Main

Contents

Collaboration wiki
- Build Forge/RAFW
- DOORS Next
- Insight
- RAM
- RQM
- RRC
- RTC
-- RTC for System z
-- RTC for Power
- SCD

---------------

- Collaborative ALM
- Design Management
- Jazz Foundation
- Jazz Integration Architecture

Reference wiki

Hello qsuron!
Log Out
Create personal sidebar

Main Web
Create New Topic
Index
Search
Changes
Notifications
Statistics
Preferences

Trash

You are here: TWiki > Main Web > RQMExcelWordImporter r107 - 2014-03-20 - 13:36:41 - PaulSlauenwhite

NOTE: The development wiki is a work area used by the Jazz development teams to plan and discuss technical designs and operational procedures related to the development projects at Jazz.net. Often you will find work items linking to documents in the wiki. Everyone is welcome to browse the wiki, follow along, and participate in the development process. Participation is what Jazz.net is all about! But please keep in mind that information on the wiki is "as is", unsupported, and may be outdated or inaccurate. For information on released products, consult the product documentation, support tech notes, and the Jazz.net library. See also the Jazz.net Terms of Use.

# Introduction

Microsoft Excel and Word are popular document formats used by many test organizations to author, store and maintain their test assets. These document formats are easy to use and widely available. RQM provides a utility that allows the Excel and Word applications to export test data to RQM via HTTP. Using this utility these applications can directly export test artifacts as XML file(s) in a well-defined XML format. This XML file(s) can be imported into RQM using the RQM Reportable REST API.

From the outside it might appear that a smart application could read the data from Excel or Word documents and simply convert that data into XML. Once in XML that data could be delivered directly to RQM. However, the problem is not that simple. The content of these document types can not be associated with RQM data types and fields without assistance.

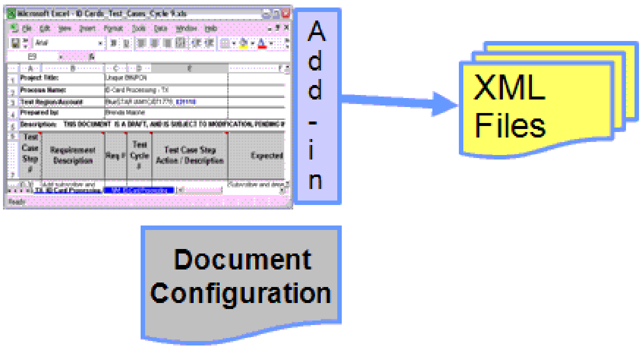
This RQM utility takes these considerations into account :

- Multiple artifact types (Test Scripts, Test Cases, Test Plans, etc)
- Relationships between record types in a single document
- Both Excel and Word document types

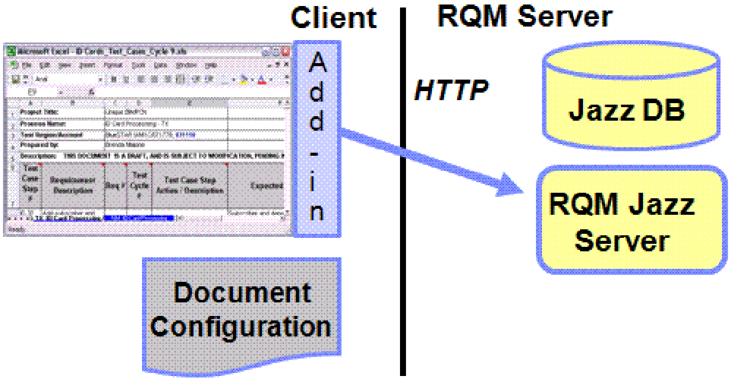
These constraints require a solution that is extensible and customizable, which is why the utility uses a document specific configuration.

The Excel and Word add-ins provide new features to these applications. The add-ins provide the following main features :

- Export the document(s) as XML files on the local file system ('Export to File')



- Export the document(s) onto the RQM repository, creating corresponding test artifacts on the RQM side ('Export to Repository')



# Recent Changes

- Please review changes to the Install Guide section.
- General improvements.
- What's new in 3.0.1
- What's new in 4.0
- Excel/Word Importer Best Practice -- **New!**

## Compatibility

The RQM Excel/Word Importer is built on the RQM Reportable REST API, which is **not** compatible between Rational Quality Manager versions.

## Permissions

The RQM Excel/Word Importer uses RQM Reportable REST API GET/PUT requests, which requires specific project-level permissions.

## Document Configuration:

The configuration describes the content of the documents being exported to RQM. This description lets this utility properly map the document content to fields in RQM records. That mapping allows coherent XML generation for import to RQM.

Note that sample configuration files are available with the installed product along with the documents that they represent. These configurations are fabricated from customer submitted specifications and tested with each release to ensure they function as designed. While configuration files can live anywhere on the file system, as a convenience it is best to align them with the documents they describe.

The sample configuration files and associated documents are located in:

RQM Excel Import Utility Samples:

```
<RQM Excel Import Utility installation directory>\Samples
```

By default, the RQM Excel Import Utility is installed in:

```
C:\Program Files\IBM\RQMExcelImporter
```

RQM Word Import Utility Samples:

```
<RQM Word Import Utility installation directory>\Samples
```

By default, the RQM Word Import Utility is installed in:

```
C:\Program Files\IBM\RQMWordImporter
```

# Install Guide

## Prerequisites

The following software is required to be installed on the computer where the Excel / Word export utility is to be installed.

- Microsoft .NET framework - v2.0 or later
- Microsoft Excel 2003 or later
- Microsoft Word 2003 or later
- Microsoft Word/Excel 2010 or later 64bit. (supported from Excel/Word Importer 4.0)
- Requires write permissions to the log directory `C:\Documents and Settings\<USER>\Application Data\Mso2Rqm`
- Requires write permissions to the registry, and permissions to do HTTP request.
- *** the best practice is to use the user with the admin permissions on windows to do the migration.**

- Starting in **CLM 4.0**, when creating a RM project area linked to a RQM project area, check "**Use a template to initially populate the project.**" and choose "Base". In addition, in the RQM Project Area management and the Overview tab, add the current user ID to the members section and "Data Migration Administrator" role has been added to the user.

* Note: For Windows XP users, please install .NET Framework before Office gets intalled. Otherwise the RQM Importer Add-in may not be loaded properly by Office. If for whatever reasons Office is installed first, please install Office Primary Interop Assemblies by using a redistributable package by looking into this page How to: Install Office Primary Interop Assemblies

## Add-In Installation

1. `NEW` – You must first install the Microsoft interop support onto your system. Please refer to **`How to: Install Office Primary Interop Assemblies`** for instructions on installing this support.
   - Note that the prerequisite for the interop assemblies is .Net framework v2.0, however choosing to install the most recent version is the only alternative in the Software Updater and is compatible with the installation of the interop assemblies.
   - Note also that the .Net framework v1.1 should also be installed from the Software Updater if not already available. This version is needed by the interop assemblies.

2. Extract the installation files from `RQM-Extras-RQMExcelWordImporter-<version>.zip` to a temporary location.
   - `RQM-Extras-RQMExcelWordImporter-<version>.zip` is available in the **Extras** section of the **All Downloads** tab on the Rational Quality Manager `<version>` download page.
   - Note: `RQM-Extras-RQMExcelWordImporter-3.0.1.zip` is compatible with a Rational Quality Manager 2.x server.

3. **cd** to the temporary location
4. Run the install setup file, **`RQMExcelImporterSetup.exe`** for Excel and **`RQMWordImporterSetup.exe`** for Word
5. Verify that the **`RQM`** Add-in (the **`RQM`** menu) is available with the chosen utility

Note: From RQM 4.0.5, RQM Word/Excel importer starts supporting TLS 1.2. To use this feature, users need to install .NET Framework 4.5 or above and install RQM Word/Excel importer with the installer named RQMExcelImporterSetupForDotNET45

## Removing the RQM Add-in

1. Open **Control Panel > Add or Remove Programs**
2. Select **`RQMExcelImporter`** or **`RQMWordImporter`**
3. Click the **Remove** button and confirm desire to remove the application
4. Delete all the log files in the log directory `C:\Documents and Settings\<USER>\Application Data\Mso2Rqm`

- Note: Windows 7 users will find the log directory at `C:\Users\<USER>\AppData\Roaming\Mso2Rqm`

## RQM menu does not appear in Excel or Word after installation?

Although the installation goes through without a hiccup, sometimes the RQM menu does not appear. This is mainly due to the fact that Microsoft Office sometimes fails to load the add-ins, if the add-in is built upon .NET framework v2.0 or above. See the article http://support.microsoft.com/kb/948461 for more details.

Alternatively, you can try the following options :

- Install Microsoft Office 2003 SP3 on top of Microsoft Office 2003
- Upgrade to Microsoft Office 2007 or 2010

## RQM popup does not appear in Excel or Word when clicking on a menu entry

After clicking on a menu entry it could that the popup does not appear. This is an issue with the locale setting on the machine. Importer works for English only and using other languages may result in unexpected behaviors. When meet this issue, please try below actions:

**Windows 7**

Go to Control Panel -> Region and Language -> Formats, change Format to 'English (United States) '. Also change

Location to 'United States'.

**Windows XP**

Change the locale settings to 'United States' by changing the value under Control Panel -> Regional and Language Options -> Location to 'United States'. Additionally change the 'Standards and formats' to 'English (United States)'.

# User Experience

## Overview

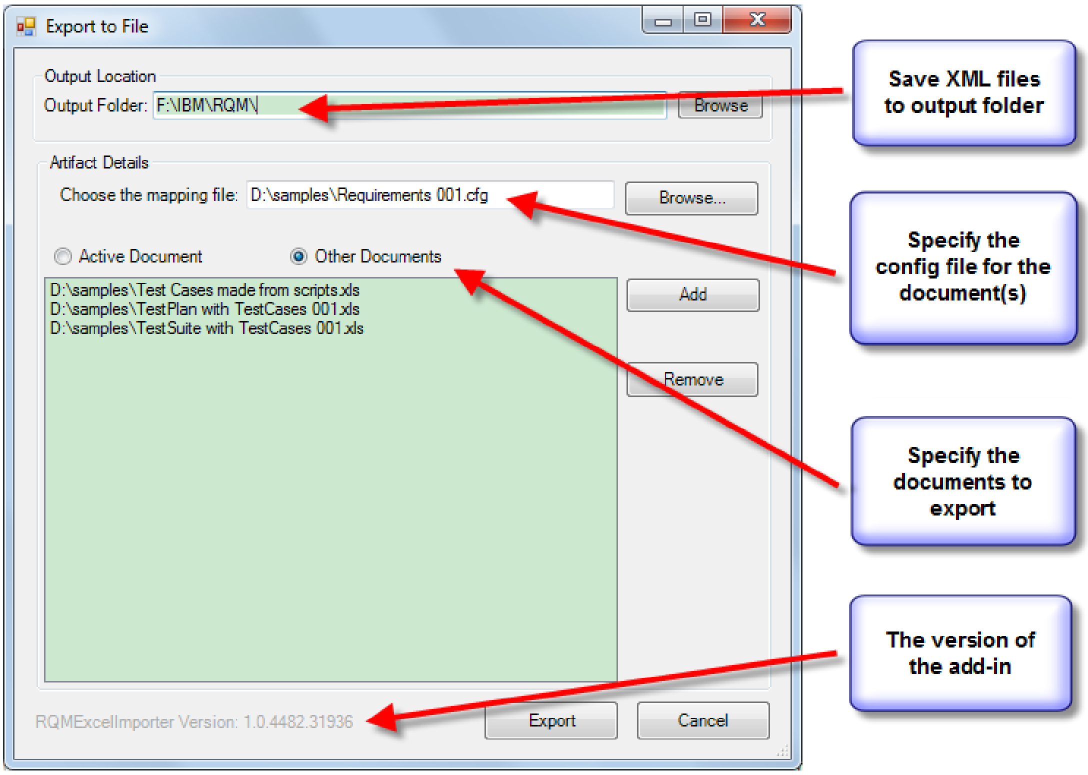
The Excel / Word Export utility has two action :

- **Export to File**
- **Export to Repository**

The Excel and Word add-ins contribute a new menu called ‘RQM’ that has the above options.

## Export to File

Clicking on the menu option ‘RQM > Export to File’ shows the following dialog.



Output Folder: F:\IBM\RQM\

Choose the mapping file: D:\samples\Requirements 001.cfg

○ Active Document ● Other Documents

D:\samples\Test Cases made from scripts.xls
D:\samples\TestPlan with TestCases 001.xls
D:\samples\TestSuite with TestCases 001.xls

RQMExcelImporter Version: 1.0.4482.31936

Save XML files to output folder

Specify the config file for the document(s)

Specify the documents to export

The version of the add-in

The user can export the currently open document (by selecting the option ‘Active Document’), or export one or more other documents (by selecting the option ‘Other Documents’), which provides the capability for a user to export multiple documents at once.From 4.0.0.1 Word Importer can add files with the extension ".doc", ".docx", ".rtf" and for Excel Importer, support add files with the extension ".xls", ".xlsx".

In either case, the user **must** select a configuration file that describes the content of the file being exported. The configuration file is required for the export utility to decipher the contents of the document.

**Hint :** the install includes several sample documents with associated configuration files that can be used as patterns to build your configuration file from.

In case of exporting multiple documents, the user can add one or more documents to be exported by clicking on the ‘Add’ button. The selected documents show up in the list. Also, the user can select one or more of these list elements and remove them by clicking on the ‘Remove’ button.

Finally, clicking on the ‘Export’ button will trigger the actual export operation. The selected documents are converted to XML based on content format described in the configuration file and saved as XML file(s) in:

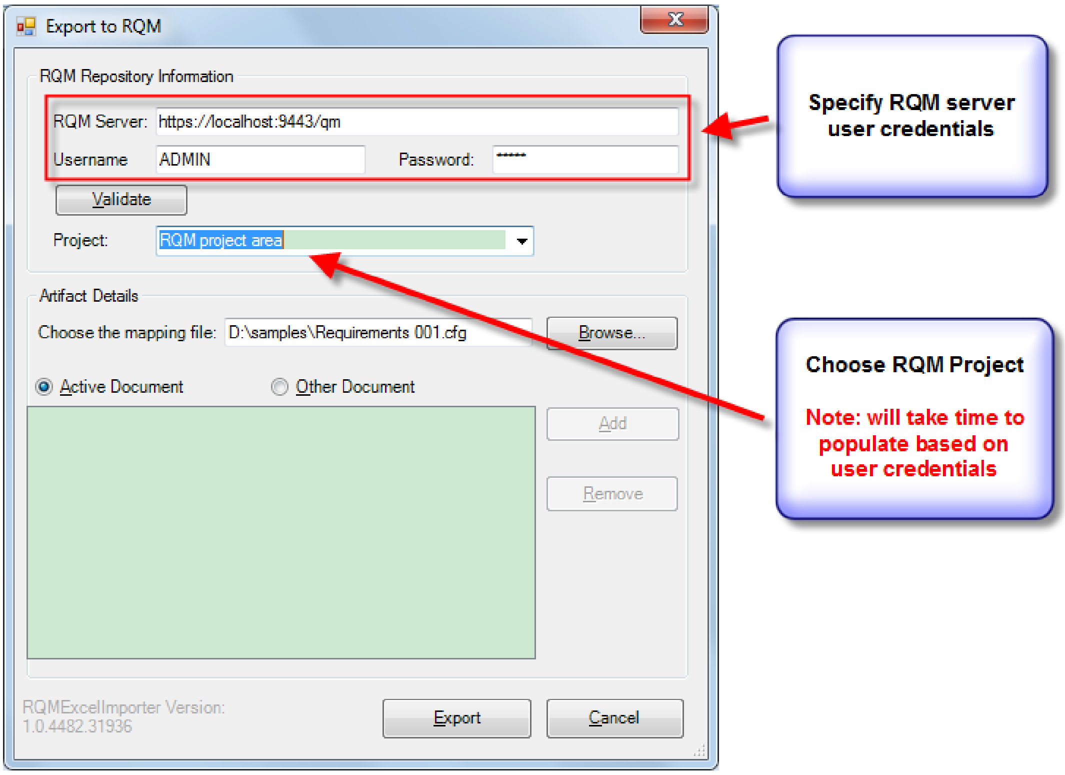
```
<output folder>\<resource>\< default external ID>
```

For example:

```
C:\excel export\executionresult\Execution Results_Sheet1.xml
```

## Export to Repository

Clicking on the menu option ‘RQM > Export to Repository’ shows the following dialog.



RQM Server: https://localhost:9443/qm

Username ADMIN Password: *****

Project: RQM project area

Choose the mapping file: D:\samples\Requirements 001.cfg

● Active Document ○ Other Document

RQMExcelImporter Version: 1.0.4482.31936

Specify RQM server user credentials

Choose RQM Project

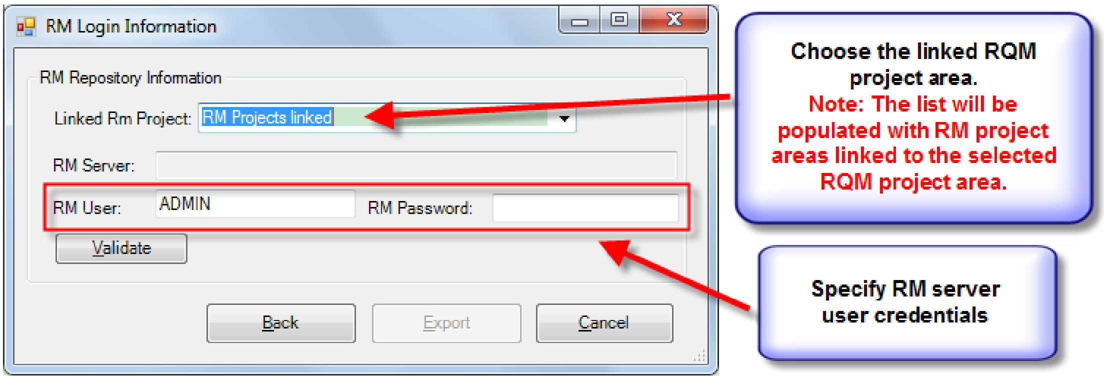
Note: will take time to populate based on user credentials

The behavior with regard to artifact selection is same as in the case of **Export to File** dialog. The primary difference being that this operation requires the RQM repository related details (the server URL, the port number) and the user credentials, instead of the output file location. When selecting the `Project` field the specified user credentials are utilized to dynamically populate this list. Note, if the user credentials are invalid or incorrect the `Project` selector will fail to populate.

Upon clicking ‘Export’ button, the selected artifacts are converted into the appropriate XML representation and sent over a HTTP connection to the RQM server, instead of being persisted on the local file system.

**Rational Quality Manager 4.0:**

When Requirement artifacts are included in export, clicking the Export button will display the RM Login Information dialog:



Linked Rm Project: RM Projects linked

RM Server:

RM User: ADMIN RM Password:

Choose the linked RQM project area.

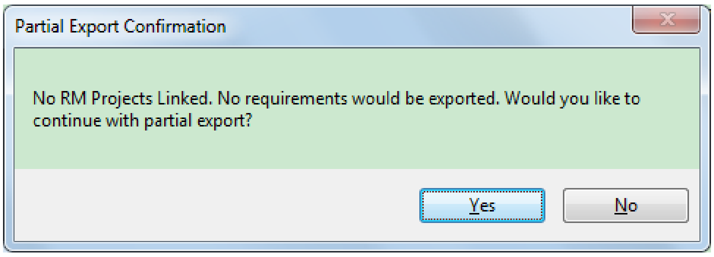
Note: The list will be populated with RM project areas linked to the selected RQM project area.

Specify RM server user credentials

Linked RM Projects drop down list will be auto-filled with list of RM projects that are linked to the selected QM project. Select desired RM project. Note that RM server field is a Read Only field. It will be auto-filled with server url of the selected RM project area.

Fill in the RM server username and password and click on ‘Export’ button. All the artifacts other than requirement will be exported to selected RQM server. Requirements will be exported to selected RM server.

If there are no linked RM project areas available, then the following confirmation dialog will be displayed:



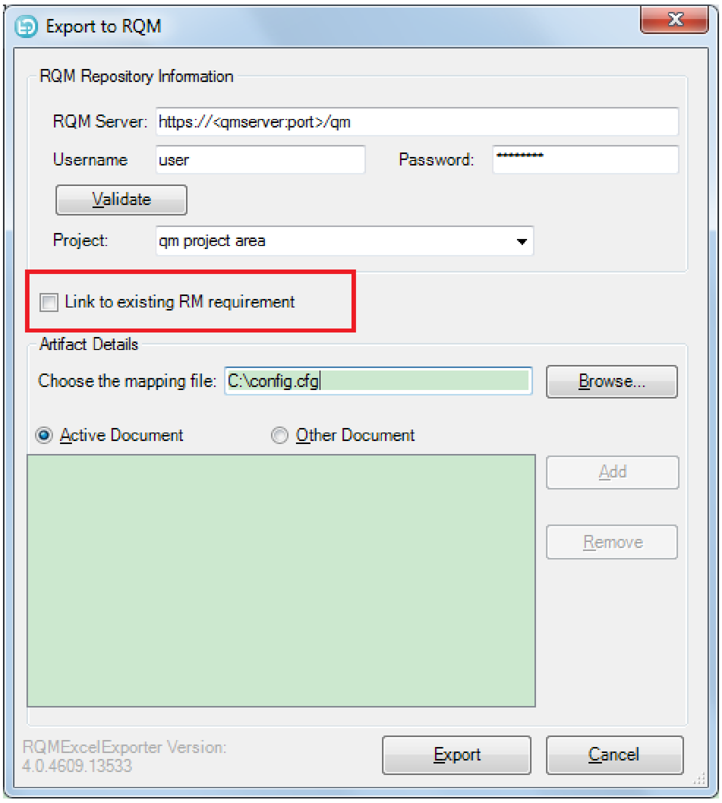
No RM Projects Linked. No requirements would be exported. Would you like to continue with partial export?

Choosing 'No' will cancel the export. ‘Yes’ will export all the artifacts other than requirements. Links to Requirements cannot created since the Requirements are not available.

**From RQM Excel Importer 4.0.1: supporting linking test artifact with existing RM requirement artifact.**

**Known limitation: time out problem when there are a large number(such as more than 500) of requirements on the RM server. (tracked in 88033)**

Clicking on the menu option ‘RQM > Export to Repository’ and check the "Linking to existing RM requirement" check box.



Export to RQM

RQM Repository Information

RQM Server: https://<qmserver:port>/qm

Username: user

Password: *********

Project: qm project area

□ Link to existing RM requirement

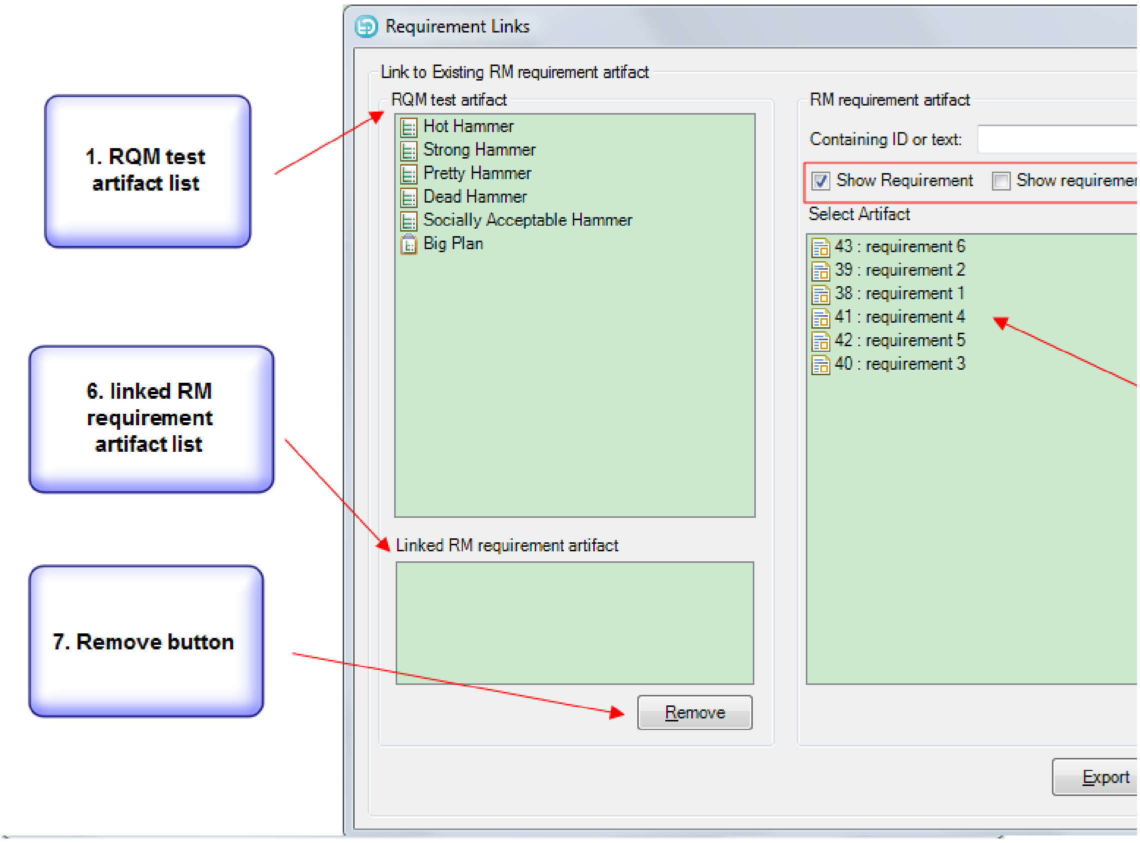
Artifact Details

Choose the mapping file: C:\config.cfg

● Active Document ○ Other Document

RQMExcelExporter Version: 4.0.4609.13533

press "Export" button, then "RM login information" dialog will display, input the correct RM server information then press "Export", the following Requirement links dialog will display.



Requirement Links

Link to Existing RM requirement artifact

RQM test artifact:
- Hot Hammer
- Strong Hammer
- Pretty Hammer
- Dead Hammer
- Socially Acceptable Hammer
- Big Plan

Linked RM requirement artifact

RM requirement artifact

Containing ID or text:

☑ Show Requirement □ Show requiremen

Select Artifact:
- 43 : requirement 6
- 39 : requirement 2
- 38 : requirement 1
- 41 : requirement 4
- 42 : requirement 5
- 40 : requirement 3

1. RQM test artifact list
6. linked RM requirement artifact list
7. Remove button
Remove
Export

1. "RQM test artifact" list will list all the testcases and testplans that created from the excel file.

2. "Search Box" provides a search function either by requirement id or text contained in the requirement title.

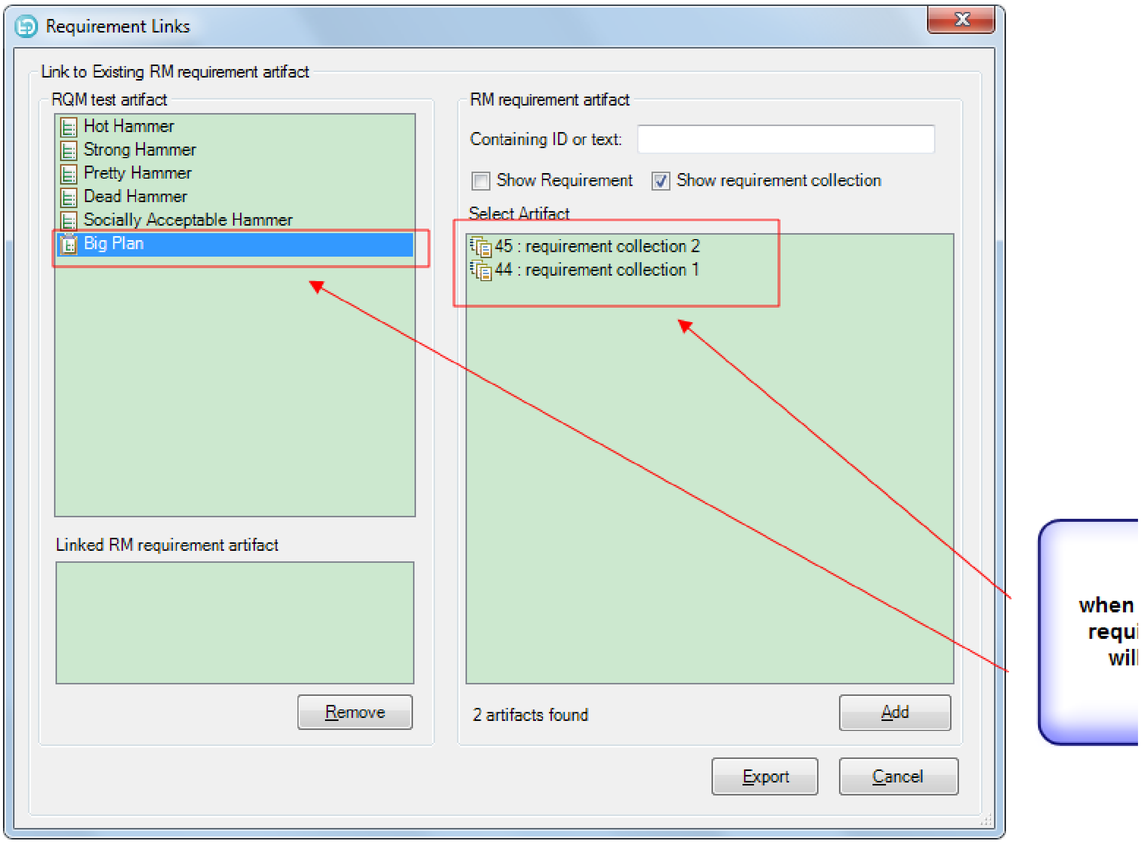
3. "Requirement and requirement collection check box" - when "Show requirement" is checked only requirement artifact will be listed, and when "Show requirement collection" is checked, only requirement collection will be listed, and when testcase in "RQM test artifact" list is selected, the default list will be requirement, and when testplan in "RQM test artifact" list is selected the requirement collection will listed by default.

4. "Requirement / requirement collection list" will list the requirements that in the search result, if no text is entered in the search box, all requirements will list by default.

5. "Add" button, will add requirement to the "Linked RM requirement artifact" list, before pressing "Add" button, one test artifact must be selected, and one or more requirement must be selected, and the selected requirements will be associcated with the test artifact selected.

6. "linked RM requirement artifact list" will list the associated requirement artifacts with the test artifact selected.

7. "Remove" button, will remove the requirement artifact from the "linked RM requirement artifact list".



Requirement Links

Link to Existing RM requirement artifact

RQM test artifact

- Hot Hammer
- Strong Hammer
- Pretty Hammer
- Dead Hammer
- Socially Acceptable Hammer
- Big Plan

Linked RM requirement artifact

RM requirement artifact

Containing ID or text:

□ Show Requirement ☑ Show requirement collection

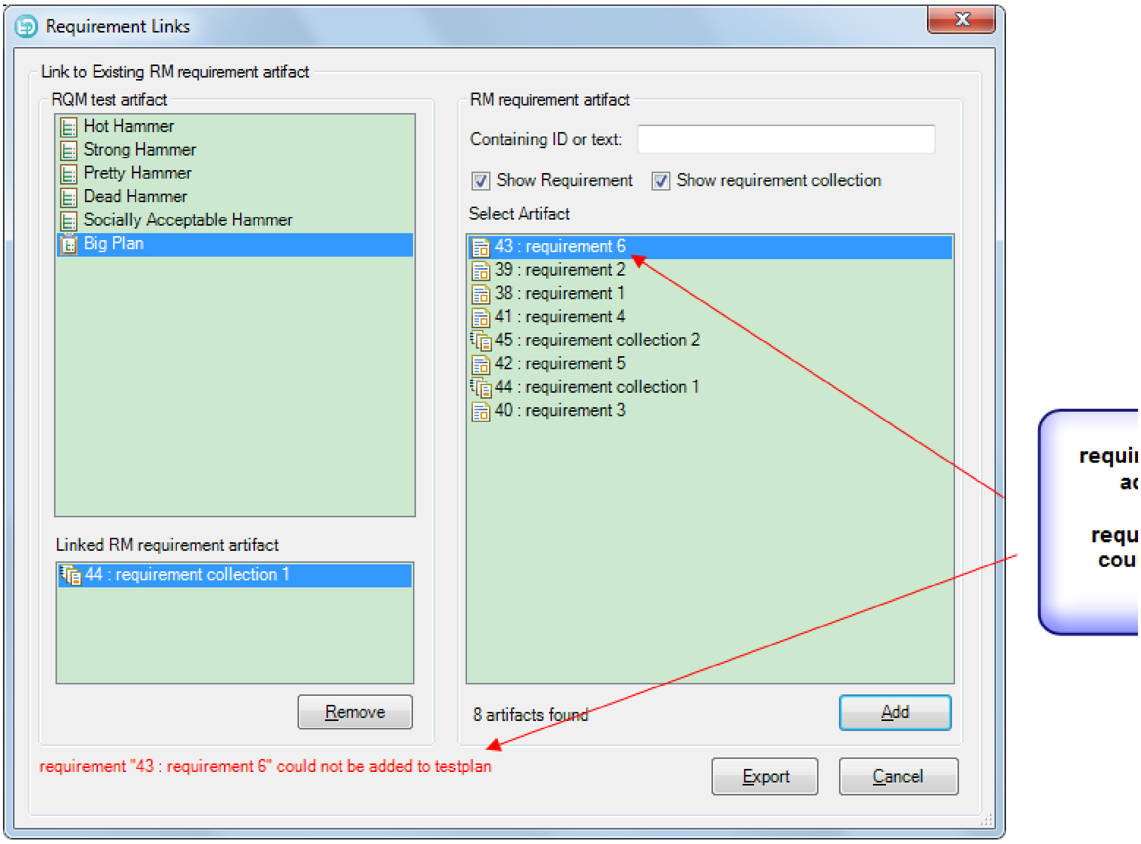
Select Artifact

- 45 : requirement collection 2
- 44 : requirement collection 1

2 artifacts found

there are some restrictions, e.g. requirement should not be added to testplan, and requirement collections should not be added to testcase, and one requirement should not be added to test artifact more than once, and and press "Add" button, one test artifact must be selected and one or more requirement artifacts must be selected.

If the restriction is not met, then error message will display.



Requirement Links

Link to Existing RM requirement artifact

RQM test artifact
- Hot Hammer
- Strong Hammer
- Pretty Hammer
- Dead Hammer
- Socially Acceptable Hammer
- Big Plan

Linked RM requirement artifact
- 44 : requirement collection 1

RM requirement artifact

Containing ID or text:

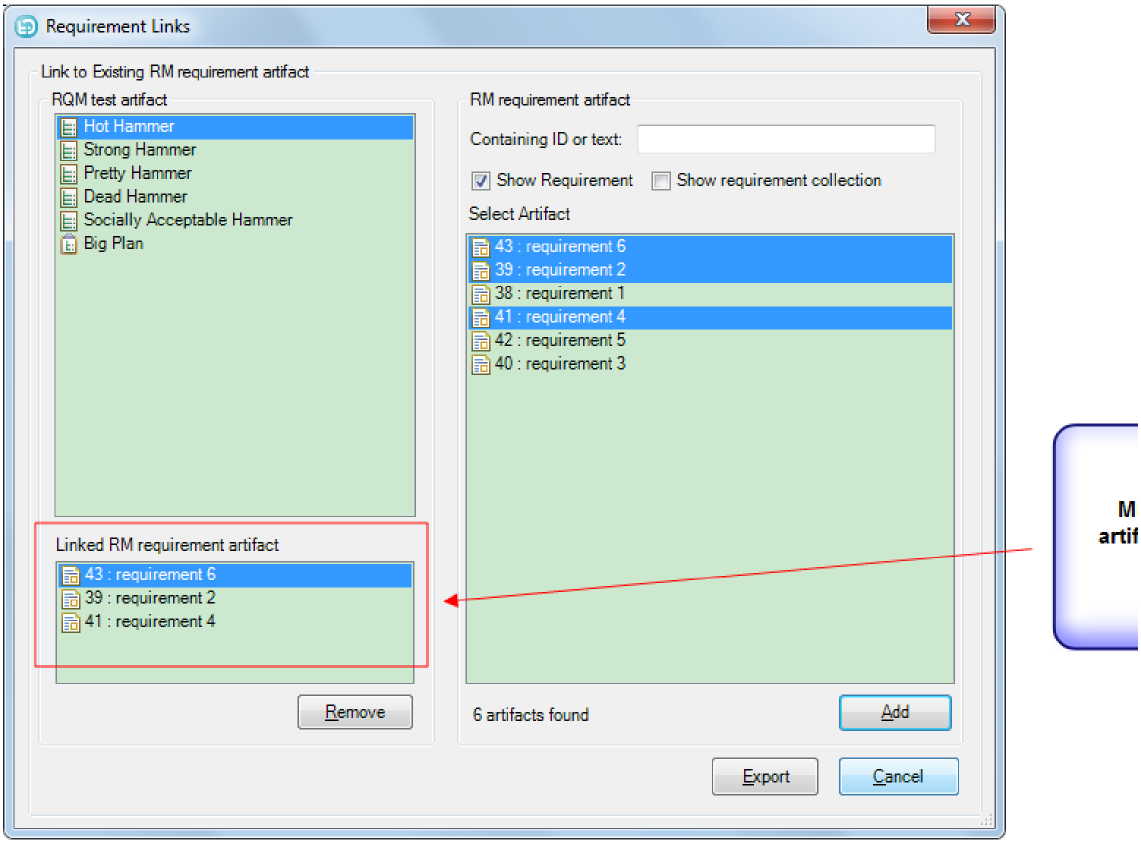
☑ Show Requirement ☑ Show requirement collection

Select Artifact
- 43 : requirement 6
- 39 : requirement 2
- 38 : requirement 1
- 41 : requirement 4
- 45 : requirement collection 2
- 42 : requirement 5
- 44 : requirement collection 1
- 40 : requirement 3

8 artifacts found

requirement "43 : requirement 6" could not be added to testplan

one or more requirement artifacts can be associated with one test artifact.



Requirement Links

Link to Existing RM requirement artifact

RQM test artifact
- Hot Hammer
- Strong Hammer
- Pretty Hammer
- Dead Hammer
- Socially Acceptable Hammer
- Big Plan

Linked RM requirement artifact
- 43 : requirement 6
- 39 : requirement 2
- 41 : requirement 4

RM requirement artifact

Containing ID or text:

☑ Show Requirement □ Show requirement collection

Select Artifact
- 43 : requirement 6
- 39 : requirement 2
- 38 : requirement 1
- 41 : requirement 4
- 42 : requirement 5
- 40 : requirement 3

6 artifacts found

## Troubleshooting

When an import fails, the 'Export to RQM' dialog displays an error and the location of the event log file. In the event of an import error, see the debug log file ( `C:\Documents and Settings\<USER>\Application Data\Mso2Rqm\Mso2Rqm_Debug.log`) for more information. In the event of a server error, see Troubleshooting.

In addition, export to a file and attempt to import the XML file(s) into RQM using the RQM Reportable REST API for a detailed HTTP error response.

# Creating Configuration Files

## Configuration file structure

The configuration file is constructed using a few basic premises:

1. Each line is by itself. What this means is that previous lines do not modify the subsequent lines, and has significance only to itself.
2. The left side of an equals sign, =, is comprised of a combination of keywords and RQM artifact names and fields separated with periods.
3. The right side of the equals sign is comprised of the location to find the data that goes into the field on the left of the equals sign.
4. Blank lines and lines that start with // are ignored. Comments can be inserted after // on an otherwise blank line.
5. External IDs may be specified for a resource in the configuration file using the `XLSArtifactID` keyword (Excel **only**).
6. Date/time values are represented as XML dateTime values.
7. The RQM Excel/Word Importer is built on the RQM Reportable REST API.
8. When using DBCS characters in the config file make sure to set the file encoding to "UTF-8".

## Supported Resources

| Resource | Excel | Word |
|---|---|---|
| configuration | X | X |
| executionresult | X | X |
| executionworkitem | X | X |
| labresource | X | X |
| remotescript | X | X |
| requirement[1, 2] | X | X |
| requirement collection[1, 2, 3] | X | |
| resourcegroup | X | X |
| testcase | X | X |
| testphase | X | X |
| testplan | X | X |
| testscript | X | X |
| testsuite | X | X |
| iteration | X | X |
| categoryType | X | X |
| category | X | X |
| keyword | X | X |

[1] For more information on Migrating work-item-based requirements, see the Collaborative Lifecycle Management 3.0.1 Information Center Migrating work-item-based requirements topic.

[2] When Rational Quality Manager requirements reference resources in Rational Requirements Composer, the following configuration is required:

1. The Quality Management project area in Rational Quality Manager must have a "uses and requirements" association to a Requirements Management project area in Rational Requirements Composer. Note, multiple Quality Management project areas can be associated with a single Requirements Management project area.
2. Set the following Requirement ( `com.ibm.rqm.requirement.service.internal.RequirementService`) advanced server configuration properties (see *Configuring advanced properties* in the Collaborative Lifecycle Management 3.0.1 Information Center):

- `Requirements Composer User Name`
- `Requirements Composer User Password`

[3] Requirement collection is supported from Excel Importer 4.0.3

See the RQM Schema Documentation for the resource properties and the Create Tip to resolve an example XML representation of the resource.

## Supported Attributes and Configuration Syntax

### Test Case

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | testcase.dc:title | **Testplan_Testcase_Testscript_TER_Environment_Iteration_AllInOne.xls** |
| Description | testcase.dc:description | |
| Estimate | testcase.estimate | |
| Locked | testcase.locked | |
| Priority | testcase.priority | |
| Owner | testcase.alm:owner | |

| | | |
|---|---|---|
| Suspect | testcase.suspect | |
| Weight | testcase.weight | |
| Creation Date | testcase.creationDate | |

**Test Script**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | testscript.dc:title | **Testplan_Testcase_Testscript_TER_Environment_Iteratio** |
| Description | testscript.dc:description | |
| Step Type | testscript.steps.type="Execution"/"Reporting" | |
| Step Title | testscript.steps.title | |
| Step Description | testscript.steps.description | |
| Step Expected Result | testscript.steps.expectedResult | |
| Step Comment | testscript.steps.comment | |

**Test Case Execution Record**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | executionworkitem.dc:title | **Testplan_Testcase_Testscript_TER_Environment_Iteration_AllInOne.** |
| Description | executionworkitem.dc:description | |
| Estimate | executionworkitem.estimate | |
| Time Spent | executionworkitem.timeSpent (in ms) | |
| Owner | executionworkitem.alm:owner | |

**Test Case Result**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | executionresult.dc:title | **Testplan_Testcase_Testscript_TER_Environment_Itera** |
| Description | executionresult.dc:description | |
| Actual Result | executionresult.jzalm:state | supported value syntax: value is case insensitive<br>"pass", "passed", "true", "yes" -> com.ibm.rqm.execution.cor<br>"failed", "fail", "false", "no" -> com.ibm.rqm.execution.commor<br>"blocked" -> com.ibm.rqm.execution.common.state.blocked<br>"Partially Blocked", "Partially_Blocked","PartiallyBlocked", "pa<br>"partblocked" -> com.ibm.rqm.execution.common.state.part_l<br>"error" -> "com.ibm.rqm.execution.common.state.error"<br>"deffered" -> "com.ibm.rqm.execution.common.state.deferrec<br>"paused" -> com.ibm.rqm.execution.common.state.paused"<br>"Permanently Failed", "PermFailed", "perm_failed" -> com.ibm.rqm.execution.common.state.perm_failed<br>"incomplete" -> com.ibm.rqm.execution.common.state.incomp<br>"inprogress", "in progress", "in_progress" -> com.ibm.rqm.execution.common.state.inprogress<br>other values -> com.ibm.rqm.execution.common.state.inconc |
| Start Time | executionresult.starttime | |
| End Time | executionresult.endtime | |
| Step Result Type | executionresult.er:stepResults.er:steptype | |
| Step Result Description | executionresult.er:stepResults.er.description | |
| Step Result Expected Result | executionresult.er:stepResults.er:expectedResult | |
| Step Result Actual Result | executionresult.er:stepResults.er:actualResult | |
| Step | | |

| | |
|---|---|
| Result Comment | executionresult.er:stepResults.er:comment |

**Requirement**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | requirement.dc:title | **Requirements 001.xls** |
| Description | requirement.dc:description | |
| Primary Text | requirement.primaryText | |
| Requirement Type | requirement.resourceShape | |

**Requirement Collection**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | requirementCollection.dc:title | **Test Plan Requirement Collection Test Case and Requirements.xls** |
| Description | requirementCollection.dc:description | |
| Primary Text | requirementCollection.primaryText | |
| Requirement Type | requirementCollection.resourceShape | |

**Test Phase**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | testphase.dc:title | **Testplan_Testcase_Testscript_TER_Environment_Iteration_AllInOne.xl** |
| Description | testphase.dc:description | |
| Expected Total Points | testphase.expectedTotalPoints | |
| Expected Defects | testphase.expectedDefects | |
| Expected Validity Rate | testphase.expectedValidityRate | |
| Expected Start Date | testphase.expectedStartDate | |
| Expected End Date | testphase.expectedEndDate | |
| Start Time | testphase.starttime | |
| End Time | testphase.endtime | |

**Test Plan**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | testplan.dc:title | **Testplan_Testcase_Testscript_TER_Environment_Iteration_AllInOne.xls** |
| Description | testplan.dc:description | |
| Locked | testplan.locked | |

**Test Suite**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | testsuite.dc:title | **TestSuite with TestCases .xls** |
| Description | testsuite.dc:description | |
| Weight | testsuite.weight | |
| Estimate | testsuite.estimate | |
| Locked | testsuite.locked | |
| Sequential Execution/Parallel Execution | testsuite.sequentialExecution="true"/"false | |

**Key Word**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | keyword.dc:title | keyword sample.xls |
| Tags | keyword.tags | |

**Category**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | category.dc:title | **Categories and Types.xls** |

**Category Type**

| attribute name | syntax of the configuration | related samples |
|---|---|---|

| | | |
|---|---|---|
| Title | categoryType.dc:title | **Categories and Types** |
| Required | categoryType.required | |
| MultiSelectable | categoryType.multiSelectable | |
| Scope | categoryType.scope | |

**Lab Resource**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | labresource.title | **Lab Resources.xls** |
| Description | labresource.dc:description | |
| Owner | labresource.alm:owner | |
| Type | labresource.type="Machine"/"PhysicalMachine"/"VirtualMachine" | |
| Operation System | labresource.osType | |
| Memory | labresource.memory | |
| Free Disk Space | labresource.diskspace | |
| IP Address | labresource.ipAddress | |
| Total Disk Space | labresource.diskspace | |
| Fully Qualified Domain Name | labresource.fullyQualifiedDomainName | |
| Host Name | labresource.hostname | |
| Primary MACAddress: | labresource.primaryMacAddress | |

**Remote Script**

| attribute name | syntax of the configuration | related samples |
|---|---|---|
| Title | remotescript.dc:title | **Testcase with Remote Scripts 001.xls** |
| Description | remotescript.dc:description | |
| Remote Script Type | remotescript.type | |
| Owner | remotescript.alm:owner | |
| Command | remotescript.rs:command | |
| Argument | remotescript.rs:argument | |
| Properties | remotescript.rs:properties | |
| Adapter id | remotescript.adapterid | |
| Script Location | remotescript.manageadapter="true"/"false"<br>true->Use test resources that are local to a test machine<br>false->Use test resources from a shared location | |
| Shared Resource Location | remotescript.shareprefix | |
| Relative Script Path | remotescript.relativepath | |
| Script Path | remotescript.fullpath | |
| Script arguments | remotescript.rs:ScriptArguments | |
| Execution arguments | remotescript.rs:ExecArguments | |

# Excel Migration Configuration

## Data locations and self referential links

The right side of the equals sign of a configuration can consist of three possibilities.

1. A column or cell definition i.e. `D` or `D4`
2. A literal string i.e. "`execution`"
3. A reference to another artifact i.e. `testscript.description`

To define a row or cell, simply put the column identifier or cell identifier:

```
testscript.dc:title=G
```

To define a literal string, simply wrap it with quotes:

```
testscript.steps.type="execution"
```

To define a self reference, simply put in the information from the left side of the equals from a previously defined line:

```
testscript.dc:title=G
```

```
testcase.dc:title=testscript.dc:title
```

**Note:** Self references are used to create artifacts that will use data already being used by another artifact and will create a link to that artifact.

**Note:** Self references cannot be used with the `XLSLink` keyword.

## Keywords and special cases

There are several keywords that go on the left side of the equals sign. They all start with "XLS" and take the place of a field after the artifact. They are defined as:

`XLSStartRow` defines what row the artifact data starts after.

`XLSDelimeter` defines what denotes the next artifact (currently only \n is supported).

`XLSLink` specifies that this artifact should be linked to all artifacts of the specified type within the worksheet.

`DOCLink` specifies that this artifact should be linked to all artifacts of the specified type within the word documnet.

`XLSLinkReverse` specifies that the tool put the link XML into the artifact, rather than the one making the link. Only used with self referential links. Used when creating an artifact based on another (e.g. `requirement.dc:title=testcase.dc:title` ) but need the link to be created in the artifact that is the original (the one to the right of the equals). Specifying `requirement.XLSLinkReverse=true` will put the link in the main artifact, rather than the one specifying the self-referential link.

`XLSArtifactID` specifies the external ID of the artifact. If external ID of the artifact is not specified, a default external ID is assigned using the following format:

```
<Excel file name>_<Excel worksheet name>[_<artifact count when 2 or more artifacts in the same Excel
worksheet>].xml
```

Note, if an artifact exists with the same external ID, it is updated. For example, multiple export operations of the same Excel/configuration files to the same repository.

**Warning**: if using artifact.XLSArtifactID="urn:com.ibm.rqm:testplan:2" when the value is specified a string not a cell, make sure there's no blank sheet or use artifact.XLSWorksheetID to specify the worksheet to read the ArtifactID.

To create new artifact(s) with multiple export operations of the same Excel/configuration files to the same repository:

- Use the XLSArtifactID keyword to specify a unique external ID for each export operation.

or

- Change the Excel file name or Excel worksheet name.

or

- Export to File, change the name of the XML file(s), and import the XML file(s) into RQM using the **Planning > Import Test Plans**, **Construction > Import Test Cases**, or **Construction > Import Test Scripts** menus. Note, test plan, test case, and test script **only**.

`XLSWorksheetID` specifies the name of the worksheet to process. Only a literal string is accepted.
From Excel Importer 4.0.1, XLSWorksheetID support specifying multiple work sheets, the syntax is testcase.XLSWorksheetID=""Acme Hammers"" : "Acme Rockets", use ":" to separate the worksheet names.

From RQM4.0 the syntax for testsuite linked with testcases has changed from*testsuite.testcases=testcase* to ***testsuite.XLSLink=testcase***

Example:

```
testscript.XLSStartRow=7

testscript.title=B

testscript.steps.type="execution"

testscript.steps.description=B

testscript.steps.title=B

testscript.steps.name=B

testscript.steps.expectedResult=F

testscript.XLSDelimeter=\n

testcase.dc:title=C1

testcase.dc:description=C2

testcase.XLSLink=testscript
```

`testcase.XLSArtifactID=D1` or `testcase.XLSArtifactID=D1`="test_case_1"

The script steps, categories, sections, exporting images, dynamically searching for a word or phrase and accessing the spreadsheet properties are special cases. A script step requires that it has several fields for each step. Since a script can have many steps, each line until a blank line is encountered will be considered the next step.

The syntax for exporting RM requirement primary text is

```
requirement.primaryText=Div(C)
```

note: the cell/column/ must be wrapped by Div modifier, and only **raw** text will be export to RRC requirement, and primaryText is case sensitive.

## SectionsIds

Sections are special cased to simplify their use and to keep their definitions from interfering with the period notation used otherwise. To update a section, simply use the following format:

```
testcase.Section("myns:com.ibm.rqm.planning.editor.section.testCaseAcceptanceCriteria", "RQM-KEY-TC-ACCEPT-CRITERIA-TITLE")=C
```

The ID and extension display name for test plan, test suite, and test case sections are list below:

Test Case:
**Test Case Design:**
id: com.ibm.rqm.planning.editor.section.testCaseDesign
extensionDisplayName: RQM-KEY-TC-DESIGN-TITLE
**Pre-Condition**
id: com.ibm.rqm.planning.editor.section.testCasePreCondition
extensionDisplayName: RQM-KEY-TC-PRE-COND-TITLE
**Post-Condition**
id: com.ibm.rqm.planning.editor.section.testCasePostCondition
extensionDisplayName: RQM-KEY-TC-POST-COND-TITLE
**Expected Results**
id: com.ibm.rqm.planning.editor.section.testCaseExpectedResults
extensionDisplayName: RQM-KEY-TC-EXP-RESULTS-TITLE
**Notes**
id: com.ibm.rqm.planning.editor.section.testCaseNotes
extensionDisplayName: RQM-KEY-TC-NOTES-TITLE
**Test Preparation**
id: com.ibm.rqm.planning.editor.section.caseTestPreparation
extensionDisplayName: RQM-KEY-TC-TEST-PREP-TITLE
**Test Description**
id: com.ibm.rqm.planning.editor.section.caseTestDescription
extensionDisplayName: RQM-KEY-TC-TEST-DESCRIPTION-TITLE
**Test Case Scope**
id: com.ibm.rqm.planning.editor.section.caseTestScope
extensionDisplayName: RQM-KEY-TC-TEST-SCOPE-TITLE

Test Suite:
**Test Suite Design**
id: com.ibm.rqm.execution.editor.section.design
extensionDisplayName: RQM-KEY-EXE-DESIGN-TITLE
**Pre-Condition**
id: com.ibm.rqm.planning.editor.section.testSuitePreCondition
extensionDisplayName: RQM-KEY-TSUITE-PRE-COND-TITLE
**Post-Condition**
id: com.ibm.rqm.planning.editor.section.testSuitePostCondition
extensionDisplayName: RQM-KEY-TSUITE-POST-COND-TITLE
**Expected Results**
id: com.ibm.rqm.planning.editor.section.testSuiteExpectedResults
extensionDisplayName: RQM-KEY-TSUITE-EXP-RESULTS-TITLE

Test Plan:
**Business Objectives**
id: com.ibm.rqm.planning.editor.section.planBusinessObjectives
extensionDisplayName: RQM-KEY-TP-BUSS-OBJ-TITLE
**Test Objectives**
id: com.ibm.rqm.planning.editor.section.planTestObjectives
extensionDisplayName: RQM-KEY-TP-TEST-OBJ-TITLE
**Instruments and Test Equipment**
id: com.ibm.rqm.planning.editor.section.planTestEquipment
extensionDisplayName: QM-KEY-TP-TEST-EQUIPMENT-TITLE
**Test Bed**
id: com.ibm.rqm.planning.editor.section.planTestBed
extensionDisplayName: RQM-KEY-TP-TEST-BED-TITLE
**Test Data**
id: com.ibm.rqm.planning.editor.section.planTestData
extensionDisplayName: RQM-KEY-TP-TEST-DATA-TITLE
**Test Conditions**

id: com.ibm.rqm.planning.editor.section.planTestCondition
extensionDisplayName: RQM-KEY-TP-TEST-CONDITION-TITLE
**Test Identification**
id: com.ibm.rqm.planning.editor.section.planTestIdentification
extensionDisplayName: RQM-KEY-TP-TEST-IDENTIFICATION-TITLE
**Test Strategy**
id: com.ibm.rqm.planning.editor.section.planTestStrategy
extensionDisplayName: RQM-KEY-TP-TEST-STRATEGY-TITLE
**Test Plan Scope**
id: com.ibm.rqm.planning.editor.section.planTestScope
extensionDisplayName: RQM-KEY-TP-TEST-SCOPE-TITLE
**Software Test Environment Details**
id: com.ibm.rqm.planning.editor.section.planTestEnvDetails
extensionDisplayName: RQM-KEY-TP-TEST-ENV-DETAIL-TITLE
**Sampling Strategy**
id: com.ibm.rqm.planning.editor.section.planSamplingStrategy
extensionDisplayName: RQM-KEY-TP-SAMPLING-STRATEGY-TITLE

This example will populate the acceptance criteria section that is built into the RQM product.

To determine the name/display name of predefined sections:

1. Update the predefined section of a resource in the Rational Quality Manager UI.
2. Read the resource as XML with `abbreviate=false`.
3. Locate the element in the XML document with the section name (e.g. `com.ibm.rqm.planning.editor.section.testCaseAcceptanceCriteria`).
4. Locate the `extensionDisplayName` attribute in the element containing the section display name (e.g. `RQM-KEY-TC-ACCEPT-CRITERIA-TITLE`).

To define a custom section, simply replace the first parameter with a valid name, and the second with a display name:

```
testcase.Section("myns:mysection","my custom section")=C
```

Categories are defined with the category term=<category name>.value=<category value> notation:

```
testcase.category term="Function".value=H
```

from Excel Importer 4.0.1, support category with multiple values, user can specify mulitple values in one cell and the values can be speareated by new lines.

```
Adding custom attributes to an artifact
if we have 2 customAttributes  testcaseCustomAttribute1 , and testcaseCustomAttribute1 in one
testcase, the customAttributes testcaseCustomAttribute1 and testcaseCustomAttribute2 should be
created in RQM UI before running the RQM Excel Importer.

we can config the .cfg file like

//for testcaseCustomAttribute1 put identifier in Excel Cell M3 and name in Cell N3 and type in
Cell M4.
testcase.customAttributes identifier=M3.name=N3.type=M4.value=O

or using testcase.customAttributes
identifier="Custom_attribute_0_".name="testcaseCustomAttribute1".type="SMALL_STRING".value=O

//for testcaseCustomAttribute2 put identifier in Excel Cell Q3 and name in Cell R3, if type
isn't specified, the "SMALL_STRING" is by default.
testcase.customAttributes identifier=Q3.name=R3.value=Q


the result will be:

<customAttributes>
    <customAttribute type="SMALL_STRING">
        <identifier>Custom_attribute_0_</identifier>
        <name>testcaseCustomAttribute1</name>
        <value>1111111111111111111111111111</value>
    </customAttribute>
    <customAttribute type="SMALL_STRING">
        <identifier>Custom_attribute_1_</identifier>
        <name>testcaseCustomAttribute2</name>
        <value>3333333333333333333333333</value>
    </customAttribute>
</customAttributes>
```

new sample TestCases 001 with custom attributes.xls and TestCases 001 with custom attributes.cfg will be added in 4.0

we also add custom attributes supporing in Test Case Execution Result for  "Run Offline Execution".

********************************************************************************

There are several ways to dynamically set a row, column, or cell.

```
requirement.XLSStartRow=FindRow(“Start on this row”)
```

```
requirement.dc:title=FindColumn(“Title Header”)
```

```
requirement.jzalm:owner=FindCell(“Owner”)
```

Constants can also be added to the function calls.

So FindRow(“Data that is in the second row”) + 1would return 3

Also FindCell (“Data that is in A1”) + (1, 1)would return B2.

Lastly if you just add one number to a FindCell it adds it to the row. For Examaple FindCell(“Data of A1”) + 1would return A2.

The related sample xls/cfg files are titled“Requirements 001 with dynamic directive setting.”

Along the same lines, you can access the properties of the spreadsheet that you are working on. Examples:

```
requirement.dc:title=property(“Title”)
```

```
requirement.dc:creator=property("Author")
```

```
requirement.jzalm:owner=property(“Company”)
```

The properties accessible are built into Microsoft Office are found at http://msdn.microsoft.com/en-us/library/microsoft.office.core.documentproperty.aspx

There are also three other properties available, “Name,” “Full Name,”and “Location.” If the property has not been set or is not supported by Excel, property (“property name”) will return the empty string:“”

The related sample xls/cfg files are titled “Requirements001 using document properties.”

The location of images in excel are represented by the cell that the top left hand corner of the image is in, so even if an image takes up multiple cells, only the cell that the top left corner of the image is in knows that the image is there and will be able to export it. Also there is no limit on how many images can be put in a cell. It will export all of them as long as the top left hand corner of the image is in the cell in question.

There are two ways to export images in Excel. The first is by using the image function. Image is setup as a function to put after the equals sign in the configuration file. images can be specified by the cell or by the row. This will insert an image into the rich text field that it was assigned to. If the attribute does not use a rich text field the image function will be ignored.

```
testscript.steps.expectedResult=image(G)
```

```
testscript.steps.description=image(C4)
```

The second is by using the attachment attribute.

```
testcase.attachment=C4
```

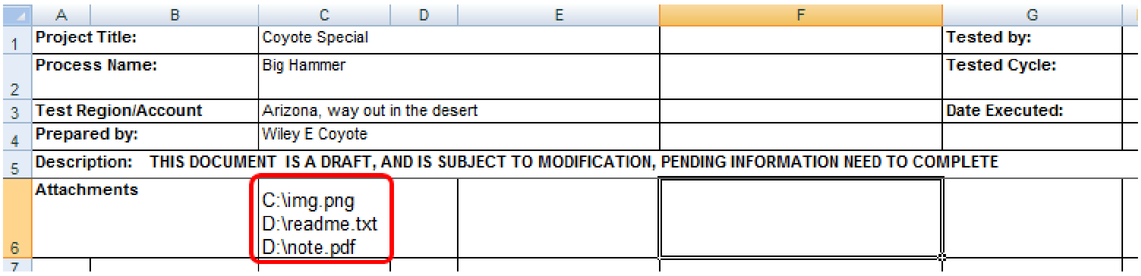
```
testcase.attachment=C
```

This will add the images as attachments to the artifact rather than inserting the image itself into rich text fields.

The related sample xls/cfg files are respectively titled “Test Cases made from scripts with images“ and "Test Cases made from scripts with attachments.”

From Excel Importer 4.0.3, in addition to export attachments in image format, you can also export attachments specified in absolute file path. The attachment file paths should be specified in one cell and if there are multiple paths, they should be separated by a new line "\n". Attachment types such as txt, pdf, etc. are supported.

Here is an example:



| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | Project Title: | | Coyote Special | | | | Tested by: |
| 2 | Process Name: | | Big Hammer | | | | Tested Cycle: |
| 3 | Test Region/Account | | Arizona, way out in the desert | | | | Date Executed: |
| 4 | Prepared by: | | Wiley E Coyote | | | | |
| 5 | Description: THIS DOCUMENT IS A DRAFT, AND IS SUBJECT TO MODIFICATION, PENDING INFORMATION NEED TO COMPLETE | | | | | | |
| 6 | Attachments | | C:\img.png<br>D:\readme.txt<br>D:\note.pdf | | | | |
| 7 | | | | | | | |

The cfg file is defined like:

```
testcase.dc:title=C1
```

```
testcase.dc:description=C2
```

```
testcase.attachment=C6
```

The 3 attachments specified in cell C6 will be exported and referenced in the testcase.

**Execution Variable Support (4.0.1 and above)**

Execution variables are now supported in the configuration files of the import tool. The following format can be used to include execution variables:

Excel spreadsheet with testcase execution variables (name is found in cell A1 and value is found in cell B1)
`testcase.variables.name=A1.value=B1`

Excel spreadsheet with testcase execution variables (name is found in column A and value is found in column B)
`testcase.variables.name=A.value=B`

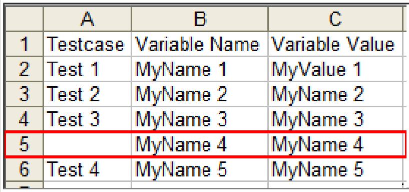
Excel spreadsheet with testcase execution variables (name is found in column A and value is always 'MyValue')
`testcase.variables.name=A.value="MyValue"`

Excel spreadsheet with testcase execution variables (name is always 'MyName' and value is found in column B)
`testcase.variables.name="MyName".value=B`

Notes:

- Providing an execution variable setup with both a literal name and a literal value is not supported (E.G. `testcase.variables.name="MyName".value="MyValue"`)

- Execution variables provided columns are a one-to-one mapping, meaning that variables in the next row are presumed to belong to the next artifact)
  To include multiple execution variable per artifact, additional lines must be added to the configuration file pointing to additional rows/columns in the spreadsheet.
  -

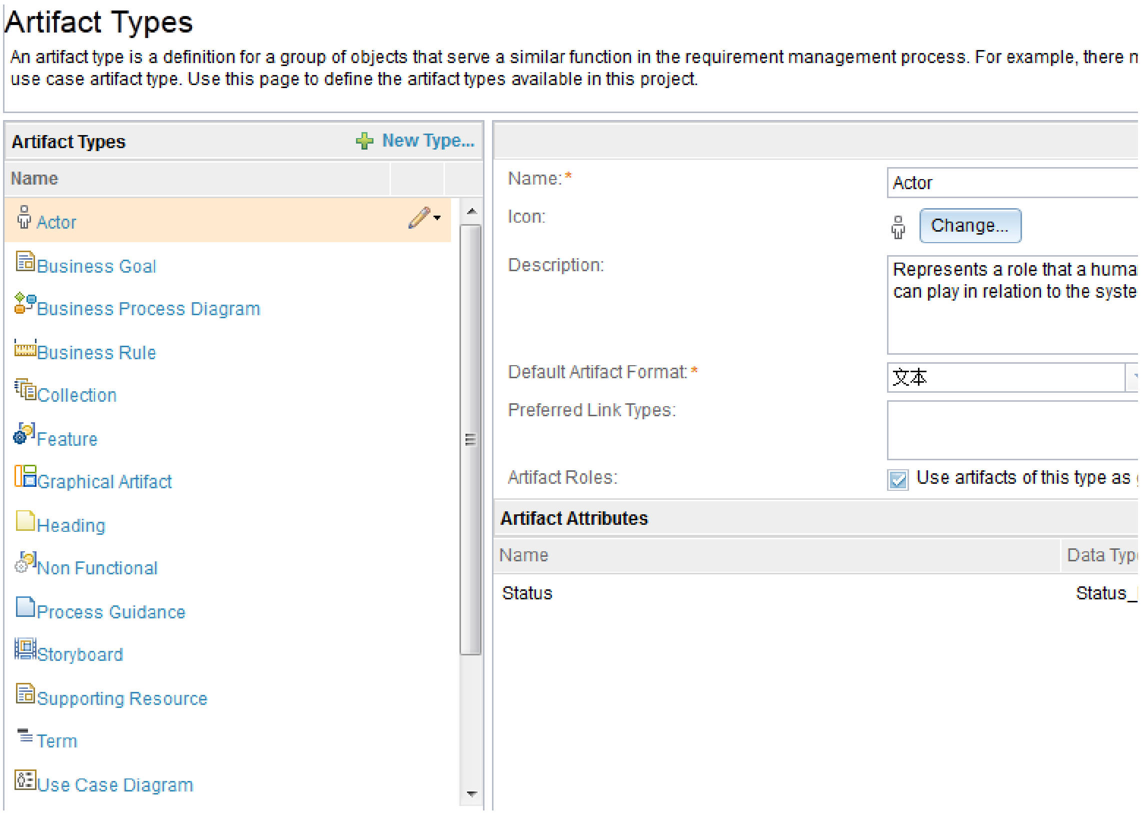
<table>
<tr><td>Excel Spreadsheet</td><td>Configuration File</td></tr>
<tr><td>
<table>
<tr><th></th><th>A</th><th>B</th><th>C</th></tr>
<tr><td>1</td><td>Testcase</td><td>Variable Name</td><td>Variable Value</td></tr>
<tr><td>2</td><td>Test 1</td><td>MyName 1</td><td>MyValue 1</td></tr>
<tr><td>3</td><td>Test 2</td><td>MyName 2</td><td>MyName 2</td></tr>
<tr><td>4</td><td>Test 3</td><td>MyName 3</td><td>MyName 3</td></tr>
<tr><td>5</td><td></td><td>MyName 4</td><td>MyName 4</td></tr>
<tr><td>6</td><td>Test 4</td><td>MyName 5</td><td>MyName 5</td></tr>
</table></td><td><code>testcase.dc:title=A</code><br><code>testcase.variables.name=B.value=C</code><br><code>testcase.XLSStartRow=2</code></td></tr>
<tr><td colspan="2">Generates five tests cases, one for each execution variable. The test case from row 5 is generated solely due to the existence of the execution variable.</td></tr>
</table>

- If the name field of an execution variable is blank, no execution variable is generated for the artifact.

- If the value field of an execution variable is blank, the execution variable is presumed to have an empty value and is still generated.

**#Exporting requirement with customized types (4.0.2 and above)**

- define requirement type on RM project area.



## Artifact Types

An artifact type is a definition for a group of objects that serve a similar function in the requirement management process. For example, there n
use case artifact type. Use this page to define the artifact types available in this project.

**Artifact Types** ✚ New Type...

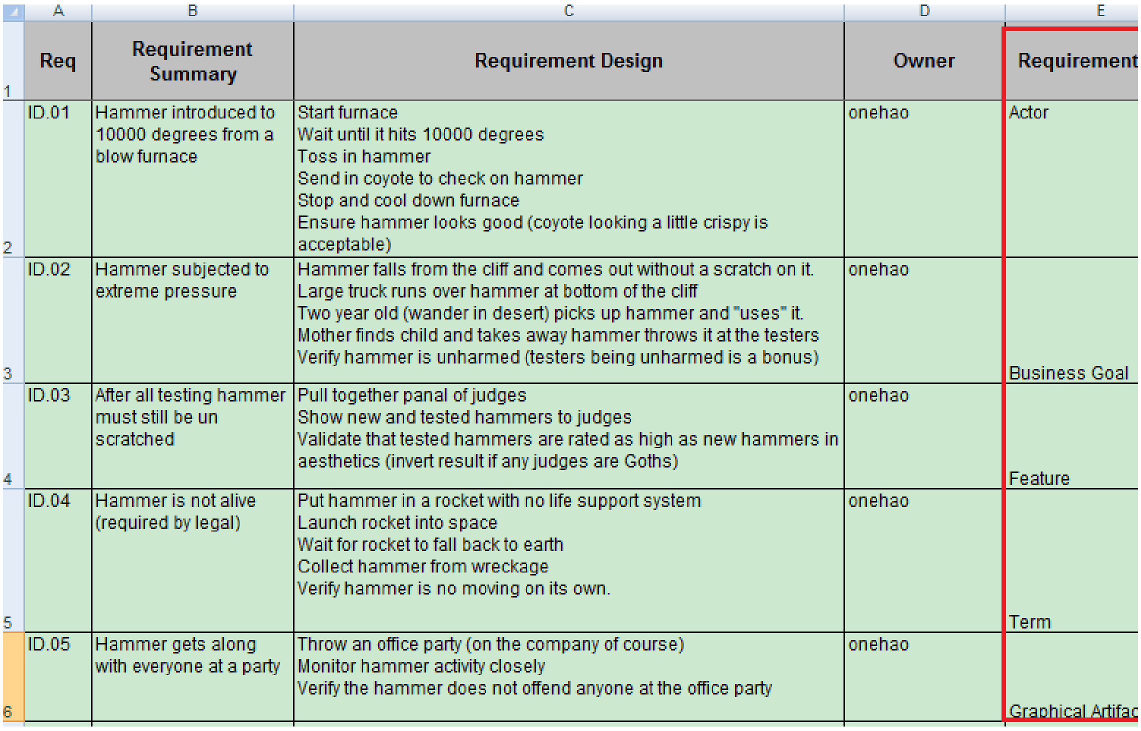
| Name |
|---|
| Actor |
| Business Goal |
| Business Process Diagram |
| Business Rule |
| Collection |
| Feature |
| Graphical Artifact |
| Heading |
| Non Functional |
| Process Guidance |
| Storyboard |
| Supporting Resource |
| Term |
| Use Case Diagram |

| | |
|---|---|
| Name:* | Actor |
| Icon: | Change... |
| Description: | Represents a role that a huma<br>can play in relation to the syste |
| Default Artifact Format:* | 文本 |
| Preferred Link Types: | |
| Artifact Roles: | ☑ Use artifacts of this type as |

**Artifact Attributes**

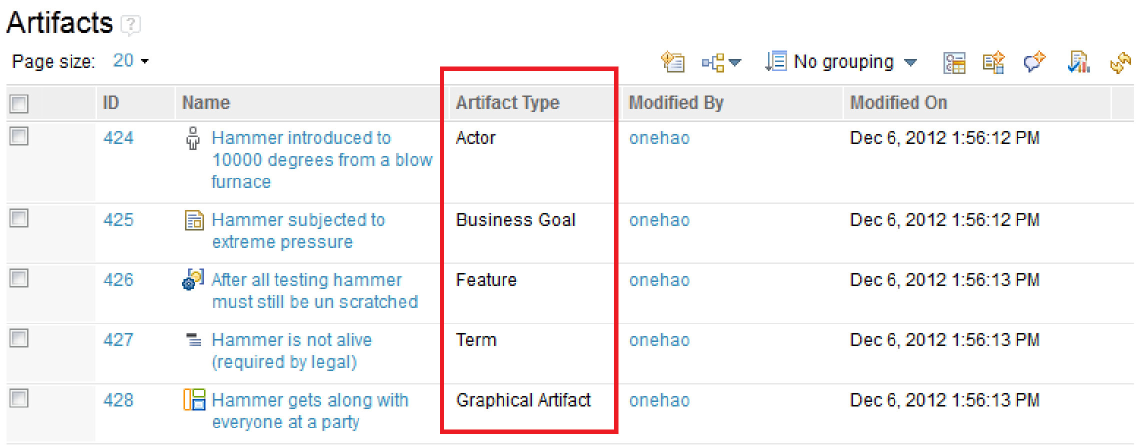
| Name | Data Typ |
|---|---|
| Status | Status_ |

- specify the requriement type in Excel, and update the config file with requirement.**resourceShape**=E



| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | **Req** | **Requirement Summary** | **Requirement Design** | **Owner** | **Requirement** |
| 2 | ID.01 | Hammer introduced to 10000 degrees from a blow furnace | Start furnace<br>Wait until it hits 10000 degrees<br>Toss in hammer<br>Send in coyote to check on hammer<br>Stop and cool down furnace<br>Ensure hammer looks good (coyote looking a little crispy is acceptable) | onehao | Actor |
| 3 | ID.02 | Hammer subjected to extreme pressure | Hammer falls from the cliff and comes out without a scratch on it.<br>Large truck runs over hammer at bottom of the cliff<br>Two year old (wander in desert) picks up hammer and "uses" it.<br>Mother finds child and takes away hammer throws it at the testers<br>Verify hammer is unharmed (testers being unharmed is a bonus) | onehao | Business Goal |
| 4 | ID.03 | After all testing hammer must still be un scratched | Pull together panal of judges<br>Show new and tested hammers to judges<br>Validate that tested hammers are rated as high as new hammers in aesthetics (invert result if any judges are Goths) | onehao | Feature |
| 5 | ID.04 | Hammer is not alive (required by legal) | Put hammer in a rocket with no life support system<br>Launch rocket into space<br>Wait for rocket to fall back to earth<br>Collect hammer from wreckage<br>Verify hammer is no moving on its own. | onehao | Term |
| 6 | ID.05 | Hammer gets along with everyone at a party | Throw an office party (on the company of course)<br>Monitor hammer activity closely<br>Verify the hammer does not offend anyone at the office party | onehao | Graphical Artifac |

- the requirement with the specified types will be created.



Artifacts

Page size: 20

| □ | ID | Name | Artifact Type | Modified By | Modified On |
|---|---|---|---|---|---|
| □ | 424 | Hammer introduced to 10000 degrees from a blow furnace | Actor | onehao | Dec 6, 2012 1:56:12 PM |
| □ | 425 | Hammer subjected to extreme pressure | Business Goal | onehao | Dec 6, 2012 1:56:12 PM |
| □ | 426 | After all testing hammer must still be un scratched | Feature | onehao | Dec 6, 2012 1:56:13 PM |
| □ | 427 | Hammer is not alive (required by legal) | Term | onehao | Dec 6, 2012 1:56:13 PM |
| □ | 428 | Hammer gets along with everyone at a party | Graphical Artifact | onehao | Dec 6, 2012 1:56:13 PM |

**#Control whether to set ".xml" extension in the test artifact external id (Excel 4.x:4.0.2 and above, Excel Importer 3.x : 3.0.1.6 and above)**

testcase.XMLExtension=true/yes or no/false
if set true/yes Excel Importer will add the .xml extension to the external id and if not then will not add it.

**#new samples (4.0.2 and above)**

Scripts that create testcases with TCER.xls, Testplan_Testcase_Testscript_TER_Environment_Iteration_AllInOne.xls, Testcase with Remote Scripts 001.xls

## Modifiers of data (Used to the right of the equals sign)

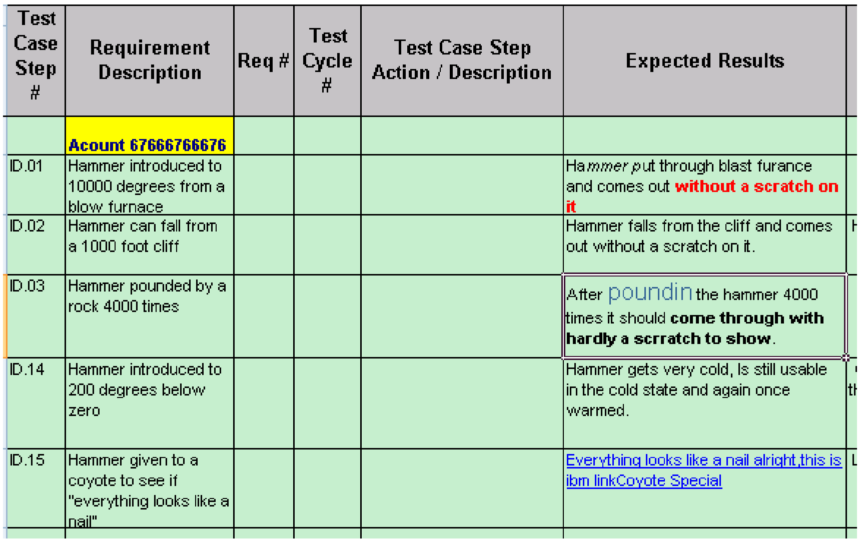
Some data can be modified by the tool to facilitate creation of artifacts. These are used on the right side of the equals sign. Except for the ampersand ( `&` ) the notation is like a function. The name of the modifier then the data wrapped in parentheses, for example: `testcase.attachment=Pre(C4)`

The data within the parentheses can be defined as anywhere else, and the list of modifiers are:

`Pre()` causes the data to be wrapped in an XHTML pre tag. This tells RQM that the data within is preformatted, and so the new lines are preserved. Note: that in most circumstances the migration tool will automatically convert new lines into XHTML paragraph markers to preserve the formatting.

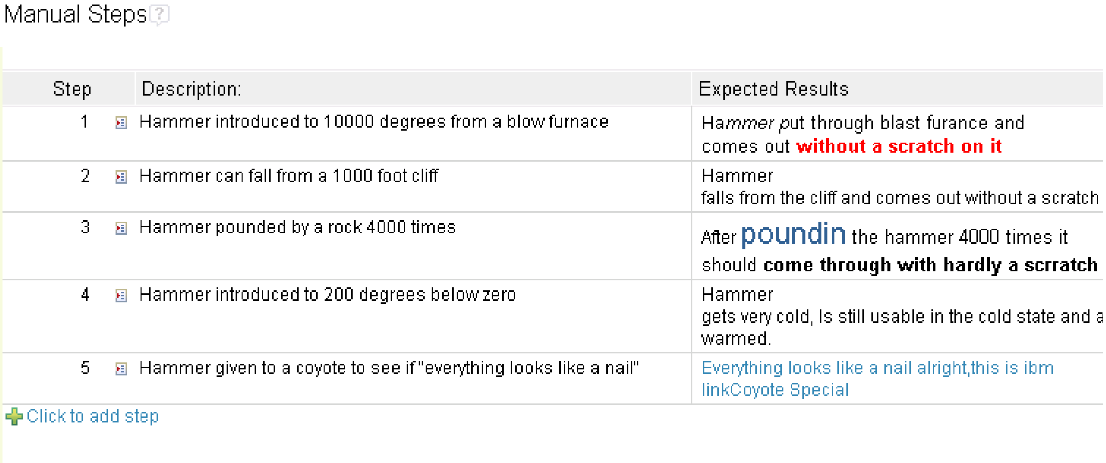
`Div()` Similar to the `Pre()` modifier, except that it only wraps the data in the XHTML tag. This will still preserve the line breaks, by replacing them with the XHTML <br/> tag.

RichText() used to display the word/excel document format in the RQM rich text area. The corresponding RQM target attributes need to support rich text content. Details can refer to Sample "Test Cases made from scripts". This modifier is supported from RQM Excel Importer 4.0.1.



| Test Case Step # | Requirement Description | Req # | Test Cycle # | Test Case Step Action / Description | Expected Results |
|---|---|---|---|---|---|
| | **Acount 67666766676** | | | | |
| ID.01 | Hammer introduced to 10000 degrees from a blow furnace | | | | Ha*mmer pu*t through blast furance and comes out **without a scratch on it** |
| ID.02 | Hammer can fall from a 1000 foot cliff | | | | Hammer falls from the cliff and comes out without a scratch on it. |
| ID.03 | Hammer pounded by a rock 4000 times | | | | After poundin the hammer 4000 times it should **come through with hardly a scrratch to show**. |
| ID.14 | Hammer introduced to 200 degrees below zero | | | | Hammer gets very cold, Is still usable in the cold state and again once warmed. |
| ID.15 | Hammer given to a coyote to see if "everything looks like a nail" | | | | Everything looks like a nail alright,this is ibm linkCoyote Special |

convert office format to RQM rich text format



Manual Steps

| Step | Description: | Expected Results |
|---|---|---|
| 1 | Hammer introduced to 10000 degrees from a blow furnace | Ha*mmer pu*t through blast furance and comes out **without a scratch on it** |
| 2 | Hammer can fall from a 1000 foot cliff | Hammer<br>falls from the cliff and comes out without a scratch |
| 3 | Hammer pounded by a rock 4000 times | After poundin the hammer 4000 times it should **come through with hardly a scrratch** |
| 4 | Hammer introduced to 200 degrees below zero | Hammer<br>gets very cold, Is still usable in the cold state and a<br>warmed. |
| 5 | Hammer given to a coyote to see if "everything looks like a nail" | Everything looks like a nail alright,this is ibm linkCoyote Special |

Click to add step

From Excel/Word Importer 4.0.3, when migrating data in rich text format, predefined style as well as customized style is supported in Word/Excel Importer.

**NOTE: RichText() modifier is not suggested to be used when importing a large number of artifacts form one document.**

`Link()` Specifies that the content is the ID of an artifact to be linked to. This modifier can be used in several ways; the left side of the equals specifies the type of artifact to link to, `testcase.testscript=Link(C)` . If the content within the link is formatted to use the full notation, the artifact type listed to the left of the equals is ignored and the data is used raw, `testcase.testscript=Link("../requirement/myimportedreq")` . If the content of the data is a list of items to link to separated with new lines, each will be made into a link. The `XLSArtifactID` keyword is the most common way to create artifacts who's ID is known so as to be able to reference it using the Link modifier.

From Excel/Word Importer 4.0.3, Link() modifier supports string/cell/list/table concatenation, and each value should be separated by "&".
i.e.
Link("pre" & D & "end")
Link("pre" & List("*.1.1"))
Link("pre" & Table(1).B2)

`LinkExisting()` modifier - indicates a reference to the master test artifacts, content is the external/internal id or the excel cell/column that stores the external/internal id of the master artifact on RQM server.
i.e.
testplan links with testcase, then test plan is master artifact and testcase is sub/child artifact.
test case link with test script, then test case is master test artifact, and test script is child artifact.
test suite link with test case, then test suite is master test artifact, and test case is child artifact.

the `LinkExisting` modifier only deals with the support for linking master artifact and the artifact must exist on the RQM server side first.

the syntax will be
testcase.testplan=LinkExisting("testplanid"), testplanid1 is the external/internal id of the test plan on RQM server.
testscript.testcase=LinkExisting(E1)
testcase.testsuite=LinkExisting(F)

`Image()` used to import an image from the define location. The left side must refer to a rich text field as the modifier automatically creates an embedded image tag to the new image that gets imported, along with adding the image to the list of attachments.

`&` The ampersand symbol is used to add two or more bits of data to the same field. The data can be a mix of items such as: `testcase.description=”my description:” & C4 & testscript.description`

# Word Migration Configuration

## Data locations and self referential links

The right side of the equals sign of a configuration must contain either a literal string, a table or an outline definition. A table and an outline are denoted by "`Table`(" and "`List`(" respectively, and must be concluded with a right parenthesis, "`)`".

### Outline notation:

The content definition for an outline list can consist either of indentation level or single wildcard denoted by an asterisk, "`*`". An indent level is just a number, whereas the wildcard must be denoted within quotes. Either can then be optionally further defined with a specific paragraph number after a comma.

The following states that an outline who's headline match "1.*." taking only the first paragraph from that to be the testcase title:

```
testcase.dc:title=List("1.*.",1)
```

Omitting the comma and the number will put the entire content of the outline level into the title field.

Using a number without quotes will take the outline that many indentations in:

```
testcase.dc:title=List(1,1)
```

### Paragraphs without outline list

Starting from RQM 406 a new syntax for paragraphs without outline list was added. This new syntax allow users to include a range of unlisted paragraphs under a list paragraph. The following is an example.

word document:

1.Test Case 1
1.1.Test Case Design 1
I’m the first paragraph under test case 1 with no number
I’m the second paragraph under test case 1 with no number
I’m the third paragraph under test case 1 with no number
I’m the fourth paragraph under test case 1 with no number
I’m the fifth paragraph under test case 1 with no number
I’m the sixth paragraph under test case 1 with no number

2.Test Case 2
2.1.Test Case Design 2
I’m the first paragraph under test case 2 with no number
I’m the second paragraph under test case 2 with no number
I’m the third paragraph under test case 2 with no number

cfg file:

testcase.dc:title=List("*")
testcase.Section("myns:com.ibm.rqm.planning.editor.section.testCaseDesign","RQM-KEY-TC-DESIGN-TITLE")=List("*.1,[1,10]")

"[1,10]" means including paragraph 1 to 10 under a list paragraph. Since no Test Case Design has more than 10 paragraph below so all the paragraph under Test Case Design will be included in the section of testCaseDesign.

### Table notation:

A table is defined either by the number at which it exists or by the first cell's contents. A table defined by a number is static, so that each artifact using that table will always reference that exact table, where as the named table is dynamic, meaning if you have more than one table in the document that matches the text, more than one artifact will be created. After a table is defined, it must then be followed by a period and a cell or column reference, the same way an Excel cell or column is defined.

The following defines a static table and a static cell:

`testscript.dc:title=Table(1).B3`

A dynamic table matches the start of text, case sensitive, ignoring white space. The following will take information from column B from all tables who's first cell starts with "Step":

`testscript.steps.title=Table("Step").B`

## Self referential links:

To define a self reference, simply put in the information from the left side of the equals from a previously defined line:

`testscript.dc:title= Table(1).G`

`testcase.dc:title=testscript.dc:title`

**Note:** Self references are used to create artifacts that will use data already being used by another artifact and will create a link to that artifact.

## Keywords and special cases

`XLSLink keyword` specifies that this artifact should be linked to all artifacts of the specified type within the document.

`testcase.XLSLink=testscript`

The example will link all testscripts within the document to all testcases.

`DOCStart` and `DOCStop` are used for specifying what list definition to start and stop at, it is only used for list types and must be a string literal:

`testscript.DOCStart="5."`

`testscript.DOCStop="12."`

This will cause the list processing to start when the "5." list item is found, and stop when the "12." list item is found. These are exact matches, so list item "12.1" will not end processing.

`DOCTableStartRow` when importing artifacts from the word table, DOCTableStartRow used to specify which row to start constructing the artifacts.

Any target definition can be wrapped in a `Pre()` modifier:

`testcase.Section("myns:com.ibm.rqm.planning.editor.section.testCaseAcceptanceCriteria", "RQM-KEY-TC-ACCEPT-CRITERIA-TITLE")=Pre(Table(1).C)`

or

`testcase.Section("myns:com.ibm.rqm.planning.editor.section.testCaseAcceptanceCriteria", "RQM-KEY-TC-ACCEPT-CRITERIA-TITLE")=Pre("this is my text")`

Which will wrap the resulting data in div and pre tags preserving the formatting information.

Sections are special cased to simplify their use and to keep their definitions from interfering with the period notation used otherwise. To update a section, simply use the following format:

`testcase.Section("myns:com.ibm.rqm.planning.editor.section.testCaseAcceptanceCriteria", "RQM-KEY-TC-ACCEPT-CRITERIA-TITLE")=Table(1).C`

This example will populate the acceptance criteria section that is built into the RQM product.

See how to determine the name/display name of predefined sections.

To define a custom section, simply replace the first parameter with a valid name, and the second with a display name:

`testcase.Section("myns:mysection","my custom section")=Table(1).C`

Categories are define with the category term=<category name>.value=<category value> notation. Your category name goes within the quotes:

`testcase.category term="Function".value=Table(1).H`

`&` The ampersand symbol is used to add two or more bits of data to the same field. The data can be a mix of items such as: `testcase.description="my description:" & Table(1).H & testscript.description`

*

You are also able to access the document properties of the current file that is being exported. `Examples:`

`requirement.dc:title=property("Title")`

`requirement.dc:creator=property("Author")`

`requirement.jzalm:owner=property("Company")`

The properties accessible are built into Microsoft Office are found at http://msdn.microsoft.com/en-us/library/microsoft.office.core.documentproperty.aspx

There are also three other properties available, Name, Full Name, and Location. If the property has not been set or is not supported by Word, property("property name") will return the empty string "".

The related sample xls/cfg sample files are titled "Testscript in Tables using Document Properties."

**NOTE from Word Exporter 4.0**:

we changed the design and artifact in list is expected to separated by "\n" , and don't set "\n" inside one artifact.
e.g.
1. title
1.1 testcase1 title
1.1.1 testscript1 reference to the testcase1
1.1.1.1 testscript1 step1 title
1.1.1.1.1 testscript1 step1 description
1.1.1.1.2 testscript1 step1 expectresuit
1.1.1.2 testscript1 step2 title
1.1.1.2.1 testscript1 step2 description
1.1.1.2.2 testscript1 step2 expectresuit
1.1.1.3 testscript1 step3 title
1.1.1.3.1 testscript1 step3 description
1.1.1.3.2 testscript1 step3 expectresuit
1.1.1.4 testscript1 step4 title
1.1.1.4.1 testscript1 step4 description
1.1.1.4.2 testscript1 step4 expectresuit

1.2 testcase2 title
1.2.1 testscript2 reference to the testcase2
1.2.1.1 testscript2 step1 title
1.2.1.1.1 testscript2 step1 description
1.2.1.1.2 testscript2 step1 expectresuit
1.2.1.2 testscript2 step2 title
1.2.1.2.1 testscript2 step2 description
1.2.1.2.2 testscript2 step2 expectresuit
1.2.1.3 testscript2 step3 title
1.2.1.3.1 testscript2 step3 description
1.2.1.3.2 testscript2 step3 expectresuit
1.2.1.4 testscript2 step4 title
1.2.1.4.1 testscript2 step4 description
1.2.1.4.2 testscript2 step4 expectresuit

The blank line between the two testcase titles denotes the testcase separation

# Frequently Asked Questions

* Frequently Asked Questions

▼ Attachments

| I | Attachment | Action | Size | Date | Who | Comment |
|---|---|---|---|---|---|---|
| | ExcelImporterLinkingToRMrequirement.png | manage | 35.5 K | 2012-08-24 - 08:57 | HaoWan | Linking to RM requirement check box |
| | ExcelRequirementLinks.png | manage | 75.1 K | 2012-08-24 - 08:58 | HaoWan | Requirement links dialog |
| | ExcelRequirementLinksDefaultList.png | manage | 54.4 K | 2012-08-24 - 09:00 | HaoWan | check box will auto change according to the test artifact |
| | ExcelRequirementLinksErrorMsg.png | manage | 60.7 K | 2012-08-24 - 09:02 | HaoWan | error msg |
| | ExcelRequirementLinksOneMoreLink.png | manage | 52.8 K | 2012-08-24 - 09:03 | HaoWan | one more link |
| | ExcelSpreadsheet1.png | manage | 3.1 K | 2012-10- | ChipSenkbeil | Example of test case being |

| | | | | | |
|---|---|---|---|---|---|
| | | | 01 - 19:43 | | generated 'unexpectedly' from exec variable |
| OfficeFormatConvert.png | manage | 24.6 K | 2012-08-24 - 12:26 | HaoWan | convert office format to RQM rich text format |
| PartialExportConfirmationDialog.png | manage | 19.8 K | 2012-04-10 - 03:02 | HaoWan | Partial Export confirmation dialog |
| Predefined_requirement_types.png | manage | 41.8 K | 2012-12-06 - 06:02 | HaoWan | Predefined requirement types |
| RMLogin.png | manage | 46.9 K | 2012-04-10 - 03:05 | HaoWan | RM Server Login dialog |
| Requirement_exporting_results.png | manage | 31.3 K | 2012-12-06 - 06:04 | HaoWan | Requirement exporting results |
| Specified_requirement_types_in_Excel.png | manage | 47.1 K | 2012-12-06 - 06:06 | HaoWan | Specified requirement types in Excel |
| WordExcelExportToFile.gif | manage | 11.9 K | 2009-07-28 - 15:31 | WilliamTobin | Import to File |
| WordExcelExportToRQM.gif | manage | 18.1 K | 2009-07-28 - 15:31 | WilliamTobin | Import to RQM |
| WordExcelUXToFile.png | manage | 70.2 K | 2012-04-10 - 03:06 | HaoWan | Export to File |
| WordExcelUXToRQM.png | manage | 56.4 K | 2012-04-10 - 03:07 | HaoWan | Export to RQM |

Edit | WYSIWYG | Attach | Printable | Raw View | Backlinks: Web, All Webs | History:
r107 < r106 < r105 < r104 < r103 | More topic actions