1. DOCUMENT CHANGE HISTORY

Version Number	Date	Description
1.0	August 1, 2005	Initial Draft
1.1	September 16, 2005	Added Requirements Traceability Matrix
1.2	September 26, 2005	Added output of usability testing to be report of requested features and suggestions as requested by Dartmouth.. Removed csv as an input format as requested by Dartmouth. Added comparison of raw to reduced SELDI-TOF dataset as an initial test for Q5
1.3	January 2006	Final Version

Q5

Test Plan

Version 1.3

FINAL

TABLE OF CONTENTS

2. INTRODUCTION

This Test Plan prescribes the scope, approach, resources, and schedule of the testing activities. It identifies the items being tested, features to be tested, testing tasks to be performed, personnel responsible for each task, and risks associated with this plan.

2.1 SCOPE

This document provides instruction and strategy for incorporating Software Testing practices and procedures into the Q5 project. This document demonstrates the application of testing on this project and provides guidelines for implementing procedures supporting this function.

The current Q5 application is an implementation of a probabilistic classification algorithm with demonstrated utility in classifying expression-dependent proteomic data from mass spectrometry of human serum. The Q5 project aims to extend the current implementation in the following ways i) implement the algorithm in an open source software environment (using the R statistical package and ii) ensure the implementation is compatible with caBIG. It was also agreed by the developer and adopter site that Q5 should be tested on higher resolution data (MALDI).

2.1.1 Identification

Q5 package implemented in R

2.1.2 Document Overview

This Test Plan defines the strategies necessary to accomplish the testing activities associated with Q5. Testing procedural instructions are included in the Standard Operating Procedures (SOPs).

The remaining Test Plan sections are organized as follows:

- **Section 2: Strategy**: Describes the overall approach and techniques to be used during testing.
- **Section 3. Software Test Environment**: Describes the intended testing environment.
- **Section 4. Test Identification**: Identifies and describes each of the tests.
- **Section 5. Test Schedules**: Contains or references the schedules for conducting testing.
- **Section 6. Requirements Traceability:** Identifies traceability from this Test Plan to the Requirements.
- **Section 7. Risks**: Identifies the risks associated with the Test Plan.
- **Section 8. Notes:** Contains general information that aids in the understanding of this document.

2.2 RESOURCES

Shannon McWeeney: PI and Director of the Informatics Shared Resource at the Oregon health and Science University Q5 adopter site

Ted Laderas: Test Manager

Ranjani Ramakrishnan: Software/statistical algorithm tester

Cory Bystrom: Proteomics SME at Oregon Health and Science University

Paul Courtney: Manager of Development at Dartmouth

2.3 REFERENCED DOCUMENTS

For additional project specific information, refer to the following documents:

- Q5 Vision and Scope Document (http://cabigcvs.nci.nih.gov/viewcvs/viewcvs.cgi/qfive/caBIG_Q5_Vision_Scope_Draft.doc)
- Q5 requirements and Specifications document (http://gforge.nci.nih.gov/docman/view.php/42/420/caBIG_Requirements_and_Specification_Q5Project.doc)
- Q5 Use Case Documents (http://cabigcvs.nci.nih.gov/viewcvs/viewcvs.cgi/qfive/)
- Q5 API Document (http://gforge.nci.nih.gov/docman/view.php/42/421/q5_Api.rtf)
- Meeting Notes August 1, 2005

3. SOFTWARE TEST STRATEGY

3.1 OBJECTIVES

We propose to carry out white box (structural testing), validating the implementation and limited performance and user acceptance testing.

3.2 APPROACH

White-Box (Component) Testing of Q5 components

We will test the components of Q5 that have new implementations (such as the PCA component) of the Q5 algorithm on well-characterized datasets (such as Fisher's Iris dataset). This will allow determination that the individual algorithms are correctly implemented.

Testing to Confirm Correctness of R Port

We will compare the output of the individual components on the synthetic dataset (described below) to ensure that the port to R was successful by comparing it with the output for these components with the original implementation (Matlab). Note that the comparison will not be exact as the test/train splits calculated in the original Matlab Q5 algorithm are not user-accessible, so the averaged statistics generated by each package (Matlab and R versions) for multiple test/train splits will be compared.

mzXML Testing

We will confirm that the mzXML functionality provided for Q5 correctly interprets the synthetic data in mzXML format as described in the API document. This will be tested in two ways: the first is to by simple examination of the scan length and number of scans as output by the mzXML functionality. The second is a comparison of the Q5 statistics of identical test/train splits run on both the synthetic-data interpolated by R-Proteomics to a run with the data as loaded by the mzXML functionality.

Validation Testing

We will validate the output (i.e. spectral classification and cross-validation statistics) of the Q5 Algorithm on two biological datasets. Validation will be performed on higher resolution MALDI data as agreed by the developer and adopter.

Synthetic Dataset – This is a MALDI dataset that has been generated in-house by OHSU. It consists of four groups of 50 samples each. One purified peptide of known mass and three complex protein mixtures were used to generate the samples in this dataset. The dataset has been designed to test Q5's discriminatory power in two relevant possible clinical situations in which the algorithm might be applied.

The first situation to examine is whether the Q5 algorithm can distinguish between two groups that are different by effect size (same proteins but different concentrations of those proteins). In the table below, the effect size issue would be examined using groups 3 and 4.

The second situation is two groups that are distinguished by the presence or absence of particular proteins. One example of which this situation would occur would be the occurrence of distinct biomarkers in a cancerous population which are absent in a normal population.

Group	P1	C1	C2	C3	Sample Size
1	Absent	Absent	0.5x	1x	49
2	1x	Absent	1x	1x	50
3	1x	1x	2x	1x	50
4	1x	2x	4x	1x	51

Legend: P=Pure, C=Complex

Performance Testing

We will measure CPU usage, memory usage, and execution time for the synthetic dataset and report these results on Windows, Solaris, and Linux machines. We will also log any errors or warning messages output by R upon installation and testing of the Q5 package.

Usability Testing

We will perform usability testing of Q5 with a number of users recruited from the OHSU Cancer Institute and report their general impressions and suggestions for making Q5 useful.

3.3 DESCRIPTION OF FUNCTIONALITY

Q5 is a probabilistic classification algorithm implemented using R. It accepts as input SELDI TOF and high resolution MALDI TOF data partitioned into training and testing sets (if the dimensions of the data are reduced sufficiently) and outputs a discriminant and related statistics on the testing set. For more information, please refer to the Q5 Software Requirements and Specifications document available at: http://gforge.nci.nih.gov/docman/view.php/42/420/caBIG_Requirements_and_Specification_Q5Project.doc .

3.4 SPECIFIC EXCLUSIONS

Integration testing for the grid – This is not specified by the developer in the Q5 Requirements and Specifications document.

Functional testing – The adopters are not completely blind to the Q5 code.

Unit testing – This will be performed by the developers.

3.5 DEPENDENCIES & ASSUMPTIONS

Delivery of software by the developers.

3.6 GENERAL CRITERIA FOR SUCCESS

Correctness of implementation validated.

All tests have been executed and the outputs documented, resolved, verified, or designated for future releases.

Report for users on speed and resource utilization for different platforms.

3.6.1 Readiness Criteria

Will be communicated by the developers.

3.6.2 Pass/Fail Criteria

API testing: OHSU will confirm that the interfaces to the functions (i.e. inputs, and outputs) conform to those described in the Q5 API document. A test is considered failed if the inputs and outputs do not conform to this document.

Interface Testing: Interface testing is tied into the API testing as described above. Additionally, a pass of the installation tests on each operating system is dependent on successful deployment and utilization of the Q5 package on each operating system.

Performance Testing: Completion of the test scripts and reporting of processor time and resource utilization.

Validation of the R implementation: The R version will be considered successful if the averaged statistics are within a pre-established error threshold for different test/train splits. A test is considered failed if the statistics do not meet this error threshold.

mzXML testing: The tests will be considered successful if the original data is preserved through the translation. Deviations in performance of mzXML loaded data compared to interpolated data will be considered a failure.

3.6.3 Completion Criteria

The criteria for completion of the testing procedures is that the system produces the output (described in the Requirements and Specifications document) within expected performance requirements. Testing is considered completed when:

- The assigned test scripts have been executed.
- Defects and discrepancies are documented, resolved, verified, or designated as future changes.

3.6.4 Acceptance Criteria

4. SOFTWARE TEST ENVIRONMENT

This section describes the software test environment at each intended test site.

4.1 OREGON HEALTH AND SCIENCE UNIVERSITY

4.1.1 Software Items

R 2.2.0 (installed on all machines described below)
Microsoft Windows XP
Mac OS X 10.4.4
Red Hat Linux 2.4.2
Solaris

4.1.2 Hardware and Firmware Items

Solaris Server Configuration (Murdock 1)
Pentium^R 4 3.6 GHz, 1 GB RAM
Xeon 3.6 GHz, 6 GB RAM
Power PC 1.2 GHz, 768 MB RAM

4.1.3 Other Materials

The other major components are described below.

Generation of Synthetic Dataset

The data consists of two replicates.

Replicate 1 consists of two plates that were ionized by MALDI that contained 100 samples apiece.

The 200 samples consisting of group 1, group 2, group 3, and group 4 were unpacked from two wiff files that each contained 100 scans corresponding to these plates using the ABI Analysis QS software package. These individual wiff files were converted to mzXML using the mzStar conversion package from ISB. The individual files were then sorted into the four groups described in randomization table and design.xls file.

Note that group 1 contains 49 samples and group 4 contains 51 samples. Both groups 2 and 3 contain 50 samples apiece.

Replicate 2 consists of the exact same groups, but with different plates. There are two targets available on the spectrometer; in replicate 2, the plates are swapped to minimize variation due to differences between targets.

Randomization Design and Replicates of Synthetic Data

Plate 1

G3	G1	G3	G1	G4	G4	G2	G4	G3	G1
G1	G3	G2	G1	G2	G2	G3	G4	G1	G2
G2	G4	G3	G1	G4	G1	G2	G4	G1	G3
G4	G2	G1	G2	G4	G2	G3	G2	G3	G3
G1	G1	G3	G1	G2	G1	G2	G4	G1	G1
G2	G4	G1	G1	G4	G3	G1	G4	G2	G2
G2	G1	G4	G3	G4	G3	G3	G4	G2	G2
G1	G1	G1	G1	G2	G2	G4	G4	G2	G4
G2	G1	G3	G2	G2	G1	G3	G1	G3	G3
G3	G4	G4	G4	G3	G3	G2	G4	G2	G1

Plate 2

G1	G3	G2	G4	G4	G3	G4	G2	G2	G4
G2	G3	G2	G1	G3	G1	G2	G1	G2	G4
G4	G2	G3	G3	G1	G3	G2	G1	G4	G3
G3	G4	G4	G4	G2	G3	G3	G4	G3	G3
G1	G2	G2	G4	G4	G2	G3	G2	G1	G1
G1	G3	G4	G2	G1	G1	G3	G4	G1	G3
G4	G3	G1	G3	G3	G2	G2	G2	G4	G1
G4	G3	G4	G4	G1	G4	G1	G2	G3	G3
G3	G1	G2	G1	G1	G4	G2	G4	G2	G1
G3	G3	G3	G2	G4	G1	G4	G3	G4	G4

4.1.4 Participating Organizations

The testing group consists of the project's Test Manager (TM), and the Tester(s). The groups listed below are responsible for the respective types of testing:

Tester	**Responsibility**	**Qualification**
Shannon McWeeney	**Oversight**	**PI and Project manager**
Ted Laderas	**Testing**	**Test Manager**
Ranjani Ramakrishnan	**Testing**	**Tester**

5. TEST SCHEDULES

5.1 TIME FRAMES FOR TESTING

The Test Manager will coordinate with the Project Manager and add the planned testing activities to the master project schedule. Refer to the project SDP and schedule for additional information.

Phase 1: January 3 – Feb 15, 2006

Validation testing of R Q5

mzXML compatibility testing of Q5

Phase 2: February 15, 2006 – March 3, 2006

API/User Interface testing

Phase 3: March 6 – March 15, 2006

Training and User Acceptance Testing

6. REQUIREMENTS TRACEABILITY MATRIX

This requirements traceability matrix is current to the last Requirements document delivered to OHSU as of December 20, 2005. Please refer to Appendix B for complete descriptions of the tests to be conducted.

Test ID	Test Type	Test Description	Requirements Tested	Phase
R1	Validation of R-Port	Compare outputs of Matlab version of Q5 to R version of Q5 algorithm under similar conditions using synthetic data.	2.1, 2.2, 2.3, 2.4, 3.1	1
R2	Validation of R-Port	Compare effectiveness of Q5 algorithm to differentiate between two groups of different effect sizes of synthetic data.	2.1, 2.2, 2.3, 2.4, 3.1	1
R3	Validation of R-Port	Compare effectiveness of Q5 algorithm to differentiate between two groups of that contain different sets of protein mixtures.	2.1, 2.2, 2.3, 2.4, 3.1	1
M1	mzXML Compatibility Testing	Test mzXML compatibility of Q5 by comparing range of scan size wi	1.2, 1.4, 1.5	1
M2	mzXML Compatibility Testing	Test mzXML compatibility of Q5 by comparing number of scans in n	1.2, 1.4, 1.5	1
M3	mzXML Compatibility Testing	Compare output of Q5 algorithm using files loaded with Q5 mzXML() function to output of Q5 algorithm using RProteomics interpolated data.	1.2, 1.4, 1.5	1
I1	User Interface/API Testing	Compare output of Q5-implemented PCA algorithm to other implementations in R	2.2	2
I2	User Interface/API Testing	Test error conditions of Q5 algorithm Report how system responds when caMass not loaded Provide the algorithm with unbalanced groups for test and training sets Pass test class vector as type other than integer	1.3	2
I3	User Interface/API Testing	Test error conditions of PCA code. Provide a 1D vector of values – as row or column vector	1.3, 2.2	2
I4	User Interface/API Testing	Test error conditions of normalization code. Provide a 1D vector of values – as row or column vector	1.3	2
D1	Deployment testing - Installation	Install Q5 package on 4 different operating systems (Mac OS X, Linux, Windows 2000, and Windows XP)	1.1, 1.3, 1.4, 1.5	2
D2	Deployment Testing - Performance metrics	Report processor time and memory usage of Q5 algorithm on synthetic dataset for 5 different operating systems (Mac OS X, Solaris, Linux, Windows 2000, and Windows XP)	2.1, 2.2, 2.3, 2.4, 3.1	1
D3	Deployment Testing - User Acceptance Testing	The output of this testing will be report containing suggestions for improving Q5 including interface issues		3

7. RISKS

8. [NOTES]

APPENDIX A – ACRONYM LIST

Acronym	Description
CCR	Change control request
DBA	Database Administrator
HSTOMP	Health Services Team Organizational Management Portal
MS	Microsoft
PAL	Process Assessment Library
PM	Project Manager
RM	Requirements Manager
RTM	Requirements Traceability Matrix
SCM	Software Configuration Management
SDLC	Software Development Lifecycle
SE	Software Engineering
SEPG	Software Engineering Process Group
SM	Software Manager
SOP	Standard Operating Procedure
SPI	Software Process Improvement
SQA	Software Quality Assurance
SW	Software
TM	Test Manager
VM	Version Manager

APPENDIX B – TEST IDENTIFICATION

This section identifies and describes each test to which this Test Plan applies.

PLANNED TESTS

This section describes the total scope of the planned testing.

Testing R-Port of Q5

Test ID R1

Objective: Compare outputs of matlab version of Q5 to R version of Q5 algorithm under similar conditions.

Procedure:

1) Run Matlab version of Q5 with one hundred replicates on synthetic dataset
2) Run R version of Q5 with one hundred replicates on same dataset.
3) Report average and standard deviations of statistics.

Test ID R2

Objective: Compare effectiveness of Q5 algorithm to differentiate between two groups of different effect sizes of synthetic data.

Procedure:

1) Load dataset into memory (using groups with different effect sizes)
2) Produce 100 random partitions of data for three sets of train/test splits: 50%/50%, 75%/25%, 90%/10%.
3) Run Q5 algorithm on 100 random partitions produced above.
4) Report average and standard deviations of statistics.

Test ID R3

Objective: Compare effectiveness of Q5 algorithm to differentiate between two groups of that contain different sets of protein mixtures.

Procedure:

1) Load dataset into memory (using groups that contain different protein mixtures)
2) Produce 100 random partitions of data for three sets of train/test splits: 50%/50%, 75%/25%, 90%/10%.
3) Run Q5 algorithm on random partitions produced above step 2.
4) Report average and standard deviations of statistics.

mzXML Compatibility Testing

Test ID M1

Objective: Test mzXML compatibility of Q5 by comparing range of scan size within a mzXML file to matrix output by Q5 mzXML() function.

Procedure:

1. Load mzXML file using load.mzXML function in caMassClass package
2. For each scan in mzXML file, store length of scan
3. Load mzXML file using Q5 mzXML function
4. Note number of columns (scan length of matrix)
5. Report range of scan lengths found in step 2 and compare with number of columns of matrix in step 4

Test ID M2

Objective: Test mzXML compatibility of Q5 by comparing number of scans in mzXML file to matrix output by Q5 mzXML() function.

Procedure:

1. Load mzXML file using load.mzXML function in caMassClass
2. Count number of scans in mzXML file by using length() function
3. Load mzXML file using Q5 mzXML function
4. Count number of rows (number of scans)
5. Compare number of scans found in step 2 to number of rows in step 4.

Test ID M3

Objective: Compare output of Q5 algorithm using files loaded with Q5 mzXML() function to output of Q5 algorithm using RProteomics interpolated data.

Procedure

1) Select 10 random partitions of data file with 75/25 test train splits
2) Load interpolated data into memory and partition using random partitions
3) Run Q5 algorithm on all 10 sets of random partitions
4) Load mzXML data into memory using Q5 functionality – subset data using the same random partitions
5) Run Q5 algorithms on all 10 sets of random partitions
6) Compare statistical output from steps 3 and 5.

Interface/API Testing

Note: Q5 and mzXML functional testing is covered by tests R1, R2, R3, M1, M2, and M3.

Test ID I1

Objective: Compare output of Q5-implemented PCA algorithm to other implementations in R

Procedure:

1) Submit Fisher's Iris Dataset to Q5 PCA implementation
2) Submit Fisher's Iris Dataset to prcomp() function
3) Compare eigenvalue graphs for each implementation
4) Compare eigenvectors for each implementation.
5) Report Results.

Test ID I2

Objective: To test error conditions relating to the Q5 algorithm

Procedure : The test has been divided into three independent sub-tests to check for conditions handled by the system (see API document and SRSD document)

I **To check for issues with dependent packages not being loaded**

1) Check how the system handles essential packages not being loaded.

2) Report error message.

II **To check for invalid inputs**

1) Submit a test class vector that does not use integers to denote the class.

2) Report error message.

III **To assess the impact of unbalanced groups on algorithm performance**

1) Submit unbalanced groups for testing and training groups

2) Report on how the performance of the algorithm is impacted.

Test ID I3

Objective: To test error conditions handled by the system (see SRSD) for the PCA code

Procedure:

1) A one-dimensional vector of values as input.
2) Report error message from system.

Test ID I4

Objective : Test error conditions handled by the system (see SRSD) for normalization code

Procedure:

1) Input specified as a one-dimensional vector of values.
2) Report error message from system.

Test ID D1

Objective: Installation of Q5 package on 4 different operating systems to document any issues with installation

Procedure: For Each Operating System,

1) Install packages necessary for Q5.
2) Install Q5 package, report errors if found.
3) Load data into memory.
4) Run Q5 algorithm.
5) Report results for each operating system, along with any error messages if necessary.

Test ID D2

Objective: Report processor time and memory usage of Q5 algorithm on synthetic dataset for 4 different operating systems (Mac OS X, Linux, Windows 2000, and Windows XP)

Procedure:

For each Operating System:

1) Load synthetic dataset

2) Run Test script R5 (which runs Q5 with the pre-specified partitions, records both processor time and memory usage)
3) Report results.

Test ID D3

Objective: To obtain user feedback of the Q5 software.

Procedure:

1) Provide a biostatistician with the software package, data and installation directions
2) Document any difficulties they may have
3) Record the user's impressions and any feed back they might have.