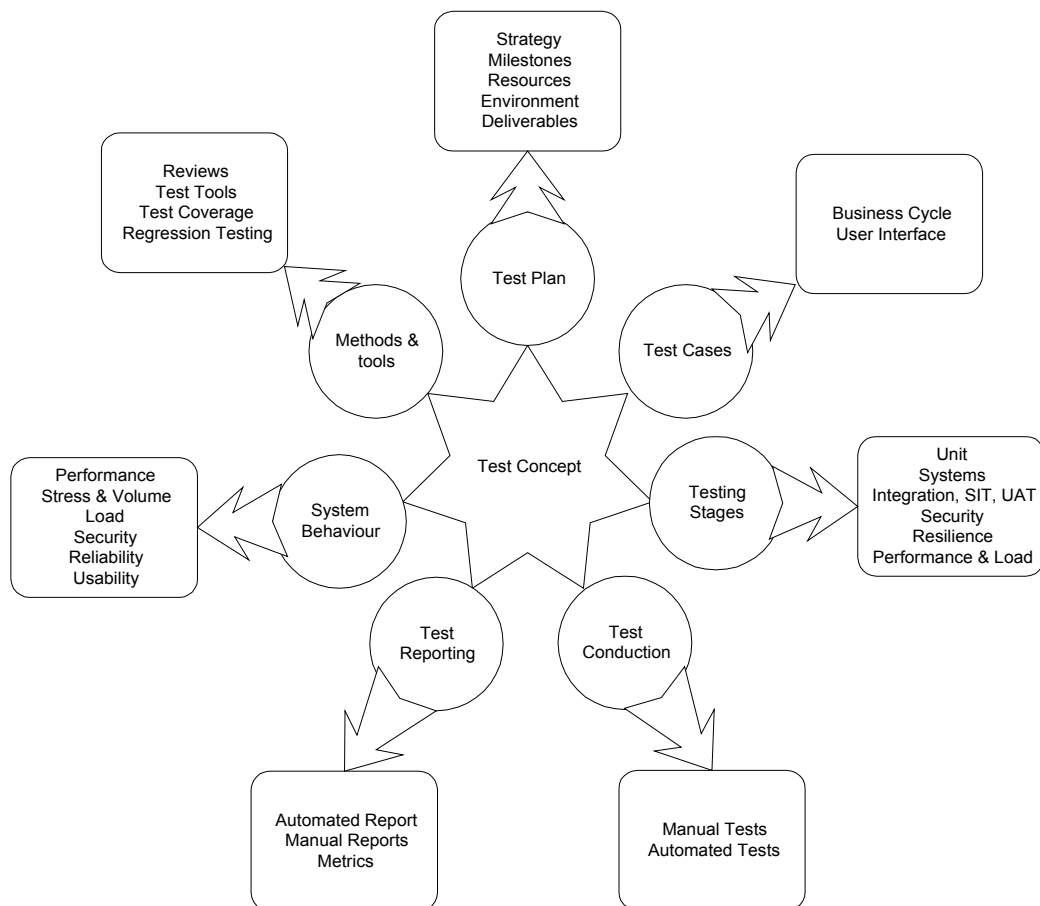
3. Testing In Context

3.1 Introduction

Although testing is usually perceived as a necessity in software development it is rarely applied as a rigorous activity. Within many projects, testing is simply omitted altogether; in others, it is executed with the distinct intent to prove that the application performs correctly under test conditions.

Software problems are can be up to 1000 times more costly to find and repair after deployment. Having software tested at several points throughout a project's life cycle is crucial, as it allows an early elimination of architectural defects and ensures the validation of software functions.

The following diagram reflects areas that must be taken into consideration when planning on implementing a testing process to a project.



Any organisation must first examine the issues involved in introducing a testing process into an organisation and the management of its successful adoption and use.

When any software development project is started the following have to be taken into account as early as possible as to ensure the overall success in a controlled manner as possible:

- Obtain management commitment and involvement
- Project needs by the business
- Establish any existing baseline software testing practices
- Identify related organisational processes
- Establish the scope and scale of the overall testing process
- What staff are required
- What the costs are
- What the quality level is to be
- What timeline must the project be based upon
- How much effort is required to implement a test strategy
- Any external factors
- Outline what training and mentoring are available
- How will your organisation monitor and improve the testing process

Each of these outlined items is detailed further below.

The first key task is to ensure that the introduction of the testing process is fully supported by the appropriate senior management within the organisation. It is important that the senior management is fully aware of the importance of a formal testing process as well as the potential cost of not following good testing practices.

Any project that is being undertaken or proposed must initially establish what the business needs from the project itself.

Once any existing levels of software testing have been assessed in an organisation then it will be possible to identify a baseline from which to govern testing practices.

Due to information technology departments, quality assurance departments and specific business areas existing in most organisations there is a need to identify what other processes already exist that may have an impact on the testing process, such as management practices and development standards. This will provide an effective means of promoting the benefits of the testing process.

When organisations consider the scope and scale of the overall testing process it must be established as to whether the intention is to implement and perform an organisation-wide testing process or a small-scale rollout of the process, these are generally decided based upon a mixture of cost and organisational difficulties.

The staff requirements must be estimated as early as possible, as to ensure that any lead times for recruitment and staff allocation can be taken into account.

The overall costs of the entire testing process definition and performance must also be taken into account, as these can quite easily escalate beyond expectation if control is not maintained.

A decision must be made to state the level of quality that is expected from the final delivery, as this is also likely to have an impact on the overall costs of the project.

Most projects will have an idea of the timescales that are required for the implementation; this is likely to make an impact on the overall staffing requirements and costs.

A certain amount of effort will have to be realised during the planning of any test strategy implementation, which is likely to comprise of many areas from within the organisation, and is certain to require external resources due to the expertise required throughout.

External factors will need to be considered, such as with external vendor developed work for additional development work on their own products, or input from external governing standards agencies or from governmental sources as a result of changes in legal requirements.

To ensure the successful implementation and use of the testing process it is essential that suitable staff training is provided as well as the allocation of a mentor, who will provide training and mentoring to other staff in need of guidance.

Once a formal testing process has been introduced it is vital that the progress is monitored so that the benefits of introducing the process can be quantified and the process can be improved. Progress reports and metrics will be used to provide this information to the senior management and other areas within an organisation.

4. Ten Fundamental points of testing

Within the realms of testing ten main fundamental points have been defined that have to be taken into account throughout the lifecycle of testing, these are as follows:

1. Testing Overview
2. Testing Principles
3. Testing Stages
4. Testing Techniques
5. Test Tracking & Control
6. Configuration Management
7. Test Process Improvement
8. Test Metrics
9. Test Documentation
10. Manual Testing versus Automated Testing

These 'Ten Fundamental points of testing' are detailed further in the sections below.

4.1 Testing Overview

Due to the process of software development maturing over the years, with the inception and use of formal methods, more formal testing methods and techniques have been adopted by testing professionals.

The most common reasons for testing software are as follows:

- To ensure that it corresponds to its design specification
- To establish what the limitations are
- To mitigate the risk of failure
- To provide confidence that it performs adequately
- To uncover defects
- To understand the risks involved in releasing a new or upgraded version

When looking at the needs of testing the overall risks must be addressed for the AUT (Application Under Test), these include:

- The business, safety or security criticality
- The commercial and/or public visibility

4.2 Testing Principles

The complete principles of testing can be fully defined under the following 5 areas:

- **Verification** is the process that confirms that a software product works correctly, by asking the question '*Are we building the product right?*'
- **Validation** is the process that confirms that a software product works according to the specification as defined by the client, by asking the question '*Are we building the right product?*'
- **Actual Testing** is reliant upon the physical running of the program
- **Desk Testing** is based upon reviews and formal analysis
- **QA** stands for Quality Assurance, which applies to every aspect of the software process, covering both development and testing areas

4.3 Testing Techniques

A number of testing techniques are employed to assist with the designing and developments of effective Test Cases. These practical and pragmatic test design techniques are defined as follows:

- **Functional Testing** techniques are used to confirm that an application meets its functional requirements, these include Equivalence Partitioning (using typical valid and invalid examples of data), Boundary Value Analysis (using strategic data based upon valid and invalid examples of business specific data limitations), Random Testing (using ad-hoc examples of valid data input), Static Testing (addresses code reviews, static analysis and complexity estimating) and Thread Testing (used to test the business logic process steps of an application)
- **Non-Functional Testing** techniques are used to verify that the application meets its non-functional requirements, including Configuration Testing, (to ensure that hardware and software have been correctly installed) Compatibility Testing (used to verify that the application being tested does not adversely impact other systems in the live environment), Documentation and Help Testing (this verifies that the user documentation and help files provide the correct information to assist the users), Fault Recovery Testing (verifies that a system can successfully be recovered after a an error or exception has been encountered), Performance Testing (addresses external,

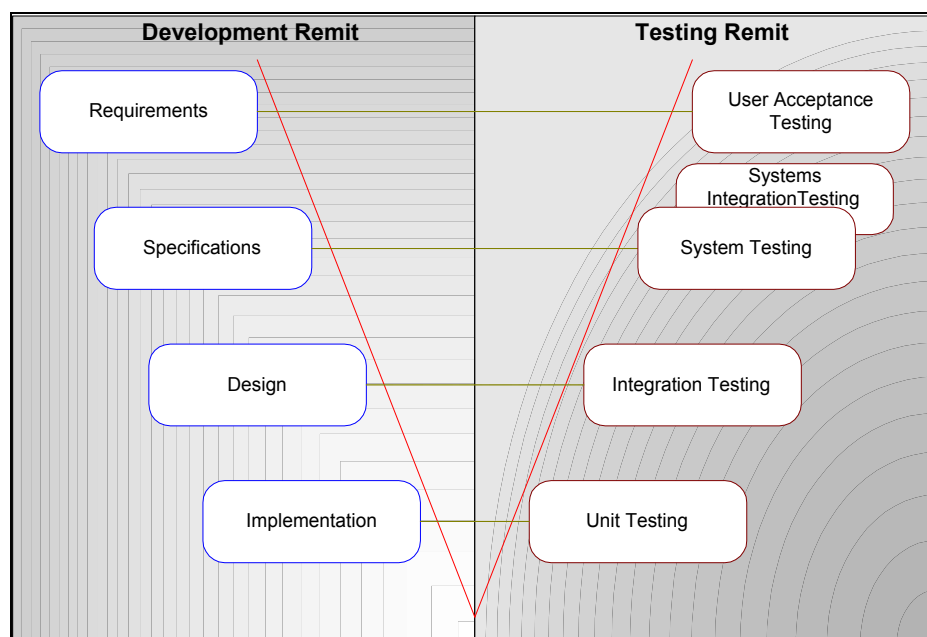
system and user response times as well as CPU and memory utilisation), Reliability Testing (ensures the reliability and robustness of the application being tested), Security Testing (verifies that data integrity, availability and confidentiality are acceptable), Load Testing (establishes that the system will perform correctly under peak loads), Usability Testing (establishes that the system works in accordance to the typical business tasks under controlled conditions) and Volume Testing (tests the capabilities of the system when under heavy volumes of data)

Within these testing techniques some specific high-level approach can be defined to ensure the best is obtained from both the Functional and Non-Functional techniques, which are as follows:

- **Positive and Negative Testing** looks at both positive and negative input into the system, such as covered by data values that are within acceptable limits and those that fall outside acceptable field validation limits
- **White Box and Black Box Testing** are based upon a test analysts level of knowledge of the system to be tested
- **Error Guessing** is a testers 'intuition' skill that can be applied in all other testing techniques to produce more effective tests

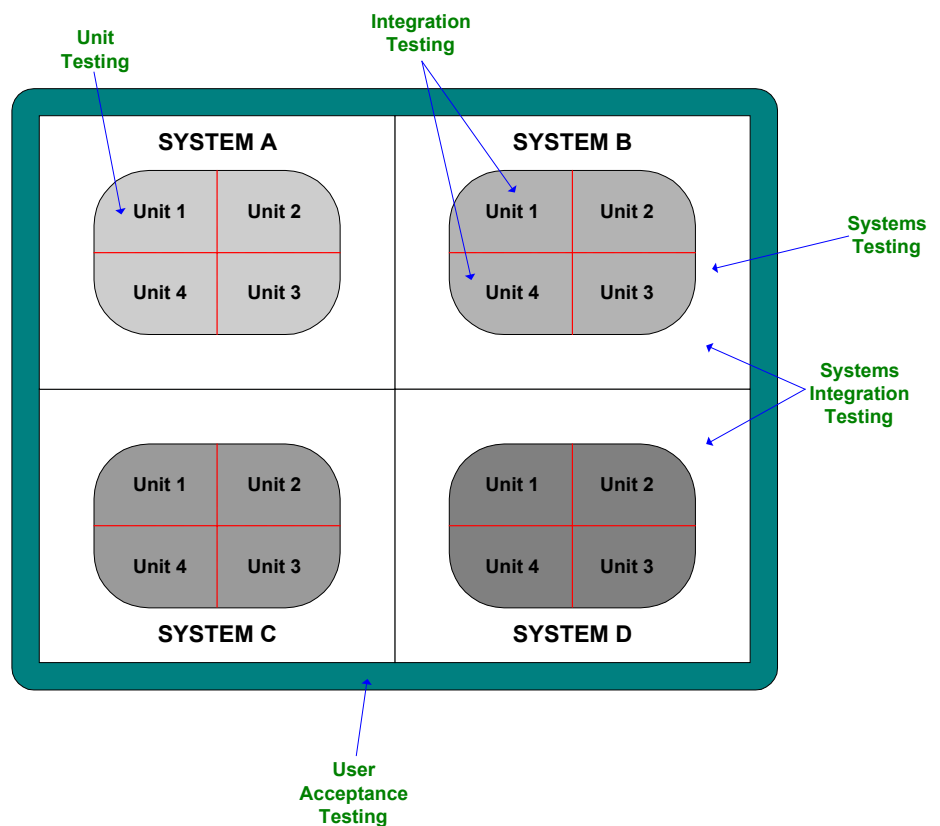
4.4 Testing Stages

All of the stages involved in the lifecycle of testing can be addressed by the use of the 'V-Model', which is a structured software development and testing model that helps to highlight the need to plan and prepare for testing.



The stages defined within the remit of testing are as follows:

- **Unit Testing** is used to ensure reliable program units are produced that meet their requirements and to identify errors in program logic. Unit Testing is conducted by the development team in the majority of projects
- **Integration Testing** is performed to demonstrate that the modules that make up the complete application being tested interface and interact together in a correct and stable manner
- **System Testing** is performed to establish confidence that the application being tested will be accepted, by addressing the functional and structural stability of the system
- **Systems Integration Testing** is performed to provide confidence that the application being testing is able to interoperate successfully with other specified software systems and does not have an adverse affect on other systems that may also be present in the live environment, or vice versa
- **User Acceptance Testing** is performed to confirm that the application being tested meets its business requirements and to provide confidence that the system works correctly and is usable before it is formally delivered to the business



4.5 Test Tracking & Control

Measure must be taken to ensure that the overall tracking of the testing process and life cycle are addressed, which will also incorporate the overall controlling mechanism.

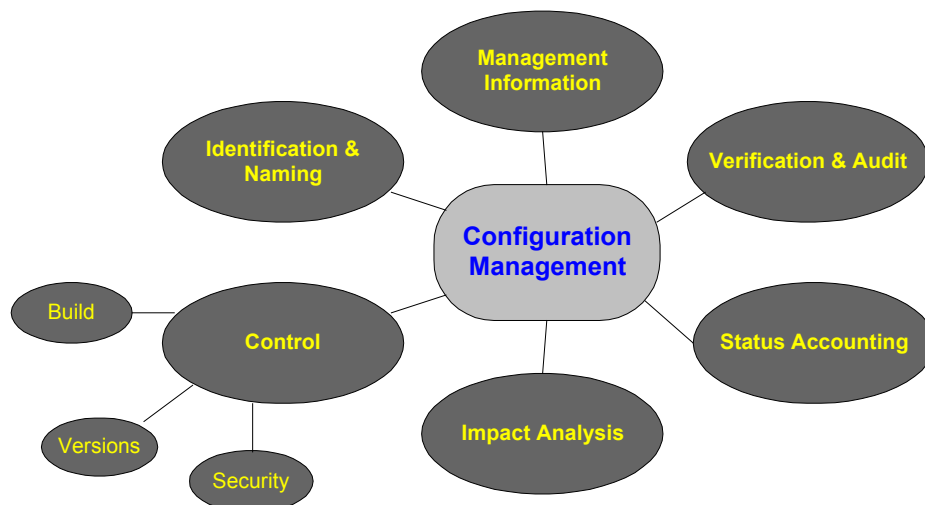
The following points must also be addressed throughout testing:

- Report on the progress against the defined Test Plan
- Implement defect reporting database or other defect logging method
- Monitor test execution
- Maintain regular test reporting to project team

4.6 Configuration Management

The primary benefit of Configuration Management is to allow an organisation to take account of what they already have.

The following diagram outlines the main areas relating to Configuration Management.



Two of the major components of Configuration Management are Impact Analysis and Control. Impact Analysis provides a whole range of information specifically addressing change and details the overall risks involved in the changes. Control addresses version control, builds, and security access restrictions.

4.7 Test Process Improvement

The need to improve a test process is always in existence, as to enable lessons to be learnt by any mistakes that are made.

One primary way to learning by experience is to collect and use metrics that are observed during the course of a project. This information collected can then be used to establish whether the testing process is improving over time.

If no formal measurements are made then there will be no way to verify the overall effectiveness of the testing process, which will therefore mean that there will be no way to improve the testing process.

The need to produce testing metrics is highlighted when the need for objective assessment of testing quality against agreed standards is taken into account.

4.8 Test Metrics

A software test metric is a standard means of measuring some attribute of the software development and testing process.

A number of test metrics can be produced, which include the following:

- Estimating the testing effort required
- Highlighting areas that may be error-prone
- Measure and track the overall cost of testing
- Measure the return on investment (ROI)
- To predict when to stop testing

A number of metrics are typically used within the testing process. These are outlined below:

- Size
- Complexity
- Cost/Effort
- Defect found during testing
- Test progress

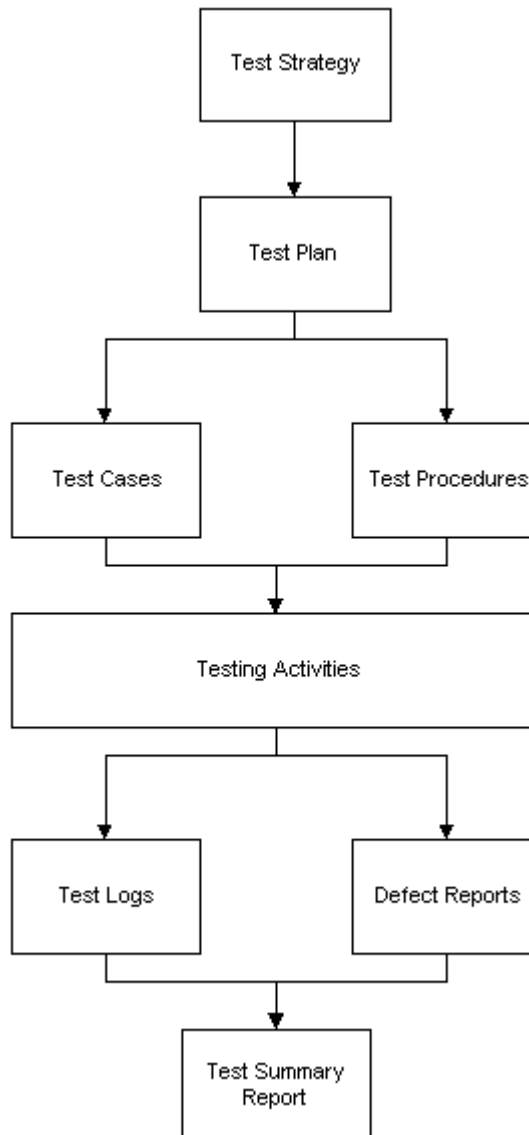
It is important that the collection of testing metrics is as unobtrusive as possible. The following requirements must be considered in setting up a typical metrics program:

- Definition of organisational objectives
- Establish roles and responsibilities

- Research and decide upon what metrics are to be recorded
- What infrastructure is needed to be set up to record the metrics

4.9 Test Documentation

The recommended minimum amount of testing related document requirements, which will then ensure a complete auditable coverage of testing, are outlined in the following diagram:



A **Test Strategy** is produced to outline the strategic approach that is being followed with regards to the overall implementation of the testing process.

A **Test Plan** details the scope, approach, resources, and schedule of testing activities. It identifies the items to be tested, the features to be tested, the testing tasks to be performed, the personnel responsible for each task, and the risks associated with the plan. It also refines the test approach and identifies the features to be covered by the design

and its associated tests. It also identifies the test cases and test procedures and specifies the feature pass/fail criteria.

Test Cases are produced to document the actual values used for input along with the anticipated outputs. A test case also identifies any constraints on the test procedures resulting from use of that specific test case.

Test Procedures describe the steps required to operate the system and exercise test cases in order to implement the associated tests. Test procedures are intended to be followed step by step and should not have extraneous detail.

A **Test Log** is used to record what occurred during test execution.

Defect Reports are produced to describe any event that occurs during testing which requires further investigation. These defects may include encountered errors, requests for enhancement, or any other event needing clarification.

A **Test Summary Report** summarises the testing activities associated with one or more test design specifications.

4.10 Manual Testing versus Automated Testing

There are plenty of both negative points as well as beneficial points that exist with both manual and automated testing methods.

Unless you create and run new test cases under an automation tool right from the start there is no huge benefit, due to most defects in software being found during manual testing methods.

The following tables outlines the various negative and positive points comparison between manual and automated testing:

MANUAL TESTING	AUTOMATED TESTING
Can be random and error-prone if not managed correctly	Very expensive to purchase and implement
Experienced testing staff can be ready to test immediately	Requires specific training in the use of automated test tools
Slow in its nature due to being Human resource intensive	Takes time to train existing staff up and to implement
Boring to perform by testing staff in certain areas	It is a software development effort on its own
More flexible	Less flexible when it comes to 'work-around' solutions
Suitable for both short-term and long term testing requirements	Primarily for long term benefit
Is as complex as the developed test scenarios that are produced	Makes testing more complex
Not too complicated to configure, mainly dependant upon close management	Complicated and hard to configure
Testing staff errors can happen during testing	Automating implemented tests can create as many problems as it solves
Frequency of testing must be closely monitored, managed and scheduled accordingly	Allows testing to happen more frequently
Careful management required to ensure test consistency	Ensure test consistency
Staff need to be more aware of the implications and needs of testing	Allows testing to be done by staff with less skills

5. Test Considerations

All of the testing areas need to be fully assessed when looking at how to approach the testing requirements to best suit the particular project being addressed.

5.1 Case Studies

The case studies outlined within this section address the following 2 main types of project:

- A Strategic Project
- A Time Constrained Project

5.1.1 Case Study 1 – A Strategic Project

Case Study 1 is following a formal structured approach to the testing requirements for a strategic project.

Scenario: Company ABC LLP has been working on a development to run an online financial management system. They have to fully test the delivered working system within 18 months, and it has to be completely structured and comprehensive in its approach to completely satisfy their strict Quality Assurance department.

Considerations: As this scenario was fully strategic in its approach some preliminary assessment of testing would have been performed throughout strategic stages of development.

5.1.2 Case Study 2 – A Time Constrained Project

Case Study 2 is following an informal approach to the testing requirements due to there being a time constraint, which ensures that any time limitations are adhered to in the overall testing life cycle.

Scenario: Company XYZ LLP has been working on a development to produce a working financial management system too. They have 6 months to test the developed system.

Considerations: As this scenario was based upon time constraints there was no time to perform any test assessment prior to the product being available for testing as the test team was put together at the last minute.

5.1.3 Case Study Conclusion

The table below reflects the overall trade-off and benefits between Case Study 1 and Case Study 2:

A TIME CONSTRAINED PROJECT vs. A STRATEGIC PROJECT		
Area Compromised	Trade-Off	Benefit
Test documentation	Quality and quantity of documentation content. Risk being that the quality and coverage is not as detailed as it could be.	Time to produce documents reduced.
Test Plan and Test Project Plan content	Amount of documented coverage. Risk being that the overall in depth details will not exist.	Reduced time to produce content.
Test tool research and selection	Usage of test tools. Risk not necessarily being an issue due to the overall timescales required for the selection, implementation and training of them.	Save time that would have been spent on test tool selection, implementation and training in the use of them.
Test coverage	Negative testing and specific testing addressing data boundaries are omitted. Risk being that the overall boundaries of data are not completely tested for specific business test scenarios.	Due to only testing a valid range of data examples the testing time will be reduced considerably.
Testing techniques	Non-functional testing is omitted, which would normally test out areas such as the configuration of the system, system recovery and performance. Risk being that non-functional based areas of the system	As specific areas of testing are focused upon, such as Integration, Systems Integration and User Acceptance testing, time can be saved, fully optimized and have a specific primary focus.
Regression testing	The re-testing if failed areas only cover failed sections and do not cover any other part of the developed system. Risk being that an area of failure may result in a failure on another section of the development.	Limited regression testing will result in a quicker re-testing process timescales of any failed areas.

Although a compromise is made by following the requirements of a time constrained project a structured, although somewhat limited, testing approach can still be followed, albeit not quite as rigorous as that of the strategic approach.