Field Session 2006 – Final Report

Prepared by:

Paul Boschert
Alan Donaldson
Chris Hill

Friday, June 23, 2006

Abstract

Avaya has data relating to Root Cause Analysis (RCA) for various projects. They lack a sufficient way to update and monitor this information whenever an error is encountered. We will implement a web-based solution to this problem.

This application will make use of Java Servlets and MySQL. The tool will implement two databases, one for the RCA information, and the other to store user-interface information. The RCA database will consist of all information relevant to an MR (Modification Request). The user-interface database will hold all GUI elements, and allow the tool to be completely administrable.

TABLE OF CONTENTS

1.0 Introduction

The scope of this project involves creating a web-based tool which implements Java Servlets and accesses a MySQL database. Specifically, the database will hold RCA (Root Cause Analysis) information. This information will consist of a Problem Record, root data and symptom data. The Problem Record holds all information regarding the problem and relates to the root data and symptom data. Root data is where the problem initially occurred, whereas symptom data contains information relating to what explicitly broke in the program.

This tool will require the user to login in order to acquire the necessary privileges; this information will be stored in the database. Three different user types exist: Developer, RCA team member, and Administrator.

The GUI will have a consistent layout regardless of the user logged in. Each user type will also have a top bar that will hold all features necessary for the current user's privileges (e.g. Generate Report, Switch Users, Logout, etc.). The GUI will also have a printer friendly layout.

The flow of this tool will consist of progressing through three different states. Each Problem Record begins in the Developer state. Whenever a breakage occurs an e-mail is sent to the Administrator, RCA team member, and the Developer that submitted the Root or Symptom Data notifying them of the problem. Once the Developer has made the appropriate modifications and submits, then the Problem Record moves to the RCA state. In this state, the Developer no longer has write privileges and can only view the Problem. Finally, after the RCA team analyzes the Problem Record and submits, the Problem Record moves on to the Complete state. In the Complete state, only the Administrator can change the Problem Record, but the RCA team can still view the Problem.

Each user type will be able to generate a report on the Problem Record that will be exportable to an Excel spreadsheet. The user will be capable of setting a time interval and selecting fields on which to report. The RCA team member, as well as the Administrator, will be able to create a group of developers on which to report. If time allows, the report will contain charts and graphs that can be used to compare performance.

2.0 Requirements

2.1 Non-functional Requirements:

- Web-based
- Implemented with Java Servlets
- Database-driven

2.2 Functional Requirements:

- Interacts with a MySQL database
- Easily administrable
    - Help text, Field labels, Groups, Group order
- Able to parse and XML document to populate fields in database record
- Able to generate a report for Microsoft Excel
- Able to view, access, and modify RCA records in the database
- Able to modify, create, and delete users

2.3 Program Language Requirements:

- Java
- JavaScript
- CSS
- HTML
- PHP

2.4 Software Requirements:

- Apache HTTP Server
- Apache Tomcat
- MySQL

2.5 User Characteristics

There are three different types of users that will access this tool: Developers, RCA team members, and an Administrator.

Developer privileges:

- Create the initial delivery

- Modify records in the Developer state
- View records associated with that developer

RCA team member privileges:

- Generate Reports
- Modify records in the RCA state
- View anything

Administrator privileges:

- Generate Reports
- Modify groups and group ordering
- Modify help text
- Modify labels
- Specify which fields are required

2.6 Assumptions and Dependencies

In this tool we will not be implementing any form of security. We are assuming that Avaya will implement their own forms of security for the rest of the data.

We also assume they have a database server setup. We will be implementing this tool on our own servers and Avaya will need to have their own server set up with Apache Tomcat and MySQL in order for the tool to have full functionality.

3.0 Analysis and Design

3.1 RCA Database Schema

Figure 3.1 – RCA Database Schema (Removed as requested by Avaya)

The schema in Figure 3.1 represents the data needed to create a problem MR (Modification Request). In the diagram, all bold entries correspond to the primary keys of the individual tables, and the arrows represent foreign keys to other tables.

The three primary tables are:

- ProblemRecord - The main table in the database; it holds all information relevant to a specific problem MR.
- Person - Contains information about each user that has access to in the system.
- RootData - Holds information pertaining to the root of the problem.

The other tables exist to enforce referential integrity.

3.2 User-Interface Database Schema

Figure 3.2 – UI Database Schema (Removed as requested by Avaya)

The UI database, represented by Figure 3.2, contains six independent tables that can be used to store data relating to the user interface. This allows for the tool to be database-driven so that it is easily modifiable by an administrator.

The six tables are:

- Required – Specifies whether a field is required by an RCA member and/or developer.
- Options – Holds which options may be selected by specific fields.
- Groups – Holds information which allows administrators to group specific fields.
- NextRCAMember – Allows us to determine which RCA member should be assigned to a problem MR.
- Help – Contains a description for each field in the tool.
- LabelOrder – Contains the individual labels of all the necessary fields in the tool.

3.3 User-Interface Layout

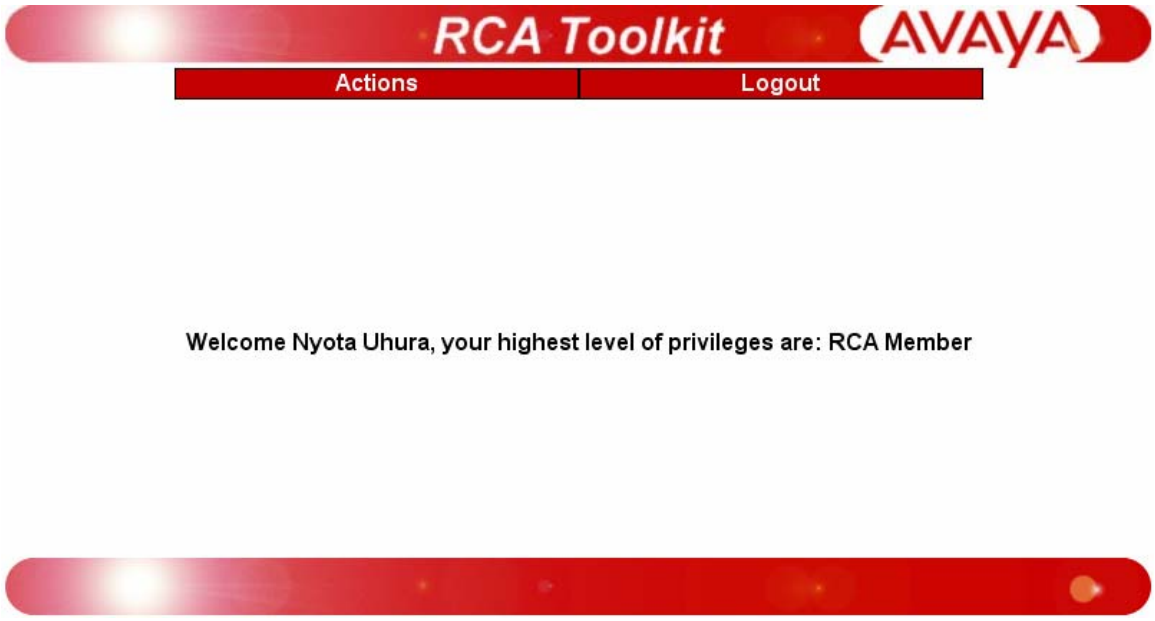

Figure 3.3 – UI Layout

Our tool will have a consistent look and feel across all users (See Figure 3.3). There will be two buttons on the top bar, Actions and Logout. The Actions button will be a drop-down menu that contains actions relevant to each specific user type.

3.4 User Types

The three user types will be:

3.4.1 Developer

- Create new problem MR – Allows the user to upload an XML document that will contain data for a new problem MR.
- View associated problem MR – Allows the user to view and modify problem MRs in the Developer state which the developer is associated with. Additionally, the user is allowed to view problem MRs in the RCA and Complete states if they are associated with them.
- Use case diagram, Figure 3.4:

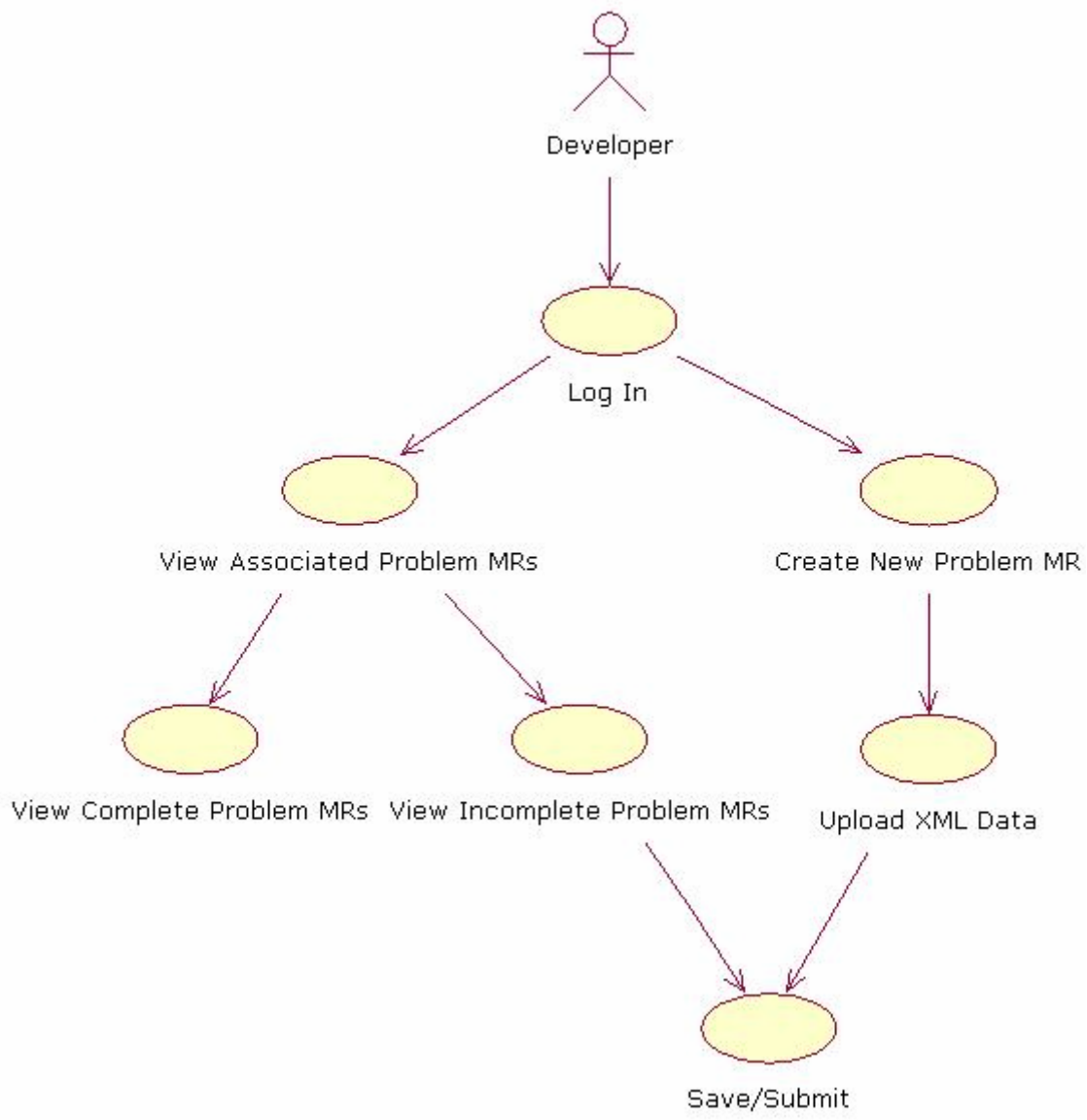

Figure 3.4 – Developer UML

3.4.2 RCA

- View incomplete unmodified problem MR – Allows the user to view and modify problem MRs that are in the RCA state but have not been modified by a user.
- View incomplete modified problem MR – Allows the user to view and modify problem MRs that are in the RCA state and have been modified by a user.
- View complete problem MR – Allows the user to view problem MRs in the Complete state and generate reports based on those problem MRs.
- Use case diagram, Figure 3.5:

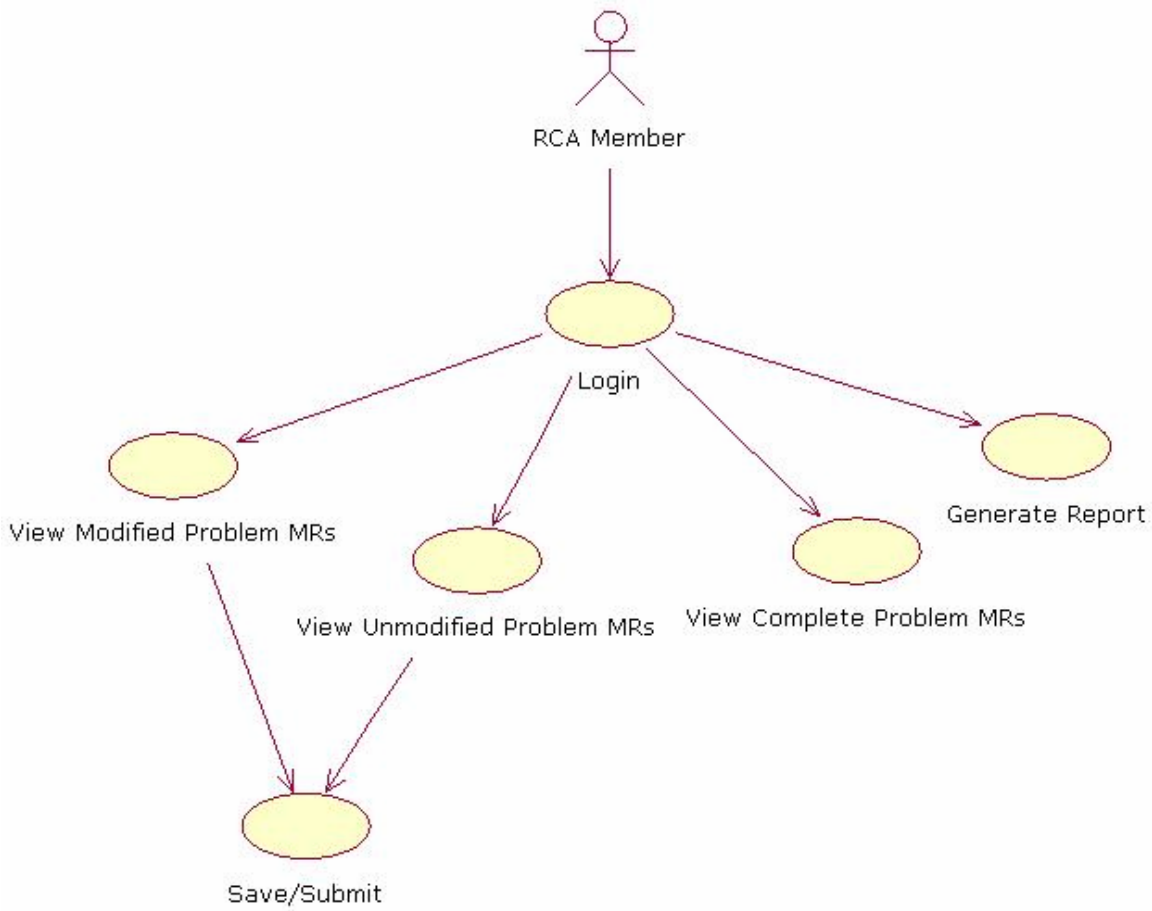

Figure 3.5 – RCA UML

3.4.3 Admin

- Create new problem MR – Allows the user to upload an XML document that will contain data for a new problem MR.
- View incomplete developer problem MRs – Allows the user to view incomplete problem MRs still in the developer state.
- View incomplete RCA problem MRs – Allows the user to view incomplete problem MRs still in the RCA state.
- View complete problem MR – Allows the user to view problem MRs in the Complete state and generate reports based on those problem MRs.
- Modify GUI – Allows the user to update the user interface. Specifically, the user will be able to change labels, help descriptions, field choices, and which fields are required.
- Edit users – Allows the admin to add and remove users from the system, as well as modify user information (e.g. e-mail address).
- Use case diagram, Figure 3.6:

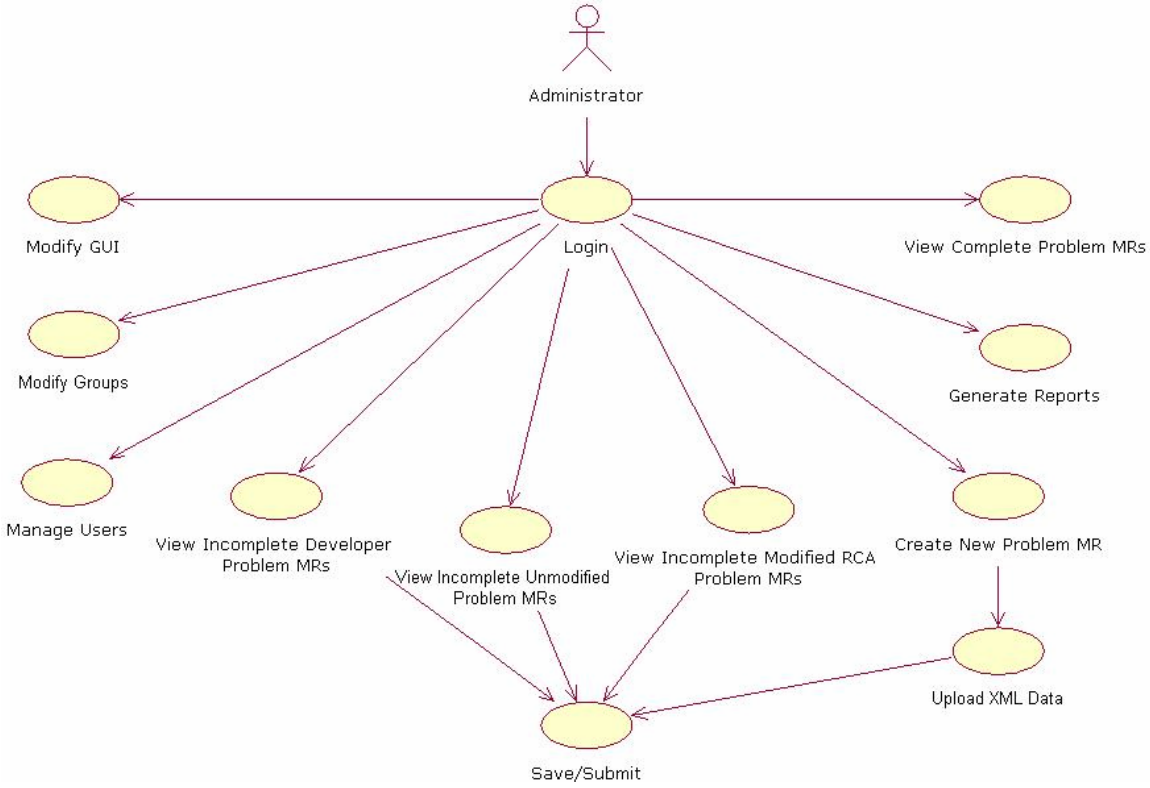

Figure 3.6 – Admin UML

4.0 Implementation

4.1 Login

This servlet logs the user into the rest of the site by storing session information, specifically a Cookie with the highest user privileges and another for the username. If a user attempts to login with a non-existent username/password combination, a relevant error message is displayed. Similarly if an inactive user attempts to log in, a relevant error message is displayed.

4.2 CreateNewMR

This servlet creates new records in the RCA database (ProblemRecord, ProjectNameType and RootData) based on information found in an XML document. The XML document is parsed by comparing the current field labels in the user-interface database to the XML tags. When a match is found, the content is verified and inserted into the appropriate field when creating the new problem record. Two fields which are required to be in the XML document are problem MR number and developer. Since the uploaded XML document is stored in memory as opposed to writing it to disk, a max file size of 512 KiB has been placed on the file upload feature. If the XML file fits these requirements, a new record will be created and the user will be redirected to a page where they can view the new record that has been created.

4.3 DevRelatedMRs

This servlet allows a Developer to view any Problem MR associated with them, whether it's in the Developer, RCA, or Complete state. It consists of two drop-down menus; one containing a list of all Problem MRs that are associated with the Developer but have not yet been submitted, and one that lists all Problem MRs the Developer has submitted. By selecting a Problem MR that has not yet been submitted, the user is redirected to the Developer servlet. By selecting a Problem MR that has been submitted, the user is redirected to the CompleteDeveloper servlet.

4.4 Developer

This servlet allows a developer to view and modify any Problem MR that is currently in the Developer state and is associated with that specific developer. Upon running the servlet, all necessary data is pulled from the database and used to populate the fields on the page. Once the page is loaded, the developer has the option to associate Root MRs with the Problem MR. When the user adds a new Root MR, the servlet checks to see if the Root MR actually exists. If it does indeed exist, then it will create an association in the RootData table using the form "CurrentMR-RootMR" where it can then store data relating to this association. If the Root MR does not exist, then this association is invalid and an error message is displayed. Each ProblemMR is permitted to have one "unknown" RootMR which allows the Developer to enter data about the RootMR without knowing its ID.

Once the user saves or submits the data, the servlet checks to see if every required option was entered before allowing the submit to proceed. If any required fields are missing, the page is reloaded with a message informing the user of which required fields were mssing. If no required fields were omitted then the database is updated with the selected data and the page is reloaded through Cookies. If the user submits the data, the state is updated to "RCA", the date is recorded signifying the date this state change occurred, a new RCA member is assigned to the MR and the Developer no longer has modification privileges on this Problem MR.

4.5 IncompleteRCA

This servlet is used to display modified and unmodified RCA pages. The status of whether the user is looking for a modified or unmodified MR appears as a parameter in the URL. The servlet then pulls this parameter and builds the drop-down menu appropriately.

4.6 RCA

This servlet is very similar to the Developer servlet except that it relates to all Problem MRs in the RCA state. This page also contains additional fields that did not exist on the Developer page including the date the Problem MR moved to the RCA state as well as the name of the RCA member assigned to the Problem MR. On this page, the RCA member is also able to associate, view, modify and disassociate Root MRs.

When the RCA team member fills in the required fields and submits the page the state is updated to "Complete", the Complete Date is updated, and the RCA member no longer has the ability to modify the Problem MR.

4.7 RootMR

When the user selects a Root MR from the Developer or RCA page, a pop-up window appears displaying all data found in the ProblemRecord of the RootMR that relates to this association. When the user clicks "submit", the servlet tests to see if all of the required fields have been entered. If they have, then the data is updated in both the RootData table and the ProblemRecord and the page is reloaded through the data stored in the Cookies.

4.8 CompleteRCA

This servlet can only be accessed by an administrator and an RCA team member. It contains a drop-down list of all the Problem MRs in the 'Complete' state. After the user selects a ProblemMR and clicks submit they are redirected to "CompleteDeveloper" where a read-only version of the RCA page is loaded.

4.9 CompleteDeveloper

This servlet is used for displaying read-only versions of the RCA and Developer pages. All fields are pulled directly from the database and the RootMRs are listed as links to CompleteRootMR.

4.10 CompleteRootMR

This servlet is used for displaying read-only versions of the RootMR. Upon calling CompleteRootMR the database is queried and all data is pulled directly from the database.

4.11 GenerateReport

This servlet handles the process of selecting the type of report to be generated, the MRs and fields to report on, and then creates the report. When arriving at the page, the user first selects which report they would like to generate: RCA data report, developer report or XML report.

4.11.1 RCA Data Report

If the user selects an RCA data report, they will be redirected to a page where they can select the problem MRs and fields to be included on the report. Once the fields and MRs have been selected, the user will be directed to a page which will prompt them to download the report in .csv file, which can be opened and modified in Microsoft Excel.

4.11.2 Developer Report

If the user selects a developer report, they will be redirected to a page where they can select the developers to be included in the report. After selecting the developers, the user must select which problem MRs and fields they wish to be included in the report. Once the fields and MRs have been selected, the user will be directed to a page which will prompt them to download the report in .csv file, which can be opened and modified in Microsoft Excel.

4.11.3 XML Report

If the user selects an XML report, they will be directed to a page which will prompt them to download an XML document. The main purpose of this feature is to allow the user to easily create a new XML document for creating new records after the field labels have been modified

4.12 ManageUsers

This servlet allows one to modify existing users, create new users, and delete existing users in the Person table of the RCA database. This servlet links to the EditUser servlet for most functionality.

4.13 EditUser

This servlet is called from the ManageUsers servlet. If the Modify, Create, or Delete button was pressed on ManageUsers, this servlet will update, insert, or delete respectively.

4.14 ModifyGUI

This servlet displays the administrable fields associated with GUI properties. Specifically, all Labels, Help Information, whether it's required for a Developer or RCA member, and links to ModifyOptions which displays any Options that a field may contain.

4.15 ModifyOptions

This servlet displays the options for a particular field and whether that option is currently marked as active or not. Also allows the creation of a new option.

4.16 ModifyGroups

This servlet displays the current Groups, allows the user to create new Groups, remove Groups, and modify the order in which the Groups are displayed.

4.17 ModifyGroupItems

This servlet displays the Group Items for any specified Group, it is called by ModifyGroup. Also allows the user to remove Items, add new Items, and modify the order in which those Items are displayed within the Group. The only new Items that may be added must not be in any other Group.

4.18 Logout

This servlet logs the user out of the system by removing all Cookies associated with the site then redirects the user to the Login page.

5.0 Testing

5.1 Login

Precondition:	User enters valid username/password.
Postcondition:	User is redirected to a Welcome page.
Status:	Pass

Precondition:	User enters invalid username/password.
Postcondition:	Relevant error message displayed, and user redirected to Login page.
Status:	Pass

Precondition:	User enters valid username/password but account is inactive.
Postcondition:	Relevant error message displayed, and user redirected to Login page.
Status:	Pass

5.2 Welcome

5.2.1 Developer

Precondition:	User with developer privileges logs in.
Postcondition:	Drop-down menu displays “Create New Problem MR” and “View Related Problem MR.”
Status:	Pass

Precondition:	Developer selects “Create New Problem MR.”
Postcondition:	Developer is redirected to CreateNewMR.
Status:	Pass

Precondition:	Developer selects “View Related Problem MRs.”
Postcondition:	Developer is redirected to DevRelatedMRs.
Status:	Pass

Precondition:	Developer selects “Logout.”
Postcondition:	Cookies are cleared and Developer is redirected to Login page.
Status:	Pass

5.2.2 RCA

Precondition:	User with RCA privileges logs in.
Postcondition:	Drop-down menu displays “View Unmodified Problem MRs,” “View Modified Problem MRs,” “View Complete Problem MRs,” and “Generate Reports.”
Status:	Pass

Precondition:	RCA Member selects “View Unmodified Problem MRs.”

Postcondition: RCA Member is directed to page containing problem MRs that haven't been viewed.
Status: Pass

Precondition: RCA Member selects "View Modified Problem MRs."
Postcondition: RCA Member is directed to page containing problem MRs that have been viewed.
Status: Pass

Precondition: RCA Member selects "View Complete Problem MRs."
Postcondition: RCA Member is directed to page containing problem MRs that are in the complete state.
Status: Pass

Precondition: RCA Member selects "Generate Reports."
Postcondition: RCA Member is directed to a page that allows the user to generate a report based on specified data.
Status: Pass

Precondition: RCA Member selects "Generate Reports."
Postcondition: RCA Member is directed to GenerateReport.
Status: Pass

5.2.3 Admin

Precondition: User with Admin privileges logs in.
Postcondition: Drop-down menu displays "Create New Problem MR," "View Incomplete RCA Problem MRs," "View Incomplete Developer Problem MRs," "View Complete Problem MRs," "Modify GUI," and "Manage Users."
Status: Pass

Precondition: Admin selects "Create New Problem MR."
Postcondition: Admin is directed to CreateNewMR.
Status: Pass

Precondition: Admin selects "View Incomplete RCA Problem MRs"
Postcondition: Admin is directed to a page containing a list of all problem MRs in RCA state.
Status: Pass

Precondition: Admin selects "View Incomplete Developer Problem MRs"
Postcondition: Admin is directed to a page containing a list of all problem MRs in Developer state.
Status: Pass

Precondition: Admin selects "View Complete Problem MRs"
Postcondition: Admin is directed to a page containing a list of all problem MRs in complete state.
Status: Pass

Precondition: Admin selects "Modify GUI Elements"
Postcondition: Admin is directed to ModifyGUI.
Status: Pass

Precondition:	Admin selects “Manage Users”
Postcondition:	Admin is directed to ManageUsers.
Status:	Pass

Precondition:	Admin selects “Modify Groups”
Postcondition:	Admin is directed to ModifyGroups.
Status:	Pass

Precondition:	Admin selects “Generate Reports.”
Postcondition:	Admin is directed to GenerateReport.
Status:	Pass

5.3 Create New Problem MR

Precondition:	User uploads valid XML file.
Postcondition:	User is redirected to Developer (where they may complete the Problem MR).
Status:	Pass

Precondition:	User uploads invalid XML file or one which doesn’t contain a Developer name and/or Problem MR.
Postcondition:	User is directed to relevant error page.
Status:	Pass

Precondition:	User uploads XML file over 512 KiB.
Postcondition:	User is directed to relevant error page.
Status:	Pass

5.4 View Related Problem MRs (Developer Only)

Precondition:	Developer selects Problem MR from Incomplete Problem MR drop-down box and clicks Submit.
Postcondition:	User is redirected to Developer (where they may complete the Problem MR).
Status:	Pass

Precondition:	Developer selects Problem MR from Complete Problem MR drop-down box and clicks Submit.
Postcondition:	Developer is redirected to read-only version of Developer.
Status:	Pass

Precondition:	Developer doesn’t select from the drop-down box and clicks Submit.
Postcondition:	Developer is redirected to the same page.
Status:	Pass

5.5 Modify GUI Elements (Admin Only)

Precondition:	Admin modifies GUI property and clicks Submit.
Postcondition:	The User-Interface database is updated.
Status:	Pass

Precondition: Admin clicks Modify Options for any relevant field.
Postcondition: The admin is redirected to ModifyOptions.
Status: Pass

5.5.1 Modify Options

Precondition: Admin modifies properties of Option (Name and Active).
Postcondition: The User-Interface and RCA database are updated.
Status: Pass

Precondition: Admin creates new Option.
Postcondition: The User-Interface and RCA database are updated and the current view is updated with the new Option.
Status: Pass

5.6 Manage Users (Admin Only)

Precondition: Admin selects user from "Edit Existing Users" drop-down box and clicks Modify.
Postcondition: Admin is redirected to EditUser.
Status: Pass

Precondition: Admin types new username in "Create New User" textbox and clicks Create.
Postcondition: Admin is redirected to EditUser.
Status: Pass

Precondition: Admin selects user from "Remove Existing User" drop-down box and clicks Delete.
Postcondition: Database is updated and successful message is displayed.
Status: Pass

5.6.1 Edit User

Precondition: Admin modifies user properties and clicks Submit.
Postcondition: Database is updated and successful message is displayed then admin is redirected to ManageUsers.
Status: Pass

5.7 Modify Groups (Admin Only)

Precondition: Admin selects "Group Items" for a specific Group.
Postcondition: The admin is redirected to ModifyGroupItems and the Items for the Group are displayed.
Status: Pass

Precondition:	Admin selects “Submit.”
Postcondition:	All items marked Remove are removed, the name of the Group is updated, and the order in which the Groups are displayed is updated.
Status:	Pass

Precondition:	Admin fills in Create New Group text box and clicks “Create Group.”
Postcondition:	New Group is created with specified name; the new Group is displayed last.
Status:	Pass

5.8 Modify Group Items (Admin Only)

Precondition:	Admin selects “Submit”.
Postcondition:	All items marked Remove are removed. And the order in which the Items are displayed is updated.
Status:	Pass

Precondition:	Admin selects a new item to add and clicks “Add Item.”
Postcondition:	The new Item is added to the end of the Item list.
Status:	Pass

5.9 Generate Reports (Admin and RCA Only)

Precondition:	User selects “RCA Data Report” and clicks “Next.”
Postcondition:	User is directed to page where they may select specific MRs and fields on which to report.
Status:	Pass
Precondition:	User selects “Developer Report” and clicks “Next.”
Postcondition:	User is directed to page where they may select specific Developers on which to report.
Status:	Pass

Precondition:	User selects “XML Report” and clicks “Next.”
Postcondition:	User is prompted to download XML document.
Status:	Pass

5.9.1 RCA Data Report

Precondition:	User selects specific MRs and fields on which to report and clicks “Next.”
Postcondition:	Report is generated and user is prompted to download comma-delaminated document.
Status:	Pass

5.9.2 Developer Report

Precondition:	User selects specific Developers on which to report and clicks “Next.”
Postcondition:	User is directed to page where they may select specific MRs and fields on which to report.
Status:	Pass

Precondition:	User selects specific MRs and fields on which to report and clicks “Next.”
Postcondition:	Report is generated and user is prompted to download comma-delaminated document.
Status:	Pass

6.0 Conclusions

6.1 Lessons Learned

Throughout the development process we continuously encountered problems that forced us to backtrack and rewrite code; among these were shifts in the requirements/design and modifications to the database schema. At first this created problems by delaying our course of action drastically, occasionally stealing entire days from our timeline. By the end of the project we were instinctively writing code that was dynamic and flexible enough so that these changes didn’t have nearly the impact they had at the start of the project.

6.2 Future Directions

We feel that we accomplished every requirement we set out to complete and that our program is fully functional. Had we been granted additional time to work on this project there are several goals we would have liked to accomplish. The first of these would be the encryption of user’s passwords. At the moment our passwords are being stored in the database in plain text format. We feel that the program would be more secure if we had encrypted these. On top of this, we were given an optional requirement to allow an RCA team member to graph the data after generating a report. This is something that would have required considerable research, but we feel that we could have accomplished it had we been given more time.