



ACCESS COMPUTER  
consulting plc

Access Computer Consulting Plc  
Access House  
25-29 Church Street, Basingstoke  
Hampshire RG21 7QQ  
tel. +44 (0) 1256 368800  
fax. +44 (0) 1256 368811  
www.accessplc.com

# *Access Computer Consulting plc*

## Testing Practice

# Introduction to the Access Quality Methodology (AQM)



Revision: *Version 1.0*

January 2002



**Document Management****Document Approval:**

| Version | Date       | Approved By | Client/Department              |
|---------|------------|-------------|--------------------------------|
| 1.0     | 04/02/2002 | Tony Head   | Access Computer Consulting plc |

**Document Information/History:**

| Author       | Status | Version | Date       | Description |
|--------------|--------|---------|------------|-------------|
| Mike Birbeck | Draft  | 1.0     | 04/02/2002 | Initial     |

**Document Owner:**

| Name         | Position             |
|--------------|----------------------|
| Mike Birbeck | Principal Consultant |

**Distribution List:**

| Name      | Company / Details                       |
|-----------|---|
| Tony Head | Access Computer Consulting plc/Director |
| John Bass | Access Computer Consulting plc/Sales    |
|           |   |
|           |   |
|           |   |

Copyright Notice © Copyright Access Computer Consulting plc Limited 2001–2002

The information contained in this document is the property of Access Computer Consulting plc. The contents of the document must not be reproduced or disclosed wholly or in part or used for purposes other than that for which the document is supplied without the prior written permission of Access Computer Consulting plc.

# Table Of Contents

- 1. MANAGEMENT SUMMARY ..... 5**
- 1.1 INTRODUCTION ..... 5
- 2. ACCESS COMPUTER CONSULTING PLC METHODOLOGY ..... 6**
- 2.1 QUALITY ASSURANCE METHODOLOGY..... 6
- 2.2 QUALITY PLANNING..... 6
- 2.3 RISK MANAGEMENT ..... 7
- 2.4 QUALITY COSTING ..... 8
- 2.5 PROCESS MODELS ..... 9
- 2.6 VERIFICATION AND VALIDATION ..... 10
- 2.6.1 *Specifications* ..... 10
- 2.6.2 *Software* ..... 10
- 2.7 SOFTWARE TESTING..... 11
- 2.7.1 *Test Strategy* ..... 11
- 2.7.2 *Test Planning*..... 11
- 2.7.3 *Test Case Design* ..... 12
- 2.7.4 *Test Procedure Design*..... 12
- 2.7.5 *Perform Testing*..... 12
- 2.7.6 *Testing Phases Definitions*..... 13
- 2.7.7 *Exit and Entry Criteria*..... 14
- 2.7.8 *Document Results*..... 17
- 2.8 TOOLS ..... 17
- 3. APPENDICES ..... 19**
- 3.1 TEST PLAN IDENTIFIER:..... 20
- 3.2 INTRODUCTION ..... 20
- 3.3 TEST ITEMS ..... 20
- 3.4 FEATURES TO BE TESTED..... 20
- 3.5 FEATURES NOT TO BE TESTED..... 20
- 3.6 APPROACH ..... 21
- 3.7 ITEM PASS/FAIL CRITERIA ..... 21
- 3.8 SUSPENSION CRITERIA AND RESUMPTION REQUIREMENTS ..... 21
- 3.9 TEST DELIVERABLES..... 21
- 3.10 TESTING TASKS ..... 21
- 3.11 ENVIRONMENTAL NEEDS..... 22
- 3.12 RESPONSIBILITIES ..... 22
- 3.13 STAFFING AND TRAINING NEEDS ..... 22
- 3.14 SCHEDULE..... 22
- 3.15 RISKS AND CONTINGENCIES..... 22
- 3.16 APPROVALS ..... 22

# 1. Management Summary

## 1.1 Introduction

This document serves as a high level introduction the Access Quality Methodology (AQM).

The document's intended audience is anyone who wishes to gain an insight into this methodology and serves as an introduction to the AQM for consultancy clients.

The full methodology is formalised in the Access Body Of Knowledge (ABOK Version 2.0) and is the primary reference for all Access Testing Personnel and Consultants. All of Access' Principal Consultants are formally trained in this methodology.

It is important to note that the full AQM is subject to continual revision as a part of the consultancy's project and programme management improvement progress. This improvement process uses feedback from all projects undertaken by Access to ensure that the methodology is continually improved.

The full methodology includes a full list of process, quality and testing templates and these remain proprietary to Access.

## 2. Access Computer Consulting plc Methodology

### 2.1 Quality Assurance Methodology

Access Computer Consulting plc has derived a formal quality model based upon an extensive legacy of project and testing management as well as by commitment to industry best practice as articulated by the Software Engineering Institutes' Body of Knowledge (PMBOK) and the Capability Maturity Model. This derived methodology is known as the Access Quality Methodology (AQM).

A primary principle of the AQM is the definition of testing as being integral within the quality management process. This definition requires strict adherence to the principles of formal risk, quality and test management.

Access Computer Consulting plc seeks, through the AQM, to work within a client's project risk management process to help define and mitigate the risk to quality.

Many of our clients have chosen to outsource the quality management process to Access Computer Consulting plc who then manage it through risk definition, measurement, mitigation, testing and reporting.

### 2.2 Quality Planning

Access Computer Consulting plc's Quality Assurance Methodology (AQM) is underpinned by the following plans:

- A Risk Management Plan
- A Quality Achievement Plan
- A Quality Control Plan
- A Quality Preservation Plan

The following schema highlights the features of these quality plans:

#### 1. Risk Management Plan

- Risks Identified
- Risk Reduction Measures Chosen

#### 2. Quality Achievement Plan

- Characterises the system to be developed/purchased
- Defines client's expectations of requirements
- Defines development methodology
- Defines chosen tool support
- Defines target environment

### 3. Quality Control Plan

- Defines planned product types
- Defines specifications and standards
- Defines quality control activities

### 4. Quality Preservation Plan

- Defines change control
- Defines configuration management control

## 2.3 Risk Management

Running parallel with the quality planning required by the AQM is the requirement for careful and empirically based risk management.

The AQM has a standard risk management process that can be summarised as follows:

- Risk Identification
- Risk Analysis
- Risk Response Planning
- Risk Resolution and Monitoring

At a high level this model can be schematically represented as follows:

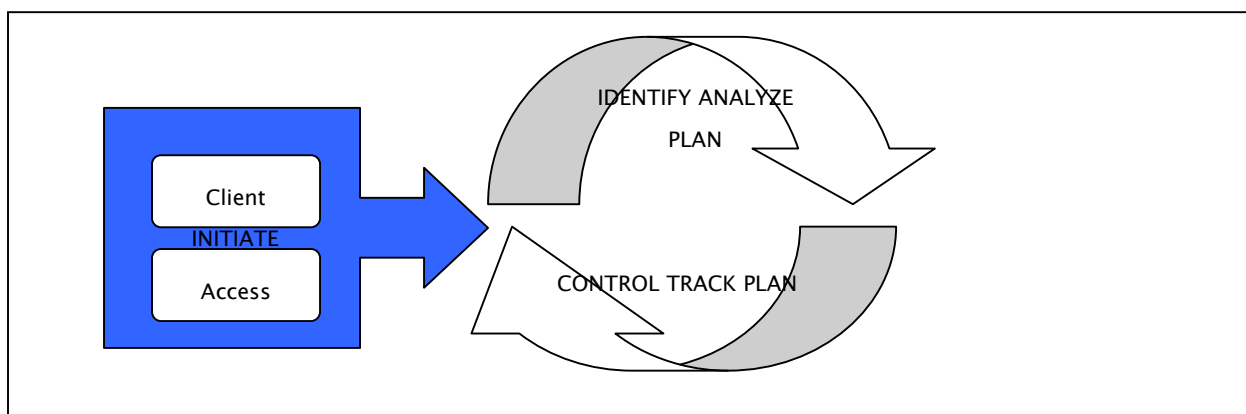


Figure 1.0

- **INITIATE** Commit to creating a team culture. Both client and Access Computer Consulting plc commit to sustain teams.
- **IDENTIFY** Search for and locate risks before they become problems.



This approach is found to be more empirical than some of the more general industry statements e.g. +- 30% of the project budget should be allocated to software quality assurance and testing.

Another benefit to our clients that accrues through the use of the AQM is that the model allows our clients to factor in the correct ratio between the use of in-house testing personnel and sub-contractors. The AQM may be adopted within a relatively small time and with support from Access Computer Consulting.

### 2.5 Process Models

The AQM allows for integration with the process model or project management methodology relevant to our client. The AQM is aligned with the traditional V model of development but is flexible enough to support Prototyping, RAD, Extreme Programming and a variety of other process models.

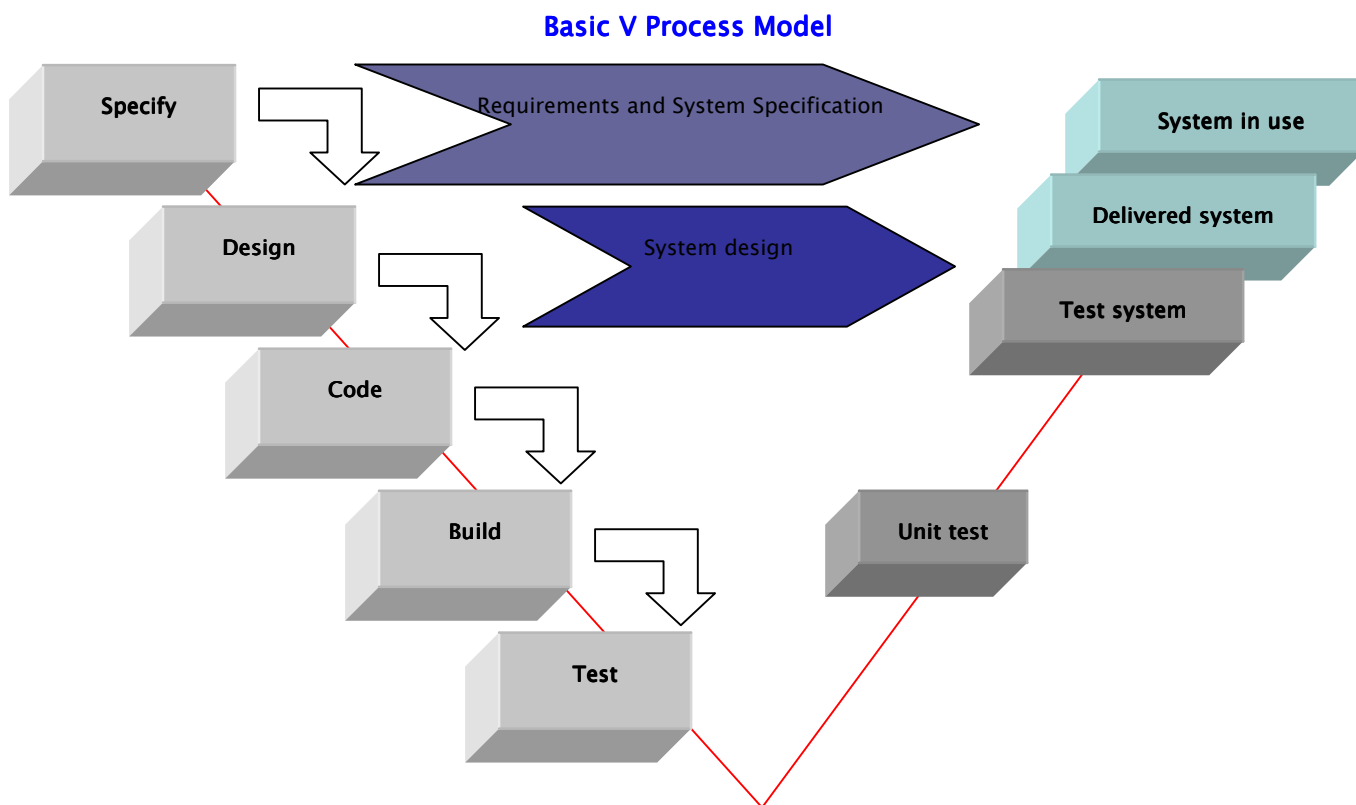


Figure 3.0

## 2.6 Verification and Validation

The AQM covers the traditional software validation and verification phases typical of other quality and testing methodologies.

**For the purposes of this document, validation is checking that we have built the system right. Verification is checking that we have built the right system.**

Within the context of the of the AQM, verification and validation is inclusive of testing but also encompasses quality checking the project management process the relevant documentation, specifications etc.

The main features verified of validated by the AQM include:



Figure 4.0

### 2.6.1 Specifications

**The AQM tests specifications for the following attributes:**

- Internal Consistency – Does the specification allow any contradictions e.g. if the specification calls for only one person to be allowed to amend a transaction at a time, does it then go to call for simultaneous validation of the transaction by another user?
- Completeness – Is the specification complete and does it fulfil all necessary quality criteria?
- Behavioural Equivalence – Does the specification cover all facets implied by a high-level requirement statement e.g. if a specification calls for 5 transaction types, does it then specify what those transaction types are?
- Standardisation – Does the specification meet the criteria outlined in the quality control plan?

### 2.6.2 Software

**The AQM tests software for the following attributes:**

- Equivalence – Does the code produced meet the specification?
- Behaviour – Does the software behave as required?
- Functionality – Does the software have all the required functional attributes?

## 2.7 Software Testing

A feature of the AQM approach to testing is the requirement for a “Chinese Wall” to exist between the developers and the testers. This implies that while developers might be responsible for **unit testing** the software, the same developers should not then be responsible for **system, integration, performance, load, stress and user acceptance** testing.

This approach is pragmatic in the sense that it recognises that a client might not have access to a dedicated testing team and that not all clients will be able (or wish) to sub-contract the testing process.

Under the AQM, members of the development team are mandated to undertake testing with the exception of user acceptance testing, with the proviso that the developers who do the testing are not the same people who did the development.

The AQM seeks to ensure that testers are well served by high quality requirement documents and specifications but makes allowance for projects where the methodology has not been followed and software has been delivered for testing (over-the-wall testing).

In less than optimal cases the AQM seeks to generate test cases by analysing high-level **Data Flow Diagrams** for a system and by using existing documentation wherever this exists.

The AQM mandates the development of a testing strategy prior to the commencement of the development of a testing plan and test scripts.

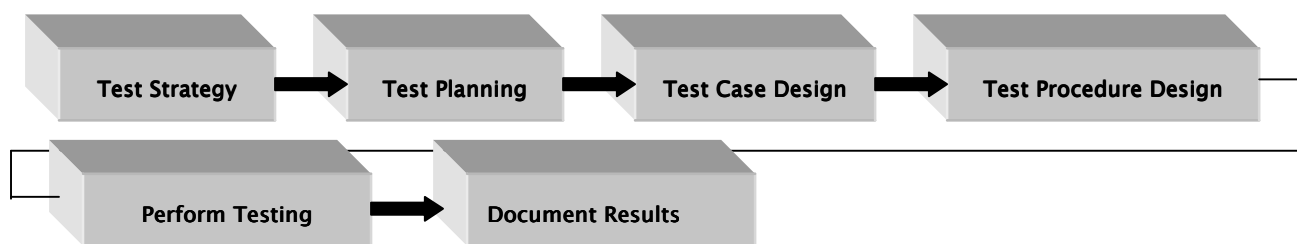


Figure 5.0

### 2.7.1 Test Strategy

The AQM test strategy defines an overall approach to testing and also identifies what levels of testing are to be applied as well as the methods, techniques and tools that are to be used. The AQM test strategy is closely aligned to the requirements detailed in the quality control plan. Access will tailor the test strategy to work within the context of existing client test strategies where these exist.

In all cases, the AQM mandates the use of a negative testing strategy i.e. the testing process is designed to find errors within the software under test and not only to prove that the software does what it was designed to do.

### 2.7.2 Test Planning

The AQM test plan requires a statement of all the following things:

- What needs to be tested
- What the level and coverage of the testing will be
- What the sequence of the testing will be
- How the test strategy will be used in testing

- How the test environment will be set up

**The AQM requires that the test plan comprises of at least the following items:**

- Unit Test Plan
- Integration Test Plan
- System Test Plan
- Acceptance Test

### 2.7.3 Test Case Design

Once the test plan has been created, the AQM requires a set of test cases for each item on the plan.

**Each test case should show:**

- How each specified requirement is to be tested. These specifications are to be linked to the specified test cases by a Verification Cross Reference Index
- What the success criteria for each case are

As designated by the AQM test strategy, the test cases should include both positive and negative cases but overall the emphasis should be on negative testing.

### 2.7.4 Test Procedure Design

The AQM mandates that for each test case there should be a test procedure showing the exact process to be followed to execute the test cases.

**Each test procedure should show:**

- A separate procedure for each test case
- All the steps required to execute the required procedures

### 2.7.5 Perform Testing

As has been outlined, the AQM requires that, at the very least, the following test phases be

performed:

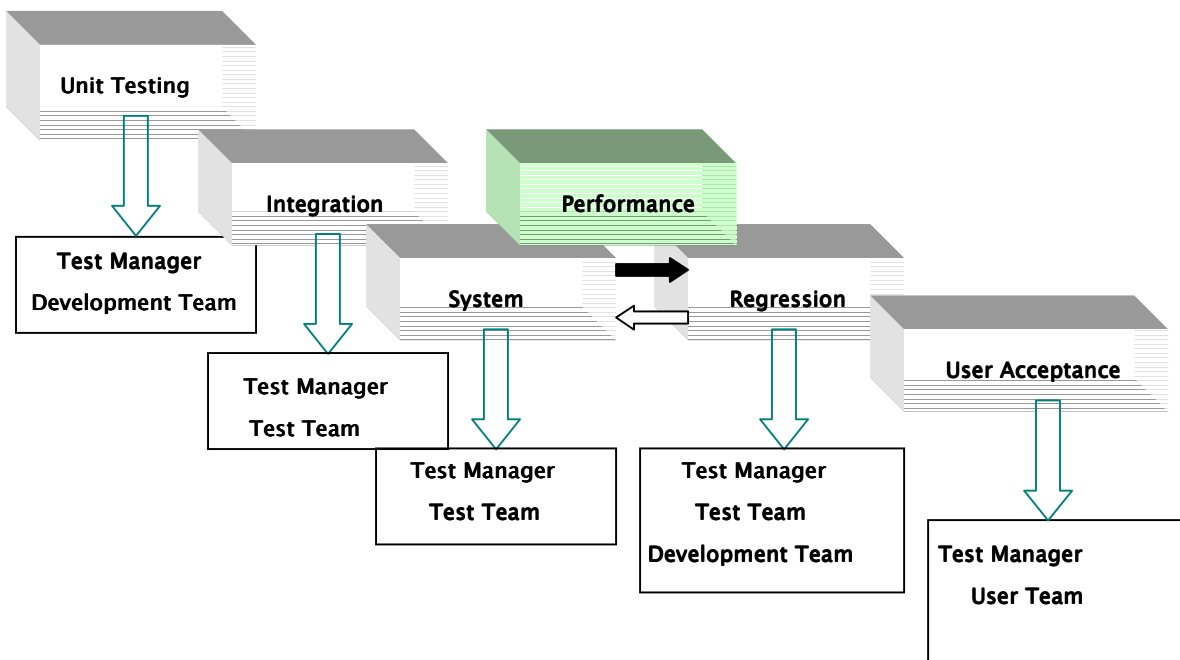


Figure 6.0

## 2.7.6 Testing Phases Definitions

### Unit Testing

The AQM defines unit testing as being the testing process required to prove that an individual software component works against a stated set of specification and criteria.

### Integration Testing

The AQM defines system testing as being the testing process required to prove that a business function works end to end over a business solution though the integration of a set of independent software components.

### System Testing

The AQM defines system testing as being the process required to verify business functionality against a stated set of specifications and criteria.

### Performance Testing

The AQM defines performance testing as being the process required to verify the system against a stated set of specifications and criteria in terms of performance (with varying volumes), stress (e.g. number of users) and operability (e.g. recovery from software failure).

### User Acceptance Testing

The AQM defines user acceptance testing as being the process required to confirm that the software meets business requirements against a stated set of specification and requirements.

### 2.7.7 Exit and Entry Criteria

The AQM specifies that a formal set of exit and entry gates be defined and signed off to allow progression from one phase of testing to the next. At the highest level these gates cover the following phase transitions and consist of the following minimum criteria:

#### Phase - Unit Testing

##### Entry Criteria

- All required plans in place (risk, quality, test etc.)
- Test strategy signed off
- Code sufficiently complete for testing (syntax checked, compiled etc.)
- Unit testing specification in place
- Test environment configured and available
- Test data available
- Test tools, test harnesses, stubs, drivers in place

##### Exit Criteria

- Unit test entry criteria met
- Test specification with actual results signed off for review by test manager
- All tests signed off
- List of outstanding issues compiled, with mitigating plans and sign off by the project manager (at the very least)

#### Phase -Integration Testing

##### Entry Criteria

- Unit test exit criteria met
- Middleware unit tested
- Integration test specification signed off
- Middleware installed in appropriate environments and available
- Test data available
- Testing staff available
- Vendor support available
- Procedure for code and library management in place
- Change control and configuration management procedures in place
- All additional criteria identified in test strategy met

**Exit Criteria**

- All tests signed off
- Software errors fixed and re-tested with an agreed action plan for each outstanding error. All test results recorded.
- Metrics recorded and documented
- Test completion report produced and signed off
- Test completion review carried out

**Phase – System Test****Entry Criteria**

- The exit criteria for integration testing have been met
- The business requirements document has been signed off
- The technical/design specification has been signed off
- The system specification has been developed and signed off
- The test environment has been set up
- Required testing tools have been configured

**Exit Criteria**

- All tests or an agreed subset, executed and signed off
- All severity 1,2 & 3 software errors fixed and re-tested plus an agreed action plan for each outstanding error in place
- All test results and evidence recorded
- Metrics collected and documented
- Test completion report produced and signed off
- Test completion review carried out

**Phase – Performance Testing (Volume, Stress)****Entry Criteria**

- All exit criteria for the system test phase have been signed off
- A performance model has been developed (i.e. how specific aspects of the system should perform)
- A “test to live” performance measurement model should have been developed
- Maximum data volume models should have been specified
- Maximum transaction throughputs should have been specified
- Performance test specifications completed and signed off
- The performance test environment has been set up
- Test data is available
- Test tools available (e.g. scripts for generating volume data, simulation of virtual

users, performance monitoring and measurement)

- Test and system support staff available
- Change management and configuration management procedures in place
- Any additional criteria specified in the testing strategy in place

#### **Exit Criteria**

- All tests, or an agreed subset, executed and signed off
- All critical problems fixed and re-testeded plus an action plan for each outstanding error in place
- All tests results and evidence collected and documented
- Metrics collected and documented
- Test completion report produced and signed off
- Test completion review signed off

#### **Phase – User Acceptance**

##### **Entry Criteria**

- All the exit criteria for the Performance testing phase have been met
- User acceptance test specification complete and signed off
- Resources (people, rooms, desks, workstations etc. available)
- Test data available
- Test environments available
- Test tools available
- Business personnel trained and given appropriate documentation
- Procedure for code library management in place
- Regression test pack available
- Additional entry criteria specified in the test environment met

##### **Exit Criteria**

- All tests, or an agreed subset, executed and signed off
- All software errors (highest severity levels) fixed and retested with action plans for outstanding errors in place
- All test results recorded
- Any changes to the business process documented and signed off
- Metrics collected and signed off
- Test completion report produced and signed off
- Test completion review carried out

### 2.7.8 Document Results

The AQM requires that all test results be formally documented in the form of a test log.

**The AQM requires that the log should:**

- Record when each test has been run
- The outcome of each test execution showing:
  - Date of execution
  - Time of execution
  - Nature of test run
  - Problem/Issue encountered

Formal management summaries should be produced from the test log.

**The AQM requires that these summaries should:**

- Show where the problem was found
- When the problem/issue was found
- Show how the problem was found
- Show the status of the problem
- Show the uniqueness of the problem
- Show importance/priority of problem/issue
- Show number of open issues
- Show number of open issues as a percentage of issues logged
- Show age of issues sorted by relative priority

## 2.8 Tools

The AQM mandates the use of tools “where appropriate”. The AQM helps the client to decide whether the use of software tools will add value to the quality/testing process by using the AQM costing model.

The following criteria are used to determine tool cost effectiveness:

- The total cost of using the specified tool (licences, user training etc.) as a percentage of the total cost of ownership of the software (whole life costs).
- The length of time required in creating testing scripts (in the case of automation tools). **It can take between 3 and 10 times as long to create, verify and minimally document some automated scripts as it does to manually run the required tests by hand.**
- The cost and availability of the specific skill sets required to support the tool over the lifetime of the software under test.
- The number of times a specific test set will be run. The more times it will be required to run a test set over the life of the software product, the more cost effective it becomes to use a software tool.

- The stability of the software under consideration. This stability is implied by the maturity of the change and configuration management process and the likelihood of changes to the software over its' lifetime. In the case of automated tools, a large number of changes to the software under test imply a significant maintenance overhead in updating and maintaining the test scripts associated with the tool.
- The relative cost of finding issues/problems later in the development lifecycle because of the use of tools. This is based up on the fact that there is a set up time associated with the use of software tool (particularly automated tools) that implies that bugs etc. may only be discovered later in the development/testing/project lifecycle with a higher attendant cost.

The AQM uses a formal weighting methodology to support tools relevant to a specific phase of testing e.g. where there is a requirement to do significant performance, load or volume testing, the weighting will favour the use of tools.

The AQM does not explicitly favour any test tools vendor but defines a number of criteria by which vendors may be judged. At a high level these criteria include:

- Licence costs
- Training costs
- Maintenance Costs
- Cost of skilled tool personnel
- Availability of skilled tool personnel
- Efficacy of tool (does it meet the requirements of the AQM).

Access Computer Consulting plc Plc is able to provide skilled resource in a number of vendor's tool kits. These vendors include Mercury, Compuware and Rational.

### 3. Appendices

**<<This page is intentionally blank>>**

## Examples of Document Standards Adopted By the AQM

# IEEE Standard for Software Test Documentation

(ANSI/IEEE Standard 829-1983)

This is a summary of the ANSI/IEEE Standard 829-1983. It describes a test plan as:

“A document describing the scope, approach, resources, and schedule of intended testing activities. It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning.”

This standard specifies the following test plan outline:

### 3.1 Test Plan Identifier:

- A unique identifier

### 3.2 Introduction

- Summary of the items and features to be tested
- Need for and history of each item (optional)
- References to related documents such as project authorization, project plan, QA plan, configuration management plan, relevant policies, relevant standards
- References to lower level test plans

### 3.3 Test Items

- Test items and their version
- Characteristics of their transmittal media
- References to related documents such as requirements specification, design specification, users guide, operations guide, installation guide
- References to bug reports related to test items
- Items which are specifically not going to be tested (optional)

### 3.4 Features to be Tested

- All software features and combinations of features to be tested
- References to test-design specifications associated with each feature and combination of features

### 3.5 Features Not to Be Tested

- All features and significant combinations of features which will not be tested
- The reasons these features won't be tested

### 3.6 Approach

- Overall approach to testing
- For each major group of features or combinations of features, specify the approach
- Specify major activities, techniques, and tools which are to be used to test the groups
- Specify a minimum degree of comprehensiveness required
- Identify which techniques will be used to judge comprehensiveness
- Specify any additional completion criteria
- Specify techniques which are to be used to trace requirements
- Identify significant constraints on testing, such as test-item availability, testing-resource availability, and deadline

### 3.7 Item Pass/Fail Criteria

- Specify the criteria to be used to determine whether each test item has passed or failed testing

### 3.8 Suspension Criteria and Resumption Requirements

- Specify criteria to be used to suspend the testing activity
- Specify testing activities which must be redone when testing is resumed

### 3.9 Test Deliverables

- Identify the deliverable documents: test plan, test design specifications, test case specifications, test procedure specifications, test item transmittal reports, test logs, test incident reports, test summary reports
- Identify test input and output data
- Identify test tools (optional)

### 3.10 Testing Tasks

- Identify tasks necessary to prepare for and perform testing
- Identify all task interdependencies
- Identify any special skills required

### 3.11 Environmental Needs

- Specify necessary and desired properties of the test environment: physical characteristics of the facilities including hardware, communications and system software, the mode of usage (i.e., stand-alone), and any other software or supplies needed
- Specify the level of security required
- Identify special test tools needed
- Identify any other testing needs
- Identify the source for all needs which are not currently available

### 3.12 Responsibilities

- Identify groups responsible for managing, designing, preparing, executing, witnessing, checking and resolving
- Identify groups responsible for providing the test items identified in the Test Items section
- Identify groups responsible for providing the environmental needs identified in the Environmental Needs section

### 3.13 Staffing and Training Needs

- Specify staffing needs by skill level
- Identify training options for providing necessary skills

### 3.14 Schedule

- Specify test milestones
- Specify all item transmittal events
- Estimate time required to do each testing task
- Schedule all testing tasks and test milestones
- For each testing resource, specify its periods of use

### 3.15 Risks and Contingencies

- Identify the high-risk assumptions of the test plan
- Specify contingency plans for each

### 3.16 Approvals

- Specify the names and titles of all persons who must approve the plan
- Provide space for signatures and dates