Applewood Computers

TEST STRATEGY FOR

ACAS

(Printed 18/01/25 21:25)
(Version 1.02)

Amendments History Sheet

Version	Date	Section Ref.	Page No.	Type of Change *	Detail
1.0	June 1987.				Original Document
1.01	April 1988				Minor grammar and typo's for ACAS v2
					and systems used for Linux.
1.02	April 2024				Font, size chg only

*** I-Insertion A-Amendment D-Deletion**

Distribution List

Recipient(s)	Department/Area of Responsibility
	Change Management -

Business Sign-Off

Business Representative	Responsibility	Signature	Date

CONTENTS

1. INTRODUCTION....5
1.1 OBJECTIVES....5
2. REFERENCES....6
3. TEST STEPS....7
3.1 UNIT TESTING....7
3.1.1 Scope and Purpose....7
3.1.2 Procedures....7
3.1.3 Who performs Unit Testing....8
3.2 LINK TESTING....10
3.2.1 Scope and Purpose....10
3.2.2 Procedures....10
3.2.3 Who performs Link Testing....11
3.3 SYSTEM AND INTEGRATION TESTING....12
3.3.1 Scope and Purpose....12
3.3.2 Procedures....13
3.3.3 Who performs System Testing....13
3.4 USER ACCEPTANCE TESTING (UAT)....14
3.4.1 Scope and Purpose....14
3.4.2 Procedures....14
3.5 VOLUMETRICS....15
3.5.1 Scope and Purpose....15
3.5.2 Procedures....15
4. TEST TOOLS....16
4.1 Unit Testing....16
4.2 LINK TESTING....16
4.3 SYSTEM AND INTEGRATION TESTING....16
5. ENTRY CRITERIA....17
5.1 UNIT TESTING....17
5.2 LINK TESTING....17
5.3 SYSTEM & INTEGRATION TESTING....17
5.4 USER ACCEPTANCE TESTING....18
6. TEST PROCEDURES....18
6.1 DSS1....18
6.1.1 ICL Based....18
6.1.2 Unix MySQL Server....19
6.2 ACAS SYSTEM PHASE 1 - TEST METHOD BASICS....20
6.2.1 GUI Front End....20
6.2.2 EOFS/ATM....20
6.2.3 Application Server....20
6.2.4 MySQL Server....20
6.2.5 Tools....20
7. HUMAN RESOURCES....21
7.1 UNIT TESTING....21

7.2 LINK TESTING....21
7.3 SYSTEM & INTEGRATION TESTING....21
7.4 USER ACCEPTANCE TESTING....21
7.5 DATA BASE CONVERSION AND TRANSFER....21
8. EXIT CRITERIA....22
8.1 UNIT TESTING....22
8.2 LINK TESTING....22
8.3 SYSTEM & INTEGRATION TESTING....22
8.4 USER ACCEPTANCE TESTING....22
9. APPENDIX A - PROJECT TEST COMPLETION FORM....23

1. INTRODUCTION

1.1 Objectives

The objectives of a Test strategy is to set out in a uniform manner the method and procedures to be adopted for the various testing processes to be used for the ACAS Project. It will be broken down by the various test steps needed, with details of resources required.

This is a live document and is updated as information regarding system functionality becomes available. There will be a cut-off point at which time the final draft will be created and acted upon.

2. REFERENCES

Document Date	Title	Author
06/1987	System Test Standards	Vincent. B. Coen
04/1988	ACAS Functional Specifications	Vincent. B. Coen

3. TEST STEPS

It is proposed that a formal test procedure be used within which, the various test building blocks are maintained. These blocks are broken down into Unit, Link, System, User Acceptance, Volumetrics & Performance.

3.1 Unit Testing

3.1.1 Scope and Purpose.

The developers are responsible for unit testing and this is defined as ensuring that:

1. Business logic for both positive and negative conditions conforming to the Functional Specifications.
2. Accessing Database tables is processed in the correct and most efficient manner, i.e., all SQL statements have been checked with the use of 'Explain' where applicable.
3. All data entered is stored correctly, in the correct dB table or file and in the right format;
4. All data required to be displayed is retrieved correctly from the right file or dB table, and in the right format;
5. All fields on all screens within the module conform to the format and validation rules as defined in the Functional Specification;
6. Navigation into, and out of, screens within a module conforms to the Functional Specification definitions;
7. Validation error messages are correct, meaningful, and appropriate to the error condition.

Unit Testing needs to test all functionality that can be checked without integrating the module, including database or file processing. Therefore, Unit Testing is limited to the module itself. There is no requirement, at this stage, to go beyond the confines of the module. However it is important to remember that a GUI module may consist of more than one form or screen, and any interaction between these forms or screens must be fully tested as well as all business logic.

3.1.2 Procedures.

Test Plans for Unit Testing should have been created following the development of the Functional Specification.

The Team Leader is responsible for giving the programmer a generic Module Unit Test Check list. The format of the Check list is not important, however the content is very important in that it must show all processing requirements. Additional check lists must be included to cover all business logic that the module is charged with. For this purpose the Functional Specification must be used to construct the check list and **not** the code. The

use of the word ‘program’ in this document should also mean module, package, sub-package or run unit where required.
Unit Testing will be conducted by the programmer on completion of the module, and prior to the release of the module to Link Testing.

As it is the originating programmer performing the Unit Testing, it should not be necessary to raise Fault Reports for any problems identified - these can be fixed immediately, and re-tested.

It is **not** recommended that Unit Testing is conducted after being released from the control of the programmer. It is the programmer’s responsibility to ensure that Unit Testing is done as faults found at this stage take less time and therefore have the lowest cost to resolve. It is recognised, however, that this is not always possible. When this happens, it will be mandatory to raise Fault Reports for any problems identified - refer to the Fault Reporting Procedure for more information.

It is not acceptable for a programmer to release a module without some **basic** structured testing having been done. The module Unit Test Check list should represent the **minimum** level of testing that must be performed.

As each element in the Check list is tested against the module, the appropriate Tick Box should be completed with one of the following responses:

- A 4, which denotes that the test has been performed, and passed;
- A 6, which denotes that the test has been performed, and failed. The programmer must then resolve the problem and re-test. A further 4 inside the Tick Box will then indicate that the problem has been resolved;
- The comment ‘N/A’ which indicates that the corresponding test is not applicable in this instance (for example the Radio Button tests for a program that does not contain any Radio Buttons).

When all Unit Testing has been completed (indicated by a Check list that now only contains ‘N/A’ or 4), the Check list should be signed, and dated, by the programmer. The program may now be released to the next stage in the Testing Cycle. The Program Unit Test Check list should be handed to the Team Leader to confirm completion, and it will then be filed in the Project Folder.
The Project Test Completion Form (Unit Test Section) must be signed and dated by the Team Leader, and filed in the Project Folder.

The Project Test Completion Form (Unit Test Section) must be signed and dated by the Team Leader, and filed in the Project Folder (see Appendix A)

3.1.3 Who performs Unit Testing.

Unit Testing should be performed by the programmer responsible for the module. Using a Unit Test Check list, the programmer should run through the appropriate tests relevant and test his or her own work. This should be done prior to releasing the module for any further

testing. It must be emphasised that the programmer has a **responsibility** for generating **quality** programs, and therefore must take on some of the testing effort. It is **not** acceptable for a program to go to the next stage of testing without this exercise being performed.

The results of the Unit Test must be recorded, and filed.

Optionally, the Unit Test may be performed by any person assigned to the task by the Project Manager, and this person may be anybody on the project, or any other project, although this is to be avoided wherever possible. As the Unit Test is a significant check on module functionality, it does require a familiarity with the system being developed.

3.2 Link Testing

3.2.1 Scope and Purpose.

The developers are expected to perform link testing prior to passing the system to the independent test team and this is defined as ensuring that:

1. Information entered, and therefore stored, in one module is successfully retrieved and manipulated by another module, in the correct format;

2. Navigation between modules is correct, both forward and backward;

3. Ticking and greying/un-greying of Menu Options where applicable is performed correctly.

Link Testing is the process of ensuring that different modules communicate with each other, and pass data between each other correctly, via the system and the database. Data entered in one module will inevitably be required by another, and it is this interaction between these modules that must be fully tested as well as all functional logic.

3.2.2 Procedures.

The Test Plans for the Link Testing stage should have been created following the development of the Functional Specification.

Formal Link Testing will be conducted by the developers or the Test Team allocated to the Project. The Test Team should consist of a group of independent individuals, dissociated from the development team. It is accepted, however, that this may not always be the case. The important aspect to ensure in this, and all ACAS stages of Testing, is that the originating programmer should not be involved. If the programmer is involved in testing his/her own programs, then assumptions will be made with regard to the functionality. This is not the purpose of Testing.

As Link Testing proceeds, any problems identified **must** be recorded on a Fault Report Form. Refer to the Fault Reporting Procedure for more information. Note that only **one** Fault is recorded per Fault Report Form. The reasons for this are:

- It is easier to respond to one problem at a time;

- It is easier to analyse, statistically, the volume of problems reported on a system;

- Not all reported Faults are program Faults. Some may be caused by the Operating System, Programming Language or Development Tool, Operator error, misinterpretation of the specification, or a fault within the test specification. In cases such as these, it may be

possible to reject a reported Fault, but not if a genuine program bug is recorded on the same Fault Report form.

As each Fault Report is completed, it should be recorded on to the Fault Reporting System.

When Link Testing has been successfully completed, the Project Test Completion Form (Link Testing Section) must be signed, and dated, by the Team Leader of the development team or the Test Team. The module may now be released to the next stage in the Testing Cycle. The Link Test authorisation should be handed to the Test Manager to confirm completion, and it will then be filed in the Project Folder.

3.2.3 Who performs Link Testing.

Link Testing **must** be performed by a member of the project team who is familiar with the integration requirements of the modules being tested. Link Testing is best performed by the programmer(s) responsible for the modules being integrated. The project manager should ensure that the relevant resources are allocated to each Link Test.

3.3 System and Integration Testing

3.3.1 Scope and Purpose.

The developers having completed unit and link testing will transfer the system together with the results of their testing (test results, expected and actual) to the independent test team.

The purpose of System and Integration Testing is to ensure that:

1. Navigation between module is correct, both forward and backward;

2. Ticking and greying/un-greying of Menu Options where applicable is performed correctly.

3. Information entered, and therefore stored, in one module is successfully retrieved and manipulated by another module, in the correct format;

4. Information entered is processed correctly by the application servers and correct results obtained.

5. Reports produced by system functions are correct and as expected.

6. Business scenarios entered through the system produce the expected results.

System Testing is the process of ensuring that different modules communicate with each other, and pass data between each other correctly, via the system and the database. Data entered in one module will inevitably be required by another, and it is this interaction between these modules that must be fully tested.

This is very similar in practice to Link Testing, but the purpose of System Testing is to ensure that the entire system works as a cohesive unit. For this reason, it is necessary to test the whole system from end to end (start to finish) with a known set of data, requiring that the MySQL database is set up with specific data that can be used against known transaction data. The same must be applied to all databases that is being supported.

System Testing is performed with a number of Case Studies supplied by the users. These will have been developed and provided before System Testing commences, and will consist of working examples of how the system should operate, with input data, and output expected results. These Case Studies therefore represent the Test Plans for this stage of testing. Additional data will be created by the test team to validate negative as well as positive logic, designed from master test cases.

3.3.2 Procedures.

The Project Manager is responsible for ensuring that the Client provides Case Studies for System Testing, but it is the responsibility of the Client to ensure that they:

- Are accurate;
- Are correct;
- Fully test most, if not all, aspects of the system being developed.

On receipt of the Case Studies, they should be checked to ensure that they are accurate. This is important in order to avoid any confusion that may arise when identifying differences between the Case Study and the System. Often, Faults are reported because the system produces a different result to that of the Case Study, and it is later discovered that the Case Study was wrong. Unfortunately, however, a lot of time has been spent in proving this. Time spent ‘up front’ in verifying the Case Studies will prevent many of these problems occurring.

The Project Manager should define how much verification should be performed, how detailed a check should be made, and which resources are allocated.

The Test Team is responsible for processing the Case Studies, and the Test Manager should allocate resources as appropriate to the task.

As System Testing proceeds, any problems identified **must** be recorded on a Fault Report Form. Refer to the Fault Reporting Procedure for more information. Note that only one Fault is recorded per Fault Report Form. The reasons for this are as stated in Link Testing.

As each Fault Report is completed, it should be recorded on to the Fault Reporting System.

When System Testing has been successfully completed, the Project Test Completion Form (System Testing Section) must be signed, and dated, by the Team Leader of the development team or the Test Team. The modules may now be released to the next stage in the Testing Cycle. The System Test authorisation should be handed to the Test Manager to confirm completion, and it will then be filed in the Project Folder.

3.3.3 Who performs System Testing.

As System Testing is concerned with the overall functionality of the system, the person(s) best suited to performing this role must have a clear and unambiguous understanding of what the system is trying to achieve. As Case Studies should have been provided by the Client for this phase of the test cycle, it is important to involve the User at this time.

Therefore, System Testing may be conducted by many people, including Users, programmers, team leader, or/and a test team specifically created for the project.

3.4 User Acceptance Testing (UAT)

3.4.1 Scope and Purpose.

This section of the test cycle will be performed by company users with some assistance of the test team to ensure that resources are used to their best effect. All user departments will have created their own Business profiles from which Business cases will have been created.

It is this process that verifies that the supplied system fulfils the user IS requirements for that specific release of the software.

3.4.2 Procedures.

User Testing may be conducted either at the Client development site, or at another location. It may also be performed either by Client Personnel alone, or with the development team and/or the Test Team in attendance. It is recommended that the development team maintain a presence during User Testing, and it is the responsibility of the Project Manager to organise this.

The procedure for processing reported Faults remains the same as in the earlier test steps. It is the responsibility of the Test Manager to ensure that, when User Testing is conducted, the Client personnel are aware of, and conform to, the Fault Reporting Procedure.

As User Testing proceeds, any problems identified must be recorded on a Fault Report Form. Refer to the Fault Reporting Procedure for more information. Note that only one Fault is recorded per Fault Report Form.

As each Fault Report is completed, it should be recorded on to the Fault Reporting System.

When User Testing has been successfully completed, the Project Test Completion Form (User Testing Section) must be signed, and dated, by the person from the Client organisation responsible for User Testing (see Appendix A). The User Test authorisation should be handed to the Test Manager to confirm completion, and it will then be filed in the Project Folder.

3.5 Volumetrics.

3.5.1 Scope and Purpose.

The purpose of Volume Testing is to ensure that the system works correctly:

- With volumes of data comparable to those to be expected in a ‘live’ environment;
- On a hardware configuration **identical** to that planned for actual use.

Volumetric Testing addresses any outstanding issues that may have been specified in the Business Requirements Specification, such as usability, response times, data migration, and upload/download of data, as well as confirming the results of previous test stages.

3.5.2 Procedures.

Volume Testing must be conducted at the Client production site. It may also be performed either by Client Personnel, or by Development and Test Team staff.

The procedure for processing reported Faults remains the same. It is the responsibility of the Project Manager to ensure that, when Volume Testing is conducted at the Client site or by Client personnel, they are aware of, and conform to, the Fault Reporting Procedure.

As Volume Testing proceeds, any problems identified **must** be recorded on a Fault Report Form. Refer to the Fault Reporting Procedure for more information. Note that only one Fault is recorded per Fault Report Form. The reasons for this are as stated in Link Testing.

As each Fault Report is completed, it should be recorded on to the Fault Reporting System.

When Volume Testing has been successfully completed, the Project Test Completion Form (Volume Testing Section) must be signed, and dated, by the person responsible for Volume Testing. As stated before, this may be a representative from the Client organisation or the Test Team Leader. If Volume Testing is undertaken by a Client representative, it is the responsibility of the Project Manager to ensure that the correct procedure is followed. The program may now be released to the next stage in the cycle which would be the handover to the operations department. The Volume Test authorisation should be handed to the Test Manager to confirm completion, and it will then be filed in the Project Folder.

It is understood that the Client in the first instance is the company.

4. Test tools

4.1 Unit Testing

It is expected that the development environment as supplied with Cobol is all that is required to complete unit testing on modules produced by Cobol.

For MySQL development, the normal tools as well as a good GUI to MySQL database tool should suffice. Over the next few weeks we hope to examine various products that may fit the requirements.

4.2 Link Testing

It is expected that the tools used for unit testing will be the same for link testing.

4.3 System and Integration Testing

In addition to the GUI/MySQL DB tool listed above, the following is required:

A fault report and management tool such as SQA Manager or Tplan for controlling fault reports, passing through Email with MS Mail/Exchange. The Microsoft products are already present on site but SQA Manager would need to be purchased with an estimated price range of £2000, subject to license requirements only one is needed. In addition, if it is required to simulate multi-users entering data a suitable product may well be SQA LoadTest for which no pricing information is available at time of writing but a guesstimate would be £10,000+ per 25 user license. This would also require the use of a similar number of workstations. It may well be possible to rent this product.

For the data transfer between the m/f and the new *nix based system, standard current products will be used such as AlterData, Query Master and a GUI MySQL DB query tool as indicated in the Unit Testing test tools.

Other products can be used as available for the Linux platforms.

5. ENTRY CRITERIA

5.1 Unit Testing

There is no requirements for Entry other than the production of a specific module or a modification to same at any stage after initial development and test.

Data requirements: Subject to individual needs of each module to be tested by the programmer.

5.2 Link Testing

Completed Unit Testing of a group of modules that form a continuous executable thread that can be Link Tested. The Project Test Completion Form duly filed in for each module under the Unit Test Section. Supporting documentation such as a Unit Test Check list for each module completed with no outstanding Fault reports, Unit Test expected and actual test results.

Data requirements: As required by individual streams but it would be a recommendation to use standing data developed for System Testing.

5.3 System & Integration Testing

Completed Link testing of all modules that form the system. The Project Test Completion Form duly filed in for each module under the Unit and Link Test Section. Supporting documentation such as a Unit Test Check list for each module completed with no outstanding Fault reports, Unit Test expected and actual test results. The Link Test results for all Link Test elements within the Link Test Section.
No outstanding Fault reports other than cosmetic and under review.

Data requirements: Standing data as well as data transferred from the ICL and IBM mainframe systems under DSS (ICL) such as Hot cards, accounts lists etc. This will be expanded when more information via the Functional Specification becomes available.

5.4 User Acceptance Testing

Completed System Testing with the Project Test Completion Form filed in for all elements of the system as under the System Test Completion section No outstanding fault reports other than cosmetic and under client reviews.

Data requirements: Subject to the needs of the UAT team but would expect that standing data used in System & Integration testing is also used . Liaison between test teams will be required to validate further requirements.

6. TEST PROCEDURES

This section outlines the methods and procedures to be used for each system element and will be expanded as more information becomes available.

6.1 DSS1

6.1.1 ICL Based

Initial Load: The mainframe will run an extract procedure against IDMS tables creating a file for each IDMS table containing all records present. The records will consist of an exact mirror image of the original plus additional key information taken from the owning record. These files will then be transferred to the Unix platform using a FTP type protocol.

Updates: The procedure is basically the same as initial load except that only new or changed records are extracted.

Test Method:

1. Tracking record counts for all data types and comparing against the import procedure running on the Unix box.

2. Using Query Master to verify data mapping from the IDMS records and the extract file against the record data layouts.

3. Using standard MySQL tools for Unix to examine first & last few records comparing against the ICL.

6.1.2 Unix MySQL Server

Initial Load: The files having arrived from the ICL are placed into a staging area where they are transformed and loaded into the cleansing area. A remapping of various record types then takes place creating new table types for the OLTP and the DSS system. Other than the receipt of files into the Unix system (staging area) the imported records are stored in the MySQL database After the remapping some (new) tables are sent back to the ICL mainframe.

Updates: Processing seems to be the same as for the initial load.

Test Method:

As all imported records are stored within various MySQL tables it is proposed to use a GUI database query tool to verify correct operation at each stage. Subject to the capability of the GUI DB query tool, a remap query will be compared to the created data.

Note *Where ever possible a GUI tool required for the Business will be used for testing to help reduce testing budgets subject to capability.*

6.2 ACAS system Phase 1 - Test Method Basics

The details described below are a light treatment for this subject. As more information becomes available from detailed design or functional specifications this area will be expanded.

6.2.1 GUI Front End.

All screens will have data entered against all variants. Checks will be made to confirm that data is transferred to 6.2.3 where relevant.

6.2.2 EOFS/ATM.

Data to be created and stored the same format as the ATM front-end data ‘collator’, so that sample transactions can be actioned through the application server. Further details about specifics will be added as more information becomes available.

6.2.3 Application Server

Verification that the Authorisation and Risk systems processes each type of transaction in a predetermined manner and compared against expected results. Further details about specifics will be added as more information becomes available.

6.2.4 MySQL Server.

That all transactions created by the application servers are stored correctly and compared against expected results for new and changed table entries. Further details about specifics will be added as more information becomes available.

6.2.5 Tools.

No special tools other than that defined for Unit & Link Testing.

7. HUMAN RESOURCES

7.1 Unit Testing

Individual developers as allocated to the project.

7.2 Link Testing

Individual developers as allocated to the project.
Light independent test team support.

7.3 System & Integration Testing

Full Test Team consisting two testers with knowledge of Windows (light) and MySQL and it's supporting tools. User representatives to assist in creation and conversion of business cases to test scripts etc., for all effected departments. Some support from developers will be expected during this phase.

7.4 User Acceptance Testing

As per UAT document, with support of Test Team as in 7.3. Some support from developers will be expected during this phase.

7.5 Data Base Conversion and Transfer

Test team with (one to two). Support from development team both ICL, IBM and MySQL. Some assistance from users may also be required.

8. EXIT CRITERIA

8.1 Unit Testing

Module source code (or equivalent) Unit test plan and evidence of completion of same with expected and actual test results.

8.2 Link Testing

Module source code (or equivalent) Unit test plan and evidence of completion of same with expected and actual test results.

8.3 System & Integration Testing

8.4 User Acceptance Testing

9. APPENDIX A - PROJECT TEST COMPLETION FORM.

Stage	Authorisation	Date
Unit Test		
Link Test		
System Test		
Volume Test		
User Test		

This form must be filed in the Project Folder.