

# U.S. Department of Education Federal Student Aid



START HERE  
GO FURTHER  
FEDERAL STUDENT AID<sup>®</sup>

## Federal Student Aid Enterprise Test Management User Guide

Version 01.01

11/01/2010

**Document Version Control**

<b>Version</b>	<b>Date</b>	<b>Description</b>
01.00	09/30/2010	Initial Release
01.01	11/01/2010	Updated decision point for in figure 6.5 to include a description of who cancels a defect if it is determined to be invalid. Minor corrections to verbiage.

## Table of Contents

Section 1. Introduction.....	1
1.1 Purpose.....	1
1.2 Scope.....	1
1.3 Intended Audience .....	1
1.4 Document Organization.....	1
1.5 References and Related Documents.....	2
Section 2. Rational Environment Configuration.....	3
2.1 Configuration of the Rational Environment .....	3
2.2 Rational Tool Integration Overview .....	4
2.3 Security Overview .....	16
Section 3. Test Management Overview .....	18
Section 4. Test Planning .....	20
4.1 Test Plans .....	20
4.2 Test Cases .....	30
4.3 Test Scripts.....	42
4.4 Test Suites.....	44
4.5 Test Data.....	47
4.6 Test Asset Reuse.....	51
Section 5. Test Execution .....	52
5.1 Test Execution Scheduling .....	52
5.2 Test Execution Assignment .....	54
5.3 Test Execution Evaluation .....	58
Section 6. Defect Management.....	64
6.1 ClearQuest Design Overview .....	64
6.2 Working with Change Requests (CRs).....	69
6.3 Working with Defects.....	73
6.4 Email Notification.....	74
Section 7. Reporting.....	75
7.1 Test Management Report.....	75
7.2 Requirements Traceability Matrix .....	82
7.3 Defect Management Reports.....	83
Appendix A: Acronyms and Abbreviations .....	A-1
Appendix B: Glossary .....	B-1
Appendix C: ClearQuest Report Samples .....	C-1
Appendix D: Security .....	D-1

## List of Figures

Figure 2-1: Requirements Section of a Test Plan .....	5
Figure 2-2: Requirements Section of a Test Case.....	6
Figure 2-3: RequisitePro Opened from RQM.....	7
Figure 2-4: Defects from RQM.....	8
Figure 2-5: Defect Creation During Test Case Execution.....	9
Figure 2-6: Defect Generated from Failed Test Case .....	10

---

Figure 2-7: Defect Associated with Failed Test Step .....	11
Figure 2-8: Defect Visible from Test Execution Result .....	12
Figure 2-9: Change Request – Requirements Tab .....	13
Figure 2-10: RequisitePro Requirements Linked to ClearQuest Records .....	14
Figure 2-11: CR Change Set .....	15
Figure 3-1: Test Management Artifact Relationships .....	18
Figure 4-1: Test Plan Summary Section .....	21
Figure 4-2: Test Plan Formal Review Section .....	22
Figure 4-3: Test Plan Requirements Section .....	23
Figure 4-4: Test Plan Test Schedules Section .....	24
Figure 4-5: Test Plan Test Case Section .....	26
Figure 4-6: Test Plan Attachments Section .....	27
Figure 4-7: Test Plan Child Test Plans Section .....	29
Figure 4-8: Test Case Summary Section .....	30
Figure 4-9: Test Case Test Case Design Section .....	32
Figure 4-10: Test Case Formal Review Section .....	33
Figure 4-11: Test Case Requirement Section .....	34
Figure 4-12: Test Case Pre-Conditions Section .....	36
Figure 4-13: Test Case-Post Conditions .....	37
Figure 4-14: Test Case Expected Results Section .....	38
Figure 4-15: Test Case Test Scripts Section .....	39
Figure 4-16: Test Case Test Results Section .....	40
Figure 4-17: Test Script Editor .....	43
Figure 4-18: Test Suite Summary Section .....	44
Figure 4-19: Test Suite Test Suite Design Section .....	45
Figure 4-20: Test Suite Test Cases Section .....	46
Figure 4-21: Sample Test Data CSV File .....	48
Figure 4-22: Creating Test Data .....	49
Figure 4-23: Entering Test Data Variables in Scripts .....	50
Figure 4-24: Test Data Execution Example .....	51
Figure 5-1: Test Execution Schedules .....	52
Figure 5-2: Expanded Test Execution Schedule .....	53
Figure 5-3: Test Suite Expanded in Test Execution Schedule .....	54
Figure 5-4: Opening Work Items .....	55
Figure 5-5: Creating a Work Item .....	55
Figure 5-6: Completing a Work Item .....	56
Figure 5-7: Linking a Work Item to a Test Suite/Case .....	57
Figure 5-8: Specifying the Work Item URL .....	57
Figure 5-9: Work Item’s Related Artifacts .....	58
Figure 5-10: Work Items in a User’s Dashboard .....	58
Figure 5-11: Finding Test Suite Execution Results .....	59
Figure 5-12: Test Suite Execution Results .....	60
Figure 5-13: View Test Suites .....	61
Figure 5-14: Test Suite View Results .....	61
Figure 5-15: Test Execution Console .....	62
Figure 5-16: TER Listing Report .....	63

Figure 6-1: Related ClearQuest Record Types .....	64
Figure 6-2: Defect Record Workflow .....	66
Figure 6-3: CR Record Workflow .....	67
Figure 6-4: ChildCR Record Workflow .....	68
Figure 6-5: Defect Resolution Flowchart .....	69
Figure 7-1: Summary of Phases by Iteration .....	76
Figure 7-2: Summary of Phases by Iteration – Details (page 1).....	77
Figure 7-3: Test Execution Report.....	79
Figure 7-4: Test Status Report for Phase .....	81
Figure 7-5: Requirements Traceability Matrix .....	83
Figure 7-6: Plan Requirements Coverage Detail Report .....	83

## List of Tables

Table 1-1: Intended Audience and Uses .....	1
Table 2-1: Rational Tools Requiring Configuration .....	3
Table 2-2: Operations Reserved for the Test Lead .....	17
Table 3-1: Steps to Create Testing Artifacts.....	19
Table 4-1: Test Plan Summary Fields.....	21
Table 4-2: Test Plan Formal Review Fields .....	22
Table 4-3: Test Plan Requirements Fields .....	24
Table 4-4: Test Plan Test Schedule Fields.....	25
Table 4-5: Test Plan Test Case Fields.....	27
Table 4-6: Test Plan Attachments Fields .....	28
Table 4-7: Test Plan Child Test Plan Fields .....	29
Table 4-8: Test Case Summary Fields .....	31
Table 4-9: Test Case Test Case Design Field .....	32
Table 4-10: Test Case Formal Review Fields.....	33
Table 4-11: Test Case Requirements Fields .....	35
Table 4-12: Test Case Pre-Conditions Fields .....	36
Table 4-13: Test Case Post-Conditions Fields.....	37
Table 4-14: Test Case Expected Results Fields.....	38
Table 4-15: Test Case Test Scripts Fields.....	39
Table 4-16: Test Case Test Results Fields.....	41
Table 4-17: Test Script Editor.....	43
Table 4-18: Test Suite Summary Fields.....	45
Table 4-19: Test Suite Design Fields.....	46
Table 4-20: Test Suite Test Cases.....	47
Table 6-1: CR Step/Action Table .....	73
Table 6-2: Defect Step-Action Table.....	73
Table 6-3: Defect Email Notification Rules .....	74
Table 7-1: Summary of Reports.....	75
Table 7-2: Summary of Phase by Iteration Data.....	77
Table 7-3: Summary of Phases by Iteration – Details (page 1) Data.....	78
Table 7-4: Test Execution Report Data .....	80
Table 7-5: Test Stats Report for Phase Data.....	82

# Section 1. Introduction

## 1.1 Purpose

The purpose of this Federal Student Aid Test Management User Guide is to provide guidance in the utilization of the Rational tools to support testing activities in the FSA Environment.

It is the responsibility of the Test Management Lead to ensure that the guidelines, rules, and procedures defined in the Federal Student Aid Enterprise Test Management Standards document and this user guide are followed.

## 1.2 Scope

The information presented in this user guide is limited to the usage of the Rational tools. All users are expected to have prior hands-on experience using RequisitePro, Rational Quality Manager, ClearCase, and ClearQuest.

## 1.3 Intended Audience

Table 1-1 lists the intended users and the purpose for which the users may utilize information in this document.

Users	Uses
Test Team	As a guide to implementing standard test management practices using the Rational suite of tools.

**Table 1-1: Intended Audience and Uses**

## 1.4 Document Organization

This document comprises the following sections:

- Section 1 – Introduction
- Section 2 – Rational Environment Configuration
- Section 3 – Test Management Overview
- Section 4 – Test Planning
- Section 5 – Test Execution
- Section 6 – Defect Management
- Section 7 – Reporting
- Appendix A – Acronyms and Abbreviations
- Appendix B – Glossary
- Appendix C – ClearQuest Report Samples

## Appendix D – Security

### 1.5 References and Related Documents

The following documents were referenced during the development of this requirements management user guide:

- ACS Directive OCIO:1-106, Lifecycle Management Framework
- FSA Enterprise Configuration Management Plan Template
- FSA Enterprise Configuration Management User Guide
- FSA Enterprise Requirements Management Plan Template
- FSA Enterprise Requirements Management User Guide
- FSA Enterprise Test Management Standard
- IBM Rational ClearCase Command Reference
- IBM Rational ClearCase Information Center (Web Based ClearCase Guidance):  
<https://publib.boulder.ibm.com/infocenter/cchelp/v7r1m0>

## Section 2. Rational Environment Configuration

While a standard Rational solution has been developed for use across all FSA system development efforts, a degree of system-specific configuration has been preserved in order to support the needs of individual systems.

### 2.1 Configuration of the Rational Environment

At the start of a system development effort, a system-specific instance of the FSA Enterprise Rational Solution will need to be created and configured. The FSA Rational Support Group (FSA RSG) will create the instance of the environment for the new system and the configuration management (CM) Lead will configure each tool in the environment based on system-specific requirements. The Test Lead for the system will coordinate with the CM Lead in order to ensure that all required customizations are implemented.

Table 2-1 lists the primary Rational tools used in every system development effort as well as the type of system-specific configuration performed by the CM Lead.

Rational Tool	Description of System-Specific Configuration
RequisitePro	No system-specific tailoring is supported.
ClearQuest	Choice list values will be updated for CR, ChildCR, and Defect Record types.
Rational Quality Manager	Test Plan and Test Case category choice lists will be tailored to system-specific requirements.
ClearCase	Directories will need to be created in the system documentation Version Object Base (VOB) and software VOBs. Throughout the development effort, base-ClearCase labels and Unified Change Management (UCM) baselines will need to be created.

**Table 2-1: Rational Tools Requiring Configuration**

Although some of the Rational tools offer multiple interfaces, the web/browser-based interface should be used whenever available. The available interfaces for each tool are as follows:

- RequisitePro: Native Windows interface accessible via Citrix, Web/Browser-based interface.
- ClearQuest: Native Windows interface accessible via Citrix, Web/Browser-based interface.
- Rational Quality Manager: Web/Browser-based interface.
- ClearCase: Native Windows interface accessible via Citrix, Thin Client (http/https)-based interface (ClearCase Remote Client), and Integration with Eclipse-based Rational tools.

Unless otherwise indicated, the Rational tool related functions described in this document can be performed using any of the interfaces (native-client, thin-client, and web-client) provided by the tools.

NOTE: When printing from the web interface for any of the Rational tools, the tool-specific print function should be used rather than using the browser print function directly.

A brief overview of tailoring options for the various Rational tools directly related to test management is listed below. Contact the system CM Lead for additional clarification on tailoring the FSA Enterprise Rational solution to system-specific needs or refer to the Federal Student Aid Enterprise Configuration Management User Guide.

### **2.1.1 ClearQuest Project Configuration and Tailoring**

The FSA standard implementation of ClearQuest has been tailored to allow system-specific tailoring of choice lists for the various record types including CRs, Child CRs and Defects. When a project-specific instance of ClearQuest is first deployed, and throughout the life of the project, the CM Lead will be responsible for maintaining these choice lists.

### **2.1.2 Rational Quality Manager Configuration and Tailoring**

The FSA standard RQM implementation has been tailored such that the system's CM Lead may update choice lists located in the summary sections of Test Plans and Test Cases.

## **2.2 Rational Tool Integration Overview**

While this document is primarily focused on test management, test assets such as test cases and test suites do not exist in isolation. The Rational tools have been tailored to allow artifacts managed by the different Rational tools to be used in an integrated manner.

### **2.2.1 RQM to RequisitePro**

Since the objective of any testing effort is to verify that the requirements for the system have been met, each test plan and associated test cases need to be related to a set of testable requirements. RQM satisfies this need by allowing requirements managed by RequisitePro to be linked to test plans and test cases managed by RQM.

Each RQM "project area" has been configured to connect to a corresponding RequisitePro project area in order to associate RequisitePro managed requirements with RQM managed test plans and cases. Figure 2-1 shows the requirements section of a test plan. This section is populated by test team members as the test plan is being developed.

Manage Sections

## FPDM DEMO - UAT

Test Plan Overview | [View Snapshots](#)

Discard Changes Save

Originator: Mark Kaminsky    Action: Select Action    State: Draft

Description: User Acceptance Test Plan

**Table Of Contents**

- Summary
- Formal Review
- Requirements
- Test Schedules
- Test Cases
- Attachments
- Show All Sections

### Requirements

Work Item: [Create](#)

This section lists all of the content and requirements associated with a given test plan. You can select existing requirements or define new items to cover in the test plan.

Group by: Ungrouped   

Show All    Items per page    Previous | 1 - 5 of 5 | Next

Status	ID	Source Id	Name	Description	Owner
<input type="checkbox"/>	296	PR4	PR4	The QBS system shall allow only maintenance of current savings accounts.	Unassigned
<input type="checkbox"/>	299	PR7	PR7	All other screens shall have customer information updated from the Customer Information Screen.	Unassigned
<input type="checkbox"/>	301	PR9	PR9	The QBS system shall provide the following reports:	Unassigned
<input type="checkbox"/>	304	PR9.3	PR9.3	Report: Customer listing with options to sort by name, company, interest rate, and origination date.	Unassigned
<input type="checkbox"/>	308	PR13	PR13	The QBS system shall offer a comprehensive on-line help system.	Unassigned

Previous | 1 - 5 of 5 | Next

**Figure 2-1: Requirements Section of a Test Plan**

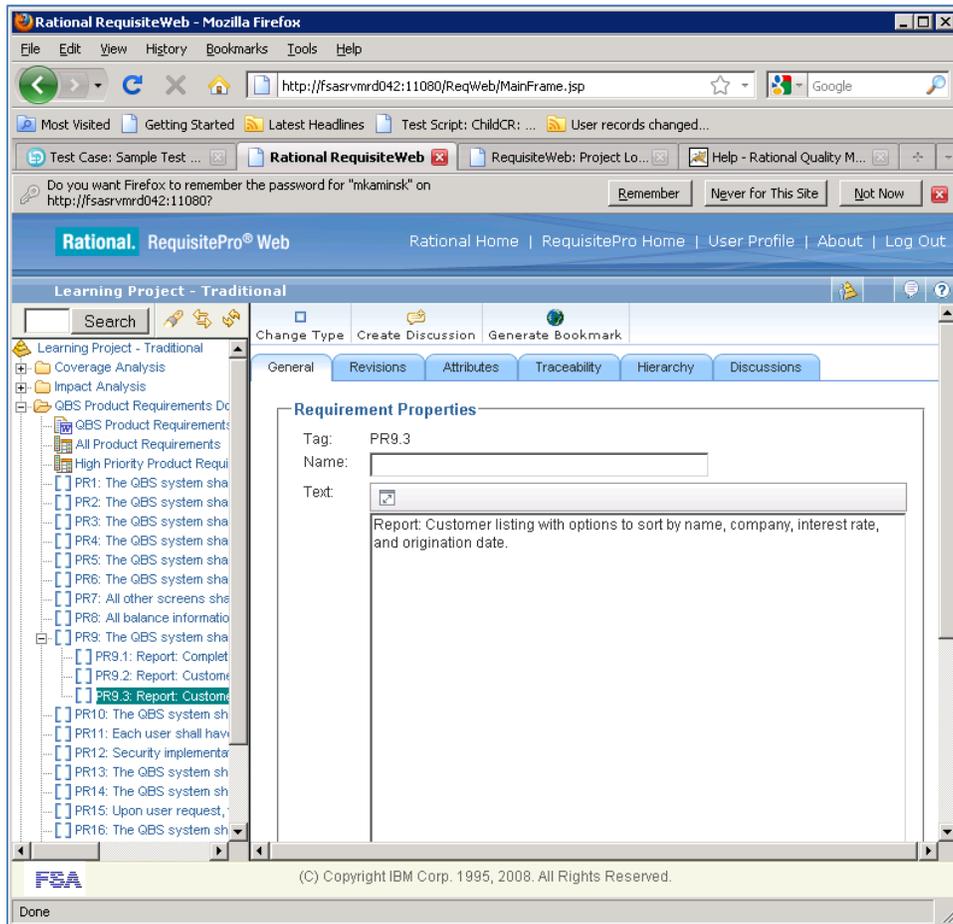
Figure 2-2 shows the requirements section of a test case. This section is populated by the test team as the test cases are being developed. Since each test case should be related to one or more test plans, RQM restricts the list of requirements that can be associated with the test case based on the requirements associated with the related test plans.

The screenshot shows the 'Requirements' section of a test case. The interface includes a sidebar with 'Manage Sections' and 'Table Of Contents'. The main area displays 'Sample Test Case 4/2/10' with a 'Requirements' section containing a table of requirements.

Status	ID	Source Id	Name	Description	Owner
<input type="checkbox"/>	301	PR9	PR9	The QBS system shall provide the following reports:	Unassigned
<input type="checkbox"/>	304	PR9.3	PR9.3	Report: Customer listing with options to sort by name, company, interest rate, and origination date.	Unassigned

**Figure 2-2: Requirements Section of a Test Case**

While RQM only displays the name (RequisitePro Tag) and description of each requirement, additional details can be viewed by opening the requirement in RequisitePro. The Requirement Name shown in the “Requirements” section of the RQM test plan and test case is a hyperlink which will open RequisitePro to the requirement being referenced. Figure 2-3 shows the RequisitePro web page produced by clicking on the requirement linke “PR9.3” shown in Figure 2-2.



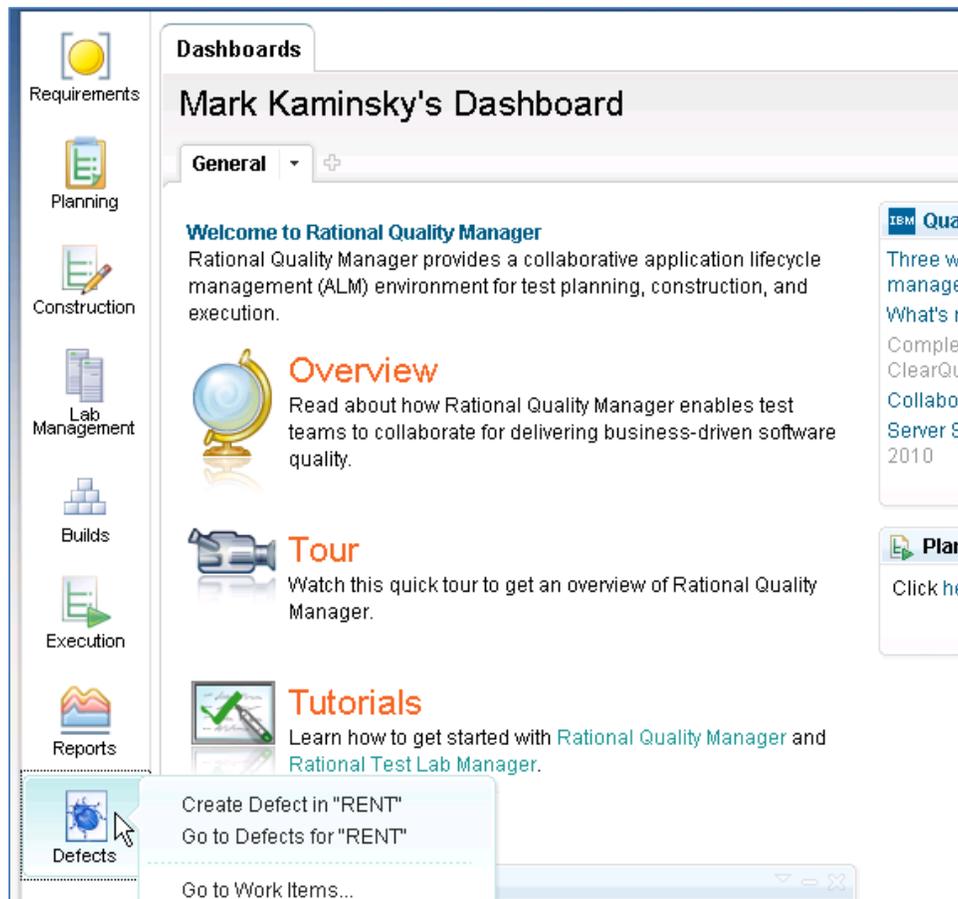
**Figure 2-3: RequisitePro Opened from RQM**

## 2.2.2 RQM to ClearQuest

The ClearQuest defect record will be used to document all defects identified during the execution of test cases managed by Rational Quality Manager. When a tester identifies a defect during the execution of a test case, the tester will generate a new ClearQuest-based defect record from inside RQM. RQM will communicate with ClearQuest and present the tester with a Defect submission form via a web interface.

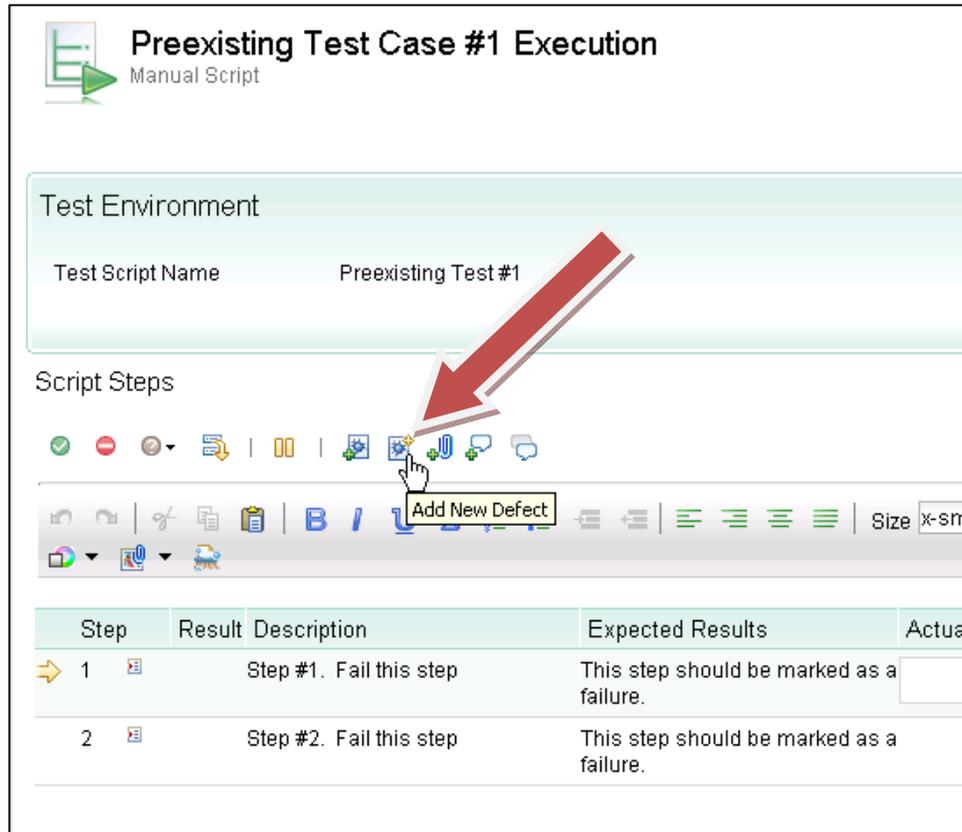
ClearQuest-based defects can be created directly in RQM. There are two approaches to creating a defect from RQM:

The first approach is to use the “Defects” icon on the leftmost pane in RQM as shown in Figure 2-4 . This approach will launch a new ClearQuest defect in a new browser window.



**Figure 2-4: Defects from RQM**

The second approach involves creating a defect during the execution of a test case. Figure 2-5 shows how to create a defect that will be tied to step 1 in the test case. This capability allows the tester to create multiple defects per test case based on the individual step that fails. It is possible that multiple defects could be created for a single step.



**Figure 2-5: Defect Creation During Test Case Execution**

After selecting the "defect" icon, ClearQuest is launched and the headline and defect description are automatically filled in with information about which test case failed and which steps will reproduce the failure. Figure 2-6 is an example of what the tester will see after clicking on the "defect" icon inside the test execution editor. Notice that the title and defect description have been automatically populated.

**Add New Defect**

Defect: RENT00000055 Save Cancel

**Main** | Related CRs | Notes | Attachments | Admin0

ID: RENT00000055 State: Submitted

Title: \*Identified On:  Opened On: 8/23/10 12:29:20 p.m.

Failing Test Case "Preexisting Test Case #1"

Full Name: Kaminsky, Mark Login ID: mkaminsk

Phone: 703.626.5014 Email: mkaminsky@trinity-software.com

\*Defect Type:  \*Severity:

Defect Type (Other)

\*Detected in Phase:

Detected In Phase (Other)

**Defect Description**

Test Script: Preexisting Test #1  
 Test Plan: Sample Test Plan

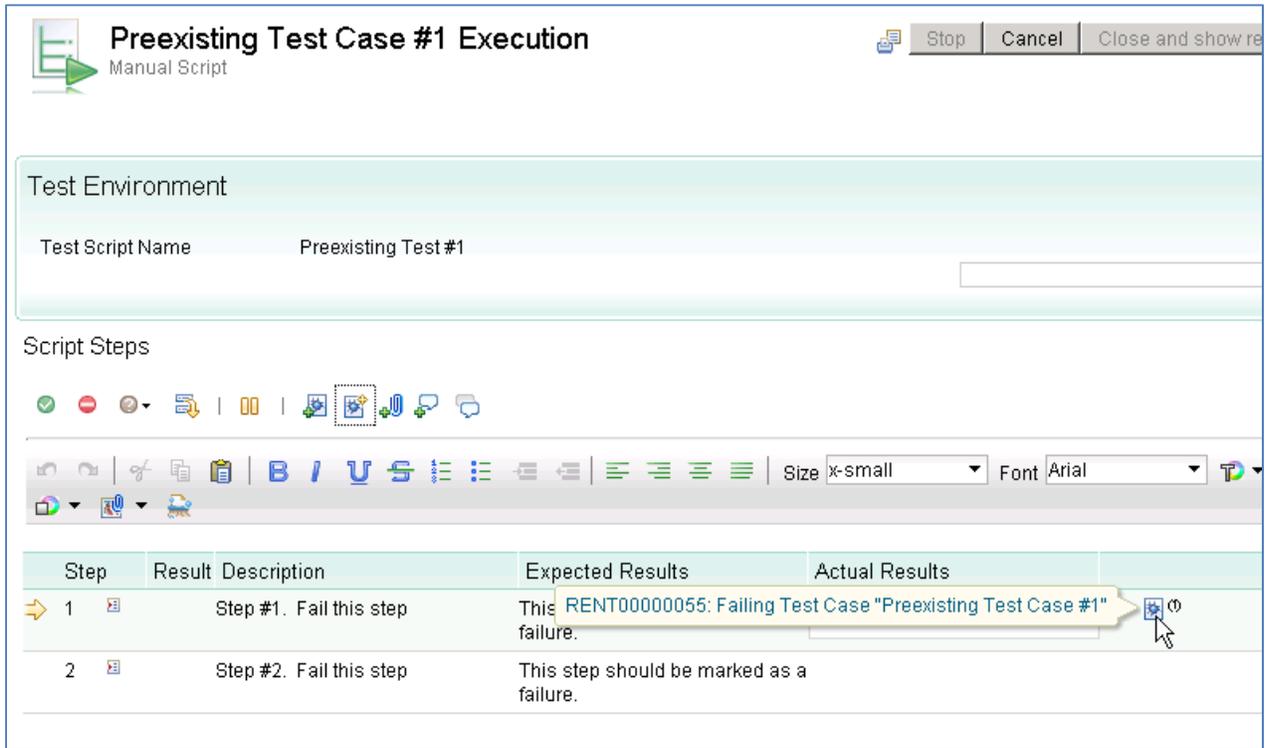
Test Case  
[http://fsarationalrqmdev:9080/jazz/oslc\\_qm/contexts/\\_aeG6kIq8Ed-RUocRhnxVsQ/resources/com.ibm.rqm.planning.VersionedTestCase/\\_z1yI1ZTOEd-wKOoFv6CzJw](http://fsarationalrqmdev:9080/jazz/oslc_qm/contexts/_aeG6kIq8Ed-RUocRhnxVsQ/resources/com.ibm.rqm.planning.VersionedTestCase/_z1yI1ZTOEd-wKOoFv6CzJw)

\*System Impact

Template:  Load

**Figure 2-6: Defect Generated from Failed Test Case**

After the defect has been submitted, the defect appears next to the failed test execution step as shown in Figure 2-7. Notice that rolling the mouse over the defect icon shows the title of the defect.



**Figure 2-7: Defect Associated with Failed Test Step**

At the conclusion of the test case execution, the execution results are presented to the tester. Any defects created during the execution of the test case appear in the “Defects” section of the test execution results. Notice in Figure 2-8 that when the mouse is rolled over the defect summary, full details of the main tab/page of the defect record is displayed and it is possible to view each tab/page without opening ClearQuest in a separate browser.

The screenshot displays the 'Execution Result' window for a manual test. The test ID is 163, and the actual result is 'Failed'. The test case is 'Preexisting Test Case #1'. A defect window is open, showing details for 'Defect RENT00000055'. The defect title is 'Failing Test Case "Preexisting Test Case #1"'. The defect type is 'Calculation problem', and the severity is 'Low'. The defect was identified on Monday, August 23, 2010, at 00:00:00 EDT and opened on the same day at 12:29:20 EDT. The defect description is 'Test Script: Preexisting Test #1' and 'Test Plan: Sample Test Plan'.

**Execution Result Details:**

- ID: 163
- Actual Result: Failed
- Host Name: Local Computer
- Owner: Mark Kaminsky
- Test Plan: Sample Test Plan
- Test Milestone: Iteration2
- Test Case: Preexisting Test Case #1
- Test Script: Preexisting Test #1
- Test Data: Unassigned
- Build: Unassigned
- Weight: 100
- Start Time: Aug 23, 2010 12:28:05 p.m.
- End Time: Aug 23, 2010 12:31:07 p.m.
- Total Run Time: 3 min 3 sec

**Defect RENT00000055 Details:**

- ID: RENT00000055
- State: Submitted
- Identified On: Mon Aug 23 00:00:00 EDT 2010
- Opened On: Mon Aug 23 12:29:20 EDT 2010
- Title: Failing Test Case "Preexisting Test Case #1"
- Full Name: Kaminsky, Mark
- Login ID: mkaminsk
- Phone: 703.626.5014
- Email: mkaminsky@trinity-software.com
- Defect Type: Calculation problem
- Severity: Low
- Priority:
- Detected in Phase: Other
- Target Release:
- Defect Closed Date:
- Defect Description: Test Script: Preexisting Test #1  
Test Plan: Sample Test Plan

**Figure 2-8: Defect Visible from Test Execution Result**

### 2.2.3 RequisitePro to ClearQuest

A ClearQuest CR can be linked to one or more requirements managed by RequisitePro from the “Requirements” tab on the CR form. Requirements are associated with a CR under one of the following circumstances:

1. A change request has been submitted against one or more requirements.
2. A change request has been written as a request for new functionality (enhancement) to the system under development. In this case, the details of the request are elaborated during a requirements effort and linked back to the CR for reference.

Figure 2-9 shows the “Requirements” tab from a CR. If this CR were linked to one or more requirements managed by RequisitePro, the requirements would appear in the “Associated Requirements” field.

View CR MSL0000349

Attachments Unified Change Management History Duplicate Admin  
Main Impact Analysis CCB Requirements Development Testing Notes

ID:  State:

Assigned Requirements Analyst

Name:  Login:

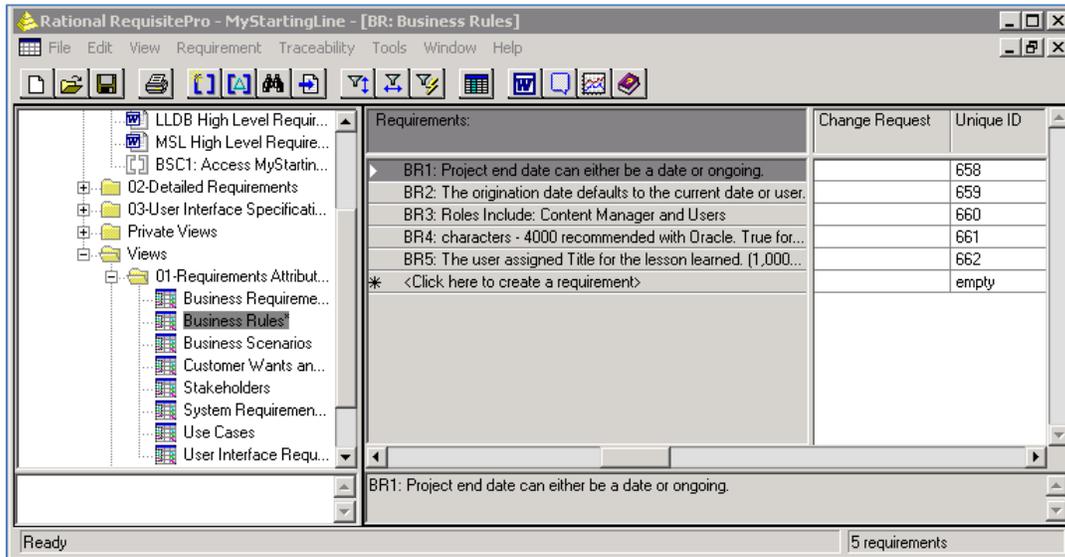
Associated Requirements: RAProject:

Tag	Name	Requirement	RAProjectName

Add from:

**Figure 2-9: Change Request – Requirements Tab**

The link between the CR and Requirements can also be viewed from RequisitePro. As shown in Figure 2-10, each requirement has an attribute labeled “Change Request”. If the requirements listed below were linked to CRs then the CR ID number would appear in the “Change Request” column.



**Figure 2-10: RequisitePro Requirements Linked to ClearQuest Records**

## 2.2.4 ClearQuest to ClearCase

Any change to a file controlled in one of the ClearCase UCM Component VOBs will require that either a CR or a ChildCR be associated with the checkout and checkin operations. That is, when a developer attempts to modify a file stored in a ClearCase Component VOB, s/he will be required to specify either a CR or a ChildCR stored in ClearQuest. Furthermore, that developer must have been assigned to work on the CR for the checkout/checkin operation to succeed.

Upon a successful checkin, the CR will be linked to the version of the file that was created. This approach ensures that all files changed in order to implement a CR are linked back to the CR. By cross referencing the files associated with each CR that was implemented for a given release, it is possible to identify all of the files that were created or changed in order to create that release. This is the standard approach to Activity Management used in the ClearCase UCM Model for software development described in the IBM Rational ClearCase Information Center: <https://publib.boulder.ibm.com/infocenter/cchelp/v7r1m0>.

Figure 2-11 shows a ClearQuest CR linked to two versioned elements. Notice that the CR shows not only what files and directories were changed while working on the CR, but also which specific versions of those files and directories were created.

View CR MSL00000049

Main | Impact Analysis | CCB | Requirements | Development | Testing | Notes  
Attachments | Unified Change Management | History | Duplicate | Admin

ID: MSL00000049 State: Development

UCM Project:  
MSL\_Project\_UAT\_2

View: Stream: mkaminsk\_MSL\_Project\_UAT\_2

Change Set:

Name	Versions
MSL\.	1
MSL\.\@\main\mkaminsk_MSL_Project_UA...	1, 2

OK  
Cancel  
Print Record  
Actions ▾

Figure 2-11: CR Change Set

## 2.3 Security Overview

Security protocols for each of the Rational tools are primarily based on a user's membership in one of the groups:

- CMTeam
- CMLead
- DevTeam
- DevLead
- FSA\_Mgmt
- OperationsLead
- OperationsTeam
- RequirementsLead
- RequirementsTeam
- SecurityTeam
- TestLead
- TestTeam

The Leads for each group will be members of both the “Lead” group and the “Team” group. For instance: If the user Bob is the CM Lead for the project WidgetDev, he will be a member of the group CM Lead and CM Team.

Rational Quality Manager, RequisitePro, and ClearQuest have built in group definitions which can be used to control access to artifacts managed by the tools. ClearCase security is based primarily on membership in active directory groups and custom triggers that specify which users have permission to perform specific actions.

The table below summarizes the operations in each Rational tool that are reserved for the system's Test Lead.

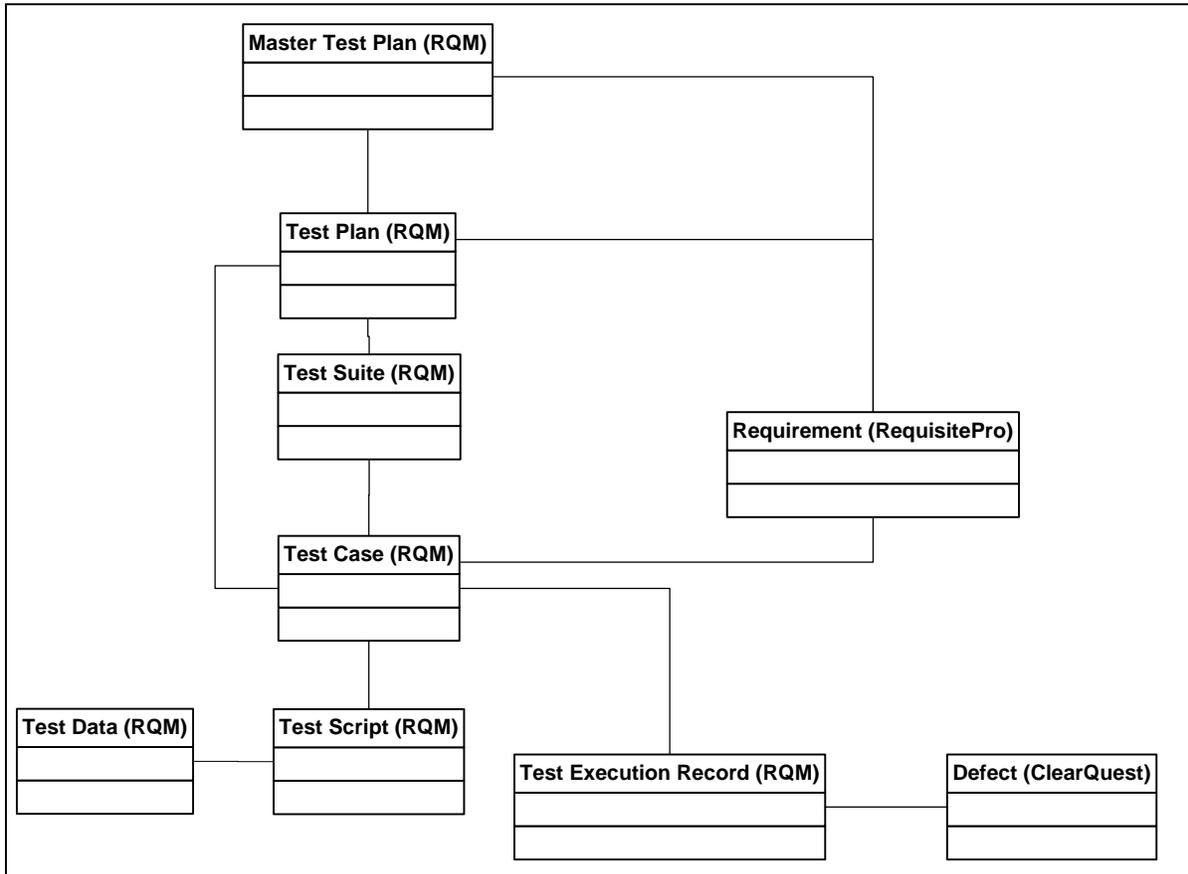
<b>Rational Tool/Component</b>	<b>Test Lead Permitted Operation</b>
RequisitePro	** No special permissions are reserved for the Test Lead.
Rational Quality Manager	Create Test Plans, Create Test Plan Schedules, Approve Test Cases.
ClearQuest/Defects	Cancel a defect.
ClearQuest/CRs	Reject a CR from UAT back to development.
ClearQuest/ChildCRs	** No special permissions are reserved for the Test Lead.
ClearCase	** No special permissions are reserved for the Test Lead.

**Table 2-2: Operations Reserved for the Test Lead**

Additional details on role/group-based permissions are described in Appendix D.

## Section 3. Test Management Overview

Test management entails the planning and execution of test cases based on a set of requirements as well as the creation of defects based on failed test cases. Figure 3-1 shows the relationships between components used to manage testing activities within the Rational tools.



**Figure 3-1: Test Management Artifact Relationships**

Table 3-1 shows who performs which actions required to construct a typical set of testing artifacts.

Person	Action
RSG Team	Creates a Rational environment (RequisitePro, ClearQuest, ClearCase, RQM) for the system under development based on the FSA enterprise solution.
CM Lead	Configures (RQM and ClearQuest) to support system specific-testing requirements.

<b>Person</b>	<b>Action</b>
Test Lead	Creates the MS Word-based Master Test Plan and Phase-specific test plans.
Test Lead	Creates RQM-based Master Test Plan and Phase-specific test plans and adds the corresponding MS Word-based plans as attachments.
Test Lead	Delegates the creation of test cases, test scripts and test suites to the testing team members by creating and assigning RQM work items. As test cases are created, the test lead associates the test cases with the test plans.
Test Team Member	For each assigned test case, the tester will create the RQM test case, and associated test script. During the creation of the test case, the tester may work with the test lead in order to create test data that can be used across multiple test cases.
Test Team Member	Construct test suites using existing test cases
Test Lead	Review the test cases and either approve or reject them
Test Lead	Create one or more test execution schedules in RQM based on test suites
Test Lead	Assign the test execution to testers using RQM work items.
Test Team Member	Execute test suites based on the assigned work items and related test execution schedule.
Test Team Member	Create defects from RQM when test execution steps fail during the execution of test suites/cases.

**Table 3-1: Steps to Create Testing Artifacts**

## Section 4. Test Planning

A Master Test Plan (MTP) will be developed by the Test Lead for most systems. Exceptions may be made based on criteria documented in the Enterprise FSA Enterprise Test Management Standards document. The MTP is created before testing begins as a high-level description of the planned testing effort and may be refined through the life of the system. For large, complex systems, individual Phase-Level test plans will supplement the MTP. For example, a System Test Plan and User Acceptance Test plan may be created to support the MTP. The MTP and related supplemental test plans must go through a formal FSA approval process.

### 4.1 Test Plans

Formal test plans will be written and delivered in Microsoft Word format. These test plans will be incorporated in system baselines (functional, design and product). RQM-based test plans will also be created in order to support linkage to other RQM-based assets such as test cases and test suites. Each formal MS Word-based test plan will be associated with a corresponding RQM test plan via the attachment field in the RQM test plan.

#### Test Plan Sections

Since the RQM test plan will, primarily, be used as a proxy for the “official” MS Word-based Test Plan, only the sections below will be visible:

- Summary
- Formal Review
- Requirements
- Test Schedules
- Test Cases
- Attachments

#### Test Plan: Summary

The test plan summary section is used to specify the name of the plan, give a brief description of the plan and to associate the plan with a system release and testing phase. This section also provides the mechanisms to duplicate, print, print to PDF, lock and manage snapshots for the test plan. Figure 4-1 shows the test plan summary section. Table 4-1 describes the fields in the test plan summary section.

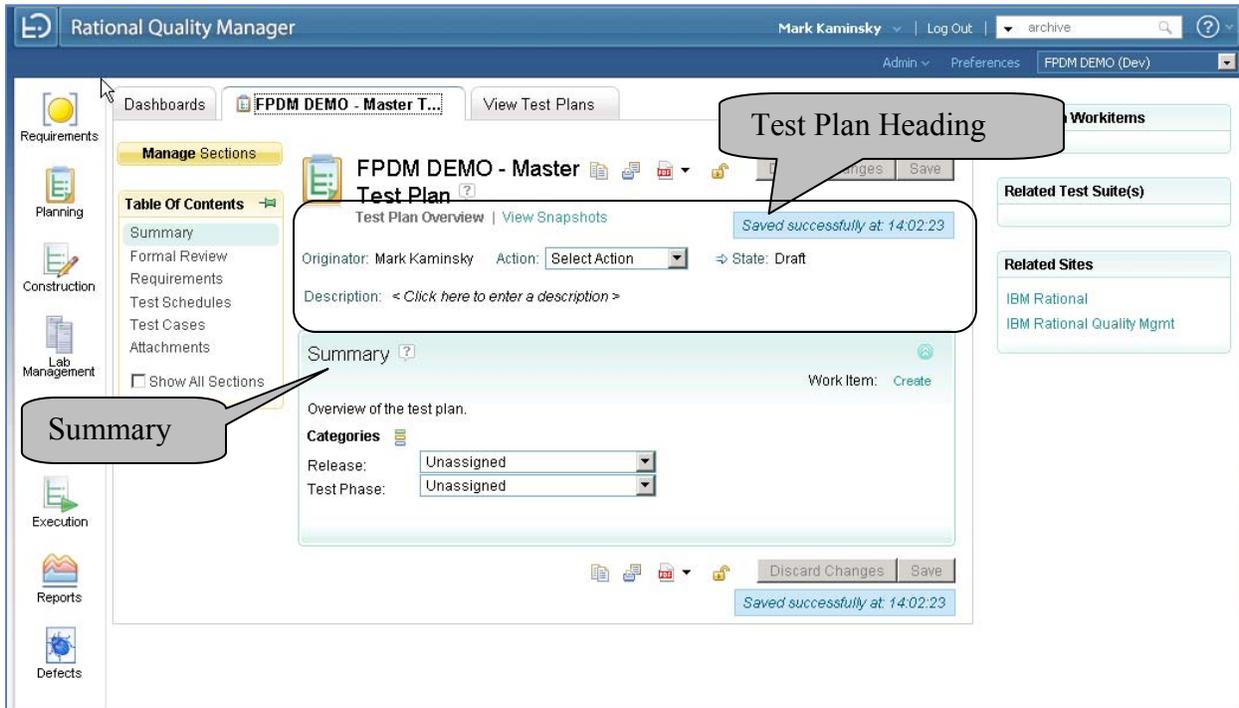


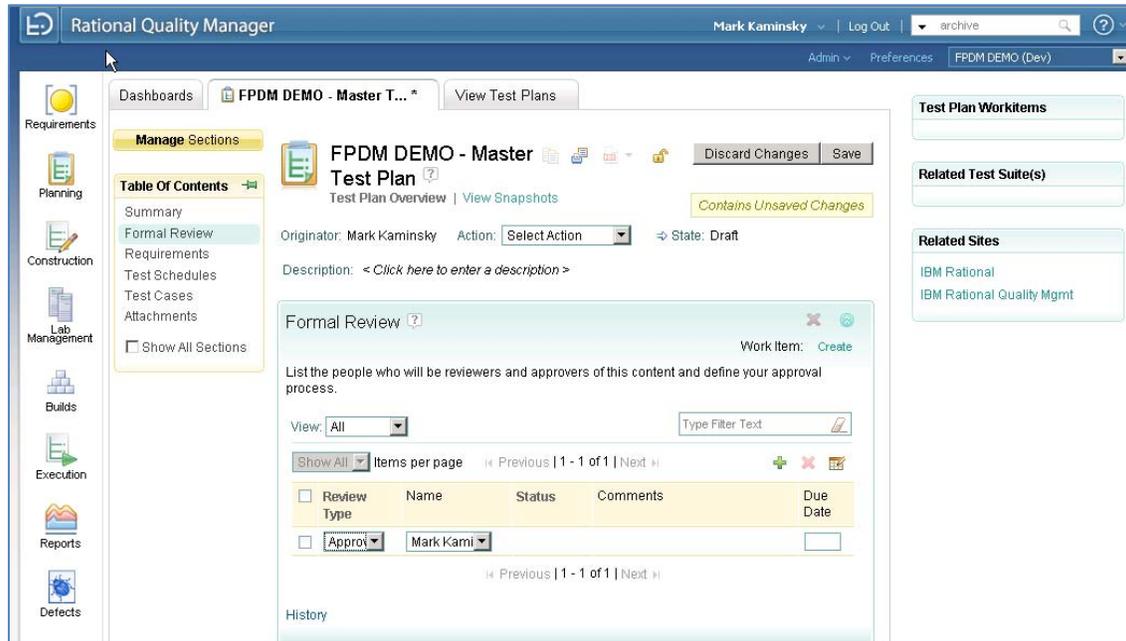
Figure 4-1: Test Plan Summary Section

Field	Description
Categories	<p>The test plan can be categorized in terms of the release that it is associated with and the phase of testing.</p> <ul style="list-style-type: none"> <li>Release: The release that the test plan is associated with. The CM Lead will update the choice list for releases as necessary. The choice list for Test Plan releases should be kept in synch with the choice list for Test Case releases and releases listed in ClearQuest.</li> <li>Test Phase: The phase of testing that this test plan addresses.</li> </ul>
Work Item	<p>The test lead may assign this section of the test plan or the entire test plan to another test team member by creating a work item in the summary section of the test plan.</p>

Table 4-1: Test Plan Summary Fields

### Test Plan: Formal Review

The formal review section of the test plan allows other team members to be assigned as either reviewers or approvers for the test plan. Typically, the test lead would populate this section in order to request that others, including FSA management, review the MS Word version of the test plan. Figure 4-2 shows a test plan Formal Review section. Table 4-2 describes the fields that are part of the test plan’s formal review section.



**Figure 4-2: Test Plan Formal Review Section**

Field	Description
Review Line Item	<p>Used to assign the task of reviewing a test plan to a team member.</p> <ul style="list-style-type: none"> <li>Review Type: Designates the type of reviewer.. Choices include “Reviewer” and “Approver”</li> <li>Name: The person performing the review</li> <li>Status: Status of the scheduled review. Choices include “Pending”, “Approved”, and “Rejected”.</li> <li>Comments: Free-form text comments entered by the reviewer</li> <li>Due Date: The date that the review should be completed by.</li> </ul>
Work Item	The test lead may assign this section of the test plan to another test team member by creating a work item in this section.

**Table 4-2: Test Plan Formal Review Fields**

### Test Plan: Requirements

The requirements section of a test plan is used to specify all of the requirements that are planned to be tested. Each requirement is a reference to a requirement stored in the System’s RequisitePro project. Figure 4-3 shows a test plan’s Requirements section. Table 4-3 describes the fields that are part of the section.

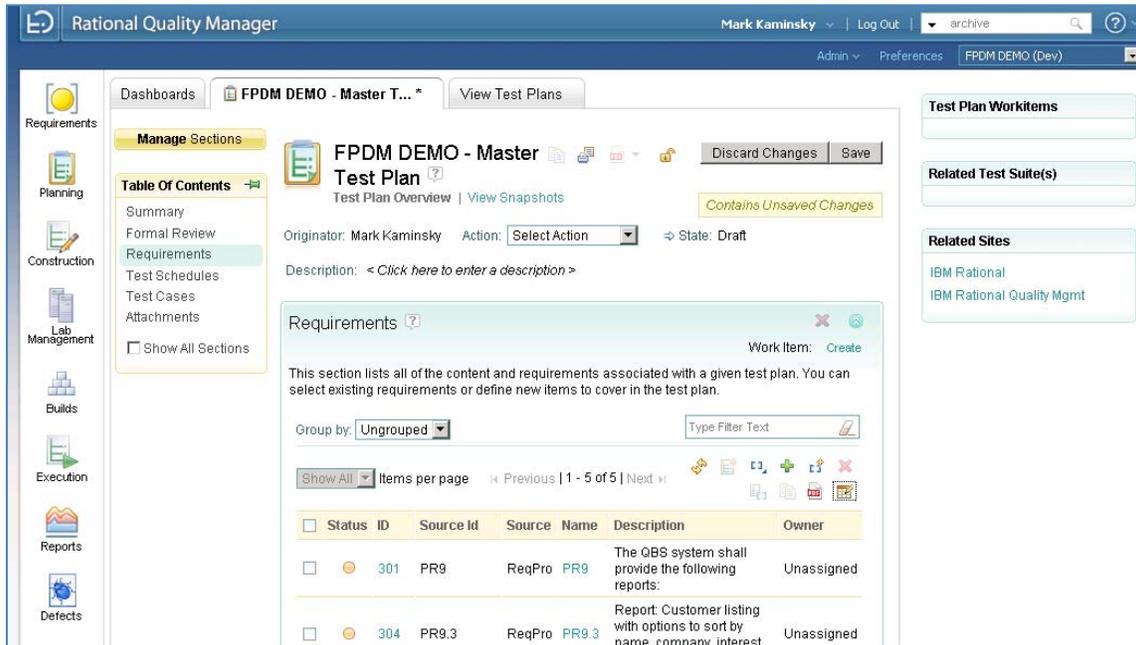


Figure 4-3: Test Plan Requirements Section

Field	Description
Requirement Line Item	<p>An individual requirement that is related to the test plan. All requirements listed in this section are references to requirements stored in RequisitePro.</p> <ul style="list-style-type: none"> <li>• <b>Status:</b> Indicates the status of the requirement in RequisitePro. Values include “Normal” and “Suspect”. The value “Suspect” indicates that the requirement has changed in RequisitePro and should be evaluated against the test plan</li> <li>• <b>ID:</b> The RQM identifier for the Requirement. ID should not be used as a reference to the requirement outside RQM. The “global” reference to the requirement is the requirement Name/Tag stored in RequisitePro.</li> <li>• <b>Source ID:</b> The requirement’s tag/ID managed by RequisitePro.</li> <li>• <b>Source:</b> Where the Requirement is managed. This should always be RequisitePro.</li> <li>• <b>Name:</b> The name of the requirement as managed by RequisitePro. This will be the same value as the short “Name” has been added to the requirement in RequisitePro, in which case the value will be the concatenation of the Source ID and the short name of the requirement. The “Name” field is a hyperlink to the requirement in RequisitePro. Clicking on this link will launch RequisitePro and open the Requirements editor for the selected requirement.</li> <li>• <b>Description:</b> The detailed description of the requirement from RequisitePro</li> <li>• <b>Owner:</b> The person responsible for the requirement from the perspective of RQM. Since all requirements are managed by</li> </ul>

Field	Description
	RequisitePro, this value should always be "Unassigned".
Work Item	The test lead may assign this section of the test plan to another test team member by creating a work item in this section.

**Table 4-3: Test Plan Requirements Fields**

### Test Plan: Test Schedules

The test schedules section of the test plan is used to schedule milestones related to the test plan. For example: A user acceptance test plan might be decomposed into multiple iterations where different test suites/cases are executed in each iteration. In this example, each iteration would be listed as a separate line item in the Test Schedule Table.

During the execution of a test case or test suite, the tester is prompted for both the test plan and the milestone that the test should be related to.

Figure 4-4 shows the test schedule section of a test plan and Table 4-4 describes the fields available in the test schedule section of a test plan.

The screenshot displays the Rational Quality Manager interface for a test plan titled "FPDM DEMO - Master Test Plan". The "Test Schedules" section is active, showing a table with the following data:

Name	Description	Planned Start Date	Planned End Date	Planned Duration	Points	Planned Defects	Actual Start Date	Actual End Date	Actual Duration
Iterati	Sample Iter	May 1,	May 8,	8 d	0	0			0 d

**Figure 4-4: Test Plan Test Schedules Section**

Field	Description
Test Schedule Line Item	Individual milestone associated with the test plan. <ul style="list-style-type: none"> <li>• Name: Short name for the milestone.</li> <li>• Description: Detailed description of the milestone.</li> <li>• Planned Start Date: Date that the testing should begin.</li> <li>• Planned end Date: Date that the testing should complete.</li> <li>• Planned Duration: Calculated based on the planned start and planned end dates.</li> <li>• Points: Calculated number of points based on the points associated with the included test cases.</li> <li>• Planned Defects: Number of defects anticipated for the testing effort.</li> <li>• Actual Start Date: The date that the testing actually started.</li> <li>• Actual End Date: The date that the testing actually completed.</li> <li>• Actual Duration: Calculated based on the actual start and actual end dates.</li> </ul>
Work Item	The test lead may assign this section of the test plan to another test team member by creating a work item in this section.

**Table 4-4: Test Plan Test Schedule Fields**

### Test Plan: Test Cases

The test cases section of the test plan lists all test cases that will be executed in order to test the requirements associated with the test plan. Figure 4-5 shows a test plan's Test Cases section. Table 4-5 describes the fields that are part of the section.

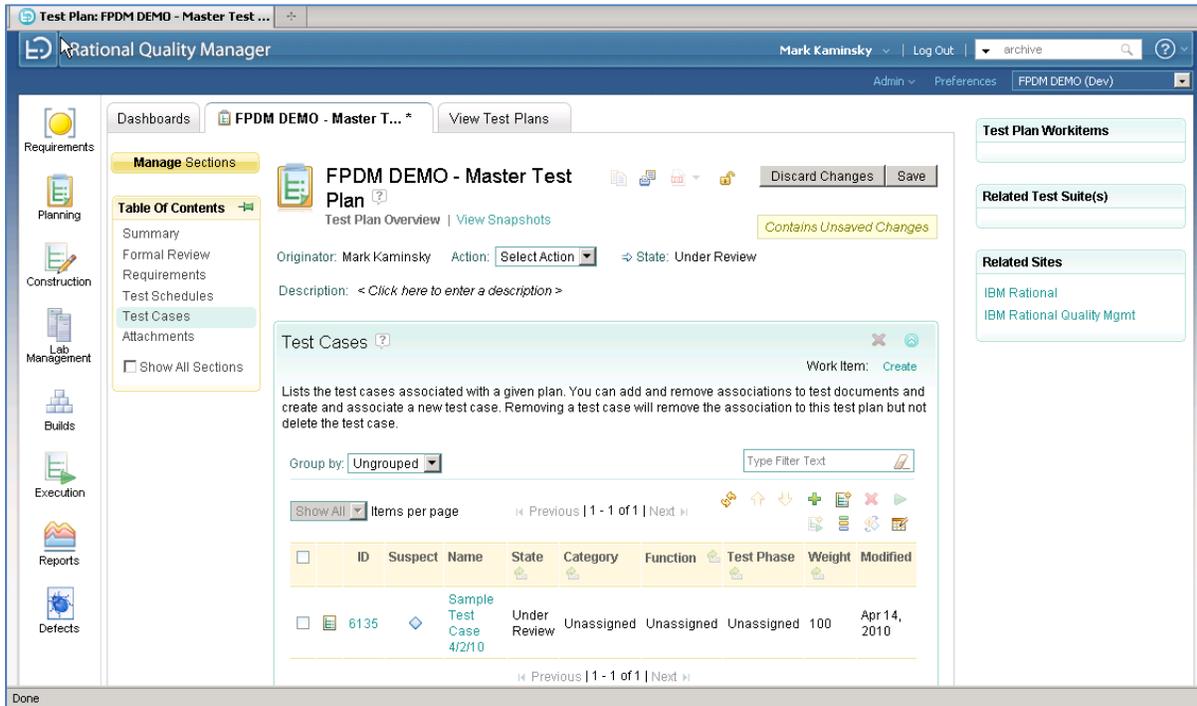


Figure 4-5: Test Plan Test Case Section

Field	Description
Test Case Line Item	<p>Each test case associated with the test plan will be displayed as a line item in this section.</p> <ul style="list-style-type: none"> <li>• ID: The RQM identifier for the test case.</li> <li>• Suspect: A test case flagged as “Suspect” indicates that one or more of the requirements associated with the test case are also suspect. The test case should be evaluated against the requirement. Values include: Not-Suspect and Suspect. This field becomes suspect when RQM determines that a related requirement is suspect. After evaluation of the test case, the tester must manually clear the suspect flag.</li> <li>• Name: Name of the test case</li> <li>• State: Current state of the test case. States include Draft, Under Review, Approved, and Retired.</li> <li>• Category: Generic category for the test case. The CM Lead populates the category list at the start of the project and updates the list as needed.</li> <li>• Function: Means of categorizing the test case based on the type of function being tested.</li> <li>• Test Phase: The phase of testing being performed. The default choice list includes the values “System”, “UAT”, and “Other” however the CM Lead may choose to update the choice list during the life of the</li> </ul>

Field	Description
	<p>system.</p> <ul style="list-style-type: none"> <li>Weight: A unit of measurement indicating the relative length of time that a test case might take to run. For example, a test case with the weight of 10 might take twice as long to run as a test case with a weight of 5. It is up to the test team to determine what unit of measurement they wish to use (points, hours, minutes, seconds, etc.)</li> <li>Modified: The date that the test case was last modified.</li> </ul>
Work Item	The test lead may assign this section of the test plan to another test team member by creating a work item in this section.

Table 4-5: Test Plan Test Case Fields

### Test Plan: Attachments

The formal MS Word-based test plans should be attached to the RQM test plan for reference. Figure 4-6 shows a test plan's Requirements section. Table 4-6 describes the fields that are part of the section.

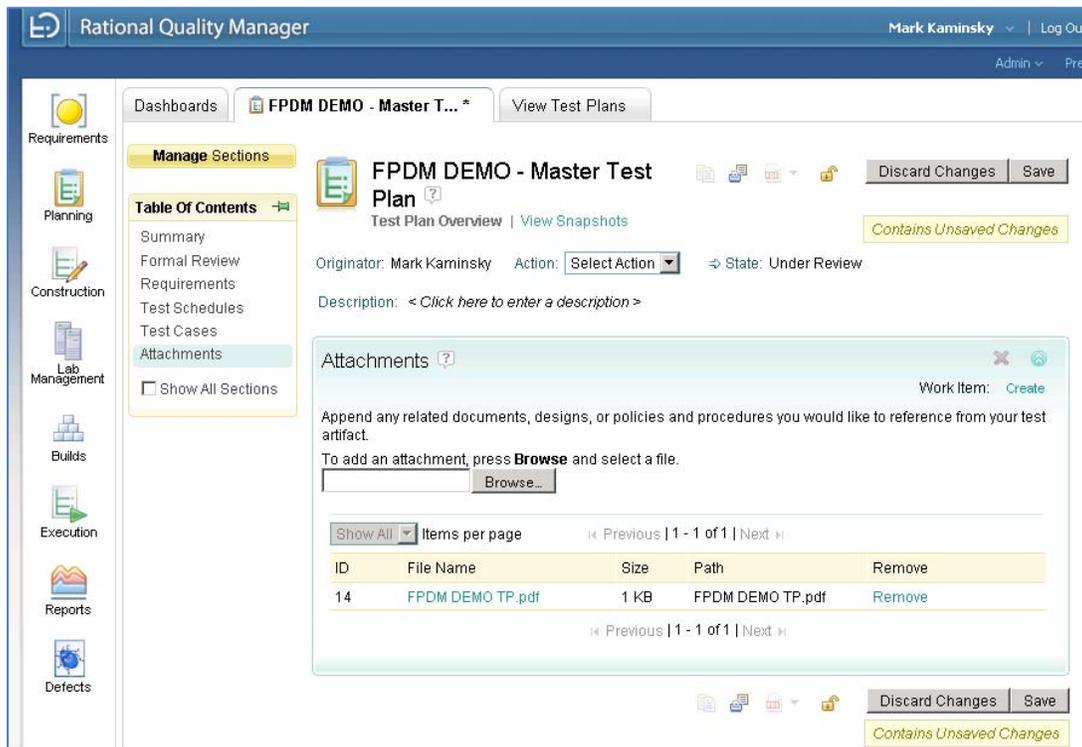


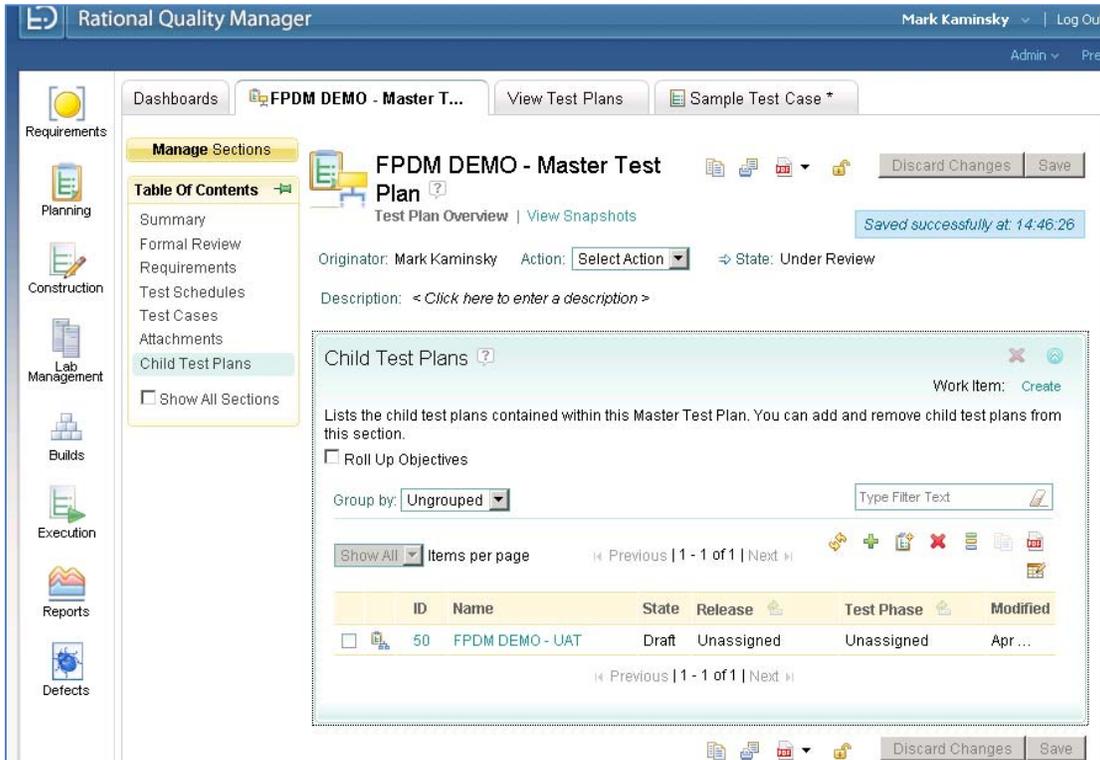
Figure 4-6: Test Plan Attachments Section

Field	Description
Attachment Line Item	<p>Each line item in the Attachments section represents a different file which has been attached/included in the test plan. For with each RQM test plan there should be a corresponding FSA test plan that is in MS Word format. It is this MS Word version of the test plan that will be delivered to FSA.</p> <ul style="list-style-type: none"> <li>• ID: The internal identifier for the attachment.</li> <li>• File Name: The file name of the RQM attachment. This field is a hyperlink to the attachment. Clicking on this link will download and open the attachment.</li> <li>• Size: The size of the attachment.</li> <li>• Path: The path to the attachment. NOTE: While entering the attachment in this section, the path will show up as the local path on the user's local workstation. After the test plan has been saved, the path will be listed as the file name.</li> <li>• Remove: Clicking on "Remove" will remove the attachment from the test plan.</li> </ul>
Work Item	The test lead may assign this section of the test plan to another test team member by creating a work item in this section.

**Table 4-6: Test Plan Attachments Fields**

### Test Plan: Child Test Plans

Child Test plans allow for the creation of a hierarchy of test plans. For large or complex systems, FSA policy requires that a master test plan (MTP) be created as well as a child test plan for each testing phase (System, UAT, etc.). Child Test plans support this policy. Figure 4-7 shows that child test plans can be associated with master test plans via the "Child Test Plans" section. Table 4-7 lists the characteristics of each child test plan that is associated with a master test plan.



**Figure 4-7: Test Plan Child Test Plans Section**

Group/Field	Description
Child Test Plan Line Item	<p>Each line item in this section refers to a child test plan. There is no specific limit to the number of child test plans which can be associated with a master test plan.</p> <ul style="list-style-type: none"> <li>• ID: RQM identifier for the child test plan.</li> <li>• Name: The name of the child test plan.</li> <li>• State: Current state of the test plan. States include Draft, Under Review, Approved, and Retired.</li> <li>• Release: The release of the system that the test plan is associated with.</li> <li>• Test Phase: The testing phase that the test plan is associated with.</li> <li>• Modified: The last date that the child test plan was modified.</li> </ul>
Work Item	The test lead may assign this section of the test plan to another test team member by creating a work item in this section.

**Table 4-7: Test Plan Child Test Plan Fields**

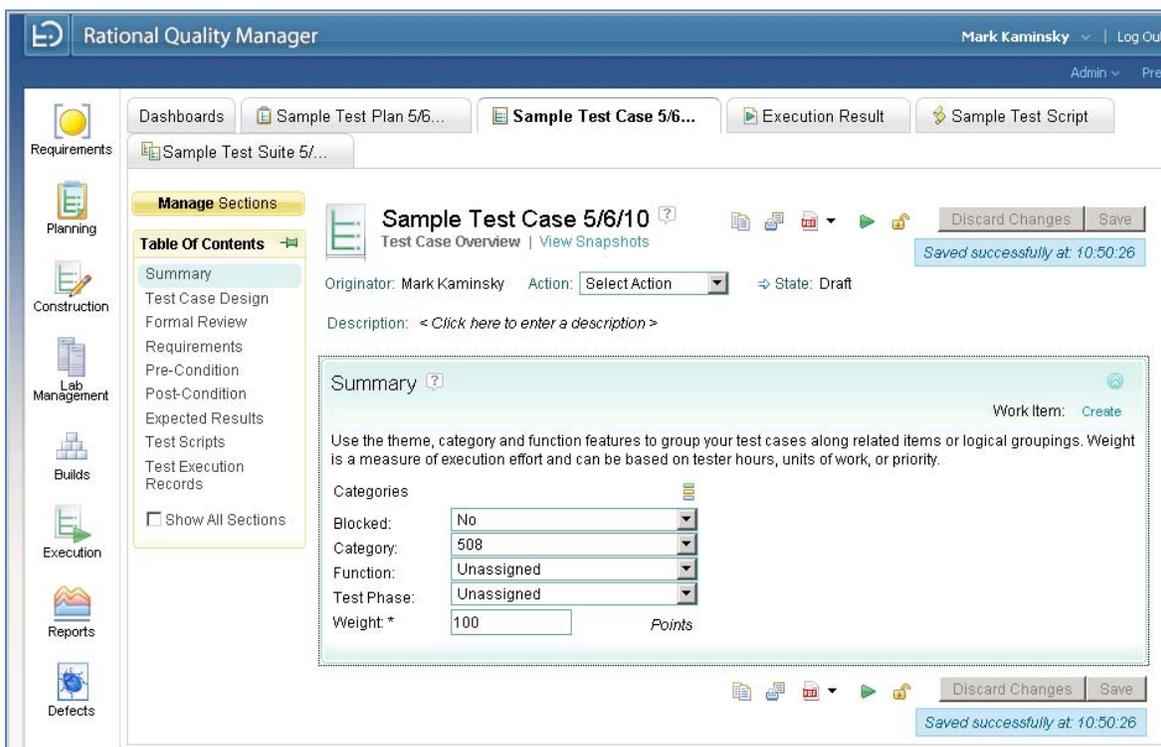
## 4.2 Test Cases

The following sections will be visible in all Test Cases:

- Summary
- Test Case Design
- Formal Review
- Requirements
- Pre-Conditions
- Post-Conditions
- Expected Results
- Test Scripts
- Test Execution Records

### Test Case: Summary

The test case summary section is used to specify the name, give a brief description, and to categorize the test case. This section also provides the mechanisms to duplicate, print, print to PDF, lock, and manage snapshots for the test plan. Figure 4-8 shows the test case summary section. Table 4-8 describes the fields in the test case summary section.



**Figure 4-8: Test Case Summary Section**

Display Name	Description
Categories: Category	<p>Generic means of categorizing the test case.</p> <ul style="list-style-type: none"> <li>• Category: A means of categorizing the test case based on a set of predefined categories. The CM Lead may choose to update the choice list for this field during the life of the system.</li> <li>• Function: Means of categorizing the test plan based on the type of function being tested</li> <li>• Test Phase: The phase of testing that the test case is associated with. If the test case relates to a single test phase and no other phases, then the test phase should be specified. If the test case relates to multiple testing phases, this value can be left unassigned.</li> <li>• Blocked: Indicates whether or not the test case is blocked by a defect.</li> </ul>
Work Item	<p>The test lead may assign this section of the test case or the entire test case to another test team member by creating a work item in the summary section of the test case.</p>

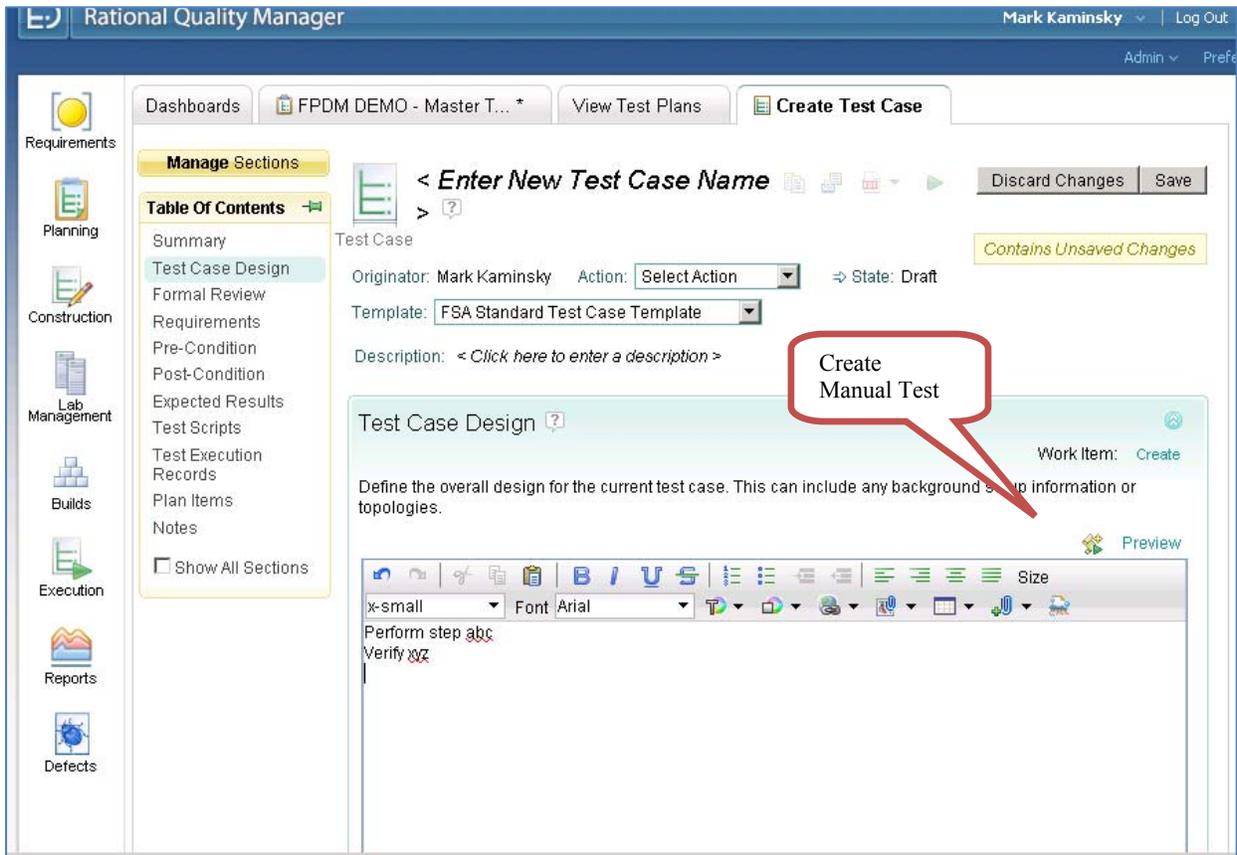
**Table 4-8: Test Case Summary Fields**

### **Test Case: Test Case Design**

The test case design section allows for the test team member to describe the steps required to execute the test case. Background and setup information may be included in this section as well as graphic images.

While the test case design is not sufficient to document the details of every step required to execute the test case, it can be used as a starting point for the creation of a manual test script by selecting the “Create Manual Test Script” icon just above the test case design editor.

Refer to the RQM online help for instructions on the creation of manual test scripts from the test case design.



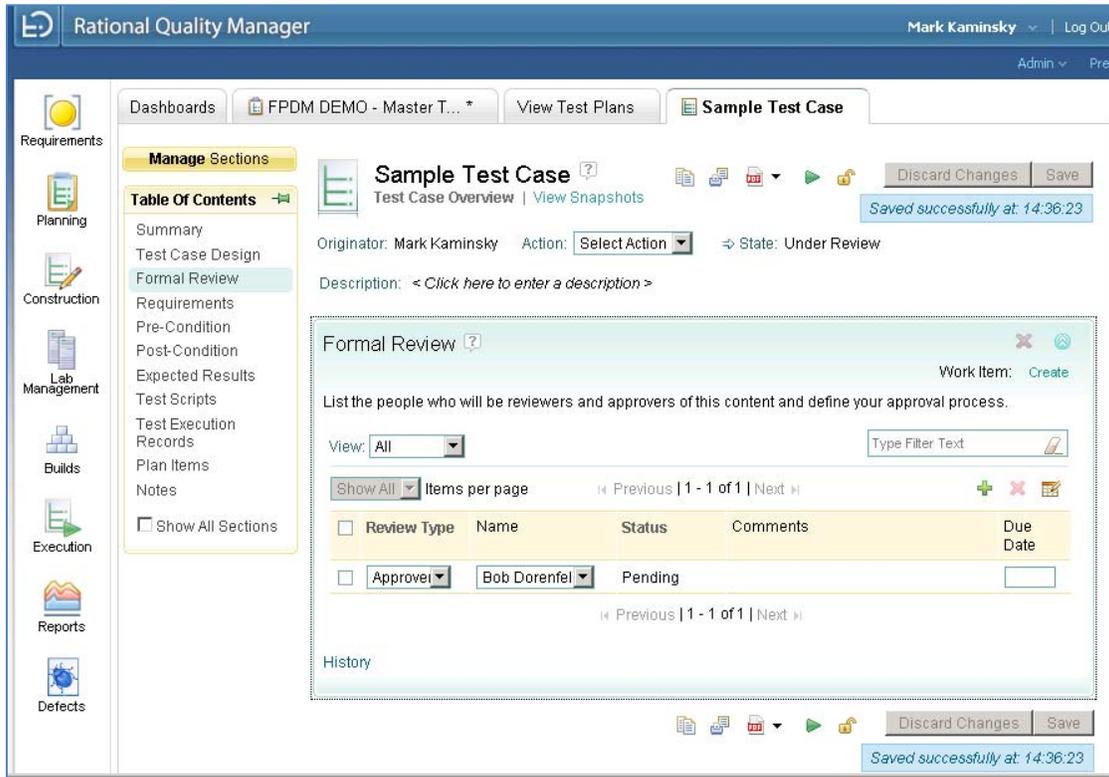
**Figure 4-9: Test Case Test Case Design Section**

Field	Description
Test Case Design Editor	A full-function editor used to document the test case design.
Work Item	The test lead may assign this section of the test case to another test team member by creating a work item in this section.

**Table 4-9: Test Case Test Case Design Field**

**Test Case: Formal Review**

The formal review section of the test case allows other team members to be assigned as either reviewers or approvers for the test case. Typically, the test lead would populate this section in order to assign other team members as reviewers of the test case.



**Figure 4-10: Test Case Formal Review Section**

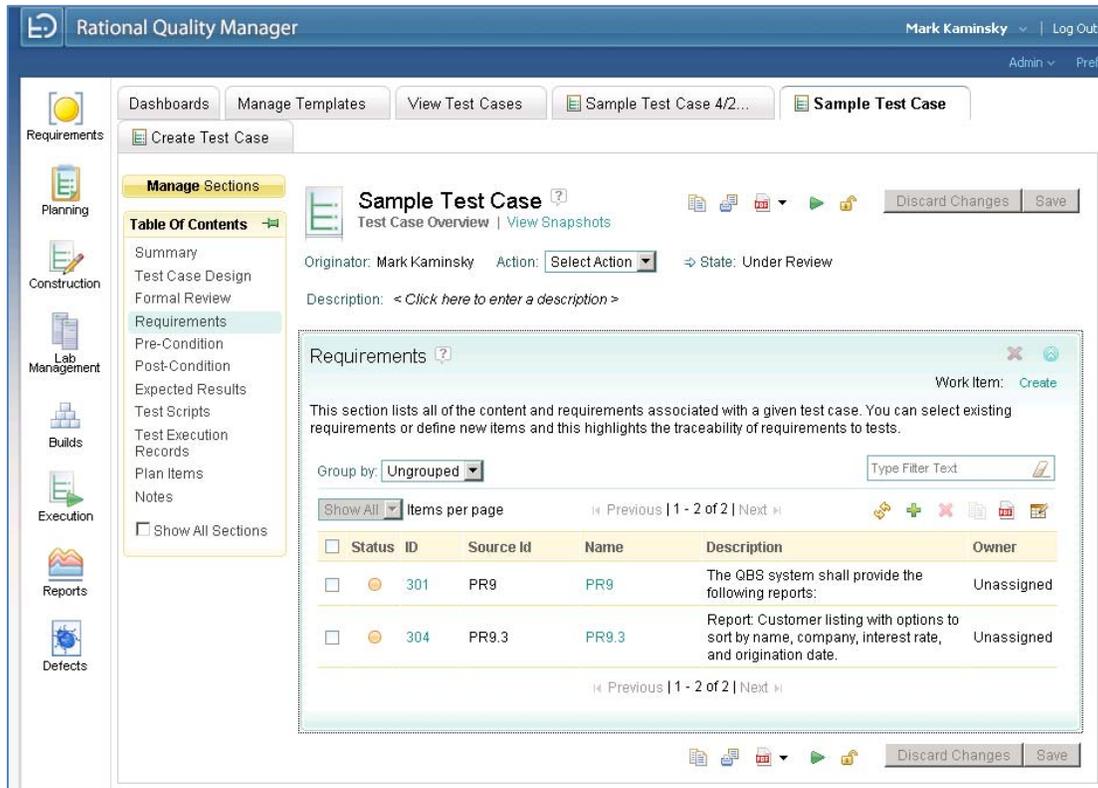
Field	Description
Review Line Item	<p>Used to assign the task of reviewing a test case to a team member.</p> <ul style="list-style-type: none"> <li>Review Type: Designates the type of reviewer. Choices include “Reviewer” and “Approver”</li> <li>Name: The person performing the review.</li> <li>Status: Status of the scheduled review. Choices include “Pending”, “Approved” and “Rejected”.</li> <li>Comments: Free-form text comments entered by the reviewer.</li> <li>Due Date: The date that the review should be completed.</li> </ul>
Work Item	The test lead may assign this section of the test case to another test team member by creating a work item in this section.

**Table 4-10: Test Case Formal Review Fields**

### Test Case: Requirements

The requirements section of a test case is used to specify all of the requirements that are being tested by the test case. The choice of requirements that can be associated with a test case is

limited to the set of requirements that have been associated with the test plan(s) that the test case is related to.



**Figure 4-11: Test Case Requirement Section**

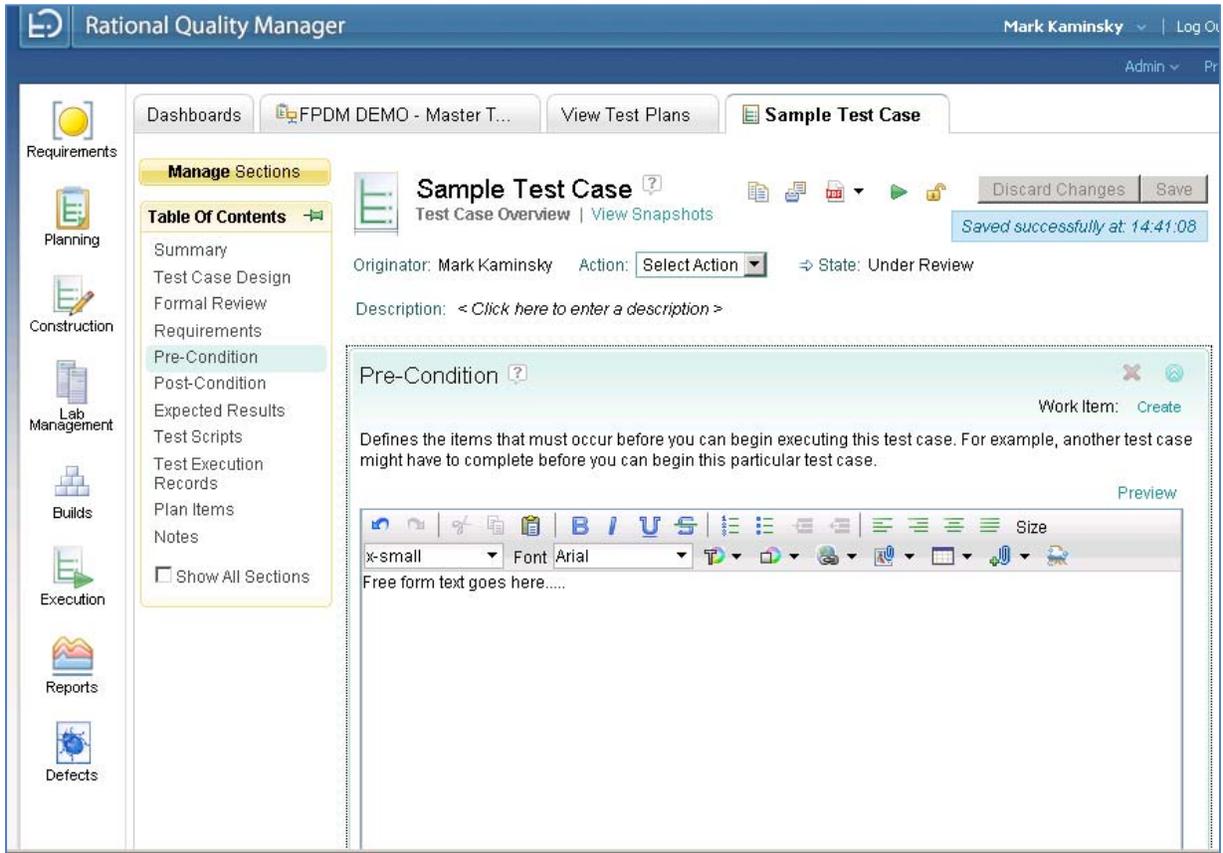
Field	Description
Requirement Line Item	<p>An individual requirement that is related to the test case. All requirements listed in this section are references to requirements stored in RequisitePro and referenced by the test plan that the test case is related to.</p> <ul style="list-style-type: none"> <li>• <b>Status:</b> Indicates the status of the requirement in RequisitePro. Values include “Normal” and “Suspect”. The value “Suspect” indicates that the requirement has changed in RequisitePro and should be evaluated against the test plan</li> <li>• <b>ID:</b> The RQM identifier for the Requirement. ID should not be used as a reference to the requirement outside RQM. The “global” reference to the requirement is the requirement Name/Tag stored in RequisitePro.</li> <li>• <b>Source ID:</b> The requirement’s tag/ID managed by RequisitePro.</li> <li>• <b>Source:</b> Where the Requirement is managed. This should always be RequisitePro.</li> <li>• <b>Name:</b> The name of the requirement as managed by RequisitePro. This will be the same value as the short ID unless a short “Name” has</li> </ul>

Field	Description
	<p>been added to the requirement in RequisitePro, in which case the value will be the concatenation of the Source ID and the short name of the requirement. The "Name" field is a hyperlink to the requirement in RequisitePro. Clicking on this link will launch RequisitePro and open the Requirements editor for the selected requirement.</p> <ul style="list-style-type: none"> <li>• Description: The detailed description of the requirement from RequisitePro</li> <li>• Owner: The person responsible for the requirement from the perspective of RQM. Since all requirements are managed by</li> </ul>
Work Item	The test lead may assign this section of the test case to another test team member by creating a work item in this section.

**Table 4-11: Test Case Requirements Fields**

### Test Case: Pre-Conditions

The pre-conditions section of the test case is used to document all pre-conditions for the execution of the test case. For example, another test case might need to execute before this test case can begin.



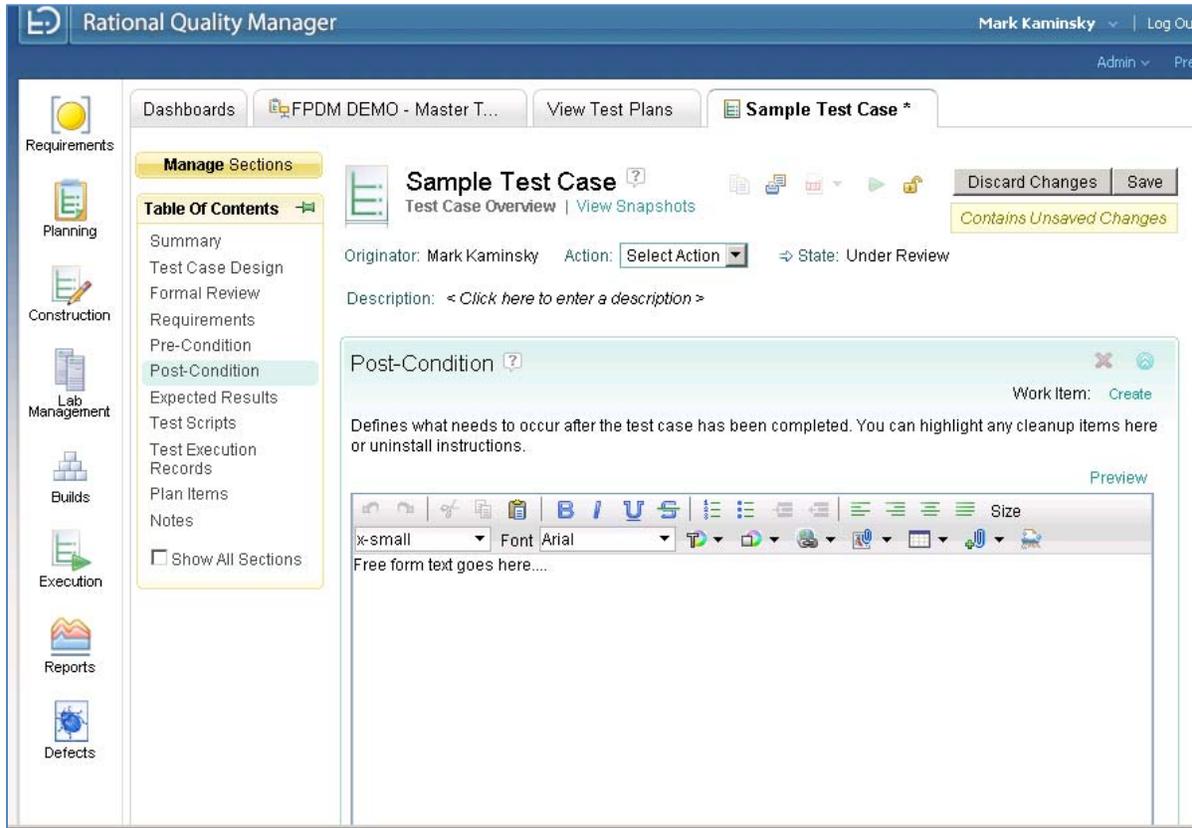
**Figure 4-12: Test Case Pre-Conditions Section**

Field	Description
Pre-Condition Editor	A full-function editor used to document the pre-conditions for the test case.
Work Item	The test lead may assign this section of the test case to another test team member by creating a work item in this section.

**Table 4-12: Test Case Pre-Conditions Fields**

**Test Case: Post-Conditions**

The post-conditions section of the test case is used to document everything that needs to be done at the conclusion of the test case.



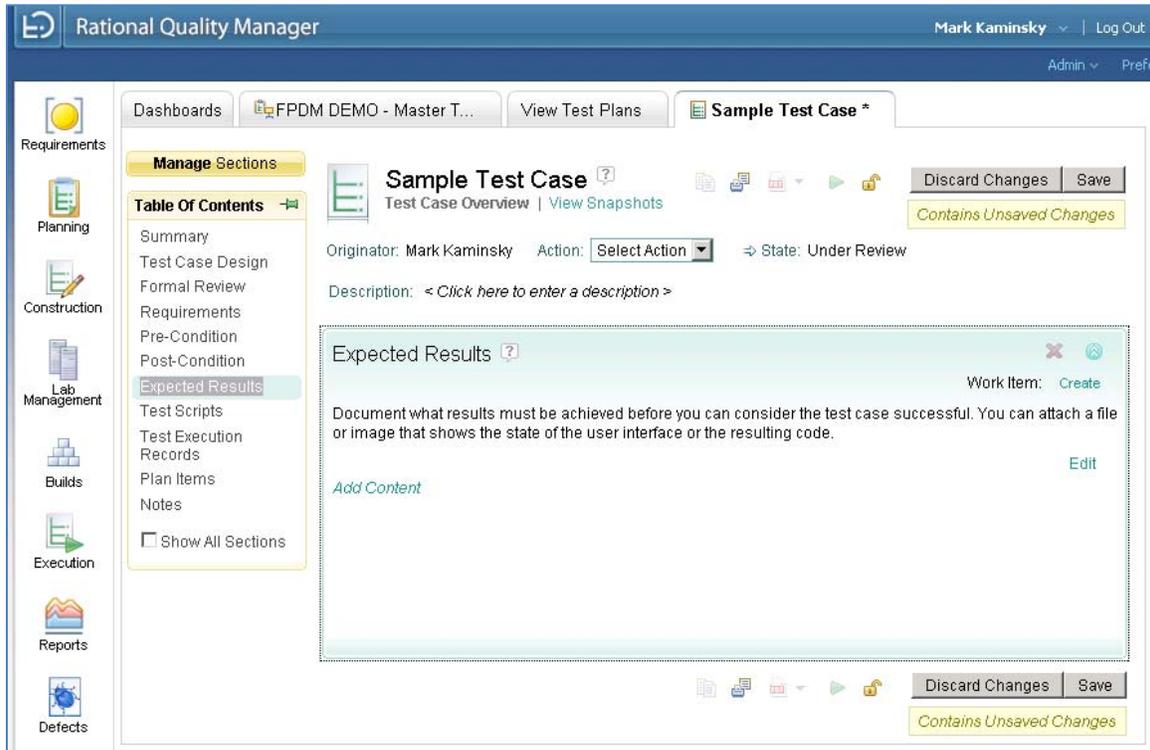
**Figure 4-13: Test Case-Post Conditions**

Field	Description
Post-Condition Editor	A full-function editor used to document the Post-conditions for the test case.
Work Item	The test lead may assign this section of the test case to another test team member by creating a work item in this section.

**Table 4-13: Test Case Post-Conditions Fields**

### Test Case: Expected Results

The expected results section of the test case is used to document the success criteria for the test case.



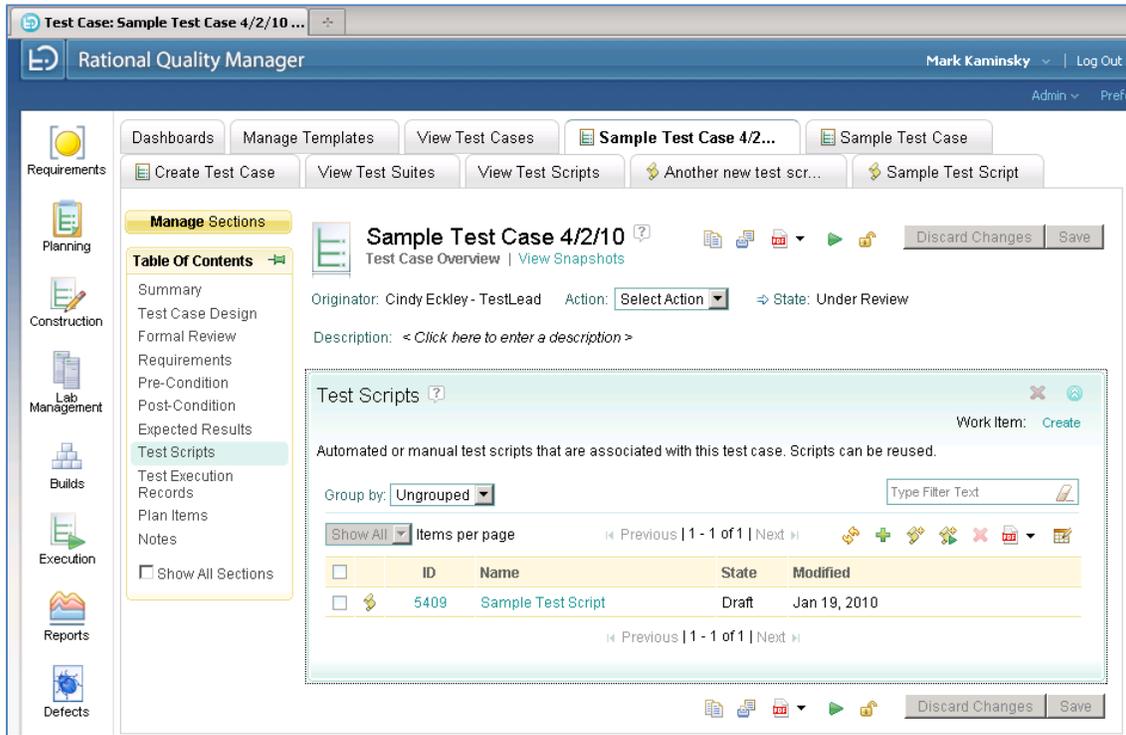
**Figure 4-14: Test Case Expected Results Section**

Field	Description
Expected Results Editor	A full-function editor used to document the expected results for the test case.
Work Item	The test lead may assign this section of the test case to another test team member by creating a work item in this section.

**Table 4-14: Test Case Expected Results Fields**

### Test Case: Test Scripts

The test scripts section of a test case is used to specify the test script(s) that are associated with the test case. The only type of test script which can currently be associated with a test case is a manual test script. While technically it is possible to associate multiple manual test scripts with a test case, in practice, only a single test script should be associated with the test case. For this reason, the test script should be named consistently with the test case.



**Figure 4-15: Test Case Test Scripts Section**

Field	Description
Test Scripts Line Item	<p>The test script that is associated with the test case.</p> <ul style="list-style-type: none"> <li>• ID: The RQM identifier for the test script.</li> <li>• Name: The name of the test script.</li> <li>• State: Current state of the test script. States include Draft, Under Review, Approved, and Retired.</li> <li>• Modified: The last date that the test script was modified.</li> </ul>
Work Item	<p>The test lead may assign this section of the test script to another test team member by creating a work item.</p>

**Table 4-15: Test Case Test Scripts Fields**

**Test Case: Test Results**

The test results section of a test case shows all of the test execution records associated with the execution of the test case. It is feasible that the same test case could have multiple test execution results based on different executions of the test case. For example, the same test case could have passed during system testing but failed during user acceptance testing. In this case, there would be two test execution records listed. Figure 4-16 shows a test case’s execution results section. Table 4-16 describes the fields that are part of the section.

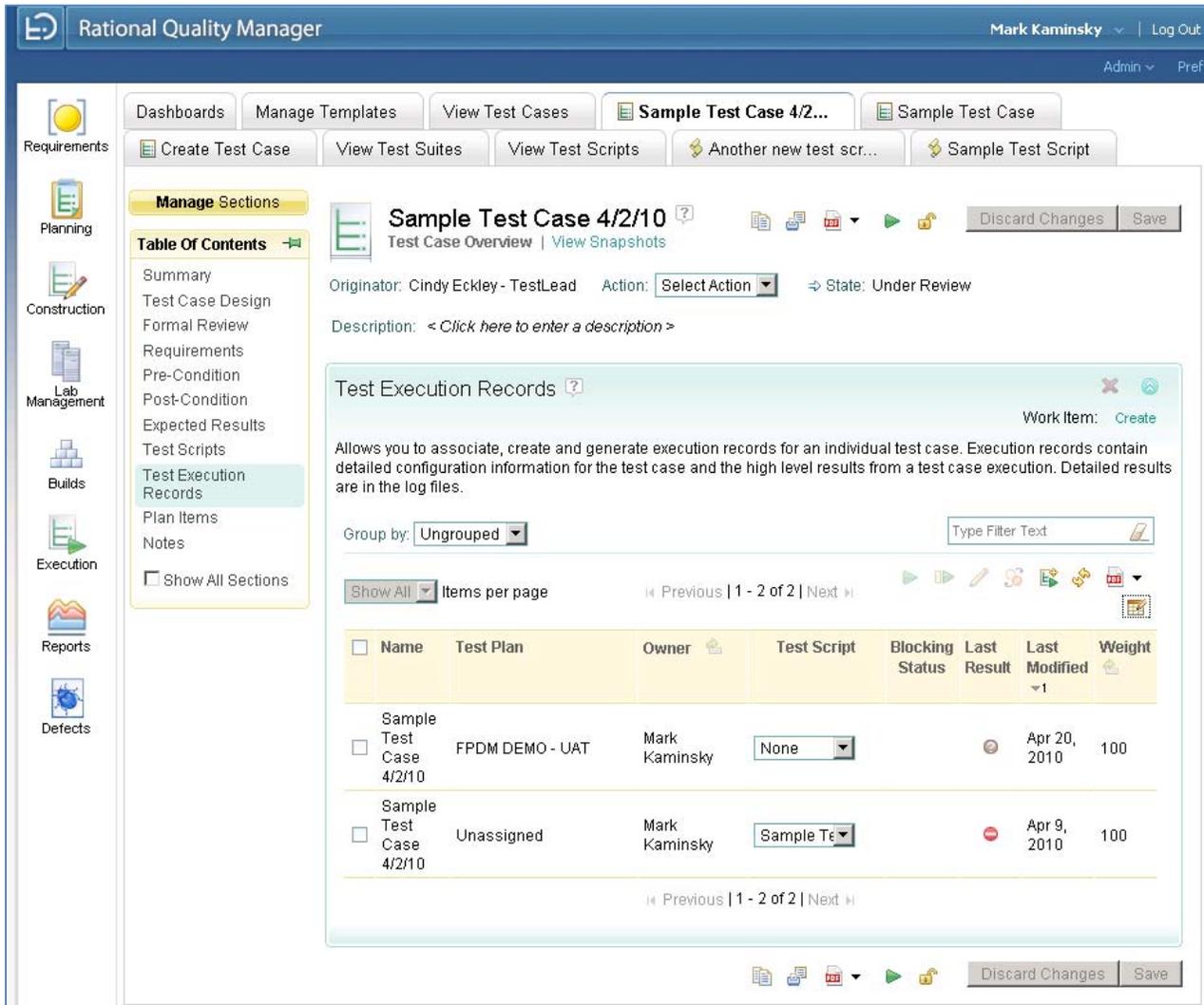


Figure 4-16: Test Case Test Results Section

Field	Description
Test Case Line Item	<p>Each line item represents an independent execution of the test case.</p> <ul style="list-style-type: none"> <li>• Name: Name of the test execution record. The name of the test execution record is typically the name of the test case concatenated with the name of the environment that the test was executed in. If no environment was specified then the test execution record name is the same as the test case name.</li> <li>• Test Plan: The name of the test plan that was selected when the test case was executed.</li> <li>• Owner: The person that executed the test case and, therefore, created the test execution record.</li> </ul>

Field	Description
	<ul style="list-style-type: none"> <li>• Test Script: The name of the test script that was executed. Since every execution of a test case should use the same test script, all test execution records for a given test case should reference the same test script.</li> <li>• Blocking Status: n/a: Blocking status will always be blank because ClearQuest is used to track defects associated with test executions. The integration between RQM and ClearQuest does not provide the ability to directly track blocking status.</li> <li>• Last Result: Indices the result of running the test case. Valid results include Passed, Failed, Incomplete, Inconclusive, Partially Blocked, Error, Blocked, PermFailed, and Deferred</li> <li>• Last Modified: The date that the test execution record was last modified.</li> <li>• Weight: The weight that was associated with the test execution. The default value for the weight of the test execution is the weight of the test case. This value can be changed if, during testing, it the test case takes more or less time than anticipated.</li> </ul>
Name	Name of the test execution record. The name of the test execution record is typically the name of the test case concatenated with the name of the environment that the test was executed in. If no environment was specified then the test execution record name is the same as the test case name.
Test Plan	The name of the test plan that was selected when the test case was executed.
Owner	The person that executed the test case and, therefore, created the test execution record.
Test Script	The name of the test script that was executed. Since every execution of a test case should use the same test script, all test execution records for a given test case should reference the same test script.
Blocking Status	n/a: Blocking status will always be blank because ClearQuest is used to track defects associated with test executions. The integration between RQM and ClearQuest does not provide the ability to directly track blocking status.
Last Result	Indicates the result of running the test case. Valid results include Passed, Failed, Incomplete, Inconclusive, Partially Blocked, Error, Blocked, PermFailed, and Deferred.
Last Modified	The date that the test execution record was last modified.
Weight	The weight that was associated with the test execution. The default value for the weight of the test execution is the weight of the test case. This value can be changed if, during testing, it the test case takes more or less time than anticipated.
Work Item	The test lead may assign this section of the test case to another test team member by creating a work item in this section.

**Table 4-16: Test Case Test Results Fields**

### 4.3 Test Scripts

Currently the only types of test script supported by RQM within FSA are manual test scripts. Each test script is typically composed of multiple test steps.

Each step required to execute the test case should be documented completely in the “Manual Steps” section of the test script editor as should the expected results for the test script steps.

Each step can be classified as either an execution step or a reporting step. An execution step instructs the tester to perform some action when running the script. A reporting step is a high-level checkpoint that might summarize the results of multiple executions steps. For example, a reporting step might ask the question: “Were you able to login?”

The test script can be associated with existing test data as described earlier. Images, files, comments, and verification text may also be attached to manual test steps.

Manual test scripts are created and maintained using the test script editor as shown in Figure 4-17. Table 4-17 describes the components of a test script. MS Word-based and MS Excel-based scripts are not supported. However both of the formats may be imported into RQM using the standalone utility: RQMExcelWordImporter. A detailed description of this utility as well as the download package is available on site <https://jazz.net>. After logging into the Jazz site, the utility can be found by searching for RQMExcelWordImporter.

NOTE: You will be required to login to the Jazz site. If you do not have a login id, you will need to register to receive one.

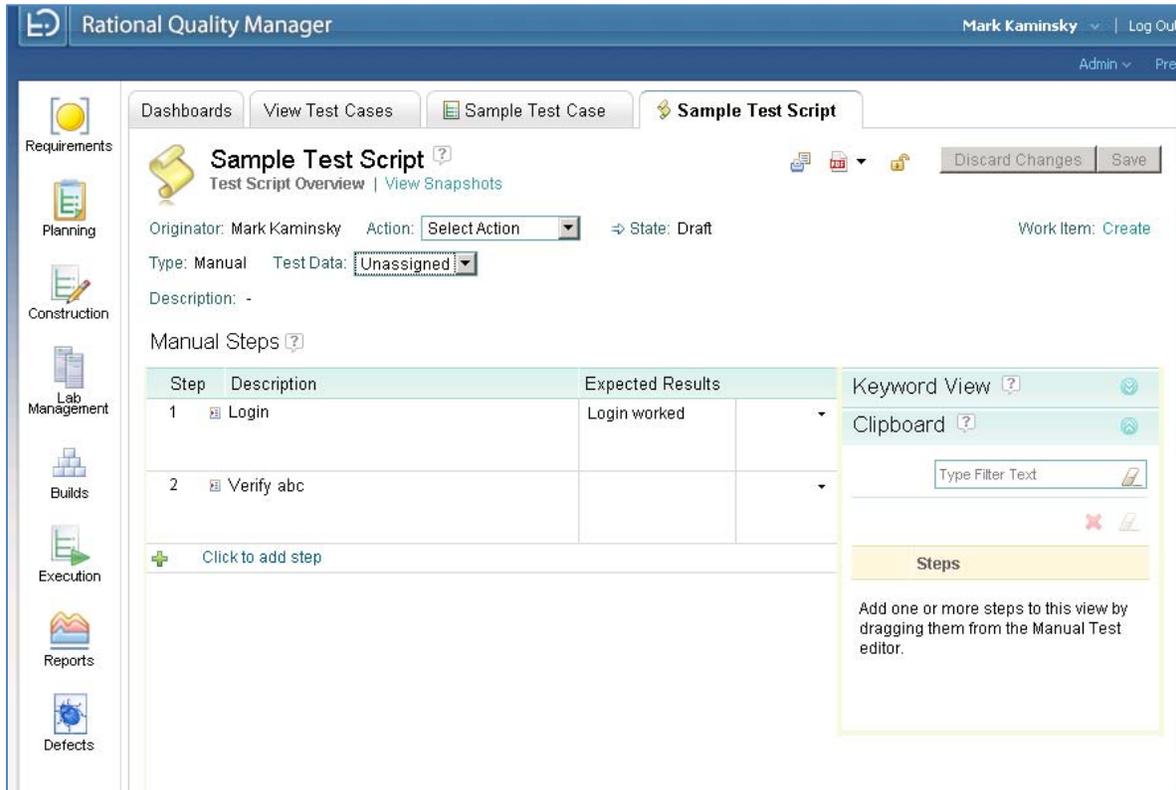


Figure 4-17: Test Script Editor

Field	Description
Manual Step Line Item	<p>The step (either reporting or execution) that is to be executed.</p> <ul style="list-style-type: none"> <li>Step: The identifier for the test step. Each test step is assigned a sequential number based on the location of the test step in the list of test steps that make up the script.</li> <li>Description: Describes what action should be taken by the tester. This field may contain text, bulleted lists, images, and test data variables.</li> <li>Expected Results: Describes the result that would lead to the successful completion of the test step.. This field may contain text, bulleted lists, images, and test data variables.</li> </ul>
Work Item	The test lead may assign this section of the test script to another test team member by creating a work item.

Table 4-17: Test Script Editor

## 4.4 Test Suites

Test Suites are used to relate a series of test cases that should be executed together. A Test suite is composed of the following sections

- Summary
- Test Suite Design
- Test Cases

### Test Suite: Summary

The test suite summary section is used to specify the name, give a brief description, and to categorize the test suite. This section also provides the mechanisms to save the test suite to PDF format and to execute the test suite. Figure 4-18 shows the test case summary section. Table 4-18 describes the fields in the test suite summary section.

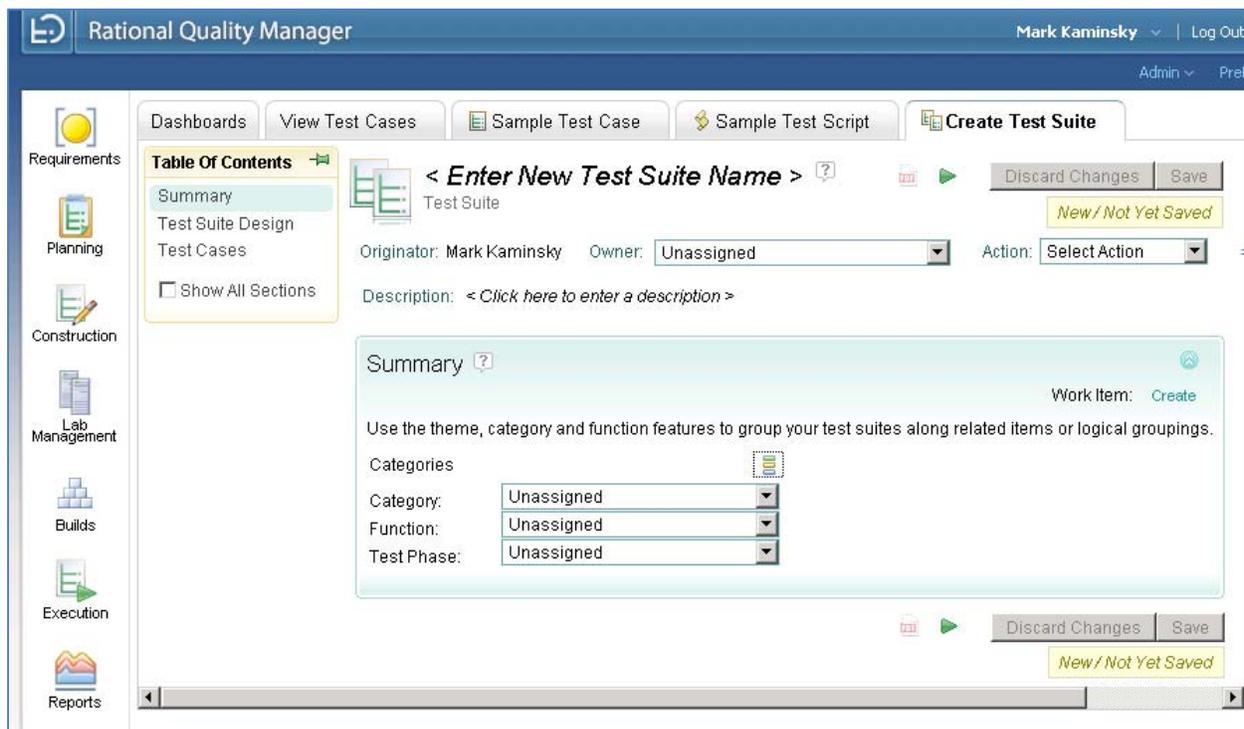


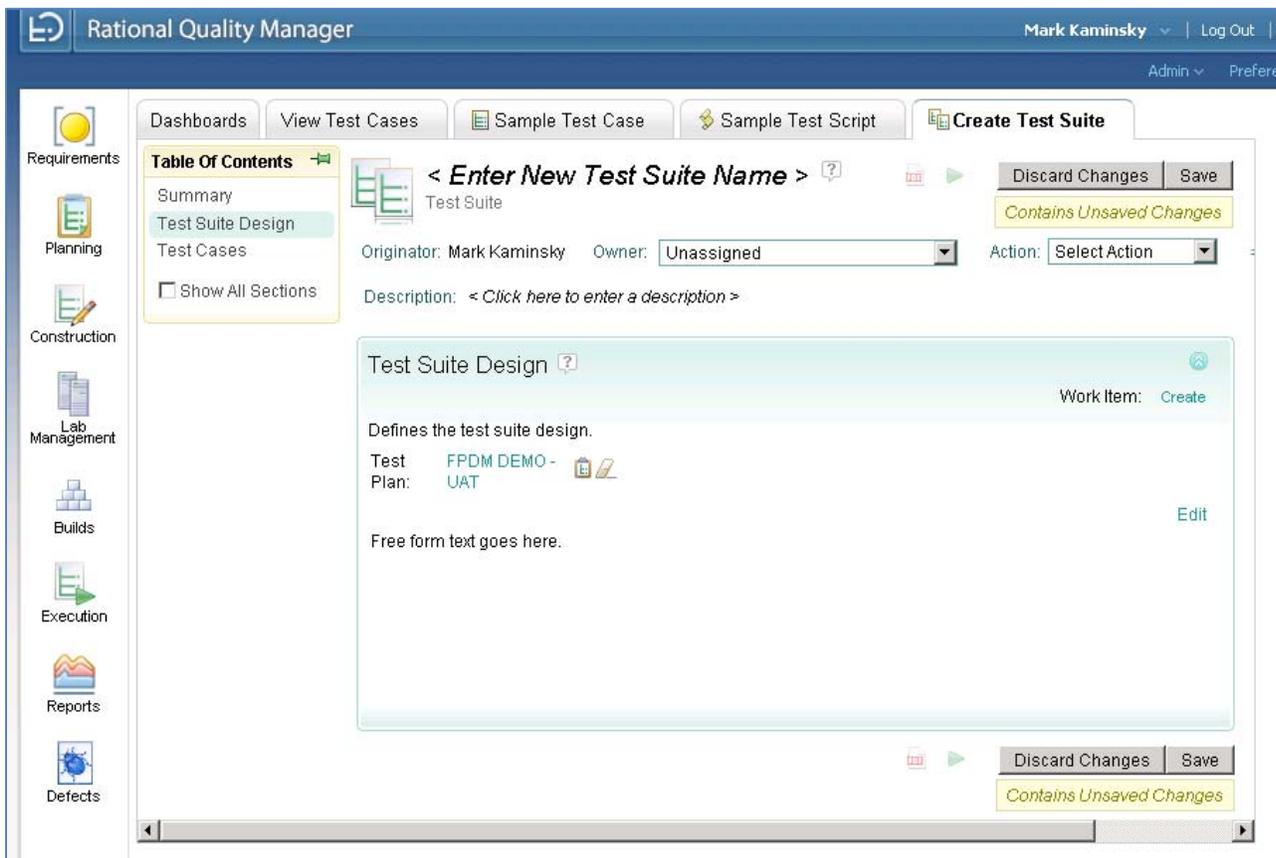
Figure 4-18: Test Suite Summary Section

Display Name	Description
Category	Generic means of categorizing the test suite. The CM Lead will maintain the choice list for this field.
Function	Means of categorizing the test suite based on the type of function being tested. The CM Lead will maintain the choice list for this field.
Test Phase	The test phase to which the test case applies. The CM Lead will maintain the choice list for this field.

**Table 4-18: Test Suite Summary Fields**

### Test Suite: Test Suite Design

The test suite design section of a test suite is used to specify which test plan the test suite is associated with. Unlike a test case, a test suite can only be associated with a single test plan. This section is also used to document the details of the test suite design in a full function editor.



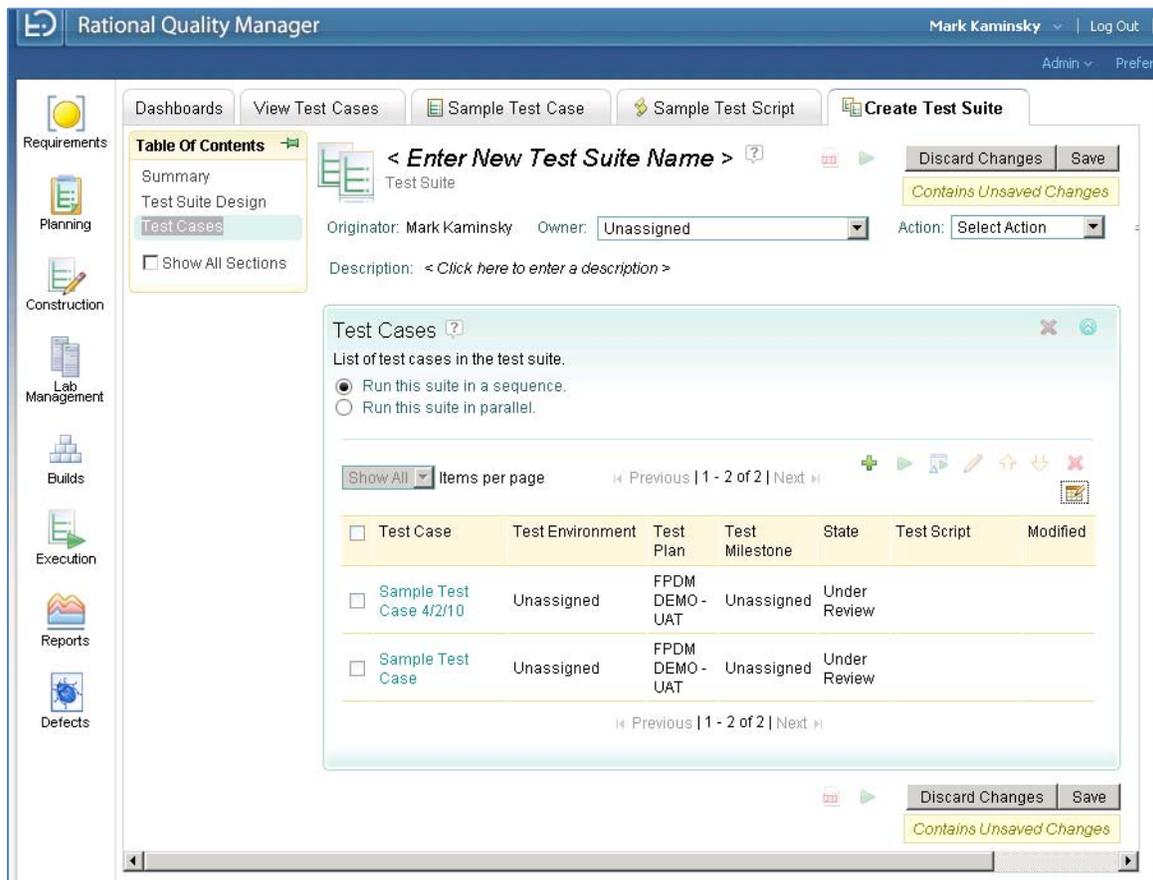
**Figure 4-19: Test Suite Test Suite Design Section**

Field	Description
Test Plan	Name of the test plan that the test suite is associated with.
Free Form Text	A detailed description of the test suite design.
Work Item	The test lead may assign this section of the test suite to another test team member by creating a work item in this section.

**Table 4-19: Test Suite Design Fields**

### Test Suite: Test Cases

The test cases section of a test suite is where the ordered set of test cases that make up the test suite is managed. For each test case listed in the test suite, the test environment, test plan, and test milestone can be specified.



**Figure 4-20: Test Suite Test Cases Section**

Field	Description
Run this suite in a sequence/in parallel	Indicates whether the test cases in the suite should be run in the order that they appear or if the tester should be allowed to run them in any order.
Test Case Line Item	<p>Each test case associated with the test plan will be displayed as a line item in this section.</p> <ul style="list-style-type: none"> <li>• ID: The RQM identifier for the test case.</li> <li>• Name: Name of the test case</li> <li>• Test Environment: The test environment that is assigned to the test case when being executed in this suite.</li> <li>• Test Plan: The Test Plan that the Test Case is associated with when being executed as part of the test suite.</li> <li>• Test Milestone: The test milestone for the test plan that the test case is associated with when being executed as part of the test suite.</li> <li>• State: Current state of the test case. States include Draft, Under Review, Approved, and Retired.</li> <li>• Modified: The date that the test case was last modified.</li> </ul>

**Table 4-20: Test Suite Test Cases**

## 4.5 Test Data

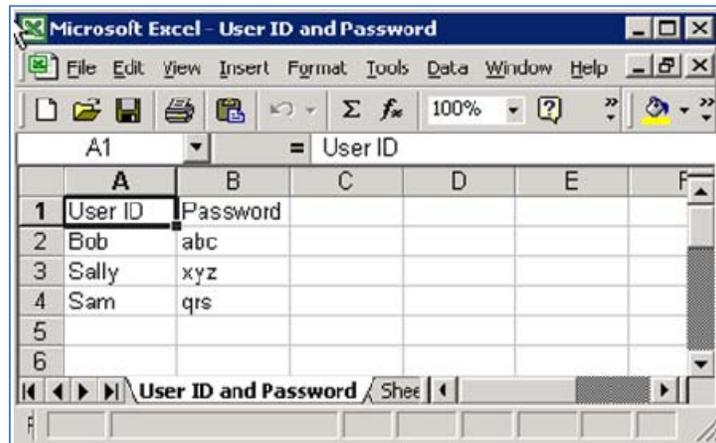
Test data is a collection of related data records that can be used to supply data values to variables in manual test scripts. For example, if a test script requires the entry of a user ID and password during execution, then test data can be created and pre-populated with multiple rows of user ID and password combinations. Test scripts can be written such that variables, rather than actual data, are contained within the script. Then, during execution of the script, the variables would be automatically replaced with the corresponding test data.

Test data is created by importing from a comma-separated file (spreadsheet). To use the test data in a test script, the tester would first associate the test data with the manual test script. Once the test data has been associated, the test data variables can be inserted into test steps within the script.

During test execution, the variables in the manual test script are replaced with the actual data from the imported CSV file. This behavior allows the data to be updated without modification to the test script.

### Creating Test Data

Before test data can be used in a test script, it must be created. Start by creating a spreadsheet that lists the rows and columns that correspond to the data that will be included in test scripts. Figure 4-21 shows a sample CSV file that could be used to supply a username and associated password to test scripts.

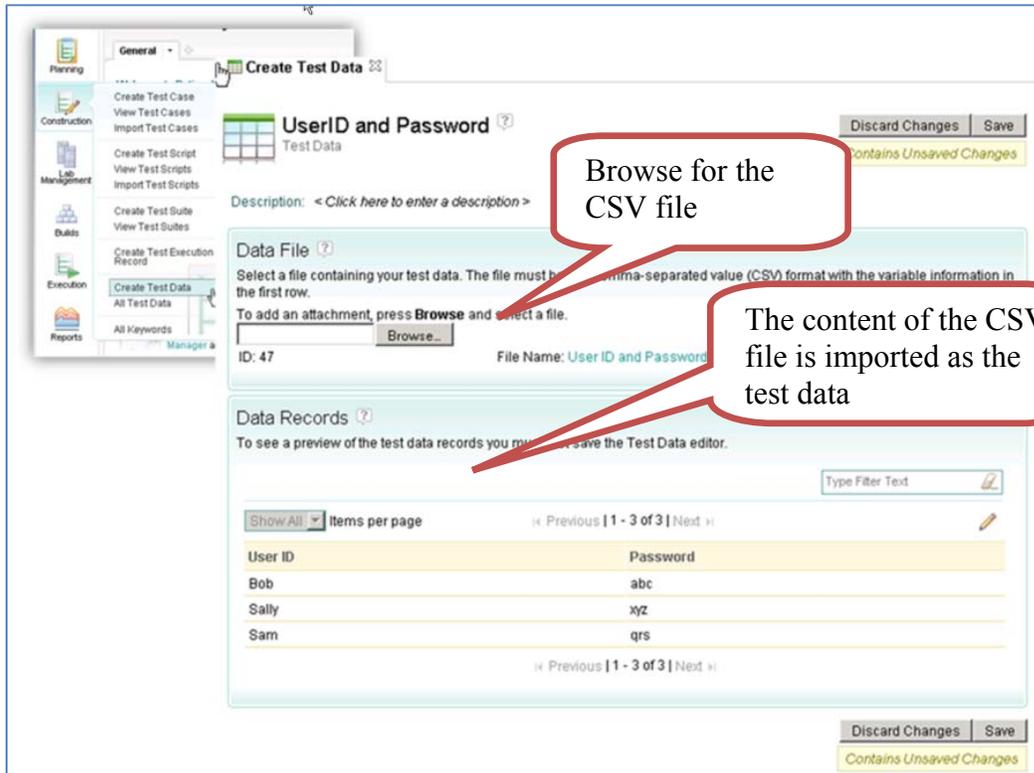


	A	B	C	D	E	F
1	User ID	Password				
2	Bob	abc				
3	Sally	xyz				
4	Sam	qrs				
5						
6						

**Figure 4-21: Sample Test Data CSV File**

After the CSV file has been created, create the test data and associate the CSV file to it. Figure 4-22 shows how a test lead could create test data. The steps to create test data are as follows:

1. Roll the mouse over “Construction=>Create Test Data”.
2. Enter the Name of the Test Data.
3. Browse for the CSV file that was previously created.
4. Verify that the rows and columns of the CSV file were imported correctly.
5. Save the Test Data.



**Figure 4-22: Creating Test Data**

Test data can be inserted into a test script by first selecting the appropriate test data from the “Test Data” choice list and then by using the test data icon to insert test data into the script as shown in Figure 4-23.

The screenshot shows a test script editor interface. At the top, the title is "Login to Application" with a "Test Script Overview" and "View Snapshots" link. The status bar indicates "Contains Unsaved Changes". The script is in "Draft" state, created by "Mark Kaminsky". The "Test Data" dropdown is set to "UserID and Password".

Two callout boxes highlight key features:

- A red callout box points to the "Test Data" dropdown menu, containing the text: "Test Data associated with the script".
- A red callout box points to the "User ID" and "Password" variables in the script steps, containing the text: "Test Data variable inserted in the test script".

The "Manual Steps" table is as follows:

Step	Description	Expected Results
1	Launch the Application	Application launches and prompts the user for user id and password.
2	Enter the user id: <code>User ID</code>	User id is accepted
3	Enter password: <code>Password</code>	The password is accepted

**Figure 4-23: Entering Test Data Variables in Scripts**

When a test script that uses test data is executed, the test data variables are replaced with the data from the corresponding test data columns. Figure 4-24 shows an example of how the variable “User ID” and “Password” were replaced by “Bob” and “abc” during the execution of a test script.

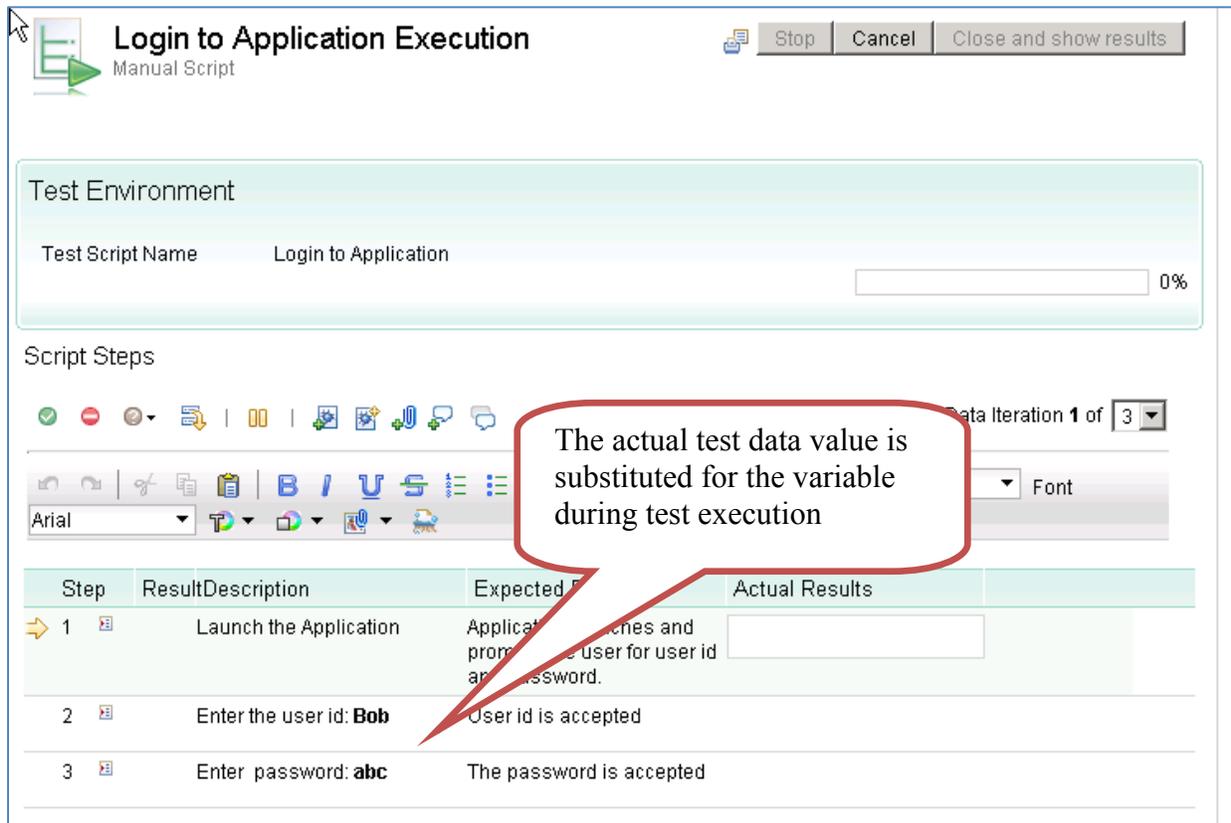


Figure 4-24: Test Data Execution Example

## 4.6 Test Asset Reuse

Test plans, test cases, test scripts, parts of test scripts, and test suites can be reused to varying degrees. There are three primary approaches to reuse:

1. **Create a Duplicate:** All test assets listed above can be duplicated and updated independently of the original. For example, a UAT test plan could be created from a copy of the system test plan and then updated as required.
2. **Test Case/Script Reuse:** Test Cases and Scripts are commonly reused across test plans. For example, a test plan that has been written for use in a system test plan or test suite should, more often than not, be acceptable without modification in a UAT test plan or suite.
3. **Keyword Substitutions:** Portions of test scripts can be copied to a global list of reusable scripts. These “keyword scripts” can then be included in any test script. Updating the global definition updates the steps that have been added to the various test scripts. Refer to the RQM online help details on how to use keywords.

## Section 5. Test Execution

### 5.1 Test Execution Scheduling

System and User Acceptance Testing should be scheduled well in advance of execution. While it is likely that the system's project schedule will reference the dates of the testing efforts, it is recommended that RQM's test execution schedule be utilized to schedule the execution of test suites/cases. Figure 5-1 shows an example of some test execution schedules for an RQM project area.

View Execution Schedules ?

Group by: Ungrouped Filter Displayed Items

Show All Items per page Previous | 1 | Next

<input type="checkbox"/>	ID	Name ^	Owner	Modified	History
<input type="checkbox"/>	3	System Testing for CM related Test Cases	Mark Kaminsky	Jul 21, 2010	
<input type="checkbox"/>	4	System Testing for TM Related Test Cases	Unassigned	Jul 21, 2010	
<input type="checkbox"/>	6	User Acceptance Testing - Iteration 1	Mark Kaminsky	3 minutes ago	
<input type="checkbox"/>	5	User Acceptance Testing - Iteration 2	Unassigned	1 minute ago	
<input type="checkbox"/>	7	User Acceptance Testing - Iteration 3	Mark Kaminsky	1 minute ago	

Showing 1-5 of 5 items Previous | 1 | Next

**Figure 5-1: Test Execution Schedules**

Figure 5-2 shows an expanded test execution schedule. In this example, the test execution is scheduled for July 27, 2010 and there are 6 test suites included in the test execution schedule under the "Steps" section of the schedule. The test execution steps can be a combination of individual test cases and test suites and are listed in the order in which they should be executed.

While RQM does not provide a means of launching (executing) the test cases and suites directly from the schedule, the schedule should be used as a guide to which test cases/suites should be executed, when they should be executed and in what order.

## User Acceptance Testing - Iteration 2 ?

Execution Schedule

Discard Changes
Save

Originator: Mark Kaminsky    Owner: Unassigned

Description: [< Click here to enter a description >](#)

**Overview** ?

Schedule an execution based on:

Date and time:

Date:

Build completion:

Select Build: Unassigned

Start execution schedule based on build status: Success (OK)

Test Cell: [Select Test Cell:](#)

**Steps** ?

Steps of the execution schedule:

Show All ▼ Items per page
« Previous | 1 - 6 of 6 | Next »

No.	Step	Executable	Target	Status
<input type="checkbox"/> 1	Test Suite	Verify RQM Asset Creation	Multiple	●
<input type="checkbox"/> 2	Test Suite	RQM: Verify Structure of RQM Assets	Multiple	●
<input type="checkbox"/> 3	Test Suite	ClearQuest: Verify Defect Primary Path	Multiple	●
<input type="checkbox"/> 4	Test Suite	ClearQuest: Verify Defect Tabs and Fields	Multiple	●
<input type="checkbox"/> 5	Test Suite	Test Management: Verify Reports	Multiple	●
<input type="checkbox"/> 6	Test Suite	Test Management: Security Verification	Multiple	●

« Previous | 1 - 6 of 6 | Next »

**Figure 5-2: Expanded Test Execution Schedule**

Figure 5-3 shows a single test suite inside of a test execution schedule expanded to show the list of test cases which are part of the test suite. This view offers a simple means of viewing the test cases associated with each test suite.

Steps ?

Steps of the execution schedule:

Show All ▼ Items per page
« Previous | 1 - 6 of 6 | Next »
+ ↑ ↓ × 📄

No.	Step	Executable	Target	Status
<input checked="" type="checkbox"/>	1	Test Suite	Verify RQM Asset Creation	Multiple <span style="color: green;">●</span>
<input type="checkbox"/>	2	Test Suite	RQM: Verify Structure of RQM Assets	Multiple <span style="color: green;">●</span>
<input type="checkbox"/>	3	Test Suite	ClearQuest: Verify Defect Primary Path	Multiple <span style="color: green;">●</span>
<input type="checkbox"/>	4	Test Suite	ClearQuest: Verify Defect Tabs and Fields	Multiple <span style="color: green;">●</span>
<input type="checkbox"/>	5	Test Suite	Test Management: Verify Reports	Multiple <span style="color: green;">●</span>
<input type="checkbox"/>	6	Test Suite	Test Management: Security Verification	Multiple <span style="color: green;">●</span>

« Previous | 1 - 6 of 6 | Next »

Test Suite ?

Test Suite details:

Show All ▼ Items per page
« Previous | 1 - 8 of 8 | Next »
✎ 📄

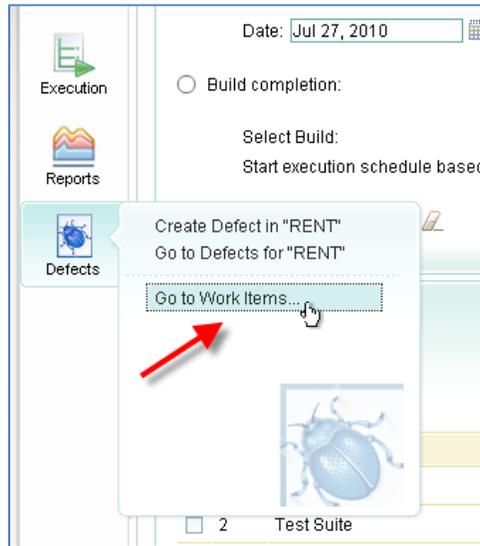
<input type="checkbox"/>	Test Case	Test Plan	Test Milestone	Test Script	Machine	Test Cell	Health
<input type="checkbox"/>	RQM: Login...	Test Manag...	UAT	RQM: Login to RQM <span style="font-size: 0.8em;">▼</span>	Local Com...		
<input type="checkbox"/>	Create Mas...	Test Manag...	UAT	RQM: Create Master Test <span style="font-size: 0.8em;">▼</span>	Local Com...		
<input type="checkbox"/>	RQM: Crea...	Test Manag...	UAT	RQM: Create a Child Test <span style="font-size: 0.8em;">▼</span>	Local Com...		
<input type="checkbox"/>	RQM: Crea...	Test Manag...	UAT	RQM: Create Test Data <span style="font-size: 0.8em;">▼</span>	Local Com...		
<input type="checkbox"/>	RQM: Crea...	Test Manag...	UAT	RQM: Create a Test Case <span style="font-size: 0.8em;">▼</span>	Local Com...		
<input type="checkbox"/>	RQM: Crea...	Test Manag...	UAT	RQM: Create Test Script <span style="font-size: 0.8em;">▼</span>	Local Com...		
<input type="checkbox"/>	RQM: Crea...	Test Manag...	UAT	RQM: Create Test Suite <span style="font-size: 0.8em;">▼</span>	Local Com...		
<input type="checkbox"/>	RQM: Link ...	Test Manag...	UAT	Link Test Plan, Case, Scr <span style="font-size: 0.8em;">▼</span>	Local Com...		

« Previous | 1 - 8 of 8 | Next »

**Figure 5-3: Test Suite Expanded in Test Execution Schedule**

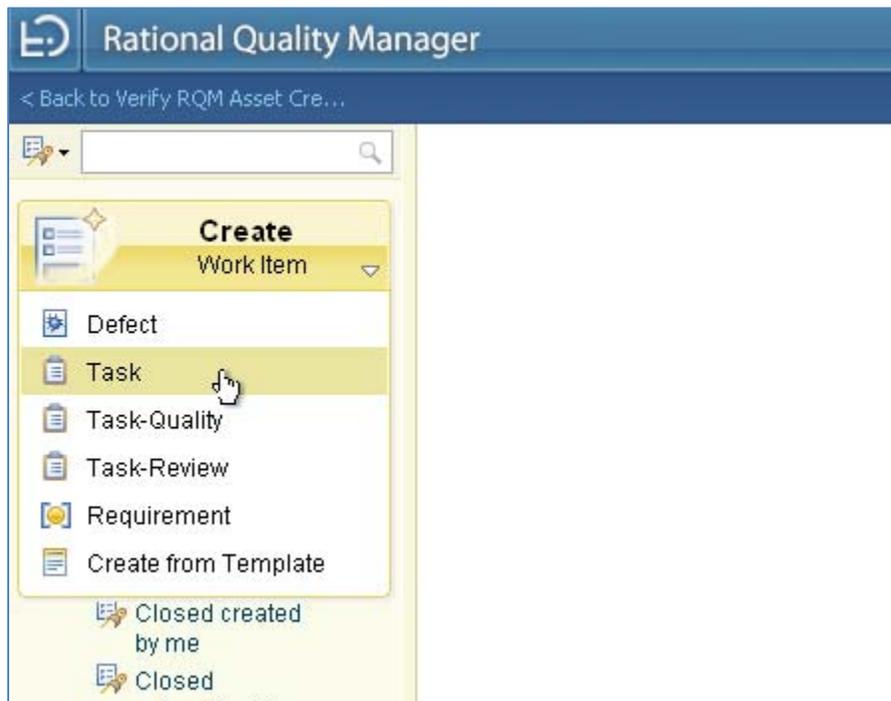
## 5.2 Test Execution Assignment

The preferred approach to task assignment in RQM is the use of a work item. The test lead may create a work item as a means of assigning any type of task to a tester; including the execution of a test case or suite. To create a work item, first select “Defects=>Go to Work Items” as shown in Figure 5-4.



**Figure 5-4: Opening Work Items**

Once in the work item view of RQM, select “Task” as the type of work item to create as shown in Figure 5-5.



**Figure 5-5: Creating a Work Item**

When the work item editor is displayed, fill in the pertinent information on the “Overview” tab as shown in Figure 5-6. At a minimum, the following information should be entered: summary, priority, due date and who will own the work item. The summary of the work item should include the name of the test case or test suite that is being assigned to be executed. The owner of the work item is the person that it is assigned to.

**Task 1128 \*** Save

Summary:  New

Loaded: Aug 20, 2010 8:39 A.M.

Overview Links Approvals History

---

**Details**

Type:	<input type="text" value="Task"/>	Priority:	<input type="text" value="Medium"/>
Filed Against:	<input type="text" value="Unassigned"/>	Planned For:	<input type="text" value="Unassigned"/>
Project Area:	FSA Rational Enterprise UAT	Estimate:	<input type="text"/>
Team Area:	FSA Rational Environment Team	Correction:	<input type="text"/>
Creation Date:	Aug 20, 2010 8:37 A.M.	Time Spent:	<input type="text"/>
Created By:	Mark Kaminsky	Due Date:	<input type="text" value="Aug 20, 2010"/>
Tags:	<input type="text"/>		
Owned By:	<input type="text" value="Mark Kaminsky"/>		

**Quick Information**  
No Links.

---

**Description** Edit

Execute the test suite as part of the scheduled test execution.

---

**Discussion** Add Comment

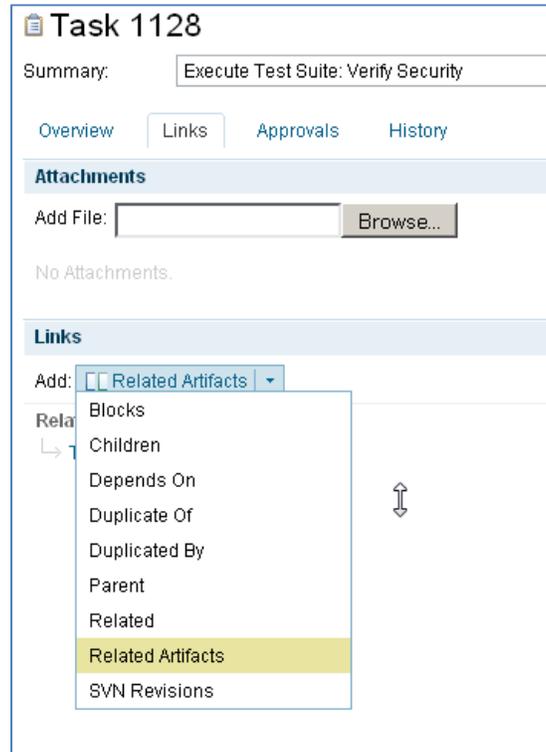
No Comments.

Add Comment

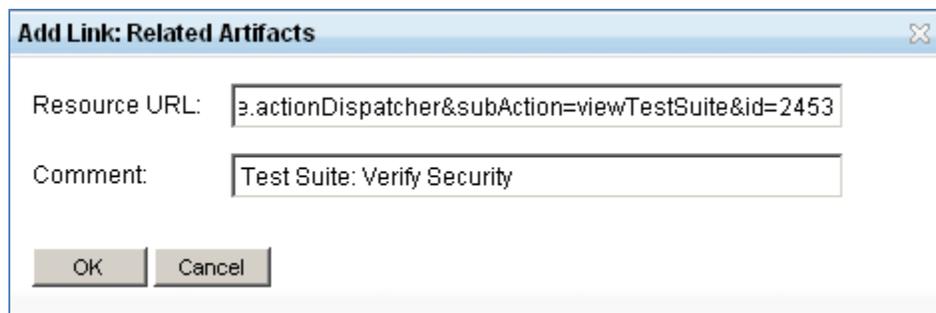
Save

**Figure 5-6: Completing a Work Item**

A link from the work item to the test case or test suite that is being assigned for execution should also be entered in the work item. The easiest way to do this is to add a link to the related artifact and specify the URL to the test case or test suite. To get the URL for the test case or test suite, find the test case or test suite in RQM and then copy the line in the address bar. This process is shown in Figure 5-7 and Figure 5-8.



**Figure 5-7: Linking a Work Item to a Test Suite/Case**



**Figure 5-8: Specifying the Work Item URL.**

Figure 5-9 shown what the work-item looks like after the related artifact (test case or suite) has been associated with it. This is extremely helpful to the person being assigned the work item because they can simply open the “links” tab and click on the test case or test suite to open it.

**Task 1128**

Summary:  New Saved: Aug 20, 2010 8:41 A.M.

Overview | **Links** | Approvals | History

**Attachments** Subscribers Add...

Add File:  Browse...

No Attachments.

**Links**

Add: Related

**Related Artifacts**

- ↳ [Test Suite: Verify Security](#)

**Figure 5-9: Work Item's Related Artifacts**

The tester will know that they've been assigned a work item because it will appear in their list of tasks under their dashboard as shown in Figure 5-10.

**Mark Kaminsky's Dashboard** Auto-save

General

**My Tasks**

Type Filter Text

Previous | 1 - 4 of 25 | Next

ID	Summary	Artifact
1128	Execute Test Suite: Verify Security	
1127	Run scheduled test cases	
1126	Execute test case abc	
1115	!Planning_AuthoringWorkItemSectionSummary!	Defect: Security Verification

Previous | 1 - 4 of 25 | Next

**Quality Manager Articles & Tutorials (6 new)**

- Three ways to migrate test assets from Rational ClearQuest test manager to Rational Quality Manager Jul 20, 2010
- What's new in IBM Rational Rhapsody, Version 7.5.2 Jun 7, 2010
- Complete the integration of Rational Quality Manager and Rational ClearQuest May 25, 2010
- Collaborative ALM interoperability May 4, 2010
- Server Sizing Guide - IBM Rational Quality Manager Version 2.0 Apr 8, 2010

Page 1 of 2

**Plan Requirements Coverage by Test Case**

Click [here](#) to configure this report.

**Welcome to Rational Quality Manager**

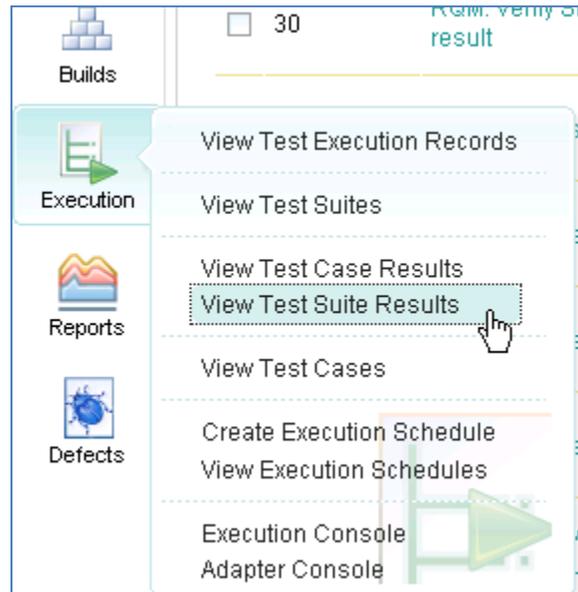
Rational Quality Manager provides a collaborative application lifecycle management (ALM) environment for test planning.

**Figure 5-10: Work Items in a User's Dashboard**

### 5.3 Test Execution Evaluation

Test Suite/Case execution results can be viewed through a number of different mechanisms. Since test cases will be executed as part of the execution of a test suite the majority of the time,

the test suite results should be viewed by selecting “Execution=>View Test Suite Results” as shown in Figure 5-11.



**Figure 5-11: Finding Test Suite Execution Results**

The test suite execution results, shown in Figure 5-12, list the test suites that were executed, when they were completed, the state, and the related test plan. Some test suites that did not finish executing have the state “In Progress”. The most likely reason for a test suite execution to be in the state “In Progress” is that the test suite was halted before completion of all related test cases. This can happen, for instance, if a tester takes a lunch break before the suite finishes executing and RQM times out.

**View Test Suite Results** ?

Group by: Ungrouped Filter Displayed Items

10 Items per page Previous | 1,2,3 | Next

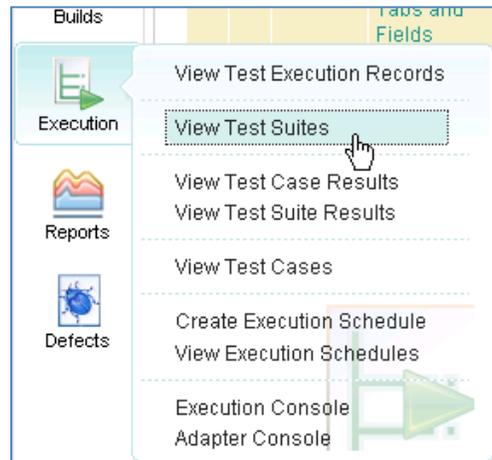
<input type="checkbox"/>	ID	Name	Completed <span>▼</span>	State	Test Plan	Test Milestone
			Any <span>↕</span>	Any <span>▼</span>	<span>↕</span>	<span>↕</span>
<input type="checkbox"/>	34	Test Management: Security Verification result	Aug 4, 2010	Failed		
<input type="checkbox"/>	30	RQM: Verify Structure of RQM Assets result	Aug 2, 2010	Failed	Test Management Test Plan	UAT
<input type="checkbox"/>	29	Verify RQM Asset Creation result	Aug 2, 2010	Failed	Test Management Test Plan	UAT
<input type="checkbox"/>	13	Verify ClearCase Operations result	Jul 8, 2010	Blocked	Configuration Management Test Plan	
<input type="checkbox"/>	12	Verify ClearCase Operations result	Jul 8, 2010	Blocked	Configuration Management Test Plan	
<input type="checkbox"/>	11	Verify ClearCase Operations result	Jul 8, 2010	Failed	Configuration Management Test Plan	
<input type="checkbox"/>	15	ClearQuest: Verify CR Primary Workflow result	N/A	Inconclusive	Configuration Management Test Plan	Sys Testing
<input type="checkbox"/>	21	ClearQuest: Verify CR Alternate Workflow result	N/A	Inconclusive		
<input type="checkbox"/>	23	Verify RQM Asset Creation result	N/A	In Progress		
<input type="checkbox"/>	33	Test Management: Verify Reports result	N/A	Inconclusive	Test Management Test Plan	UAT

Showing 1-10 of 28 items Previous | 1,2,3 | Next

**Figure 5-12: Test Suite Execution Results**

A test suite execution that is in the state “In Progress” can be restarted from where it left off. Unfortunately, it cannot be restarted from the Test Suite Execution Results shown in Figure 5-12.

In order to access the test suites that are still “In Progress”, open a list of all test suites and then search for the test suite that you would like to continue executing. The first step is to open a list of all executed test suites as shown in Figure 5-13. The results will appear as shown in Figure 5-14. Notice that the “In Progress” status appears as a hyperlink. Selecting the hyperlink “In Progress” opens the test suite execution console to the point where the test execution halted.



**Figure 5-13: View Test Suites**

**View Test Suites** ?

Group by: **Ungrouped** Filter Displayed Items

10 Items per page Previous | 1,2 | Next

<input type="checkbox"/>	ID	Name	State	Status	Owner	Category	Function	Test Phase
<input checked="" type="checkbox"/>	2417	ClearQuest: Verify Defect Tabs and Fields	Draft	Inconclusive	Unassigned	Test management	ClearQuest	Unassigned
<input type="checkbox"/>	2382	ClearQuest: Verify CR Alternate Workflow	Draft	Inconclusive	Mark Kamin	Configuration		Unassigned
<input type="checkbox"/>	2367	ClearQuest: Verify CR Primary Workflow	Draft	In Progress	Unassigned	Management	ClearQuest	Unassigned
<input type="checkbox"/>	2312	RQM: Verify Structure of	Draft	Failed	Unassigned	Test management	RQM	Unassigned

**Hyperlink to test execution console.**

**Figure 5-14: Test Suite View Results**

Clicking on the “In Progress” hyperlink as shown in Figure 5-14 opens the test execution console for the test case as shown in Figure 5-15. To continue testing from the test execution console, simply click on “Run Manual Test”. Refer to the RQM Online Help for additional information on executing test suites.

Test Suite Execution

## Test Suite Execution Console ClearQuest: Verify CR Primary Workflow

Primary Workflow ?

Close and show results

Test Suite	ClearQuest: Verify CR Primary Workflow
Execution Mode	Sequential
Completed	1 Test Cases
Running	1 Test Case
Not Run	13 Test Cases

Show All ▼ Items per page

⏪ Previous | 1 - 2 of 2 | Next ⏩

Name	Test Environment	Test Script	Host Name	Status	Progress	Modified	Result
Execution	Unassigned	Unassigned	Local Computer	✓	100% <div style="width: 100%; height: 10px; background-color: #4caf50; margin: 2px 0;"></div>	1 minute ago	
CR: Transition a CR from Submitted to Initial_Review Execution	Unassigned	CR: Transition a CR from Submitted to Initial_Review	localhost	○	Run Manual Test	Jul 28, 2010	

⏪ Previous | 1 - 2 of 2 | Next ⏩

Close and show results

**Figure 5-15: Test Execution Console**

The success or failure of individual test cases can be reported on using the Test Execution Record (TER) Report as shown in Figure 5-16.

TER Listing 

TER ID	Name	Test case	Test Environment	Test plan	Iteration	Owner	State	Last modified
2452	Defect: Cancel a DR from the Submitted State	<a href="#">Defect: Cancel a DR from the Submitted State</a>		Test Management Test Plan	System Test	mkaminsk	In Progress	
2431	RQM: Login to RQM	<a href="#">RQM: Login to RQM</a>		Test Management Test Plan	System Test	mkaminsk	Failed	
2432	Create Master Test Plan	<a href="#">Create Master Test Plan</a>		Test Management Test Plan	System Test	mkaminsk	Not Run	
2433	RQM: Create Child Test Plan	<a href="#">RQM: Create Child Test Plan</a>		Test Management Test Plan	System Test	mkaminsk	Not Run	
2434	RQM: Create Test Data	<a href="#">RQM: Create Test Data</a>		Test Management Test Plan	System Test	mkaminsk	Not Run	
2435	RQM: Create a Test Case	<a href="#">RQM: Create a Test Case</a>		Test Management Test Plan	System Test	mkaminsk	Not Run	
2436	RQM: Create Test Script	<a href="#">RQM: Create Test Script</a>		Test Management Test Plan	System Test	mkaminsk	Not Run	
2437	RQM: Create Test Suite	<a href="#">RQM: Create Test Suite</a>		Test Management Test Plan	System Test	mkaminsk	Not Run	
2438	RQM: Link Test Plan, Case, Script and Suite	<a href="#">RQM: Link Test Plan, Case,</a>		Test Management Test Plan	System Test	mkaminsk	Not Run	

Figure 5-16: TER Listing Report

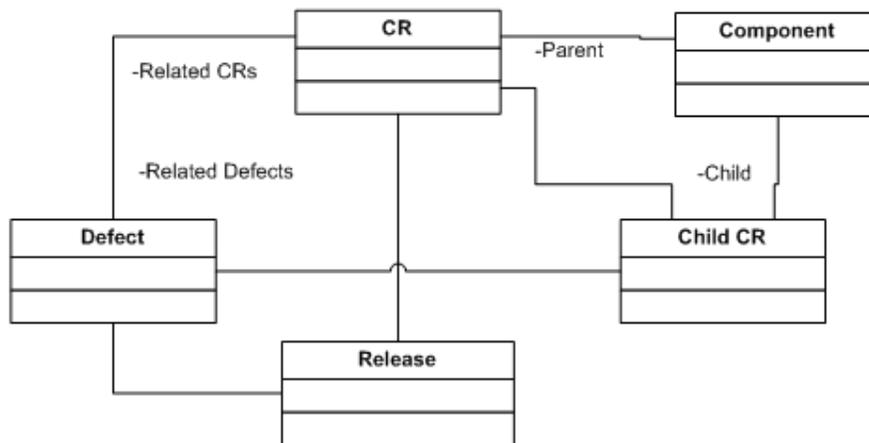
## Section 6. Defect Management

All defects identified during testing will be created from within RQM using the RQM/ClearQuest integration. Policy enforcement mechanisms have been enabled in order to ensure that only testers are able to create ClearQuest-based defect records. Defect records have been integrated with change request to ensure that only Change Requests and Child Change requests can be referenced by ClearCase in order to checkout, modify and checkin code as part of the defect correction process. In other words, defect records are used to document a problem found during formal testing and CRs and ChildCRs are used to make the necessary corrections.

Defects are NOT used to document problems found in production. Rather, a Change Request should be used to document any problem found during production.

### 6.1 ClearQuest Design Overview

The image below shows the relevant ClearQuest records used in the FSA Test Management process and their relationship to one another. The primary focus of this section is on the Defect Record type.



**Figure 6-1: Related ClearQuest Record Types**

**Defect Record:** The defect record is used by testers to track all defects found during a phase of testing. Defect records can only be created by members of the test team and should only be created while executing a test case from within Rational Quality Manager.

**Change Request (CR) Record:** The CR is the primary record used to document and manage required changes to a production release of a system. A CR is opened in ClearQuest in order to document production level enhancements, corrections, and maintenance activities. Each CR will be linked to two "Release" records and at least one "Component" record.

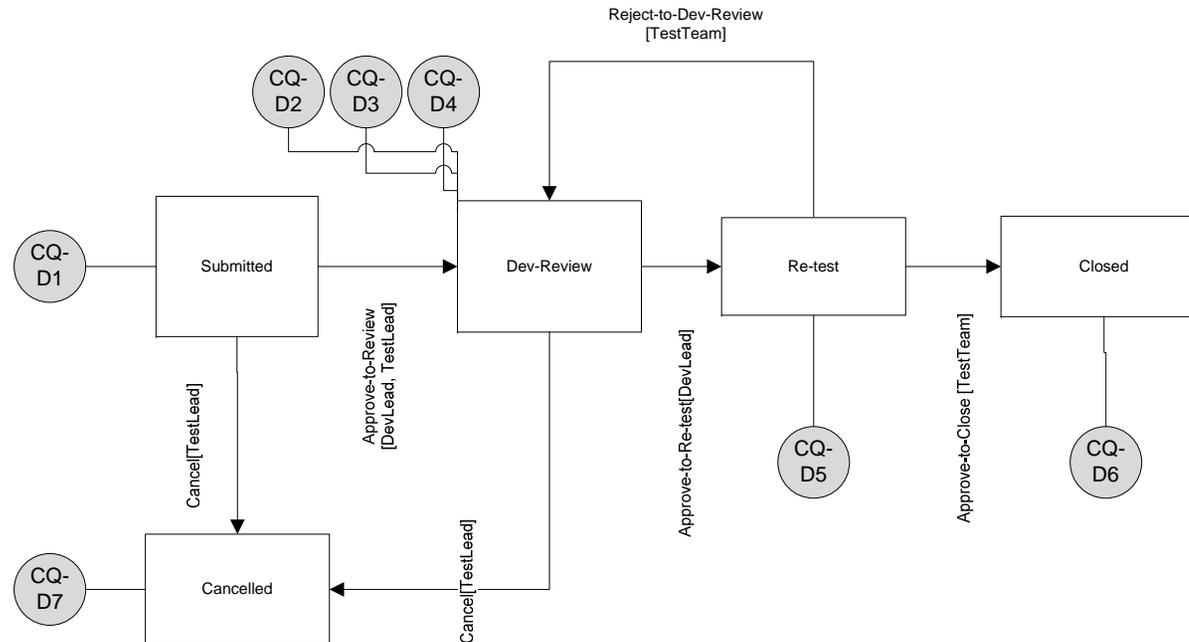
**ChildCR Record:** The ChildCR record is used to subdivide the work associated with a CR into more granular units which can be assigned to developers to implement.

**Release:** Identifies a release of the system.

**Component:** Referenced by the CR and ChildCR in order to specify which system component the change relates to.

- CR $\Rightarrow$ Component: Each CR will be related to one or more system level components.
- CR $\Leftrightarrow$ ChildCR: Each CR may be related to zero or more ChildCRs. A ChildCR will be created from the CR record form in order to subdivide the work associated with implementing a CR.
- CR $\Leftrightarrow$ Defect: Each defect identified during testing will be associated with one or more CRs.
- NOTE: In order for a CR to be associated with a defect, the following conditions must be true:
  - 1) The target release of the CR and the defect must be the same.
  - 2) The defect must be in the “Dev-Review” state.
  - 3) The CR must be in the “Development” state.
- CR  $\Rightarrow$ Release: Each CR identified will be associated with a release record indicating the release of the system that was being tested or used when the required change was found; referred to as the “Found in Release”. The CR will also be associated with a release record indicating the release of the system that the CR will be implemented in; referred to as the “Target Release”.
- Defect $\Leftrightarrow$ ChildCR: Each defect could be related to a ChildCR. This would occur in order to assign a developer the task of correcting the defect.
- Defect $\Leftrightarrow$ Release: Each defect will be related to the release that the defect is targeted to be corrected in; referred to as the “Target Release”.
- ChildCR $\Rightarrow$ Component: Each ChildCR could be associated with one or more system components. The relationship between the ChildCR and the system components indicates which part(s) of the system will be modified in order to implement the child CR.

Figure 6-2 shows the workflow (state transition model) for a Defect. The primary flow is a straight path from the Submitted state to the Closed state. Alternate transitions are possible including cancelling a defect, or transitioning the defect back to dev-review states from the re-test state. The image shows the possible state transitions as well as which user groups are allowed to initiate the transition. Each arrow is labeled as follows: <Action Name [groups allowed to perform the action]>. The off page connectors CQ-CR1 and CQ-CR2 reference the defect processing flowchart shown in Figure 6-5.



**Figure 6-2: Defect Record Workflow**

Figure 6-3 shows the workflow (state transition model) for a CR. The primary flow is a straight path from the Submitted state to the Closed state. Alternate transitions are possible including cancelling a CR, placing a CR on hold or transitioning the CR back to previous states. The image shows the possible state transitions as well as which user groups are allowed to initiate the transition.

Each arrow is labeled as follows: <Action Name [groups allowed to perform the action]>. The off page connectors CQ-CR1 and CQ-CR2 reference the defect processing flowchart shown in Figure 6-5.

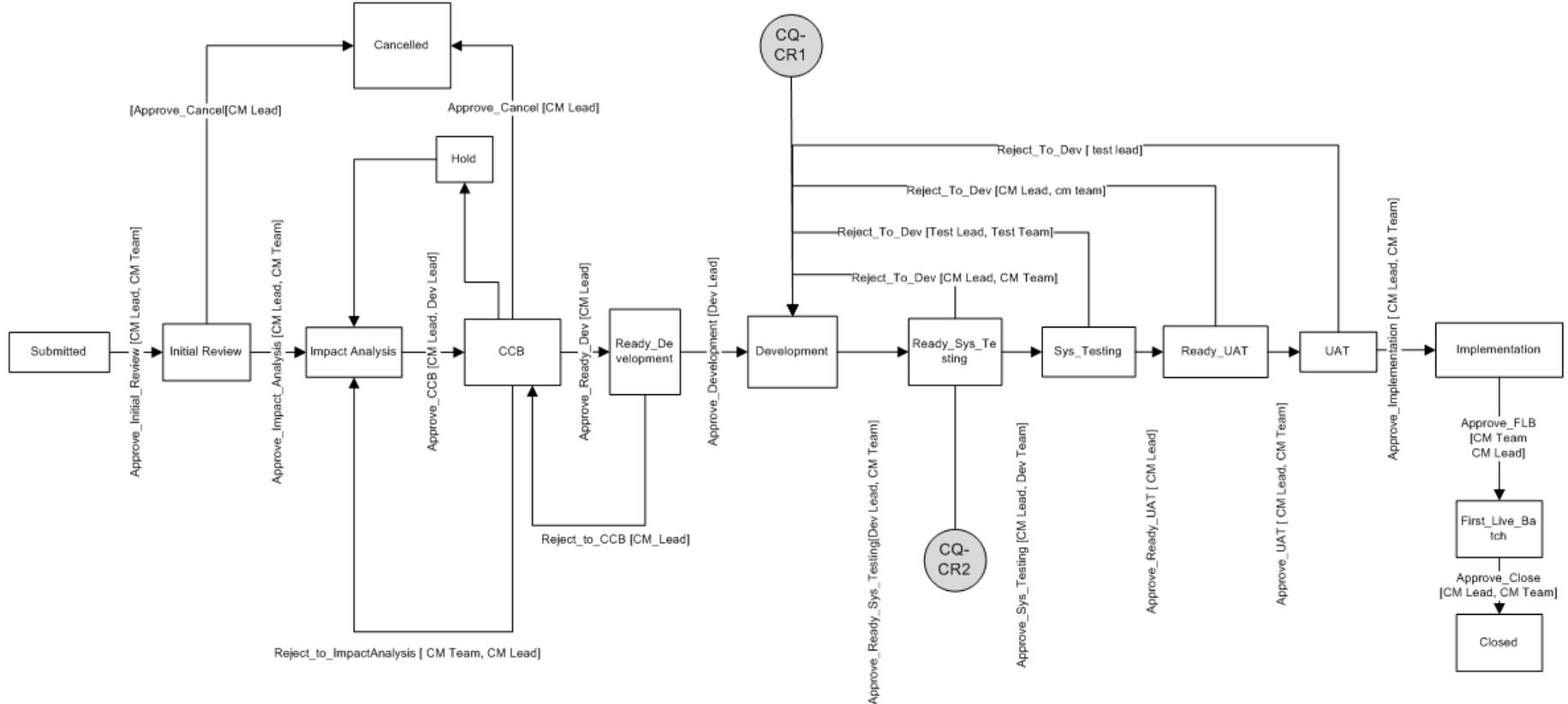
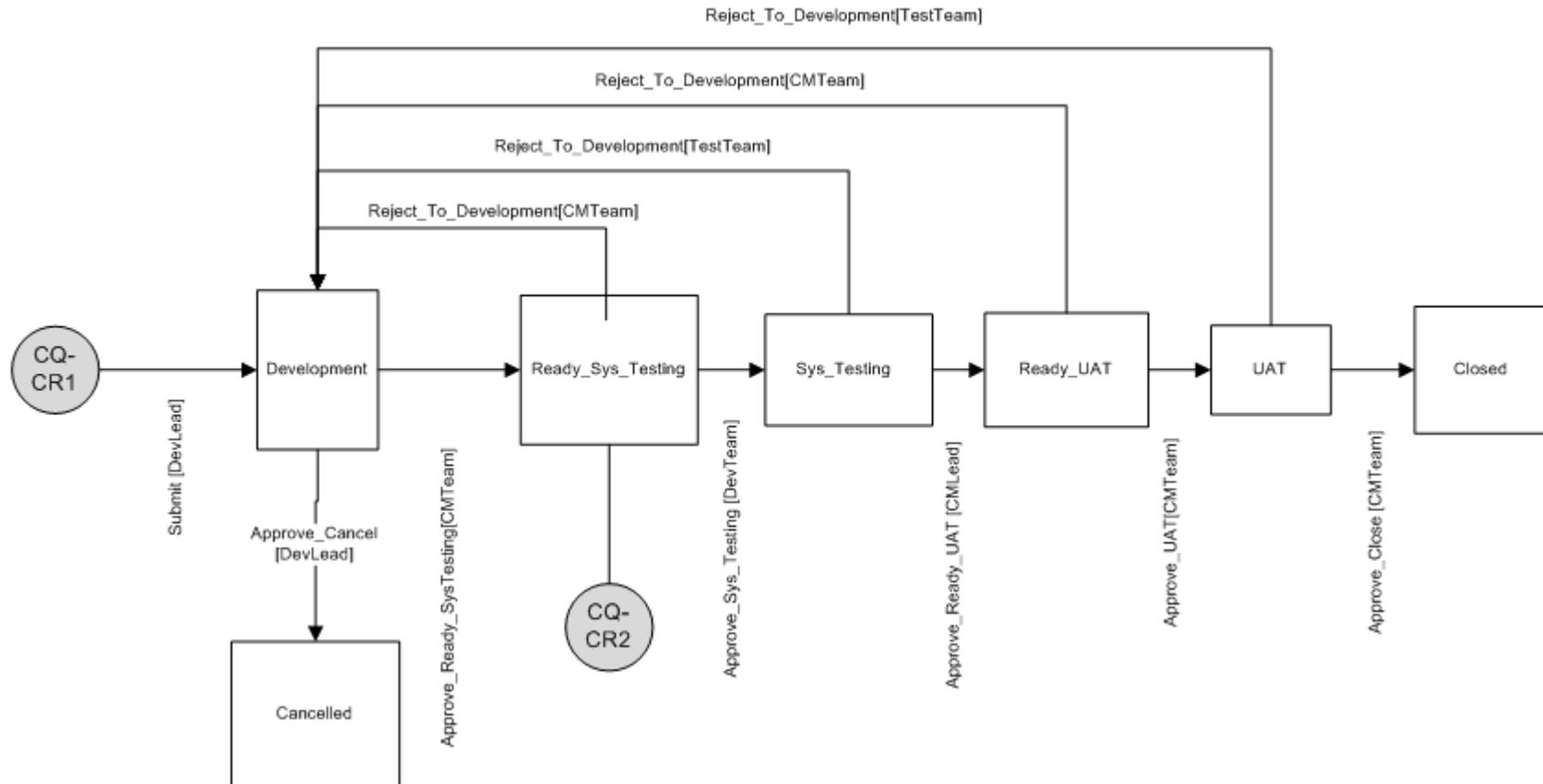


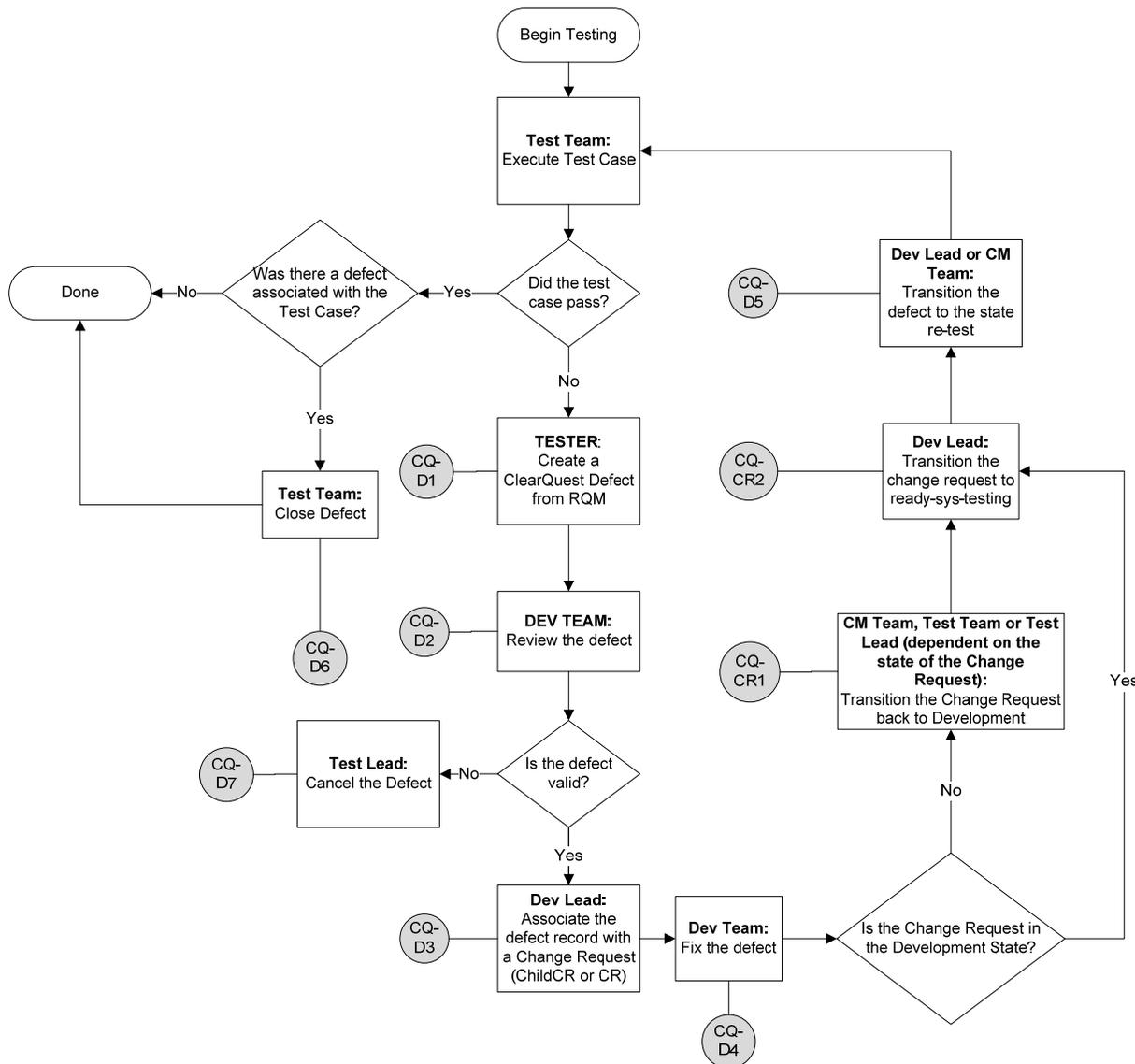
Figure 6-3: CR Record Workflow

When the effort involved in implementing a CR is likely to be large enough to require multiple developers, ChildCRs are created and linked to the “parent” CR. ChildCRs are used to subdivide work among multiple developers. Each CR will contain a list of zero or more ChildCR. The state model in Figure 6-4 shows the life of a ChildCR from the point where it is created from a parent CR through closure. The off page connectors CQ-CR1 and CQ-CR2 reference the defect processing flowchart shown in Figure 6-5.



**Figure 6-4: ChildCR Record Workflow**

Change Requests and Defects do not exist independently of one another. The flowchart in Figure 6-5 shows how defects and change requests (CRs and ChildCRs) are intertwined when a defect is discovered, analyzed, corrected, and, ultimately, closed. The off page connectors CQ-CR1, CQ-CR2, CQ-D1, CQ-D2, CQ-D3, CQ-D4, CQ-D5 and CQ-D6 show how the processes depicted in the flowchart relate to the state transition models/workflows for CRs, ChildCRs, and Defects shown on the following pages.



**Figure 6-5: Defect Resolution Flowchart**

## 6.2 Working with Change Requests (CRs)

Processing the CR entails transitioning the CR through various states where each state represents the work that is being performed relative to the CR. For example: A CR that has been transitioned to the Impact\_Analysis state means that the CR is being evaluated by the

requirements team, the development team, and the security team in order to determine the impact of implementing the CR.

By the time that the approved CR has progressed through the workflow from submission to closure, the system will have been changed and those changes will have been incorporated into a baselined release and deployed into “First Live Batch”. The step-action table in Table 6-1: CR Step/Action Table describes what happens when a CR transitions to each state.

<b>Step</b>	<b>Who Leads</b>	<b>What Happens</b>	<b>Output</b>
Submitted	Anyone with access to the automated CR system	<ul style="list-style-type: none"> <li>Complete the CR form and submit it within the system</li> </ul>	An open CR
Initial Review	CM Team	<ul style="list-style-type: none"> <li>CM Team reviews the CR to ensure the necessary information has been provided and that all information is accurate</li> <li>If information is missing, inaccurate, or not detailed enough, the CM Team manager will work with CR initiator to correct the CR</li> <li>If the CR is a duplicate of another CR already in the system then the CR will be cancelled with a notification to the requestor</li> </ul>	An accurate and complete change request that can be used for impact analysis

Step	Who Leads	What Happens	Output
Impact Analysis	Development Lead	<ul style="list-style-type: none"> <li>• Business impact analysis will determine:               <ul style="list-style-type: none"> <li>– Requirements that may need to be revised or business issues associated with the request</li> <li>– Feasibility of the time frames</li> <li>– Overall impact to the system or system components</li> </ul> </li> <li>• Technical impact analysis will determine:               <ul style="list-style-type: none"> <li>– Feasibility of the change and how to best implement it</li> <li>– Estimated effort and resources for developing, testing, and implementing the change</li> </ul> </li> <li>• Security impact analysis will determine:               <ul style="list-style-type: none"> <li>– Possible impact the change would have on the system security</li> <li>– If any security measures will need to be suspended during the implementation of the requested change</li> </ul> </li> <li>• Enterprise impact analysis will determine               <ul style="list-style-type: none"> <li>– Whether or not the change could be considered an enterprise event</li> <li>– Possible impact the change would have on enterprise systems, applications, and/or interfaces</li> </ul> </li> </ul>	An impact analysis report that includes a Security Impact Assessment
Configuration Control Board (CCB) Review and Approval	CM Manager (at the direction of the CCB)	<ul style="list-style-type: none"> <li>• Reviews the impact analysis report and the Security Impact Assessment</li> <li>• Determine one of four outcomes for CR:               <ul style="list-style-type: none"> <li>– <b>Approved:</b> implementation is authorized and may occur based upon the agreed release schedule</li> <li>– <b>Rejected:</b> a determination is made that additional information is required or that the impact analysis is incomplete</li> <li>– <b>Hold:</b> a decision is made to postpone until further notice</li> <li>– <b>Cancelled:</b> a determination is made that the request will not be performed</li> </ul> </li> </ul>	A CR that has been approved, rejected, placed on hold, or cancelled

<b>Step</b>	<b>Who Leads</b>	<b>What Happens</b>	<b>Output</b>
Ready for Development	Development Lead	<ul style="list-style-type: none"> <li>Children CRs may be created at this point in the process to address the scope of the change and the number of resources required to implement them</li> <li>Assignments are made for the Business Analyst and development staff to proceed work on the change</li> </ul>	A CR or multiple CRs are created with assigned staff
Development	Assigned Staff	<ul style="list-style-type: none"> <li>Create new or modify existing requirements</li> <li>Create/code the change request solution and update documents and materials to reflect changes</li> <li>Unit testing and integration testing is conducted</li> <li>Test Readiness Review is held to determine the start of System Test</li> </ul>	<ul style="list-style-type: none"> <li>Updates to baselines as needed</li> <li>Authority to proceed once integration testing is successfully passed</li> </ul>
Ready for System Test	CM Team	<ul style="list-style-type: none"> <li>Release is built from the CM repositories and installed into the system testing environment</li> </ul>	System changes are installed into system testing environment
System Test	Test Team	<ul style="list-style-type: none"> <li>System testing is conducted with results turned over to the CM Manager</li> <li>Test Readiness Review is held to determine the start of User Acceptance Test (UAT)</li> </ul>	<ul style="list-style-type: none"> <li>Updates to baselines as needed</li> <li>CCB Approval to proceed once system testing successfully passed</li> </ul>
Ready for UAT	CM Team	<ul style="list-style-type: none"> <li>Release is built from the CM repositories and turned over to data center staff for implementation into production</li> </ul>	System changes are installed into UAT testing environment
UAT	Test Team	<ul style="list-style-type: none"> <li>UAT is conducted with results turned over to the CM Manager</li> <li>Operational Readiness Review is held</li> </ul>	<ul style="list-style-type: none"> <li>Updates to baselines as needed</li> <li>CCB approval to proceed once user acceptance testing successfully passed</li> </ul>
Implementation	CM Team	<ul style="list-style-type: none"> <li>Functional and Physical Configuration Audits are conducted</li> <li>Release is built from the CM repositories and turned over to data center staff for implementation into production</li> </ul>	<ul style="list-style-type: none"> <li>Updates to baselines as needed</li> <li>CCB approval to proceed once user acceptance testing successfully passed</li> </ul>
First Live Batch (FLB)	CM Team	<ul style="list-style-type: none"> <li>A review of the system outputs is conducted (if corrections are required then a new CR will be opened; the product baseline will NOT be revised for the given release.)</li> <li>A results review is held to request closure of the CR</li> </ul>	<ul style="list-style-type: none"> <li>CR review results</li> <li>CCB approval to close the CR</li> </ul>

Step	Who Leads	What Happens	Output
Closed	CM Team	<ul style="list-style-type: none"> <li>▪ CR is completed</li> </ul>	

**Table 6-1: CR Step/Action Table**

### 6.3 Working with Defects

Processing the Defect entails transitioning the Defect through various states where each state represents the work that is being performed relative to the Defect. For example: A Defect that has been transitioned to the “Sys\_Testing” state means that the Defect has been evaluated by the development lead, associated with a CR that was assigned to a developer to work, corrected, and is now ready to be retested.

By the time that the approved Defect has progressed through the workflow from submission to closure, the defect will have been corrected as part of working on a related CR, incorporated into a baselined release, and passed UAT.

Table 6-1 shows the series of steps taken when a defect is identified and corrected.

Person	Action
Tester	Creates a defect record in ClearQuest in response to a failed test case
Test Lead	Transitions the defect to Dev-Review
Dev Lead	Reviews the Defect and associates a CR with the defect or contacts the Test Lead with reasons why the issue identified is NOT a defect so it can be canceled.
Test Lead	Based on Dev Lead review, either rejects the related CR back to Development or agrees to cancel the defect
Developer	Corrects the defect, documents the changes, and promotes the CR and Defect back to testing, and retest, respectively
CM Team	Promotes the CR to Ready-for-system-testing after completing a build and creating a new version of the product baseline
Tester	Retests and closes the defect after verifying that it has been corrected. The tester may also need to promote the CR to Ready-for-UAT if the defect was the result of a failed test case during UAT. The CM team would then need to also provide a build of the system and new version of the baseline for UAT for another round of UAT testing.

**Table 6-2: Defect Step-Action Table**

## 6.4 Email Notification

Email rules have been established in order to automatically notify various groups and users when a CR, Child CR, or Defect is transitioned from state to state. An email will be automatically sent to each user who is associated with a role that is relevant to the state that a record is changed to. For example, when a Defect is submitted, an email will be sent to those users in the roles of Developer and Test Lead because they will need to be aware of a new defect that may need to be corrected.

The format of the email notification will be as follows:

---

**Subject Line:** [Request ID] – [Request State Transitioning To] – [Request Title]

**Message Body:**

Created/Modified By: [User ID of Individual Taking Action]

This request may require your attention: you may be assigned to work on this request, assigned to approve someone else's work, or you are just being notified of a status change.

[[hyperlink to the ClearQuest record](#)]

---

The following email rules have been defined in order to support automatic email notification for Defects.

Actions	Roles that Email is Sent To
Submit	Developer, Test-Lead
Approve-to-Review	Dev-Lead
Approve-to-Re-Test	Test-Team

**Table 6-3: Defect Email Notification Rules**

For additional information on Automatic email notification, contact a member of the CM Team or refer to system-specific CM Training.

## Section 7. Reporting

A critical capability for any test management process is the ability to report on the status of testing related artifacts such as test plans, test cases, and test suites as well as the defects that have been identified during testing. The Rational tools provide varying capabilities with respect to automated reporting.

In general, the standard reports provided by RQM are not sufficient to provide the types of reports needed during a testing effort. Furthermore, RQM lacks the ability to create custom reports. ClearQuest reporting, on the other hand, is extremely flexible and provides a great deal of automation with respect to reporting on defects' identified and correcting. For these reasons, some of the reports required for test management activities will be manually created by accessing data from the following locations:

- Reviewing RQM assets directly in the RQM project area
- Running pre-existing RQM reports and mining relevant data
- Running ClearQuest reports that have been created to support the testing effort

Report Name	Description
Summary of Phases by Iteration	Used to summarize the number of test suites that have been assigned and completed by iteration for a specific phase of testing.
Summary of Phases by Iteration - Detail	Provides detailed information on the test suites associated with each iteration of the specified testing phase.
Test Execution Report	Summarizes test execution status of test cases associated with test suites for a given testing phase.
Test Status Report for Phase	Provides a high-level status of testing for a specific phase.
Plan Requirements Coverage Detail	Lists the requirements covered by the test cases associated with a test plan.
Defects by Component	Lists defects by related component.
Defect Detail Report	Detailed listing of all visible fields in a defect.
Defect Summary Report	Summary of defects by target release.
Defect Summary by Severity	Summary of defects grouped by release, severity, and state.

**Table 7-1: Summary of Reports**

### 7.1 Test Management Report

This section summarizes the standard test management related reports that will be produced. While RQM does provide out-of-the-box reporting capabilities, it does not provide a mechanism



Section/Field	Source of Data
	Test Execution schedule can be counted.
Assigned	Number of suites assigned to testers for the iteration. Each test suite assigned to an iteration can be examined in order to determine if the "Owner" field is set to a testers name or left "Unassigned".
Completed	Number of suites that have not been executed by testers for the iteration. Each test suite assigned to an iteration can be examined in order to determine if it has been executed.
% Complete	Total number of suites assigned / total number of suites completed. This value is calculated based on the Assigned and Completed values.
Timeline summary	Manually tracked outside of RQM.

**Table 7-2: Summary of Phase by Iteration Data**

**7.1.1.1 Summary of Phases by Iteration – Details (page 1)**

This is the second part of the Summary of Phases by Iteration report. This report provides detailed information on the test suites associated with each iteration of the specified testing phase. Figure 7-2 shows the template used to produce this report. The source of the data required is described in Table 7-3.

Summary of Phases by Iteration-Details							
Project Name:							
Testing Phase:							
Report Date:							
Iteration ID	Suite Name	Weighting	Test Cases	Assigned	Test Cases Completed	Percentage of Test Cases Completed	Weighted Percentage
2	Sample Test Suite	0.2	30	30	30	100%	20.00%
<b>Totals:</b>		0.2	30	30	30	100%	20.00%

**Figure 7-2: Summary of Phases by Iteration – Details (page 1)**

Section/Field	Source of Data
Iteration Identifier	Manually Entered
Suite Names	RQM Test Suites for the Iteration. Since each iteration equates to a test schedule that is related to a specific test plan, suite names can be obtained by reviewing the “Steps” section of a test schedule that corresponds to the iteration.
Weighting	Tracked outside of RQM since RQM does NOT allow us to associate a weighted value with a test suite. Weight is the percentage value assigned to the scenario for the total of the test phase
Test Cases	The total number of test cases associated with the suite. These are the test cases that are associated with the RQM Test Suite. The test suite would have to be opened in RQM and the test cases would need to be counted manually.
Assigned	The number of assigned test cases. The only way to get the number of assigned test cases is to look at each test case associated with the suite and verify that the “Owner” field is not “Unassigned”.
Test Cases Completed	Each test case associated with the RQM Suite would need to be opened and inspected to determine whether the test case passed or failed.
Percentage Test Cases Completed	The number of test cases / the number of test cases passed. This value can be calculated based on the number of test cases and the number of completed test cases.
Weighted Percentage	This field will be calculated based on the weighting value and percentage of test cases completed.

**Table 7-3: Summary of Phases by Iteration – Details (page 1) Data**



Section/Field	Source of Data
Number of Test Scripts Executed	The data will come from the Test Suite Execution Results associated with the RQM Test Suite.
Number of Test Scripts Passed	The data will come from the Test Suite Execution Results associated with the RQM Test Suite
Number of Test Scripts Failed	The data will come from the Test Suite Execution Results associated with the RQM Test Suite
Number of Test Scripts Blocked	The data will come from the Test Suite Execution Results associated with the RQM Test Suite
Percent attempted	Manually Calculated – number of scripts executed / (number of scripts passed + number of scripts failed)
Percent passed	Manually Calculated – number of scripts executed / (number of scripts passed)
Percent Failed	Manually Calculated – number of scripts executed / (number of scripts failed)
Percent Blocked	Manually Calculated – number of scripts executed / (number of scripts blocked)

**Table 7-4: Test Execution Report Data**

### 7.1.3 Test Status Report for Phase

This report provides a high-level status of testing for a specific phase. Most of the data required for this report can be obtained by running the ClearQuest based report “Defect Summary by Severity”. Figure 7-4 shows the sample template for this report. The source of the data required is described in Table 7-5.

Test Status Report for Phase:						
Project Name:						
Testing Phase:						
Report Period Ending Date:						
Report Date:						
<b>EVM: % Completed by Unit:</b>						
<b>Accomplishments</b>			<b>Upcomming Activites</b>			
<b>Testing Summary -</b>						
Test Cases Planned	Test Cases Executed	% Complete	Passed	Failed	Defects Opened	Defects Resolved
100	1	1%				
<b>Defect Summary (All iterations per phase)</b>						
Defect Status	Low	Medium	High	Urgent	Total	
Active						
Resolved						
<b>Active Defect Status (All iterations per phase)</b>						
Status	Low	Medium	High	Urgent		
Submitted						
Dev-Review						
Retest						

**Figure 7-4: Test Status Report for Phase**

Section/Field	Source of Data
EVM	Not tracked in RQM or ClearQuest.
Accomplishments	Not tracked in RQM or ClearQuest.
<b>Testing Summary- (Per Phase/Iteration) – example System Test Iteration 2</b>	
Test Cases Planned	Data available from one of the following RQM Reports: <ul style="list-style-type: none"> <li>TER Listing</li> <li>Execution Status using TER Count</li> </ul>
Test Cases Executed	Data available from one of the following RQM Reports: <ul style="list-style-type: none"> <li>TER Listing</li> <li>Execution Status using TER Count</li> </ul>
% Complete	Percent will be manually calculated. Test Cases planned / Test Cases successfully passed.

Section/Field	Source of Data
Passed	Data available from one of the following RQM Reports: <ul style="list-style-type: none"> <li>• TER Listing</li> <li>• Execution Status using TER Count</li> </ul>
Failed	Data available from one of the following RQM Reports: <ul style="list-style-type: none"> <li>• TER Listing</li> <li>• Execution Status using TER Count</li> </ul>
Defects Opened	Data available from the following ClearQuest Report: <ul style="list-style-type: none"> <li>• Defect Summary Report</li> </ul>
Defects Resolved (closed and cancelled)	Data available from the following ClearQuest Report: <ul style="list-style-type: none"> <li>• Defect Summary Report</li> </ul>
<b>Defect Summary – (All iterations per phase) Example: System Test Iteration 1 through xxx</b>	
Active/Resolved	Data available from the ClearQuest report “Defect Summary by Severity”.
<b>Active Defect Status – (All iterations per phase) Example: System Test Iteration 1 through xxx</b>	
Active ClearQuest State/Severity (submitted, dev-review, and retest)	Data available from the ClearQuest report “Defect Summary by Severity”.

**Table 7-5: Test Stats Report for Phase Data**

## 7.2 Requirements Traceability Matrix

The Requirements Traceability Matrix (RTM) in Figure 7-5 shows the relationship between Requirements, Test Suites/Cases, Configuration Items and other systems impacted. While this report cannot be automatically generated, the relationships between requirements stored in RequisitePro and Test Plans/Cases stored in RQM can be reported on using the RQM report shown in Figure 7-6. This information can then be used to manually obtain additional information required for the RTM



### 7.3.1 Defects by Component

**Summary:** Lists defects by related component.

**Filter:** Uses Query “Defect-By-Component”

**Fields to Display:** Component, Defect ID, Title, Target Release Number, Phase, Severity, and State.

**Sort Order:** Component, Target Release Number, Phase.

**Total/Subtotal:** Defects for each component, total unique defects in the report.

MyStartingLine Defects by Component					Federal Student Aid
As of: 08/09/10					
ID	Title	Release	Phase	Severity	State
<b>Component: MSL</b>					
00000435	Some title4	01.01.001	System Test	High	Re-Test
00000413	Some title8	01.04.010	System Test	High	Re-Test
00000462	Some title	01.06.000	System Test	Low	Re-Test
Defect count for component MSL: 3					
<b>Component: MSL Bookmarks</b>					
00000413	Some title8	01.04.010	System Test	High	Re-Test
00000462	Some title	01.06.000	System Test	Low	Re-Test
Defect count for component MSL Bookmarks: 2					
<b>Component: MSL Employee Finder</b>					
00000372	Some title6	01.03.000	UAT	Medium	Dev-Review
Defect count for component MSL Employee Finder: 1					
<b>Component: MSL IFrame Portlet</b>					
00000374	Some title		UAT	Medium	Submitted
00000398	Some title9	01.05.001	System Test	Low	Re-Test
00000375	Some title0	01.05.001	UAT	Medium	Closed
Defect count for component MSL IFrame Portlet: 3					
<b>Component: MSL Learning Coupon</b>					
00000435	Some title4	01.01.001	System Test	High	Re-Test
Defect count for component MSL Learning Coupon: 1					
<b>Component: MSL MRR</b>					
00000435	Some title4	01.01.001	System Test	High	Re-Test
Defect count for component MSL MRR: 1					
<b>Component: MSL News and RSS</b>					
00000371	Some title	01.06.000	UAT	Medium	Dev-Review
Defect count for component MSL News and RSS: 1					
<b>Component: MSL Quickr Integration</b>					
00000435	Some title4	01.01.001	System Test	High	Re-Test
Defect count for component MSL Quickr Integration: 1					
<b>Component: MSL WCM Content</b>					
00000435	Some title4	01.01.001	System Test	High	Re-Test
Defect count for component MSL WCM Content: 1					
Total Defects: 8					
1 of 1					

### 7.3.2 Defect Detailed

**Summary:** Detailed listing of all visible fields in a Defect.

**Filter:** Uses query “Defect-Query for ID”

**Fields to Display:** All visible fields in the Defect. Order the fields by Tab on the Defect record form.

**Sort Order:** Target Release (oldest first), Date Opened (oldest first)

<b>MyStartingLine Defect</b>		<b>Federal Student Aid</b>	
<b>As of: 08/02/10</b>			
<b>Defect Number: 00000035</b>			
<hr/>			
<b>Title:</b> Some title7	<b>Defect Type:</b> Data Element Problem	<b>Severity:</b> Low	
<b>State:</b> Closed	<b>Defect Type (other):</b>	<b>Priority:</b> High	
<b>Detected In Phase:</b> UAT			
<b>Target Release:</b> 01.03.000			
<b>Found By:</b> mkaminsk			
<b>Description:</b> test			
<b>System Impact:</b> test			
<b>Related Change Requests</b>			
<hr/>			
<b>Change Requests:</b>			
MSL00000039: mkaminsky test			
<b>Child Change Requests:</b>			
<b>Cancellation / Rejection</b>			
<hr/>			
<b>Cancellation Reason:</b>			
<b>Cancellation Reason (other):</b>			
<b>Cancellation Comment:</b>			
<b>Rejection Comment:</b>			
<b>Rejection History:</b>			
<b>Attachments</b>			
<hr/>			
TestAttachment.txt			
<b>Defect Number: 00000036</b>			
<hr/>			
<b>Title:</b> Some title	<b>Defect Type:</b> Calculation Problem	<b>Severity:</b> High	
<b>State:</b> Dev-Review	<b>Defect Type (other):</b>	<b>Priority:</b> High	
<b>Detected In Phase:</b> System Test			
<b>Target Release:</b> 01.05.001			
<b>Found By:</b> bsturnup			
<b>Description:</b> abc			
<b>System Impact:</b> abc			
<b>Related Change Requests</b>			
<hr/>			
<b>Change Requests:</b>			
<b>Child Change Requests:</b>			
<b>Cancellation / Rejection</b>			
<hr/>			
<b>Cancellation Reason:</b>			
<b>Cancellation Reason (other):</b>			
<b>Cancellation Comment:</b>			
<b>Rejection Comment:</b>			
<b>Rejection History:</b>			
1 of 19			

### 7.3.3 Defect Summary

**Summary:** Summary of defects by target release number.

**Filter:** Uses Query “Defect-Summary”

**Fields to Display:** ID, Title, Type of Defect, Target Release Number, Detected in Phase, Severity, Priority, State, Found By, Identified On, Opened On, Closed Date, Age of defect (number of days since the defect was identified. If the defect is closed or cancelled then the value will be the number of days between identification and closure/cancellation).

**Sort Order:** Target Release, Phase, Severity, State.

MyStartingLine Defect Summary							Federal Student Aid	
As of: 08/02/10								
ID	Title	Release	Phase	Severity	Priority	State	Age	
00000498	this is a longer title just to verify that the defect holds a long title.....		System Test	Medium		Submitted	0	
Type:	Data Element Problem	Found By: Kaminsky, Mark	Identified On: 8/2/2010	Opened On:08/02/2010	Closed On:			
00000397	Some title		System Test	Low		Submitted	88	
Type:	Design Problem	Found By: Kaminsky, Mark	Identified On: 5/6/2010	Opened On:05/06/2010	Closed On:			
00000396	Some title		System Test	High		Submitted	88	
Type:	Other	Found By: Kaminsky, Mark	Identified On: 5/8/2010	Opened On:05/08/2010	Closed On:			
00000417	Some title		System Test	Medium		Submitted	75	
Type:	Data Element Problem	Found By: CMLead user	Identified On: 5/19/2010	Opened On:05/19/2010	Closed On:			
00000378	Some title		UAT	Urgent		Submitted	104	
Type:	Design Problem	Found By: Kaminsky, Mark	Identified On: 4/20/2010	Opened On:04/20/2010	Closed On:			
00000374	Some title		UAT	Medium		Submitted	115	
Type:	Other	Found By: Kaminsky, Mark	Identified On: 4/9/2010	Opened On:04/09/2010	Closed On:			
00000075	Some title			Low	Low	Dev-Review	143	
Type:	Calculation Problem	Found By: TestTeam user1	Identified On: 3/12/2010	Opened On:03/12/2010	Closed On:			
00000475	Defect synopsis		01.00.004 System Test	Low	Medium	Closed	3	
Type:	Data Element Problem	Found By: RatAdmin	Identified On: 7/12/2010	Opened On:07/13/2010	Closed On: 07/15/2010			
00000382	Some title1		01.01.000 UAT	Low	Low	Dev-Review	98	
Type:	New Requirement	Found By: Bob Dorenfeld	Identified On: 4/28/2010	Opened On:04/28/2010	Closed On:			
00000370	Some title2		01.01.000 UAT	Medium	Low	Re-Test	124	
Type:	Gap in requirements Definition	Found By: Kaminsky, Mark	Identified On: 3/31/2010	Opened On:03/31/2010	Closed On:			
00000486	est		01.01.001 System Test	Medium	Medium	Dev-Review	18	
Type:	Data Element Problem	Found By: RatAdmin	Identified On: 7/15/2010	Opened On:07/18/2010	Closed On:			
00000466	Some title3		01.01.001 System Test	Medium	Medium	Dev-Review	34	
Type:	Design Problem	Found By: Kaminsky, Mark	Identified On: 6/29/2010	Opened On:06/29/2010	Closed On:			
00000435	Some title4		01.01.001 System Test	High	Medium	Re-Test	60	
Type:	Other	Found By: Kaminsky, Mark	Identified On: 6/3/2010	Opened On:06/03/2010	Closed On:			
00000368	Some title5		01.03.000 System Test	Medium	Low	Dev-Review	129	
Type:	Gap in requirements Definition	Found By: Kaminsky, Mark	Identified On: 3/26/2010	Opened On:03/26/2010	Closed On:			
00000372	Some title6		01.03.000 UAT	Medium	Low	Dev-Review	75	
Type:	Intersystem Interface	Found By: Kaminsky, Mark	Identified On: 5/19/2010	Opened On:04/02/2010	Closed On:			
00000035	Some title7		01.03.000 UAT	Low	High	Closed	108	
Type:	Data Element Problem	Found By: Kaminsky, Mark	Identified On: 2/22/2010	Opened On:02/22/2010	Closed On: 06/10/2010			
00000413	Some title8		01.04.010 System Test	High	High	Re-Test	76	
Type:	User Interface	Found By: TestTeam user1	Identified On: 5/18/2010	Opened On:05/18/2010	Closed On:			

1 of 2

### 7.3.4 Defect Summary by Severity

**Summary:** Summary of defects grouped by Release, Phase, Severity and State.

**Filter:** Uses Query “Defect-Summary-by-Severity”

**Fields to Display:** ID, Title, State, Phase, Severity, and Re-Test Count.

**Sort Order:** Target Release, Phase, Severity, State.

**Total/Subtotal:** Defect count for each severity level, defect count for each state, defect count for each phase, defect count for each target release, total defect count.

MyStartingLine Defect Summary Report by Severity						Federal Student Aid
As of: 08/09/10						
ID	Title	Release	Phase	Severity	State	Re-test Count
Release = 01.00.004						
Phase = System Test						
Severity = Low						
State = Closed						
00000475	Defect synopsis	01.00.004	System Test	Low	Closed	0
Total Defects in Release 01.00.004 in Phase System Test where the Severity=Low and the State=Closed : 1						
Total Defects in Release 01.00.004 in Phase System Test where the Severity=Low : 1						
Total Defects in Release 01.00.004 in Phase System Test : 1						
Total Defects in Release 01.00.004 : 1						
Release = 01.01.000						
Phase = UAT						
Severity = Low						
State = Dev-Review						
00000382	Some title1	01.01.000	UAT	Low	Dev-Review	
Total Defects in Release 01.01.000 in Phase UAT where the Severity=Low and the State=Dev-Review : 1						
Total Defects in Release 01.01.000 in Phase UAT where the Severity=Low : 1						
Severity = Medium						
State = Re-Test						
00000370	Some title2	01.01.000	UAT	Medium	Re-Test	
Total Defects in Release 01.01.000 in Phase UAT where the Severity=Medium and the State=Re-Test : 1						
Total Defects in Release 01.01.000 in Phase UAT where the Severity=Medium : 1						
Total Defects in Release 01.01.000 in Phase UAT : 2						
Total Defects in Release 01.01.000 : 2						
Release = 01.01.001						
Phase = System Test						
Severity = High						
State = Re-Test						
00000435	Some title4	01.01.001	System Test	High	Re-Test	

1 of 4

**Appendix A: Acronyms and Abbreviations**

## Appendix A: Acronyms and Abbreviations

Acronym	Definition
ATP	Authorization To Proceed
CCB	Change Control Board
CI	Configuration Identification/Item
CM	Configuration Management
CMRB	Configuration Management Review Board
COTS	Commercial Off The Shelf
CR	Change Request
ECCB	Enterprise Change Control Board
ECM	Enterprise Configuration Management
ECMM	Enterprise Configuration Management Methodology
EOCM	Enterprise Operational Change Management
FCA	Functional Configuration Audit
FSA	Federal Student Aid
GFE	Government Furnished Equipment
ITIL	Information Technology Infrastructure Library
NIST	National Institute of Standards and Technology
PA	Process Audit
PCA	Physical Configuration Audit
PM	Program Manager
QA	Quality Assurance
SCC	Standard Change Control
SE	Systems Engineering
SEI	System Engineering Institute
SR	Service Request

<b>Acronym</b>	<b>Definition</b>
UAT	User Acceptance Test
V&V	Verification and Validation
VDC	Virtual Data Center

**Appendix B: Glossary**

## Appendix B: Glossary

Term	Definition
Baseline	<p>A Benchmark used as a reference point. For example: An ITSM Baseline can be used as a starting point to measure the effect of a Service Improvement Plan.</p> <p>A Performance Baseline can be used to measure changes in Performance over the lifetime of an IT Service.</p> <p>A Configuration Management Baseline can be used to enable the IT Infrastructure to be restored to a known Configuration if a Change or Release fails.</p>
Capability Maturity Model Integration (CMMI)	<p>Capability Maturity Model® Integration (CMMI) is a process improvement approach developed by the Software Engineering Institute (SEI) of Carnegie Mellon University. CMMI provides organizations with the essential elements of effective processes. It can be used to guide process improvement across a project, a division, or an entire organization. CMMI helps integrate traditionally separate organizational functions, set process improvement goals and priorities, provide guidance for quality processes, and provide a point of reference for appraising current processes.</p>
Change	<p>The addition, modification or removal of anything that could have an effect on IT Services. The Scope should include all IT Services, Configuration Items, Processes, Documentation etc.</p>
Change Management	<p>The Process responsible for controlling the Lifecycle of all Changes. The primary objective of Change Management is to enable beneficial Changes to be made, with minimum disruption to IT Services.</p>
Change Control Board (CCB)	<p>Board that makes decisions regarding whether or not proposed changes to a software project should be implemented. The change control board is constituted of project stakeholders or their representatives. The authority of the change control board may vary from project to project, but decisions reached by the change control board are often accepted as final and binding.</p>
Configuration	<p>A generic term, used to describe a group of Configuration Items that work together to deliver an IT Service, or a recognizable part of an IT Service. Configuration is also used to describe the parameter settings for one or more CIs.</p>
Configuration Identification (CI)	<p>The Activity responsible for collecting information about Configuration Items and their Relationships, and loading this information into the CMDB. Configuration Identification is also responsible for labeling the CIs themselves, so that the corresponding Configuration Records can be found.</p>
Configuration Item (CI)	<p>Any Component that needs to be managed in order to deliver an IT Service. Information about each CI is recorded in a Configuration Record within the Configuration Management System and is maintained throughout its Lifecycle by Configuration Management. CIs are under the control of Change Management. CIs typically include IT Services, hardware, software, buildings, people, and formal documentation such as Process documentation and SLAs</p>

Term	Definition
CI Type	A Category that is used to Classify CIs. The CI Type identifies the required Attributes and Relationships for a Configuration Record. Common CI Types include: hardware, Document, User etc.
Configuration management (CM)	<p>The Process responsible for maintaining information about Configuration Items required delivering an IT Service, including their Relationships. This information is managed throughout the Lifecycle of the CI. Configuration Management is part of an overall Service Asset and Configuration Management Process.</p> <p>Configuration management (CM) is a field of management that focuses on establishing and maintaining consistency of a product's performance and its functional and physical attributes with its requirements, design, and operational information throughout its life. For information assurance, CM can be defined as the management of security features and assurances through control of changes made to hardware, software, firmware, documentation, test, test fixtures, and test documentation throughout the life cycle of an information system.</p>
Configuration Management Data Base (CMDB)	<p>A database used to store Configuration Records throughout their Lifecycle. The Configuration Management System maintains one or more CMDBs, and each CMDB stores Attributes of CIs, and Relationships with other CIs.</p> <p>A configuration management database (CMDB) is a repository of information related to all the components of an information system. Although repositories similar to CMDBs have been used by IT departments for many years, the term CMDB stems from ITIL (Information Technology Infrastructure Library). In the ITIL context, a CMDB represents the authorized configuration of the significant components of the IT environment. A key goal of a CMDB is to help an organization understand the relationships between these components and track their configuration. The CMDB is a fundamental component of the ITIL framework's Configuration Management process. CMDB implementations often involve integration with other systems, such as Asset Management Systems. These integrations may make use of either a real-time, federated design or an ETL (extract, transform, and load) solution.</p>
Custom Off The Shelf (COTS)	Application software or Middleware that can be purchased from a Third Party.
Enterprise Change Control Board (ECCB)	The official Federal Student Aid committee authorized to review, approve or reject enterprise operational changes. The ECCB oversees the Enterprise Operational Change Management (EOCM) process and supporting tools and reviews the Enterprise Master Release Schedule.
Enterprise Operational Change Management (EOCM)	The EOCM process is an enterprise-level process that operates in parallel to system/application specific CCBs within Federal Student Aid. This process includes clearly defined intersection points with a generic life cycle to ensure that communication and coordination of Enterprise Events occur across Federal Student Aid
Functional Configuration Audit (FCA)	A functional configuration audit ensures that functional and performance attributes of a configuration item are achieved

Term	Definition
Information Technology Infrastructure Library (ITIL)	<p>The Information Technology Infrastructure Library (ITIL) is a set of concepts and policies for managing information technology (IT) infrastructure, development and operations.</p> <p>ITIL is published in a series of books, each of which covers an IT management topic. The names ITIL and IT Infrastructure Library are registered trademarks of the United Kingdom's Office of Government Commerce (OGC). ITIL gives a detailed description of a number of important IT practices with comprehensive checklists, tasks and procedures that can be tailored to any IT organization.</p>
National Information Standards Technology (NIST)	<p>A measurement standards laboratory which is a non-regulatory agency of the United States Department of Commerce. The institute's mission is to promote U.S. innovation and industrial competitiveness by advancing measurement science, standards, and technology in ways that enhance economic security and improve quality of life.</p>
Physical Configuration Audit (PCA)	<p>A physical configuration audit ensures that a configuration item is installed in accordance with the requirements of its detailed design documentation.</p>
Process Audit (PA)	<p>Ensures that all required configuration management processes were correctly executed and appropriate process documentation has been maintained.</p>
Program Management (PM)	<p>The process of managing multiple interdependent projects that lead towards an improvement in an organization's performance.</p>
Quality Assurance (QA)	<p>The Process responsible for ensuring that the Quality of a product, Service or Process will provide its intended Value.</p>
Service Management (SM)	<p>Service Management is a set of specialized organizational capabilities for providing value to customers in the form of services.</p>
Service Request (SR)	<p>A request from a User for information, or advice, or for a Standard Change or for Access to an IT Service. For example to reset a password, or to provide standard IT Services for a new User. Service Requests are usually handled by a Service Desk, and do not require an RFC to be submitted. See Request Fulfillment.</p>
Standard Change Control (SCC)	<p>Formal process used to ensure that changes to a product or system are introduced in a controlled and coordinated manner. It reduces the possibility that unnecessary changes will be introduced to a system without forethought, introducing faults into the system or undoing changes made by other users of software.</p>
System Engineering (SE)	<p>An interdisciplinary field of engineering that focuses on how complex engineering projects should be designed and managed.</p>

<b>Term</b>	<b>Definition</b>
System Engineering Institute (SEI)	The Carnegie Mellon Software Engineering Institute (SEI) is a federally funded research and development center headquartered on the campus of Carnegie Mellon University in Pittsburgh, Pennsylvania, United States. SEI also has offices in Arlington, Virginia, and Frankfurt, Germany. The SEI operates with major funding from the U.S. Department of Defense. The SEI also works closely with industry and academia through research collaborations. The SEI program of work is conducted in three principal areas: management practices, engineering practices, and acquisition practices.
User Acceptance Test (UAT)	Process to obtain confirmation by a Subject Matter Expert (SME), preferably the owner or client of the object under test, through trial or review, that the modification or addition meets mutually agreed-upon requirements. In software development, UAT is one of the final stages of a project and often occurs before a client or customer accepts the new system.
Verification and Validation (V&V)	Process of checking that a software system meets specifications and that it fulfills its intended purpose. It is normally part of the software testing process of a project.

**Appendix C: ClearQuest Report Samples**





## CCB Review Report

MyStartingLine Configuration Control Board Review		Federal Student Aid	
As of: 06/10/10			
ID	Title		
00000338	sample defect from rqm Description: sample from rqm	Opened On: 03/18/2010 State: Impact_Analysis Opened By: Kaminsky, Mark Requested By: Kaminsky, Mark	Requested Priority: Emergency Assigned Priority: Routine Category: Correction Found In Release: 01.08.000 Target Release: 01.05.000
Impact Analysis			
General Impact Analysis:		Requirements Impact: Testing UCM integration	
Development Impact: Testing UCM Integration		Security Impact: Testing UCM Integration	
00000404	Add support for up to 4 shipping addresses Description: The current release of the application only supports one shipping address that is different from the billing address. Suggest that the same customer may wish to send purchases to multiple destinations	Opened On: 05/14/2010 State: Impact_Analysis Opened By: DevTeam user2 Requested By: DevTeam user2	Requested Priority: Routine Assigned Priority: Target Release: 01.00.004
Impact Analysis			
General Impact Analysis:		Requirements Impact: zzzzz	
Development Impact: zzzzz		Security Impact: zzzzz	

## CCB Target Release Report

MyStartingLine Configuration Control Board Targeted Release		Federal Student Aid		
As of: 06/16/10				
ID	Title			
0000057	Add support for up to 5 shipping addresses	Opened On: 03/08/2010	Assigned Priority: Urgent	Complexity: Unknown
Description: The current release of the application only supports one shipping address that is different from the billing address. Suggest that the same customer may wish to send purchases to up to 5 multiple destinations		State: Ready_Sys_Testing	Found In Release: 01.01.000	Assigned Developer: DevTeam user1
		Opened By: TestTeam user2	Target Release: 01.06.000	
		Requested By: TestTeam user2		
Associated Requirements:				
Associated Components:				
Related Change Requests:				
00000312	Link to a JSP on thestartingline site	Opened On: 12/29/2009	Assigned Priority: Urgent	Complexity: Medium
Description: A user is trying to access a jsp page from thestartingline.gov website.		State: Implementation	Found In Release: 01.04.010	Assigned Developer: RSG Test Account
The Data Analysis link on PCNet no longer works. I've been relying on this when I need help with PEPS queries. When I clicked on the data analysis link today, it took me here:		Opened By:	Target Release: 01.06.000	
http://thestartingline.ed.gov/INTRANETWebApp/FSAnetApps/down.jsp?URI=%22application_eligibility_delivery/school_eligibility/data_analysis/%22&QUERY=%22		Requested By: Rong Xu		
Associated Requirements:				
Associated Components:				
Related Change Requests:				

## Child CR Summary Report

### MyStartingLine Child Change Request Summary

### Federal Student Aid

As of: 06/16/10

ID	Related CR	Title	Found In Release	Target Release	Opened On	Assigned Developer	State
00000423	00000330	Update the Help Text		01.01.001		Kaminsky, Mark	Sys_Testing
00000407	00000405	A new ChildCR	01.06.000	01.04.000		DevTeam user1	Closed
00000416	00000415	Sample Child CR	01.04.000	01.04.000		Kaminsky, Mark	Closed
00000406	00000405	Son of CR 406	01.06.000	01.04.000		DevTeam user2	Development
00000022	00000019	Testing 3.14	01.04.001	01.04.001		CM Team user	Ready_Sys_Testing
00000383	00000349	Update the graphical user interface	01.04.001	01.04.010		DevTeam user1	Development
00000384	00000349	Apply changes to the backend Oracle database	01.04.001	01.04.010		DevTeam user2	Development
00000423	00000349	Update the Help Text	01.04.001	01.04.010		Kaminsky, Mark	Sys_Testing
00000053	00000041	Do X	01.04.001	01.05.000		DevTeam user1	Development
00000428	00000426	06/03/10: Another CR #2	01.01.000	01.06.000		Betteann Sturup	Closed
00000432	00000426	06/03/10: Sample for testing reports	01.01.000	01.06.000		Shah, Maulikkumar	Closed
00000427	00000426	6/3/10: Child CR Sample	01.01.000	01.06.000		Kaminsky, Mark	Development
00000425	00000319	testing another one	00.00.000	01.06.000		Kaminsky, Mark	Development
00000451	00000426	testing submit date	01.01.000	01.06.000	06/16/2010	Kaminsky, Mark	Development
00000437	00000426	Test	01.01.000	01.06.000		Betteann Sturup	UAT

Child Change Request Total: 14



**Appendix D: Security**

## Appendix D: Security

This appendix provides additional security details relating to Rational Quality Manager, ClearQuest and ClearCase.

### RQM Security

The table below shows various RQM objects and the permissions to operate on those objects that have been granted to the RQM users.  
NOTE: Leads will always be included in the related team group. For example, the Test Lead for a system development effort will be included in the Test-Team group.

RQM Artifact	Action	Test Lead	Test Team	CM Lead	CM Team	Dev Lead	Dev Team	Rqmts. Lead	Rqmts. Team	FSA PM	Project PM	FSA RSG Admin
Requirements	Import into Test Plan	X										X
	View		X		X		X		X	X	X	X
Requirements Work Item (should never be used)	View											
	Create											
	Edit											
	Delete											
	Archive											
Test Plan	View		X		X		X		X	X	X	X
	Create	X										X
	Edit	X										X
	Delete			X								X
	Archive				X							X
	Create Section											X

RQM Artifact	Action	Test Lead	Test Team	CM Lead	CM Team	Dev Lead	Dev Team	Rqmts. Lead	Rqmts. Team	FSA PM	Project PM	FSA RSG Admin
	Remove Section											X
	Snapshot				X							X
Test Schedule	Create	X										X
	Edit	X										X
Test Case	View		X		X		X		X	X	X	X
	Create		X									X
	Edit		X									X
	Delete		X	X								X
	Archive				X							X
	Execute		X									X
	Create Section											X
	Remove Section											X
	Snapshot					X						
<b>Category Types for Test Plan and Test Case</b> **Test Plan Examples include: Release, Test Phase. ** No one other than the FSA RSG group should be able to create new categories or remove existing categories.	View		X		X		X		X	X	X	X
	Create											X
	Edit											X
	Delete											X
	Archive											X
<b>Categories (Test Plan and Test Case)</b> Represents the choice lists for the defined categories. For example: The choice list for the category "Test	View		X		X		X		X	X	X	X
	Create			X								X
	Edit			X								X
	Delete			X								X

RQM Artifact	Action	Test Lead	Test Team	CM Lead	CM Team	Dev Lead	Dev Team	Rqmts. Lead	Rqmts. Team	FSA PM	Project PM	FSA RSG Admin
Phase” will initially contain “UAT” and “System” however it may be necessary to add other test phases.	Archive				X							X
Test Plan/Case Templates	Create											X
	Edit											X
	Archive											X
	Set Default											X
Test Script	Create		X									X
	Edit		X									X
	Archive				X							X
Test Data	Create		X									X
	Edit		X									X
	Archive				X							X
Build Definition	View		X		X		X		X	X	X	X
	Create				X							X
	Edit				X							X
	Delete			X								X
	Archive				X							X
Build Record	View		X		X		X		X	X	X	X
	Create				X							X
	Edit				X							X
	Delete			X								X
	Archive				X							X
Execution Results	View		X		X		X		X	X	X	X
	Save		X									X
	Edit		X									X

RQM Artifact	Action	Test Lead	Test Team	CM Lead	CM Team	Dev Lead	Dev Team	Rqmts. Lead	Rqmts. Team	FSA PM	Project PM	FSA RSG Admin
	Archive	X										X
Test Suite	Create		X									X
	Edit		X									X
	Archive				X							X

## RequisitePro Security

For every RequisitePro project, the following groups and associated privileges have been defined:

Group	Privileges
Requirements Team	Read and Write access to all requirement assets located in the RequisitePro project
Development Team	Read access to all requirement assets located in the RequisitePro project
Test Team	Read access to all requirement assets located in the RequisitePro project
Management Team	Read access to all requirement assets located in the RequisitePro project

## ClearQuest Security

ClearQuest records are transitioned from one state to another by various actions. In each state, users perform actions such as “Modify” to change the contents of a record or by moving it to another state. The actions menu lists the actions that can be performed on the record while it is in a given state, including assigning, modifying, and opening and resolving change requests.

### CR Actions Permission

The table below lists the actions which transition a CR from one state to another and the users that are permitted to perform the action.

Action	Permission	Description
Submit	Any project team member	MSL members use this action to create a record or request for change.
Approve_Initial_Review	CMTeam	The CM Team starts the preliminary review of the CR.
Approve_Impact_Analysis	CMTeam	The CM Team member determines that the CR is valid and should be analyzed further.
Approve_CCB	CMLead, DevLead	After an impact analysis has been completed, the CR is sent to the CCB for review by either the CM Lead or the Dev Lead.
Approve_Ready_Development	CMLead	The CCB approves the CR. This is a handoff to the Dev Lead.
Approve_Development	DevLead	Assigns the CR to a developer and transitions it to Development.
Approve_Ready_Sys_Testing	DevLead, CMLead	Development has completed so the CR is passed to the test lead.
Approve_Sys_Testing	CM Lead, DevLead	System Testing begins

Action	Permission	Description
Approve_Ready_UAT	CM Lead	System Testing completed and the CR is being staged for UAT.
Approve_UAT	CMTeam	UAT begins.
Approve_FLB	CMTeam	First Live Batch begins.
Approve_Close	CMTeam	First live batch completes and the CR is closed.
Approve_Cancel	CMLead	The CM Lead determines that the CR is not valid.
Approve_Hold	CMLead	The CM Lead determines that the CR should be placed on hold until specified conditions are met.
Reject_to_Development	Test Team, CMTeam	The CR fails one of the states of testing and is turned over to the development team.
Reject_to_Impact_Analysis	CMTeam	During the CCB Review, it is determined that the CR needs additional analysis.
To-Admin-Central-State		ClearQuest administrative use only.

### ChildCR Action Permission

The table below lists the actions which transition a ChildCR from one state to another and the users that are permitted to perform the action.

Action	Permission	Description
Submit	DevLead	The DevLead will submit a new ChildCR in order to subdivide the work associated with a CR across multiple developers.
Approve_Ready_SysTesting	CMTeam	A CM Team member indicates that the ChildCR is ready for system testing
Approve_Sys_Testing	CMLead	The CM Lead approves the ChildCR to be system tested.
Approve_Ready_UAT	CMTeam	A CM Team member indicates that the ChildCR is ready for user acceptance testing.
Approve_UAT	CMTeam	The CM Lead approves the ChildCR to undergo user acceptance testing.

Action	Permission	Description
Close	CMTeam	The ChildCR passes user acceptance testing and is closed
Reject_to_Development	CMTeam	The ChildCR is rejected back to development because it either fails one of the test phases or it is determined that it is not ready to move into testing.
To-Admin-Central-State	RatlAdmin	***Rational Administrator use only***

### Defect Action Permissions

The table below lists the actions which transition a Defect from one state to another and the users that are permitted to perform the action.

Action	Permission	Description
Submit	TestTeam	A test team member creates a defect from Rational Quality Manger when a test case fails.
Approve-to-Review	DevLead, TestLead	The defect is determined to be valid and therefore move into a development review state. At this point, the defect will be corrected via modification to the application. NOTE: A CR or ChildCR will be linked to the defect.
Approve-to-re-Test	DevLead	The defect has been corrected and is ready to be retested.
Approve-to-Close	TestTeam	The test passed and the defect can be closed.
Cancel	TestLead	The defect is determined to be invalid.
Reject-to-Dev-Review	TestTeam	The defect failed the test case so it is sent back to development to be worked on.

### ClearCase Security

The security protocols pertaining to the use of ClearCase that have been established are as follows:

- **VOB Access:** Read and write access to elements stored within the system's ClearCase VOBs is controlled by FSA active directory group membership. For each VOB, there will be two groups; one which allows read access and one which allows write access:
  - <system name>\_<abbreviated VOB name>
  - <system name>\_<abbreviated VOB name>\_w
- **System name:** This is the abbreviated name for the system being developed. For example: UWF for the fictitious project Universal Widget Foundation.
- **Abbreviated VOB name:** a short name for the VOB. For example: The abbreviated VOB name "Doc" would be used for the VOB \UWF\_Documentation

A complete example is as follows:

System UWF has 2 contractors, C1 and C2, supporting the development effort. Each are working on a different component of the system in separate VOBs. C1 is working on the Database for the system and C2 is working on the graphical user interface. In this example we have the following 3 VOBs:

\UWF\_Documentation: stores the project documentation

\UWF\_Database: The C1 developers are working on the DB related code for the system.

\UWF\_GUI: The C2 developers are working on the graphical user interface for the system.

Given this scenario, we have the following groups:

uwf: top level group containing all project team members

uwf\_doc: membership indicates read access to the \UWF\_documentation VOB

uwf\_doc\_w: membership indicates write access to the \UWF\_documentation VOB

uwf\_DB: membership indicates read access to the VOB \UWF\_Database

uwf\_DB\_w: membership indicates write access to the VOB \UWF\_Database

uwf\_GUI: membership indicates read access to the VOB \UWF\_GUI

uwf\_GUI\_w: membership indicates write access to the VOB \UWF\_GUI

The table below lists logical groups for a system as well as read vs. write access to multiple ClearCase VOBs used in the system development effort.

Logical Groups	Read Access to System Documentation VOB	Write Access to System Documentation VOB	Read Access to Contractor 1's System Software VOB(s)	Write Access to Contractor 1's System Software VOB(s)	Read Access to Contractor n's System Software VOB(s)	Write Access to Contractor n's System Software VOB(s)
CM Team	X	X	X	X	X	X
Development Team	X		X	X	X	

Logical Groups	Read Access to System Documentation VOB	Write Access to System Documentation VOB	Read Access to Contractor 1's System Software VOB(s)	Write Access to Contractor 1's System Software VOB(s)	Read Access to Contractor n's System Software VOB(s)	Write Access to Contractor n's System Software VOB(s)
(Contractor 1)						
Development Team (Contractor N+1)	X		X		X	X
FSA Management	X		X		X	
Operations Team	X					
Requirements Team	X		X		X	
Security Team	X		X		X	
Test Team	X		X		X	

### CM Policy Enforcement

The following rules are enforced within the FSA Enterprise Solution:

- Only CM team members will be permitted to deliver changes to the project integration stream.
- Only CM team members will be permitted to deliver changes to the release-specific integration stream.
- Only CM Team members will be permitted to create baselines on the project integration stream.
- Only CM Team members will be permitted to create baselines on the release-specific integration stream.
- ClearCase UCM baselines (labels) may be created on the release-specific development stream by the development team member.
- User specific child stream off any of the prescribed streams will NOT be permitted.