Process Release Management with Rational Method Composer

Table of contents

1 Introduction

This article will show how to explain how to plan for, deliver, and manage releases of processes using Rational Method Composer. This article is intended to supplement the general guidance in the Method Development Practice included with the IBM Practices Library.

It assumes that you are using Rational Team Concert (Jazz) source control for your library files, and that you have BeyondCompare 2, a file comparison utility provided by http://www.scootersoftware.com/.

There are other source control options and difference tools that can be used, but they are not covered in this article.

2 Release process considerations

The following are issues that you need to consider.

After each issue, IBM's approach is described. You may choose a different approach.

1. How will releases be named/numbered?

   IBM approach:

We include the RMC product release number in our library release number. So if a library was created with RMC 7.5.2, the library release is 7.5.2-1.

2. Will you be forking?

   A fork occurs when you maintain parallel streams.

   IBM approach: We do not fork our library. Note that it is difficult to merge forked processes.

3. Will users of the existing process be able to continue using that process?

   IBM approach: Once a process website has been released, it remains deployed in the same location until we are certain no one is using it. For minor changes that don't cause links to break, we over-write the existing website and don't change the version number. Release notes and version control describe what happened so that you can roll back if needed.

   We maintain an alias URL to the latest, for those who always want access to the latest.

   For example, if the website is deployed to:
   http://<server>/<configuration_name><release_number>

   Then we have another URL for:

   http://<server>/<configuration_name>_latest

4. What tests are required before a new process can be released?

   IBM approach: See this testing guideline. Also see the Guideline: Testing a Method Asset (in the Method Development Practice of the IBM Practices Library.

5. What approvals are required?
   IBM approach: Product team and development team approval is required.

6. What do we assign version numbers to and take snapshots of? The entire library? Configurations? Specific groupings of plug-ins? Individual plug-ins? Specific elements?

   IBM approach: We group plug-ins that are released together and put each group in a separate Jazz Source Control component or separate repository so that they can be easily versioned as a set.
   For the IBM Practices library, we have 3 components:

   a. EPF, which are the open source plug-ins. They are versioned in GIT on the Eclipse website.
   b. IBM shared – these are plug-ins which are shared with other non-commercial libraries in IBM.
   c. Rational – these are plug-ins that are not shared with other libraries in IBM.

   We assign a version number to the library, to practices, and to configurations. Open source (EPF) Practices are also versioned at the same time, but are given a different version number that corresponds to the EPF Composer versions.
   The configuration version information is captured as a release page in the corresponding "publish" plug-in.

   A practice has a release page as well – the EPF version information and IBM version information are combined using "contributes" variability. To see the release information,

browse the All Practices configuration, and open the last element in the treebrowser.

7. What related assets need to be updated?

   For example:

   a. Work item templates in Rational Team Concert with links to the current process

   b. Microsoft project templates with links to the current process

   c. Links from other tools / websites / documents

   IBM approach: Rational Team Concert process templates include the name and version of the corresponding configuration. Process and tooling changes are released together.

8. Where will release notes be kept?

   IBM approach: The configuration version information is captured as a release page in the corresponding "publish" plug-in.

   A practice has a release page as well – the EPF version information and IBM version information are combined using "contributes" variability. To see the release information, browse the All Practices configuration, and open the last element in the treebrowser.

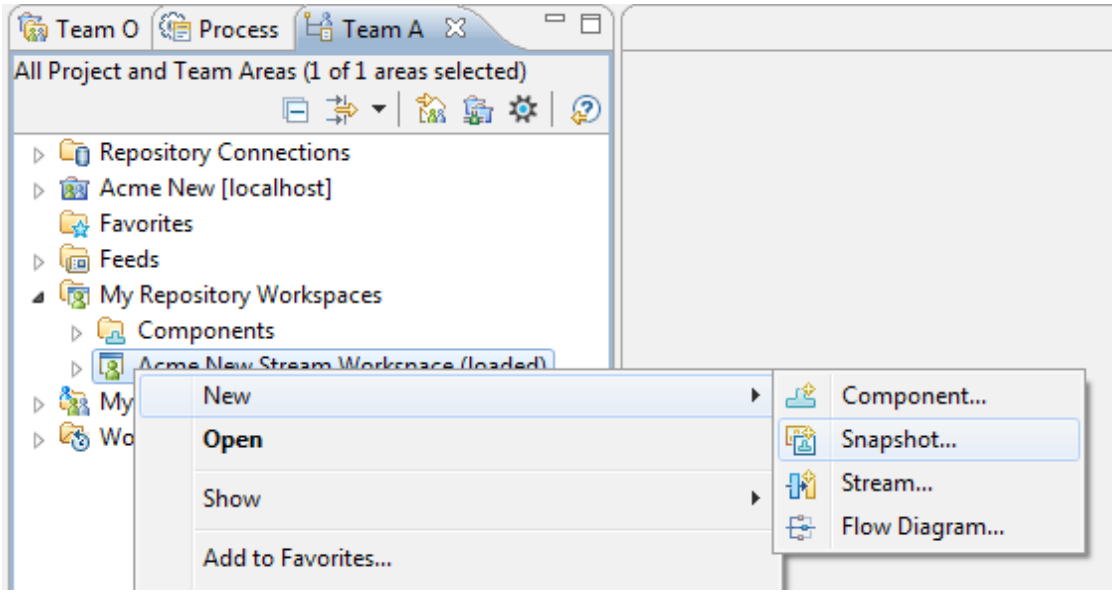
   Note that although plug-ins and content elements have attributes for Version, Change date, , Change description, and Authors. We do not to use this feature, because this information is not published in the website.

3 Creating a release

In order to be able to track what changes from one release to another, it is important to take a snapshot of the library and/or groups of plug-ins that are versioned together.

A snapshot baselines each of the components in your workspace repository. Name the snapshot according to your release naming guidelines.

To take a snapshot, in the *Jazz Administration* perspective, and *Team Artifacts* view, right-click on a repository workspace and select New > Snapshot.



Give the snapshot a name and click OK.

In addition to taking a snapshot, it is generally convenient to zip up the library, and published websites, and place those in source control as well. That way you can easily access and open any released library version.

4 Creating release notes

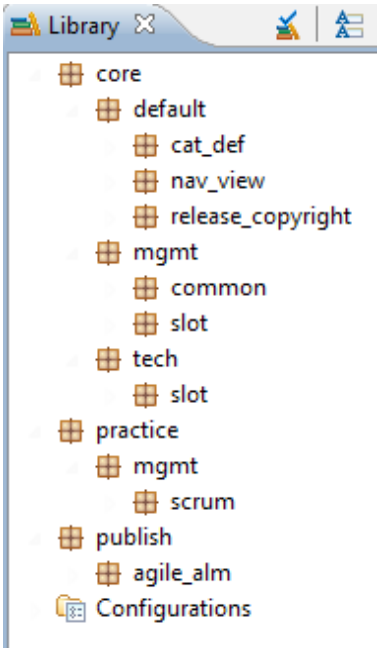
Good release notes should describe the significant differences from one release to the next.

If you take a snapshot for each release, you can compare those snapshots to identify what has changed, including:

- View the work items that were delivered from one release to the next, including changesets (the files delivered with each work item).
- Compare RMC Library Reports – These reports can be created out of the box and contain a summary of the library in excel format. Compare these reports to see additions, deletions, renames, and changes to brief description text.
- Compare HTML Exports – HTML files can be exported from RMC. Compare these files to see changes to rich text fields.
- Compare Published Configurations – These can be compared to see configuration changes.
- Compare RMC Library Files – These can be compared for a quick but imprecise view of differences.

In order to demonstrate different techniques for generating release notes, we have created an example consisting of two releases.

Release 1 – Acme Library



This is the first released version of the Acme library. The Acme Library is a process based on Scrum. A snapshot was created called “Acme Library Release 1.”

Changes to Content Packages:

In practice.mgmt.scrum.base-ibm:

1. added role Business Owner
2. added task communicate_business_context
3. added relationship: role Business Owner to communicate_business_context
4. added rich text to daily_scrum
5. removed rich text from plan_sprint

In practice.mgmt.scrum.extend-acme:

1. changed name and presentation name of role UX_Designer to experience_architect
2. changed name and presentation name of task create_ux_design to create_experience_design
3. change just the presentation name of work product ux design to experience design
4. delete role UX_Specialist
5. change the brief description of the task create_ux

In practice.mgmt.scrum.assign-acme:

1. Contributing task share_product_vision.assign_role was renamed to document_vision.assign_role
2. deleted contributing task monitor_sprint_progress.assign_role
3. removed role relationship from contributing task plan_release.assign_role (removed scrum master)
4. added relationship: product backlog added as Input to populate_product_backlog.assign_role

In Configurations:

1. remove check from plug-in core.tech.slot.base

These changes were checked-in and delivered with the comment “new changes to Acme project” and to the tasks “Change UX Development to Experience Design” and “Make updates for upcoming release.”

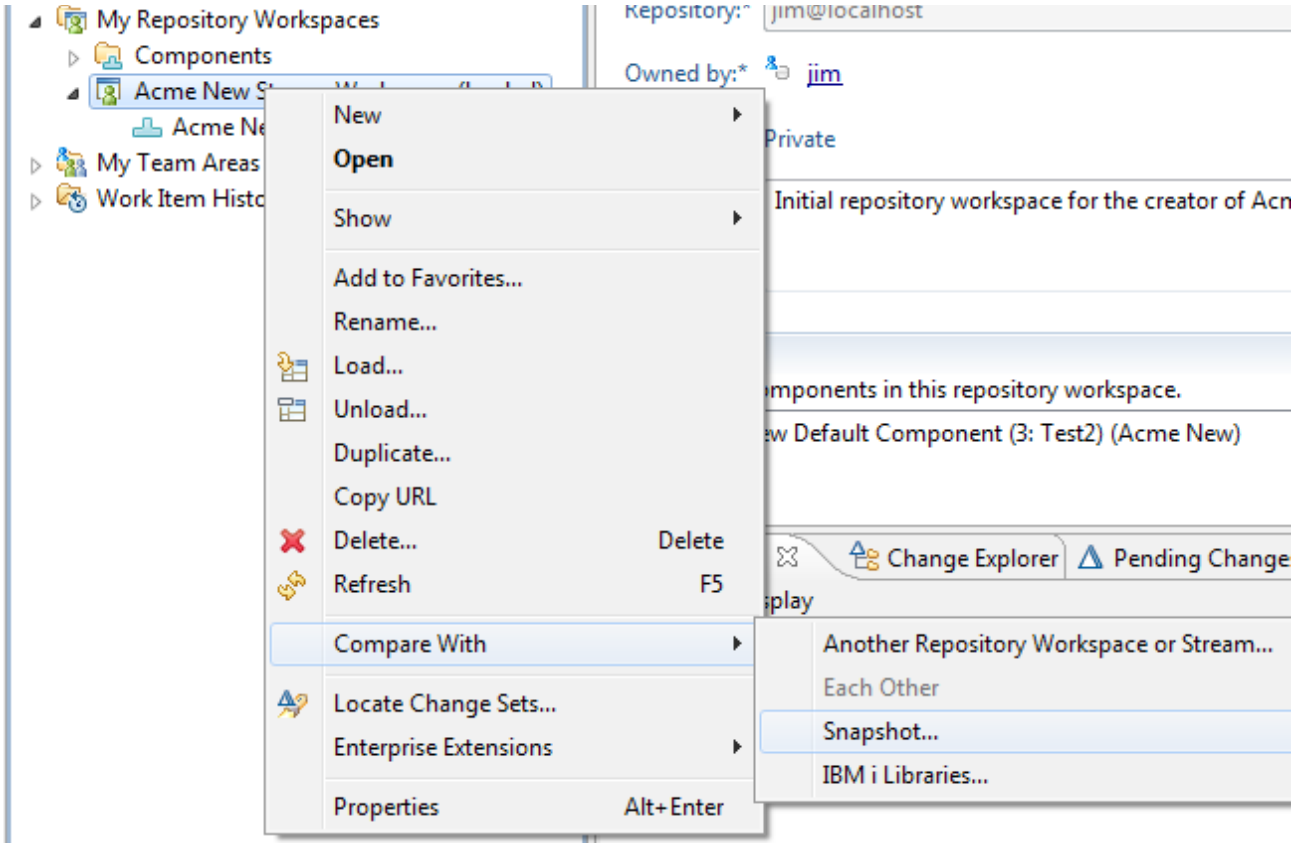
Release 2 – Acme Library

After making these changes, we take a snapshot, named Acme Library Release 2

4.1 Viewing changesets

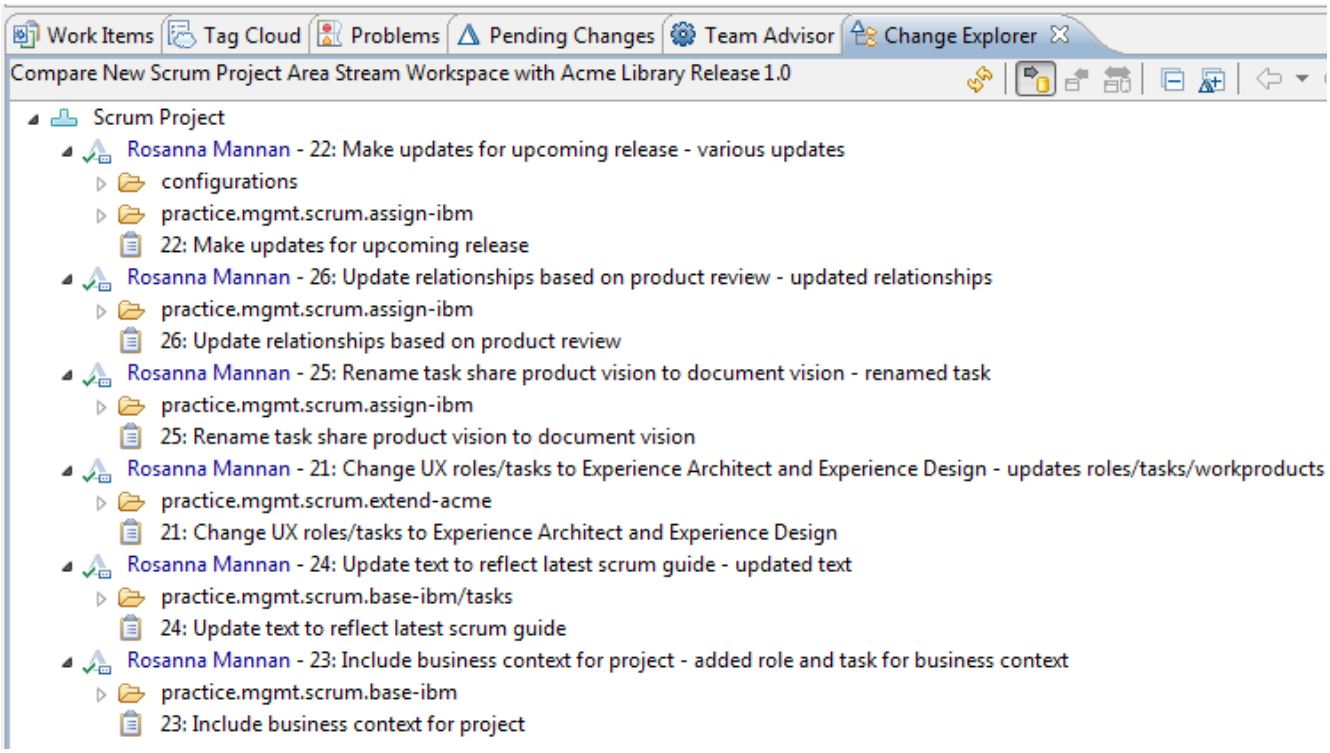
The changesets that were delivered from one release to the next will identify the work items that were delivered, the files that were changed for each work item, and comments that were delivered when the files were delivered. If work item titles and comments are descriptive, it is easy to write release notes.

To see the changesets that were delivered from one release to the next, you can do the following: right-click on the workspace and select Compare With > Snapshot.



Select the stream or workspace, then click Next. Select the snapshot, then click Finish. The Change Explorer will show the changes between the workspace and the snapshot.

For our example, you can see the following information:



We can see informative work item titles, such as “Rename task share product vision to document vision”, and a less informative title “Make updates for upcoming release”. We can see informative comments (these appear after the dash), such as “added role and task for business context”. This underscores the value of informative work item names and comments.

4.2 Recreating a snapshot

If you want to use 3rd party tools to compare releases, you will need to recreate the files for the previous release. If you zipped up and stored the library in a separate location, you can use that. Another approach is to recreate the snapshot from source control.

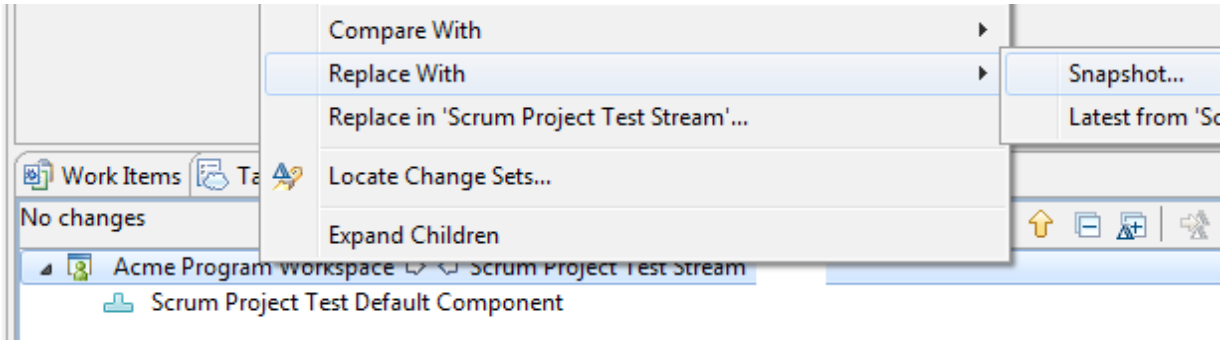
To recreate a snapshot using Jazz source control, follow these steps:

1. Open RMC using a new workspace
    a. Create a shortcut for RMC that uses a new workspace
        i. Click **Start > All Programs > Rational Method Composer**
        ii. Right click on **Rational Method Composer**
        iii. Click *copy*
        iv. Right click on the desktop and click *paste shortcut.*
        v. Rename the shortcut to “RMC-release1_snapshot”
        vi. Right click on the shortcut and click *properties*
        vii. Append “ -data <workspace_path>\ release1_snapshot” (without quotes, including an initial space, and using an appropriate workspace path) to the *target*. This needs to be an empty folder on your machine. For example, “ -data

C:\RMC\workspace\release1_snapshot." Click OK.

b. Launch RMC using the new shortcut.
c. Select **File>Open>Method Library**.
d. Check the box for *Open method library from workspace*, then click Finish.

2. Create a workspace repository (see instructions here)
3. Load the snapshot into the workspace repository
    a. In the Pending Changes tool window, right-click the workspace and click Replace With > Snapshot.



Select your workspace and click Next. Then, select the Acme Library Release 1 snapshot and click Finish.

The snapshot files are now ready for comparison with either the files in your current workspace, or with another snapshot workspace.

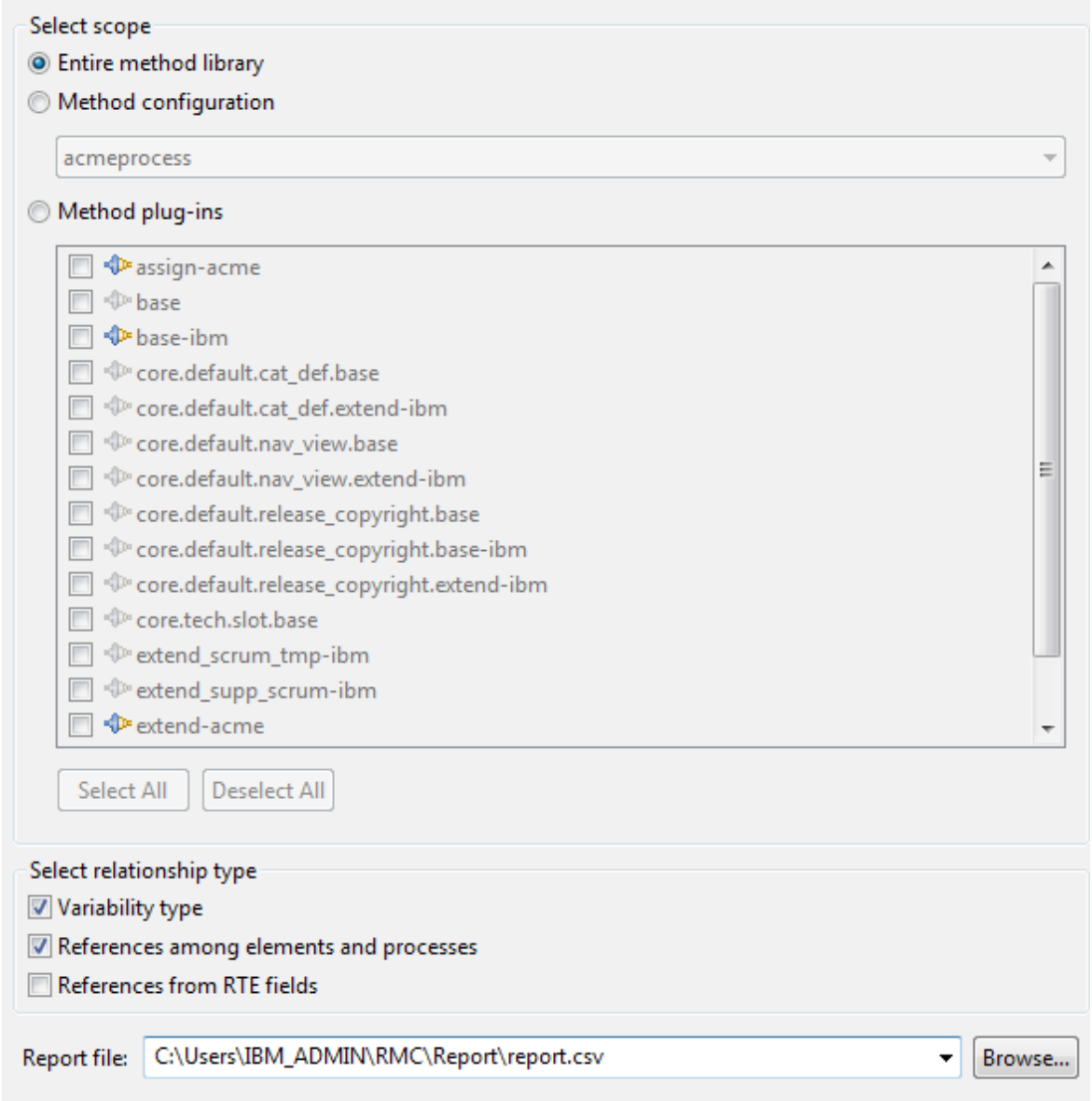
4.3 Comparing RMC library reports

RMC provides a report that exports basic information about elements in the library, including names, ids, and relationships. Comparing the report files generated for each release yields useful information about differences, most importantly:

- Additions of new elements
- Deletions
- Changed names
- Changes to brief descriptions
- Changed relationships

4.3.1 Generate the Report

Go to File > Report. Under Select scope, select Entire method library, and under Select relationship type, check the first two check boxes. Click Finish. Do this for both workspaces.



You should now have two reports, one for Release 1 and one for Release 2.

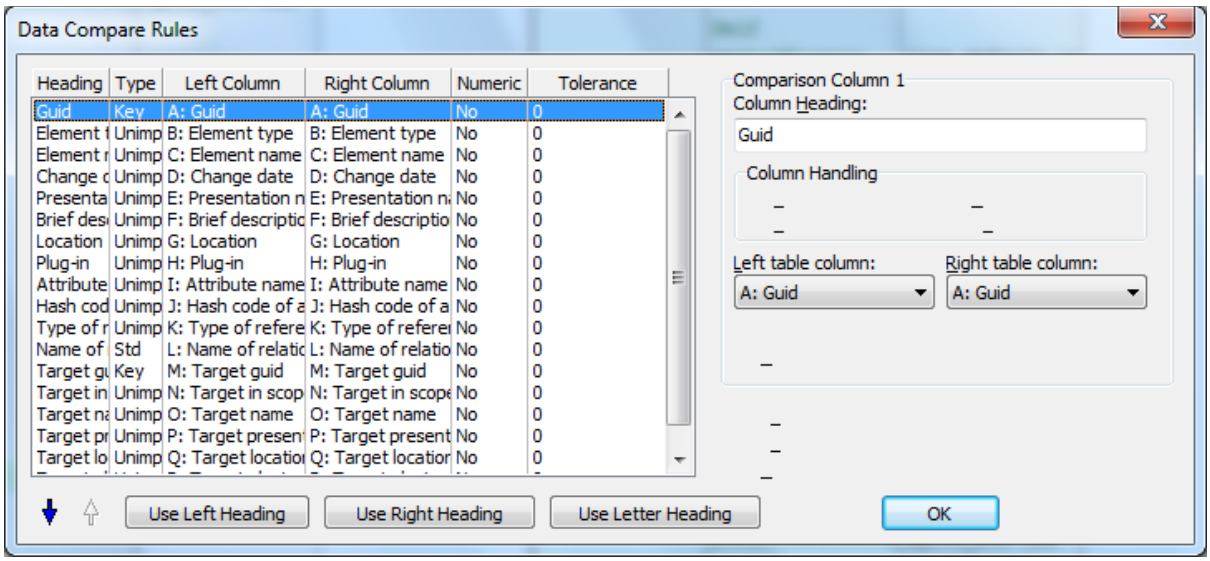
4.3.2 Comparing using BeyondCompare 2

After installing and launching BeyondCompare 2, select New File Comparison. Select both reports for a comparison.

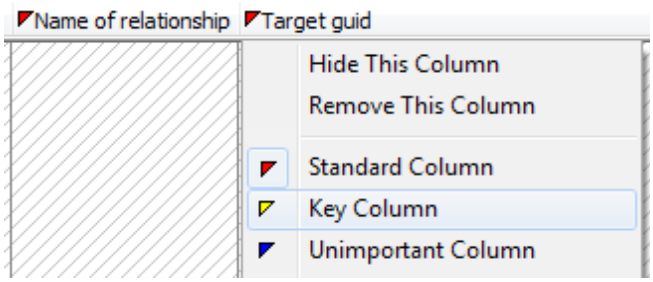
1. Recommended settings

Use the following recommended settings:

1. Use the Data Viewer
2. When prompted for rules:



1. Click on Use Right Heading
2. Set the following columns to “Key”
    a. Guid
    b. Target Guid



3. Leave the following columns as “Standard”
    a. Element name
    b. Presentation name
    c. Brief description
    d. Name of relationship
4. Set all other columns to “Unimportant”.

    You can go to Tools > Save Rules to save these rules for future comparisons.

3. When viewing the results, go to View > Just Differences or click the Only Mismatches icon ≠
4. Go to View > Hide all Matching Columns
5. You can hide and unhide specific columns by clicking on a column.

    Unhide only these columns:
    a. Element name
    b. Presentation name

c. Brief description
d. Name of relationship
e. Target name

Go to Tools > Save Rules to save these rules for future comparisons.

2. Additions/deletions/renames of elements

Hide all but column E (Presentation name). We can see the following:

Presentation name	Presentation name
	Business Owner
	Communicate Business Context
	Communicate Business Context
UX Design	Experience Design
UX Designer	Experience Architect
UX Designer	Experience Architect
Create UX	Create UX
Create UX	Create UX
Create UX	Create UX
UX Specialist	

We can see that:

- elements that were added appear only on the right and are green
- elements that were renamed (presentation name) are red
- elements that were deleted appear only on the left, and are green

There are multiple entries because there is a row for each relationship that an element has.

If we look at column C (Element Name), we can see changes to element names.

Element name	Element name
share_product_vision.assign_role	document_vision.assign_role
share_product_vision.assign_role	document_vision.assign_role
share_product_vision.assign_role	document_vision.assign_role
monitor_sprint_progress.assign_role	
monitor_sprint_progress.assign_role	
monitor_sprint_progress.assign_role	
monitor_sprint_progress.assign_role	
monitor_sprint_progress.assign_role	
plan_release.assign_role	
	populate_product_backlog.assign_role
	business_owner
	communicate_business_context
	communicate_business_context
ux_design	ux_design
ux_designer	experience_architect
ux_designer	experience_architect
create_ux	create_ux
create_ux	create_ux
create_ux	create_ux
ux_specialist	

This is similar to the previous image, except that it also shows changes to contributing elements (they have no presentation name, and so don't appear in the previous image).

3. Changes to brief description

To see changes to the brief description, you need to view column F (Brief Description).

As before, there are multiple entries because there is a row for each of the task's relationships.

Brief description	Brief description
The purpose of this task is to create the UX Design.	This task creates the user experience.
The purpose of this task is to create the UX Design.	This task creates the user experience.
The purpose of this task is to create the UX Design.	This task creates the user experience.

4. Relationship changes

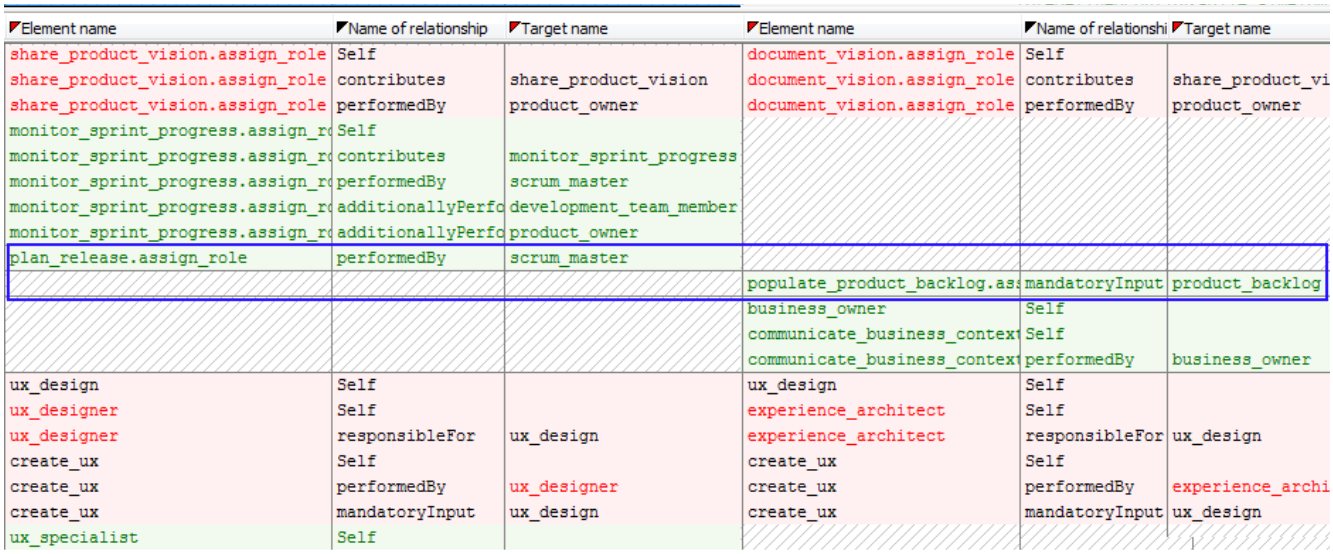
To see relationship changes, view these columns:

- Element name
- Name of relationship
- Target name

In the results below, the relationship differences are highlighted in blue.

The other differences are not relationship differences. You can tell this, because each block has a

"Self" relationship change, which means the element itself was added, deleted, or renamed.



Element name	Name of relationship	Target name	Element name	Name of relationshi	Target name
share_product_vision.assign_role	Self		document_vision.assign_role	Self	
share_product_vision.assign_role	contributes	share_product_vision	document_vision.assign_role	contributes	share_product_vi
share_product_vision.assign_role	performedBy	product_owner	document_vision.assign_role	performedBy	product_owner
monitor_sprint_progress.assign_r	Self				
monitor_sprint_progress.assign_r	contributes	monitor_sprint_progress			
monitor_sprint_progress.assign_r	performedBy	scrum_master			
monitor_sprint_progress.assign_r	additionallyPerfo	development_team_member			
monitor_sprint_progress.assign_r	additionallyPerfo	product_owner			
plan_release.assign_role	performedBy	scrum_master			
			populate_product_backlog.as	mandatoryInput	product_backlog
			business_owner	Self	
			communicate_business_contex	Self	
			communicate_business_contex	performedBy	business_owner
ux_design	Self		ux_design	Self	
ux_designer	Self		experience_architect	Self	
ux_designer	responsibleFor	ux_design	experience_architect	responsibleFor	ux_design
create_ux	Self		create_ux	Self	
create_ux	performedBy	ux_designer	create_ux	performedBy	experience_archi
create_ux	mandatoryInput	ux_design	create_ux	mandatoryInput	ux_design
ux_specialist	Self				

4.4 Comparing HTML exports

An RMC library can be exported to HTML files that exclude relationship information.

Comparing HTML export files is the best way to see differences in the rich text.

To export to HTML, do the following for each release:

1. Go to File > Export
2. Select HTML under Method Authoring.
3. Select Export Entire Library and select a destination directory.
4. Click Finish.

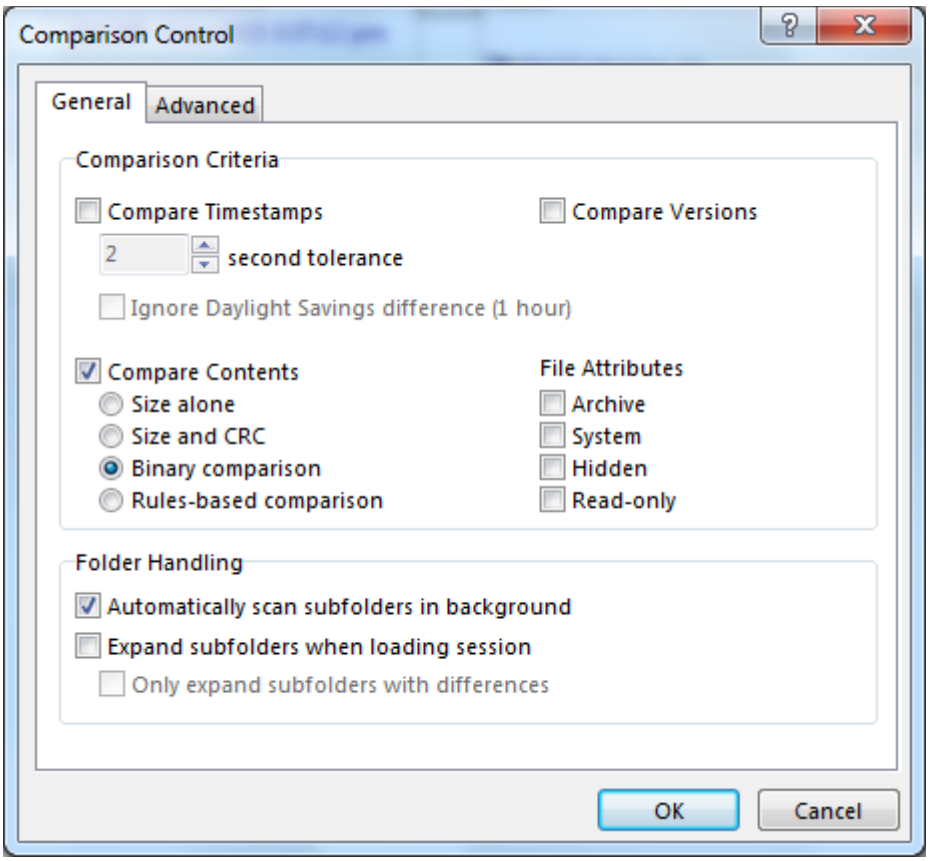
You should now have two folders with HTML files for each release. In BeyondCompare, select New Folder Comparison and then select each folder.

Make sure to click on the Only Mismatches icon ≠

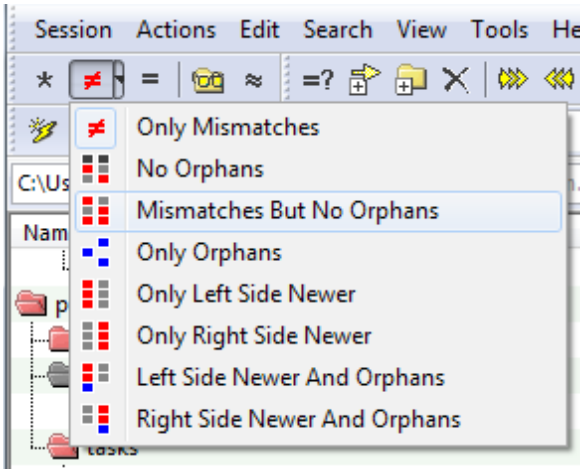
(this is to avoid seeing file additions, deletions, and renames).

Click the scales icon and use the following settings:

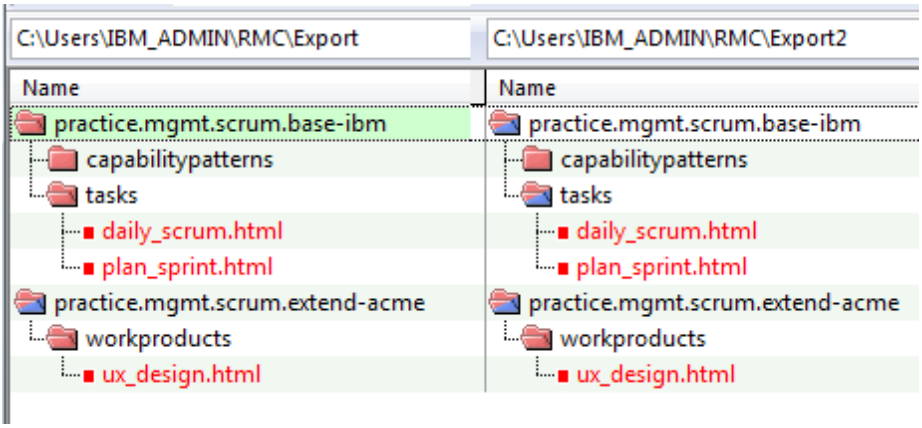
- deselect "Compare timestamps"
- select "Binary comparison"



When viewing the results, select Mismatches But No Orphans.



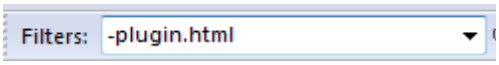
The results are shown below.



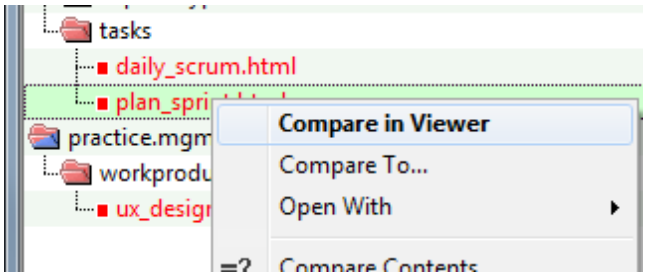
We can see that:

- elements that were renamed (presentation name) or had textual changes in rich text fields are red

Changes to to *plugin.html* files can be ignored, as these files only contain presentation name and brief description text changes, which we covered previously. You can filter out *plugin.html* files by setting them as excluded files. Under Filters, type "-plugin.html"



To see the actual changes in a file, right-click and select "Compare in viewer".



The results below are a comparison of plan_sprint.html.

The text in red was deleted.

4.5 Comparing Published Configurations

4.5.1 Configuration Changes

If a configuration has changed (such as a plug-in added or removed), this is easiest to see by comparing the published website files.

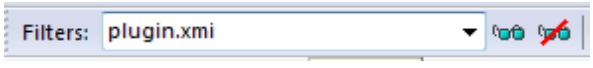
Other changes are also visible by drilling down into specific folders, such as added and removed elements. Changes to elements are more difficult to isolate in this kind of comparison because a single presentation name change can ripple through many files.

Note: The skin you select can affect how useful this report is. For example, if you used a skin that only published a single attribute, and nothing else, then your report would show only differences in that attribute.

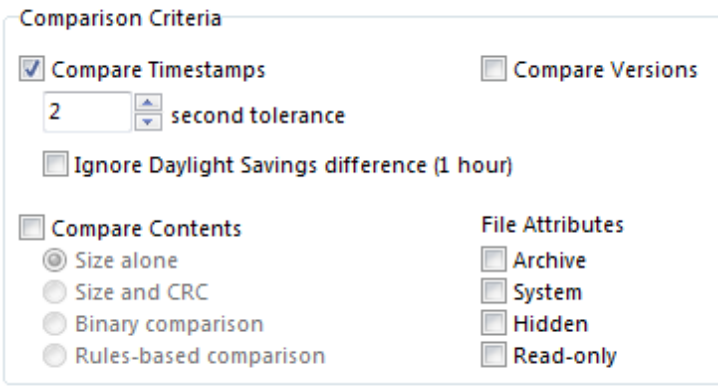
Publish each version of the configuration to create two folders and compare using the same BeyondCompare settings used previously.

Use the following recommended filter settings:

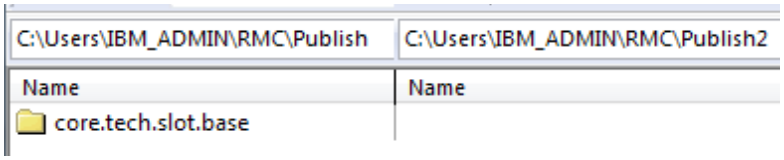
1. Set the include files filter to “plugin.xmi”



2. De-select “Compare Contents” and select “Compare Timestamps” in the Comparison Control feature (click on the Scales icon)



In the results below, we can see that the plug-in *core.tech.slot.base* has been removed from the configuration.



4.6 Comparing RMC Library Files

The RMC library files are either stored in a specific directory, or in the case of a workspace-based library, are stored in your workspace directory. Information is stored as follows:

- plugin.xmi - relationships, names, presentation names, brief descriptions
- library.xmi - list of plugins and configurations in the library
- <element name>.xmi - rich text associated with <element name>
- model.xmi - descriptors included in a capability pattern/delivery process
- content.xmi - rich text for the capability pattern/delivery pattern
- diagram.xmi - diagrams for the capability pattern/delivery process
- configurations/<configuration name>.xmi - configuration information
- user_defined_types.xml
- user_defined_types.xml - additions and modifications to RMC types definitions
- <process builder wizard name>.xmldef - a process builder definition file
- <element type>/resources/<filename> - files (typically graphics) attached to an element of <element type>

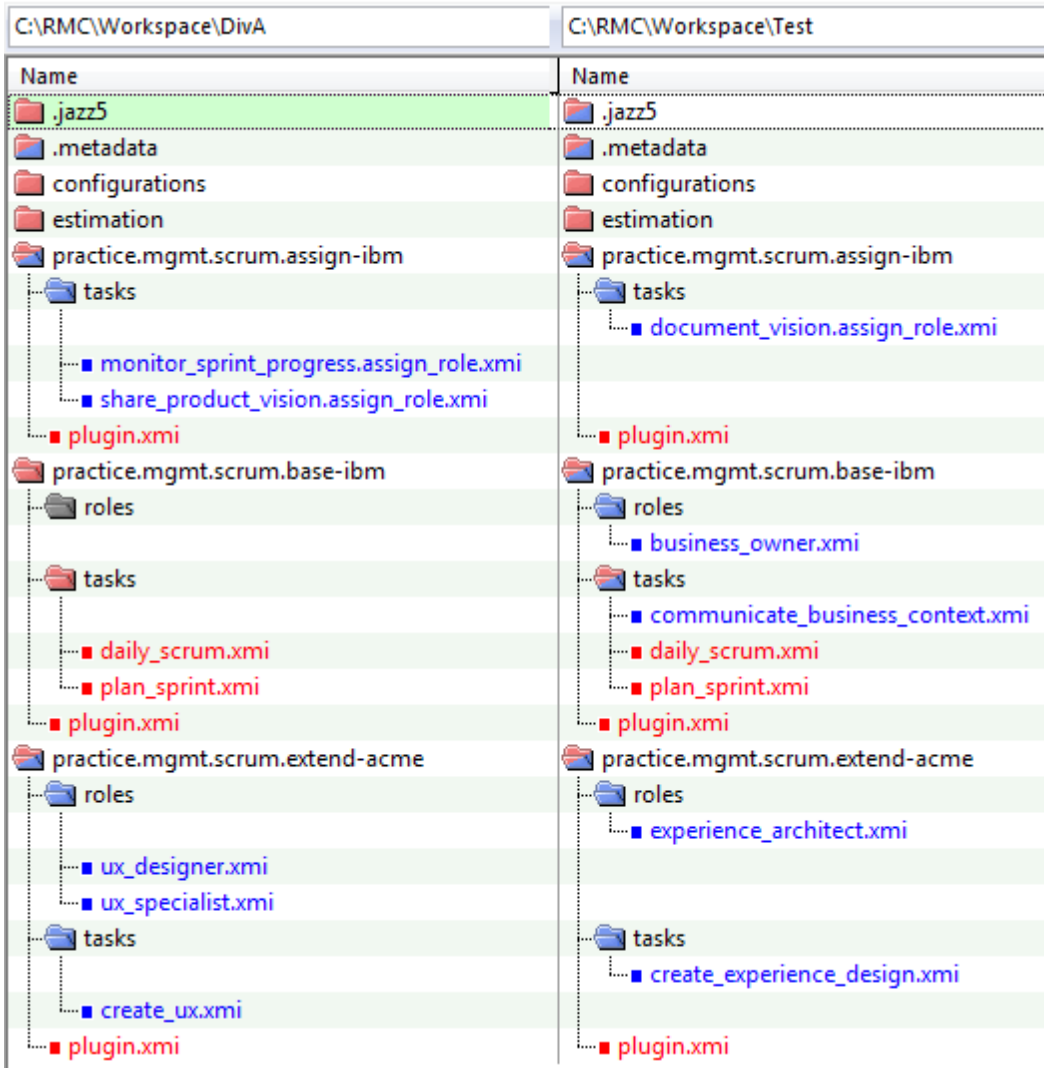
Compare the directories of two libraries using the same BeyondCompare settings used previously.

This is a quick but imprecise way to see which plug-ins have changed, and to see additions, deletions,

and renamed elements.

Note that comparing library files is not a good way to identify relationship or textual changes, and you can't easily distinguish between an element being renamed vs. one deleted and a new one added.

The results for our example are shown below:



We can see:

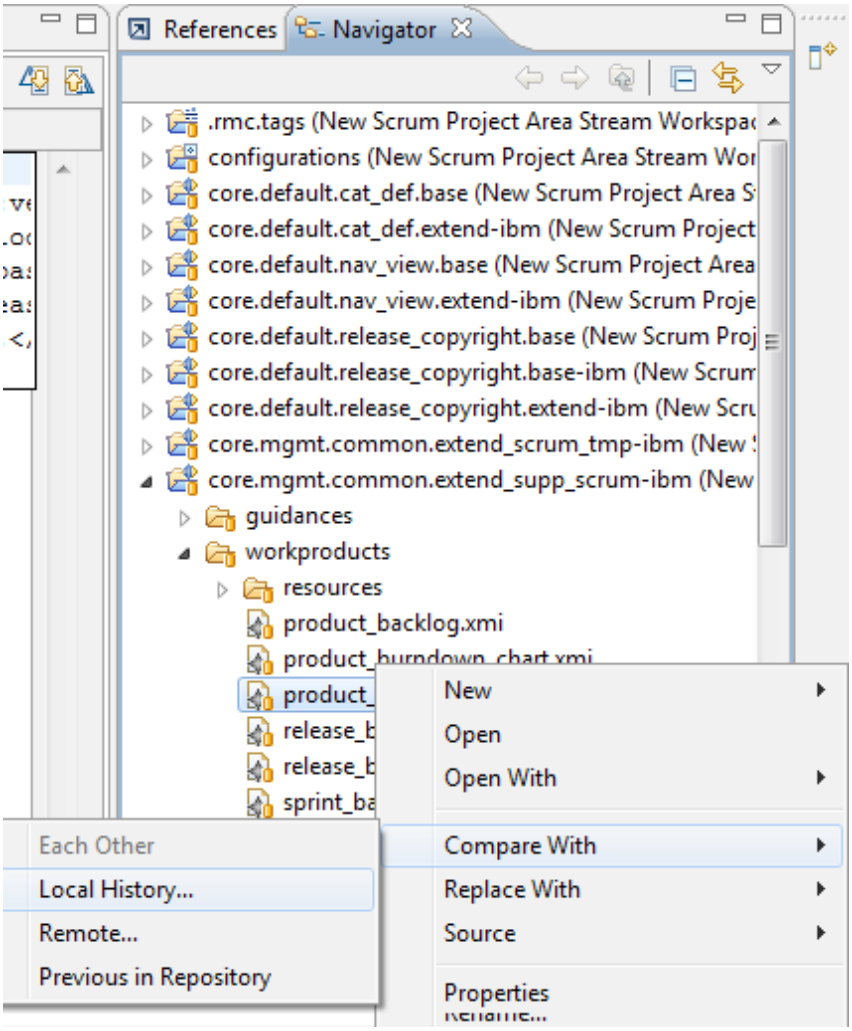
- deletions/additions/renames are blue
- changes are red

5 Appendices

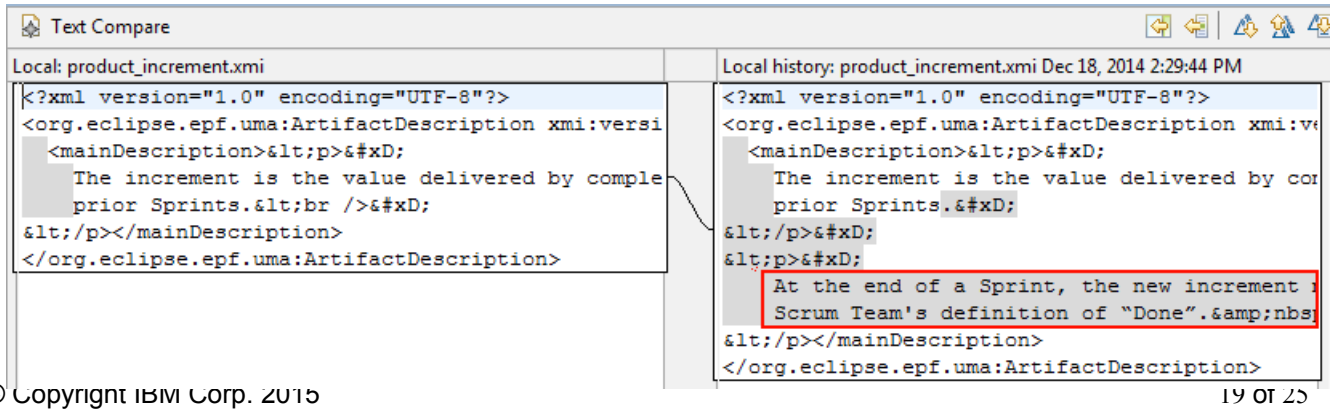
The following appendices go into more detail into other ways to compare libraries, additional tips on configuring your environment, and more detail on identifying differences.

5.1 Comparing Using Eclipse

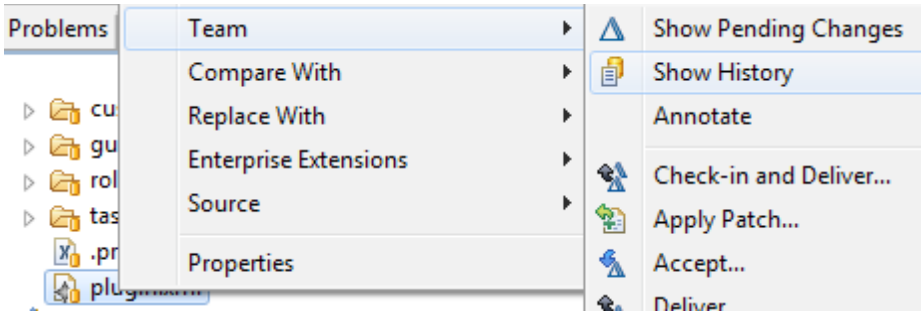
1. Go to the *Navigator* view, select an element, and then right-click and select Compare With > Local History. Then, choose the revision by its history. This compares the selected resource to one that is in the local history, which is maintained when you save changes. You can also configure your Local History settings by going to Window > Preferences > General > Workspace > Local History.



Comparing the element product_increment with its local history, we can see the text we removed from this work product.



If there isn't a local history available, right-click on Team > Show History.



Then, right-click on a revision and select Compare with Previous or Compare with Local Version.

5.2 Storing RMC Library Reports in your Workspace

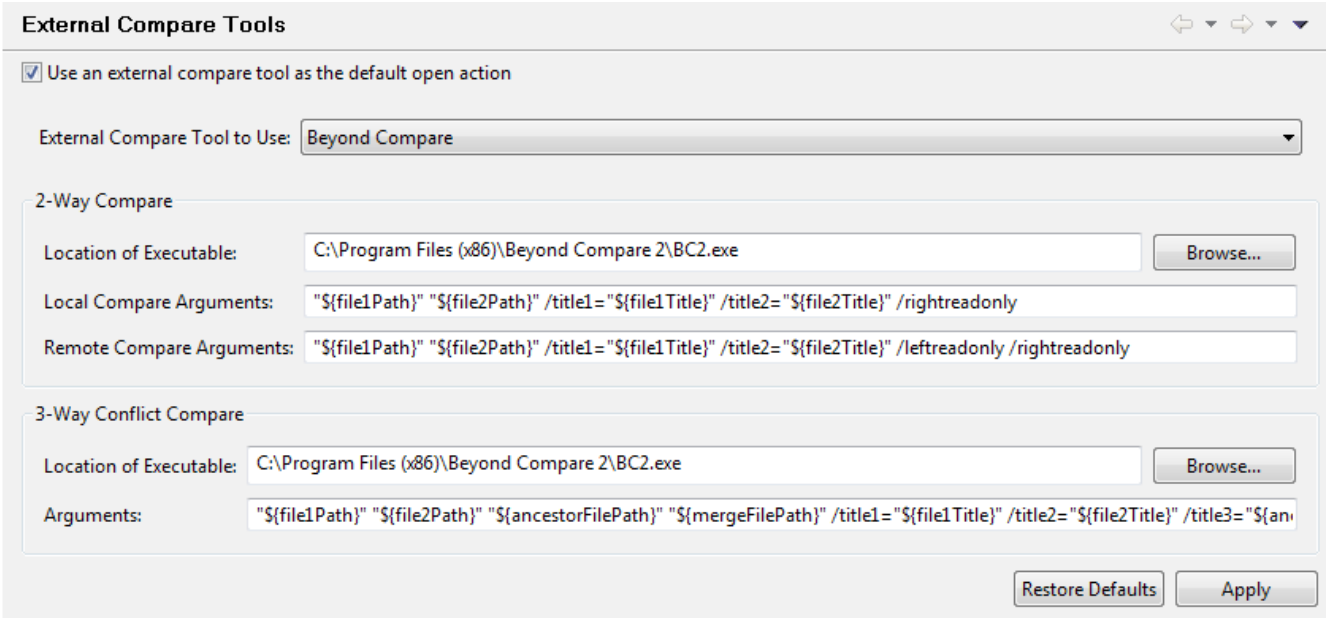
RMC Library reports can be stored in your workspace and versioned with the library that they were created from. This allows you to compare library reports using the version history, rather than recreating the whole library.

5.3 External Compare Tools

You can set Beyond Compare as the the default Compare tool in your RMC Eclipse environment, so that you can do Beyond Compare comparisons in RMC, rather than having to go to Windows®.

a. Go to Window > Preferences > Team > Jazz Source Control > External Compare Tool

b. Select Beyond Compare as the External Compare tool

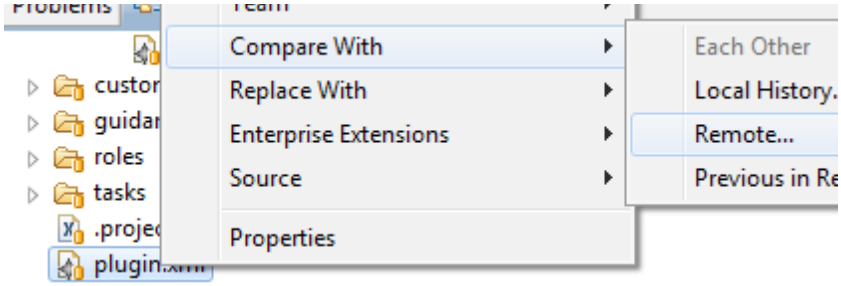


c. RMC will now use Beyond Compare to compare files.

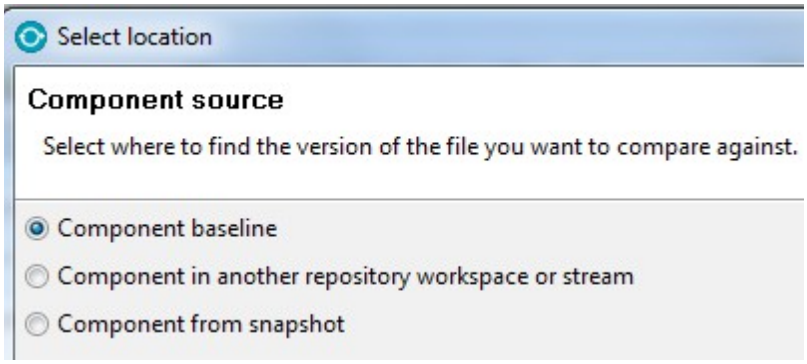
5.4 Comparing using RTC

5.4.1 Comparing Individual Files

1. Go to the *Navigator* view, select an element, and then right-click and select Compare With > Remote.



2. Next, select a component source. It is recommended that you select compare with a component baseline.



5.5 Capability pattern / Delivery process changes

Changes to capability patterns and delivery processes can be evaluated in a similar way. Each of the sections below describe a change, and how that change can be identified.

1. Add a task descriptor to a process

In the example below, the Configure_project task descriptor is added to the Start Project activity. This displays as a new element, plus related relationships.

C:\Users\IBM_ADMIN\RMC\Report\report2.csv

Element type	Element name	Presentation name	Name of relationship	Target presentation nam
Activity	Start Project	Start Project	breakdownElements	Configure project
TaskDescriptor	configure_project	Configure project	Self	
TaskDescriptor	configure_project	Configure project	superActivities	Start Project
TaskDescriptor	configure_project	Configure project	Task	Configure project

The first line shows that a breakdown element was added to the activity, the second line that a new task descriptor was created, and the others are new relationships created as part of the change

2. Delete an activity (with task descriptors)
For example, deleting the Start Project activity, with its two task descriptors Create project and Configure project.

C:\Users\IBM_ADMIN\RMC\Report\report3.csv | C:\Users\IBM_ADMIN\RMC\Report\report4.csv

Element type	Presentation name	Name of relationship	Target presentation name	Element type	Presentation name
ProcessPackage		Self			
Activity	Start Project	Self			
Activity	Start Project	superActivities	Project Initiation		
Activity	Start Project	breakdownElements	Create project		
Activity	Start Project	breakdownElements	Configure project		
TaskDescriptor	Create project	Self			
TaskDescriptor	Create project	superActivities	Start Project		
TaskDescriptor	Create project	Task	Create project		
TaskDescriptor	Configure project	Self			
TaskDescriptor	Configure project	superActivities	Start Project		
TaskDescriptor	Configure project	Task	Configure project		
CapabilityPattern	Project Initiatio	breakdownElements	Start Project		

We see the deleted activity as a "self" relationship in green on the left. Other relationships that are deleted as a side effect also show in green on the left. Also, every activity has a corresponding process package that is also deleted.

3. Create an activity
For example, add an activity Post Sprint.

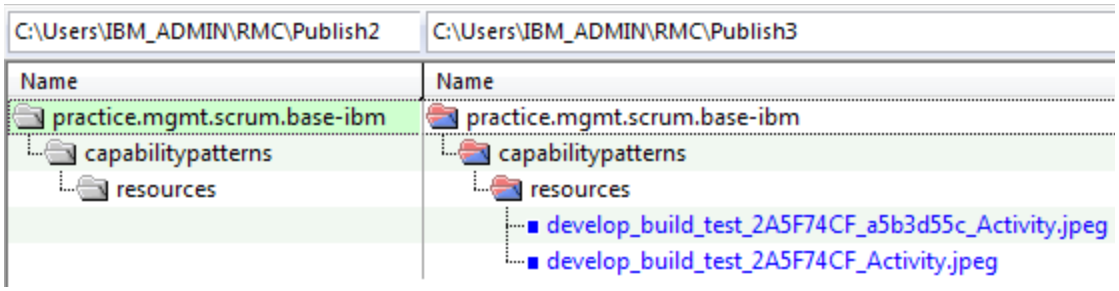
C:\Users\IBM_ADMIN\RMC\Report\report10.csv

Target presentation name	Element type	Presentation nai	Type of reference	Name of relationship
	ProcessPackage		Self	Self
	Activity	Post Sprint	Self	Self
	Activity	Post Sprint	Reference	superActivities
	CapabilityPattern	Sprint	Reference	breakdownElements

We see the added activity as a "self" relationship in green on the right. Other relationships that are added as a side effect also show in green on the right.
Also, every activity has a corresponding process package that is also added.

4. Create/Change an activity diagram
For example, creating the Develop, Build, Test activity.

Creating or changing activity diagrams are not visible in the reports. To detect that an activity diagram was created or changed, compare the published configuration files. Here is an example of what you might see after adding an activity diagram and comparing in Beyond Compare with the option Only Orphans.



Note that RMC generates two instances of each activity diagram.

5. Rename an activity
For example, Start Project activity to Begin Project.

The change appears as a change to the presentation name in all rows where the element is listed.

C:\Users\IBM_ADMIN\RMC\Report\report6.csv

Element type	Presentation name	Name of relationship
Activity	Start Project	Self
Activity	Start Project	superActivities
Activity	Start Project	breakdownElements
Activity	Start Project	breakdownElements

C:\Users\IBM_ADMIN\RMC\Report\report7.csv

Element type	Presentation name	Name of relationship
Activity	Begin Project	Self
Activity	Begin Project	superActivities
Activity	Begin Project	breakdownElements
Activity	Begin Project	breakdownElements

6. Rename a task used in a capability pattern
For example, conduct_sprint_review to sprint_review

C:\Users\IBM_ADMIN\RMC\Report\report.csv

Element type	Presentation name	Name of relationship
Discipline		tasks
Task		contributes
Task	Conduct Sprint Review	Self
Task	Conduct Sprint Review	mandatoryInput
Task	Conduct Sprint Review	output
Task	Conduct Sprint Review	output
Task		contributes
TaskDescriptor	Conduct Sprint Review	Task

C:\Users\IBM_ADMIN\RMC\Report\report2.csv

Element type	Presentation name	Name of relationship
Discipline		tasks
Task		contributes
Task	Sprint Review	Self
Task	Sprint Review	mandatoryInput
Task	Sprint Review	output
Task	Sprint Review	output
Task		contributes
TaskDescriptor	Conduct Sprint Review	Task

Notice that no change to any process elements appear. This is because the capability pattern is updated lazily – it synchronizes on publishing, or on update.

C:\Users\IBM_ADMIN\RMC\Report\report2.csv

Element type	Presentation name	Name of relationship	Target presentation name
TaskDescriptor	Conduct Sprint Review	Self	
TaskDescriptor	Conduct Sprint Review	superActivities	Sprint
TaskDescriptor	Conduct Sprint Review	Task	Sprint Review
TaskDescriptor	Conduct Sprint Review	mandatoryInput	Product increment
TaskDescriptor	Conduct Sprint Review	output	Product Backlog
TaskDescriptor	Conduct Sprint Review	output	Release Backlog
CapabilityPattern	Sprint	breakdownElement	Conduct Sprint Review

C:\Users\IBM_ADMIMC\Report\report3.c

Element type	Presentation nam	Name of relationship	Target presentation na
TaskDescriptor	Sprint Review	Self	
TaskDescriptor	Sprint Review	superActivities	Sprint
TaskDescriptor	Sprint Review	Task	Sprint Review
TaskDescriptor	Sprint Review	mandatoryInput	Product increment
TaskDescriptor	Sprint Review	output	Product Backlog
TaskDescriptor	Sprint Review	output	Release Backlog
CapabilityPatte	Sprint	breakdownElement	Sprint Review

7. Remove a relationship to a work product from a task used in a process
For example, remove product_increment as a work product input to task sprint_review

C:\Users\IBM_ADMIN\RMC\Report\report7.csv

Element type	Presentation name	Type of refer	Name of relationship	Target name	Element type	Presentation name
Task	Sprint Review	Reference	mandatoryInput	product_increment		

Notice that no change to any process elements appear. This is because the process is updated lazily – it synchronizes on publishing, or on update.

If we do an update to the capability pattern (can be anything), then we will see the effect of the task change:

C:\Users\IBM_ADMIN\RMC\Report\report4.csv

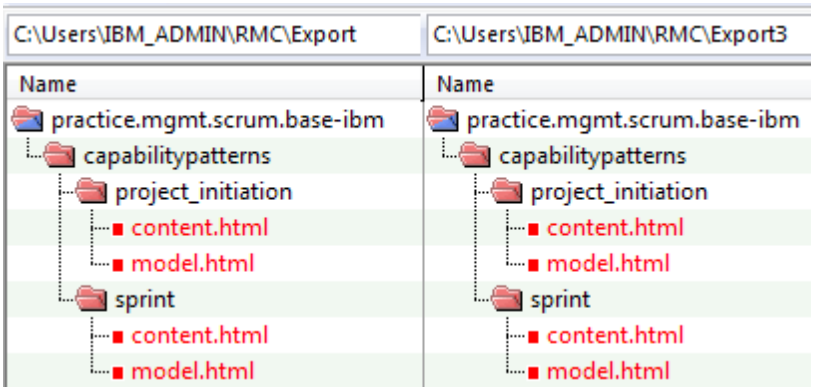
Element type	Presentation name	Type of reference	Name of relationship	Target presentation nar	Element type	Presentation name
TaskDescriptor	Sprint Review	Reference	mandatoryInput	Product increment		

By editing the capability pattern, we cause the process to be updated to reflect the task changes.

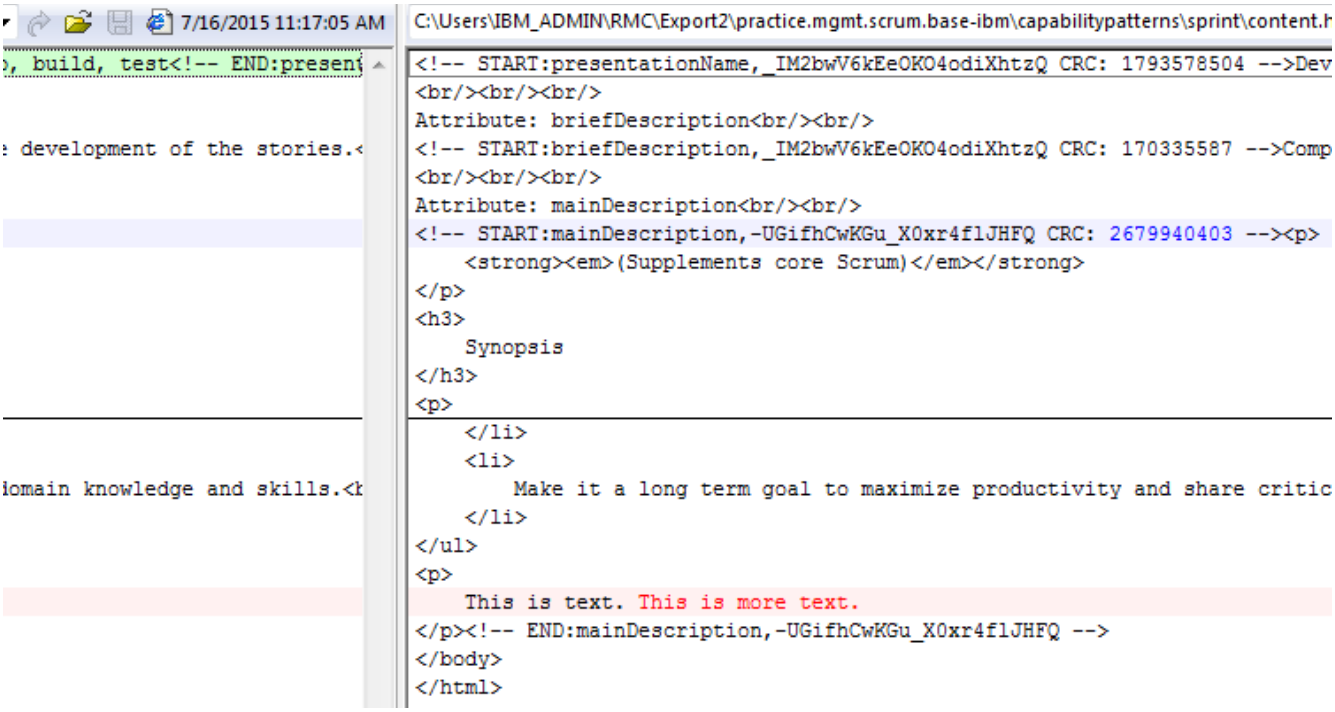
8. Add text to an activity

For example, adding text to the Develop, Build, Test activity.

Adding rich text to an activity is not visible in the reports. To detect that the addition of rich text, compare the exported HTML. For example, in the image below, we can see that changes have been made to the capability patterns.



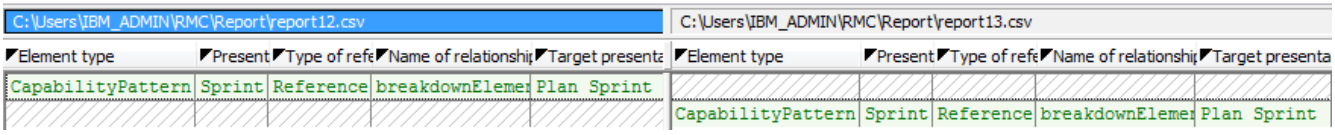
To see the actual changes, right-click on a red .html file and select “Compare in viewer”. The results below are a comparison of content.html from sprint.



In the image above, the text in red is what was added.

9. Reorder task descriptors

For example, moving up the plan_sprint task descriptor in the Sprint capability pattern.



C:\Users\IBM_ADMIN\RMC\Report\report12.csv

Element type	Present	Type of refe	Name of relationshi	Target presenta
CapabilityPattern	Sprint	Reference	breakdownElemer	Plan Sprint

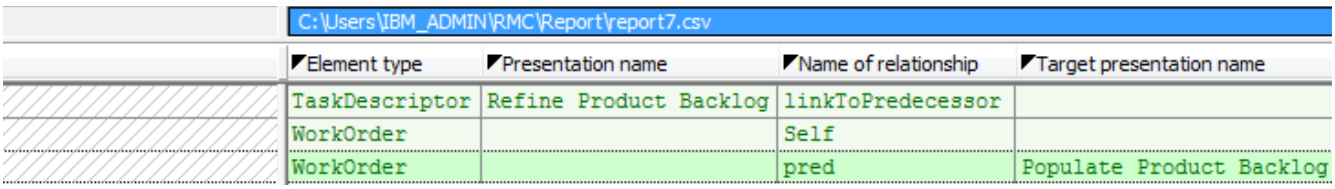
C:\Users\IBM_ADMIN\RMC\Report\report13.csv

Element type	Present	Type of refe	Name of relationshi	Target presenta
CapabilityPattern	Sprint	Reference	breakdownElemer	Plan Sprint

Comparing the two report files, there are no differences when sorted by GUID and then by Target GUID. So the lesson learned is that ordering changes aren't necessarily visible in the reports, but if there are no other changes other than ordering changes, they likely will show up as "add/remove" changes if you only sort by GUID and not by target GUID.

10. Add a predecessor relationship

For example, adding populate_product_backlog as a predecessor for task descriptor refine_product_backlog.



C:\Users\IBM_ADMIN\RMC\Report\report7.csv

Element type	Presentation name	Name of relationship	Target presentation name
TaskDescriptor	Refine Product Backlog	linkToPredecessor	
WorkOrder		Self	
WorkOrder		pred	Populate Product Backlog

The Work Order element type distinguishes the predecessor that was added.