Lab 2 Understand the Process Development Lifecycle

In this lab you'll become familiar with the cycle of describing and enacting a process. Some sections of this lab focus on what the Project Lead does, and some sections focus on what the Process Engineer does. You're stepping through the entire lifecycle so you'll switch back and forth between both roles.

Lab Scenario

Acme Corporation has a standard process for software development. The Nifty project has decided to tailor that process for their project.

In this lab you will:

- Create the Nifty Project Process, based on the ACME Process, by:
  - adding a role “Chief architect”
  - adding this role as a participant in the “Share vision” task.
  - Deploying the new process for use by the project.
- Update Rational team Concert to match by:
  - adding the new role to the Nifty project area
- Deliver the changed process assets into version control (using Jazz source control)
- Deploy the changes to the production environment

Lessons Learned:

- Tailor an existing process.
- Understand the cycle of process development.
- Harvest real-world process changes and make them available to other projects.
- Keep the process description and process definition synchronized.

Terminology Alert!

Unfortunately, the word “workspace” is overloaded.

- Eclipse uses the term “workspace” to refer to the location of the project you're working on (the local files).
- RTC expands this concept and maintains a workspace on the server that mirrors the local Eclipse workspace. You can define multiple workspaces on the server that can be swapped in and out of your Eclipse workspace.
- RMC uses the term “workspace library” to refer to an RMC library that's maintained in the Eclipse workspace. A workspace library maintains the library.xmi file locally instead of in SCM. Workspace libraries are more flexible than RMC static libraries and we use them in this workshop.

Be aware of the context when you see or hear the word “workspace” when using Eclipse, RTC, and RMC.

The word “plugin” is also overloaded. An RMC plugin is not related to an Eclipse plugin in any way.

- RMC uses the term “plugin” to refer to a component of an RMC library that contains method content.
- An Eclipse plugin is a software component used to extend the Eclipse product.

In a workspace library like you're using, the RMC plugin is also an Eclipse project. This makes it easier to manage RMC plugins in SCM.

2.1 Tailor an Existing Process for a Project

Use the projects you created during setup to perform minor additions to your process description.

Project Lead

The Project Lead or Team Lead is usually the person who makes the initial changes to a process for the team. The Project Lead only requires knowledge of the Team Process perspective in RMC to make these changes.

__1. Preparation:

__a. Log on with userid *root* and password *password.*

__b. Start the Jazz Team server

__i. Click **Computer > Start the Jazz Team Server**.

__c. Launch *Rational Method Composer*

__i. Double-click the **RMC-lab1ws** shortcut on the desktop.

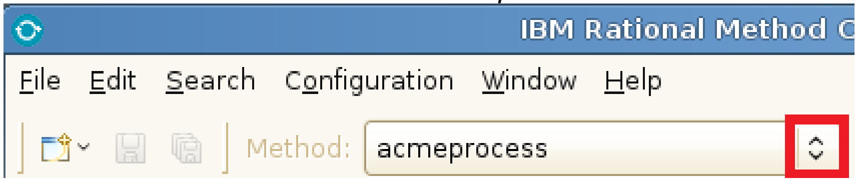
__d. Whenever prompted for a secure storage password, enter *password.*

__2. Create a process for the Nifty Project.

__a. Switch to the *Team Process* perspective.

__i. Click *Window > Open Perspective > Other > Team Process*

__b. Next to *Method* in the toolbar select *acmeprocess*

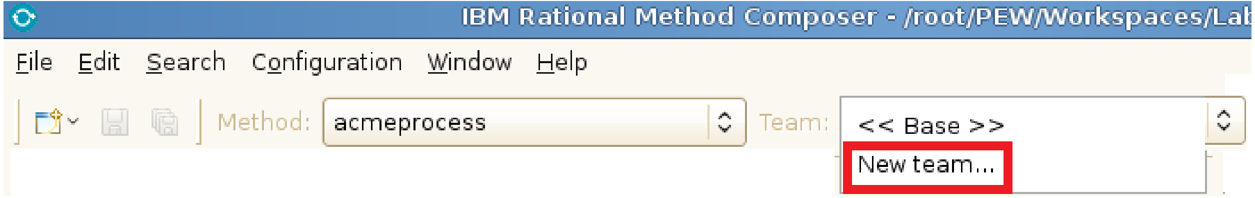


Method: acmeprocess

There's nothing in the Method selection dropdown!

Eclipse sometimes has issues with drop-down lists on the toolbar. If the drop-down to select the OpenUP method is blank, select **Window > Reset Perspective...** That should reload the perspective and populate the drop-down lists.

__c. Next to *Team*, select **New Team...**

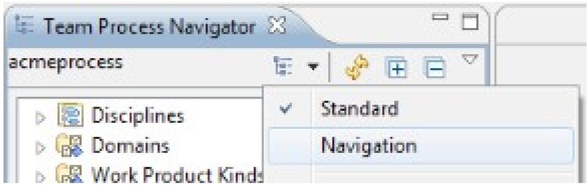

IBM Rational Method Composer - /root/PEW/Workspaces/Lab
File Edit Search Configuration Window Help
Method: acmeprocess
Team: << Base >>
New team...

__i. Name the new team *Nifty Team* and select **OK**.

Changing Elements

You can't change content in the <<Base>> process using the Team Process perspective. You can only change content for teams or projects that extend the <<Base>> process. If you need to change the base process, use the Authoring perspective.

__d. On the *Team Process Navigator*, select the option to display the treeview in **Navigation** mode



__3. Add new content that will be used by the Nifty Project.

__a. Expand Acme Process. Right click on *Welcome to Acme Process* and click **Edit**.

__b. Change the presentation name to “Welcome to Nifty”.

Names

The “name” becomes the file name where this information is stored. The “presentation name” is the name that is displayed when viewing the process.

Method Development Information

Inside RMC you are now editing a new method element. Scroll down to the Content Variability section and you'll see it uses Extends and Replaces. Your new element is a copy of the original element, but each non-empty section of the new element will replace the original section when the element is published.

So if you leave a section unchanged, the content from the original element will appear in that section when you publish your process. If you remove all content from a section, the existing content will also be displayed in the published process. And if you enter new information into a section, your new content will be displayed in that section of the published page.

This feature allows a non-method developer like a Team Lead to easily make changes to existing process content without worrying about the underlying mechanisms used to organize and re-use process content.

__c. Add a new role.

__i. Back in the treeview, right-click on *Roles* and select **New...**

__ii. For Element Type select *Role*, then select **OK**.

__iii. A new content element appears for the role. Enter the following for the role:

__a. Name: *chief_architect*

__b. Presentation name: *Chief Architect*

__c. Brief Description: *Maintains the architecture.*

__iv. Select **Save** on the toolbar.

__d. Add the role to the *Share Product Vision* task.

__i. Back in the treeview, under *Tasks>Requirements*, right-click on the *Share Product Vision* task and select **Edit**.

__ii. In the task editor, select the *Roles* tab.

__iii. Next to *Additional performers*, select **Add...**

__iv. Select the chief architect role, and select **OK**.

__v. **Save**, then switch to the *Preview* tab. You'll see the *Chief Architect* link has been entered under *Additional Performers*.

__4. Publish the changes to the server.

__a. Select **Configuration > Publish...**

__b. Make sure *Don't review publish options* is selected (this will assure that the process will publish with the options defined in the configuration). Select **Next**.

__c. For Directory type */root/PEW/Downloads/niftyprocess* (or browse and create any missing folders as needed).

__d. Make sure the option **Java EE web application** is selected with the following options, then select **Finish**.

__i. Web application name: *niftyprocess1*

__ii. Package format: WAR

> **WAR File**
>
> A WAR file that contains the published process will appear in the destination directory when publishing is complete. This is what we'll deploy to the application server, which is Tomcat in this workshop. This is the same Tomcat application server we are using for RTC.

__e. When publishing completes, close the window.

__f. Move the *niftyprocess1.war* file from *root/PEW/Downloads/niftyprocess* to /root/PEW/IBM/JazzTeamServer/server/tomcat/webapps. The WAR file will deploy.

__5. Verify the changes

__a. In your browser navigate to *https://clm.process.ws/niftyprocess1*.

__b. In the treeview, select **Tasks > Requirements > Share Product Vision**. Confirm that Chief Architect is listed under “Additional performers”.

2.2 Modify the Process Template

You have modified the process description and published that description for everyone to read. Now you need to modify a process template to match that description. This will help the team by automating and enforcing critical aspects of the process.

In this section you will create a new RTC process template based on the existing Agile ALM with Scrum template. You will modify the template so it matches the process description. Modifying the template enables you to share the changes you've made with teams across your organization.

Generally, the Process Engineer will update the process template and the Project Lead will configure projects that are created from the template.

__1. Create a Project Area to customize the process:

> **Why create a project rather than customize the process template?**
>
> You could create a process template for Nifty by directly customizing a template, rather than creating a project area. However, creating a project area is often easier because:
>
> - The out-of-the-box process templates in RTC come with externalized strings to support different languages. Dealing with the translation symbols is messy. It's easier to work with a template that contains strings in your own language rather than the translation symbols. (See Translatable Process Templates in Rational Team Concert if you need to create multi-language templates.)
> - After any customization you perform in a process you will want to test your changes. We are testing the changes in a temporary project before standardizing it as a process template for wide use in the enterprise.
> - There are more user interface options for customizing project areas, including features provided by RMC.

> Please note that the template and library used in this lab are not included in the product. For download instructions, refer to Lab 1 setup instructions. .

__a. In the browser, navigate to *https://clm.process.ws/ccm-pew/admin*

If prompted to log in, userid is *jim* and the password is *jim.*

__b. Click *Project Areas* > *Create* > *Project Area*

__c. Enter the following values:

__i. Name: Acme Nifty Process 1.0

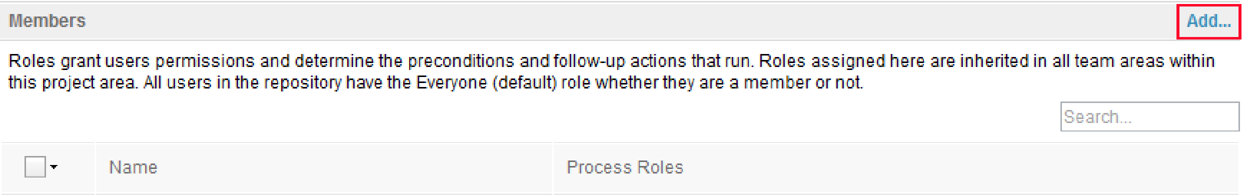
__ii. Summary: project area for customizing the process template for the Nifty project.

__d. Select the *Scrum/Agile ALM(local practices)* process template

__e. Select **Save**. The project area is created and the project area editor is opened.

__2. Add yourself (Jim) to the project

__a. In the *Overview* tab, find the **Members** area of the *Acme Process Development* window. Select **Add**.



Roles grant users permissions and determine the preconditions and follow-up actions that run. Roles assigned here are inherited in all team areas within this project area. All users in the repository have the Everyone (default) role whether they are a member or not.

__b. Enter *jim* in the search bar.

__c. *jim* appears as a matching user. Select the user jim, then the **arrow** button to add jim to the Selected Users. Then select **Next**.

__d. Add *Scrum Master* to the Selected Roles. Select **Finish**. Then select **Save.**

__3. Add a role from RMC to RTC.
(Return to RMC, which should still be in the *Team Process* perspective)

__a. Right click on the *Acme Process > Chief Architect* role.

__b. Click *Create Jazz Role*.

__c. If *Acme Nifty Process 1.0* is visible, click to select it. Otherwise right click on *jim@clm.process.ws* then click log in. Then click *Acme Nifty Process 1.0* .

__d. Under *context root* change *the /rmc to /niftyprocess1,* then select **Finish**.

Context root

The context root is just the URL of the RMC website published previously. Because we are hosting on the same Jazz server, we use an abbreviated URL. If hosted on a different server location, such as *http://acme.com/niftyprocess1*, then you would use that URL.

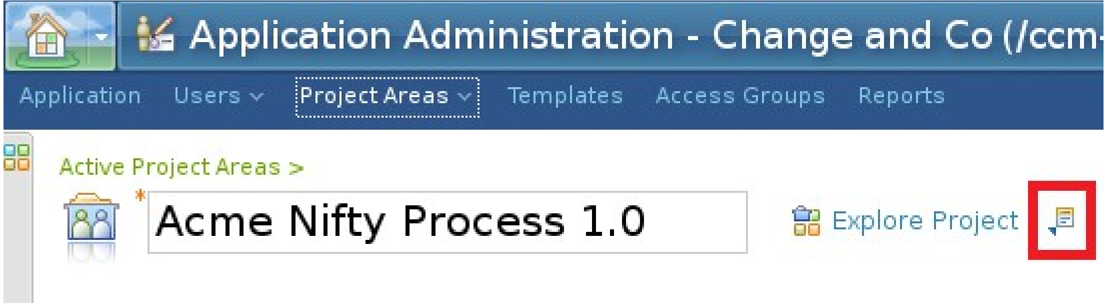
__4. View the role in RTC

__a. In the browser (still viewing the *Acme Nifty Process 1.0* project area), refresh the page, then click on Roles

__b. Confirm that *Chief architect* is in the list.

__5. Create the Nifty process template.
(So far we have been working in a test area. The new process template will be used later to create the real Nifty Project area in the software developer's CCM application.)

__a. Click the *Export Process Template* icon to the right of *Explore project*.



__b. Enter the following values, then select **OK**.

__i. Name: Nifty

__ii. ID: acme.nifty.processtemplate

__iii. Summary: Nifty process based on ACME standard Scrum/AgileALM.

__iv. Save the file to *root/PEW/Workspaces/Lab1/Acme Templates .*

Also copy the *niftyprocess1.war* file from */root/PEW/IBM/JazzTeamServer/server/tomcat/webapps* to *root/PEW/Workspaces/Lab1/Acme Templates.*

__6. Switch to RMC to check the template and war file into source control.

__a. Switch to the *Work items* perspective
Click *Window > Open Perspective > Work items*

__b. Open the *Pending Changes* view as follows:

__i. Click *Window > Show View > Other...*

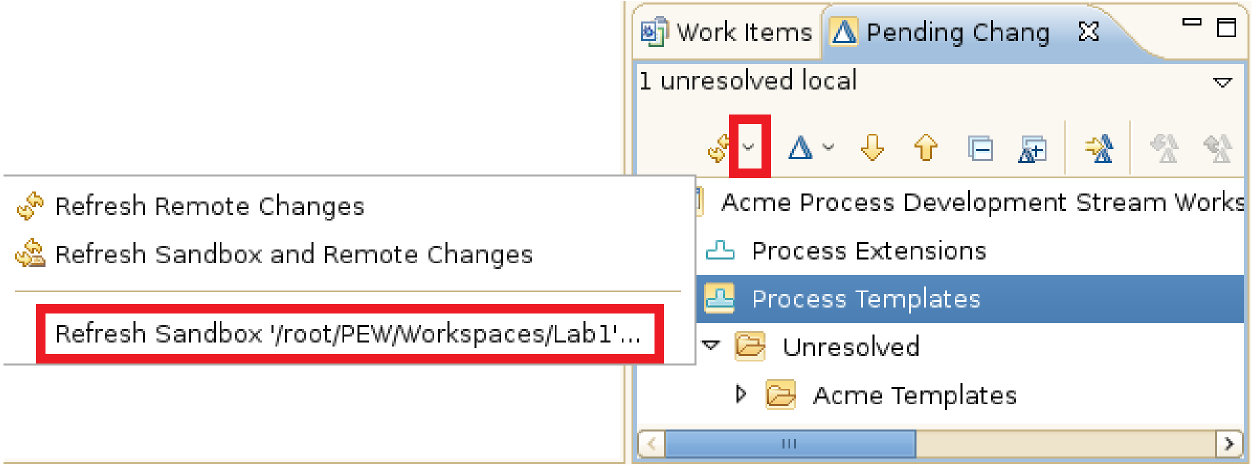
__ii. Click *Jazz Source Control > Pending Changes*

__iii. Click **OK**

Pending Changes view

In the setup for this Lab, we created a sandbox for configuration management (source control) of templates and other files. The *Pending Changes* view shows changes that we have made in the sandbox, but have not yet delivered for the rest of the team to see.

__c. Expand the Refresh button and click **Refresh Sandbox**



Select **OK** when prompted so you refresh the whole sandbox.

__d. Right-click the *Process Templates* > *Unresolved* folder in the *Pending Changes View* and click **Check-in and Deliver.** Type a comment (such as “Created V1”) and click **Finish**.

You have modified your process description by adding a role to both an RMC process website and to an RTC process template. You will now deploy these changes to your production environment.

2.3 Deploy and Customize a Project

We deploy the new process template to the SoftwareDev CCM app by exporting it from */ccm-pew*, then importing it to */ccm*.

One could also move the process website to another webserver, to make it officially available.

These steps help keep things consistent:

- Publish the website in its final location, but don't advertise it.
- Test the process template links to make sure they work.
- Make the process template available and advertise both the template and the process website.

In our example, we will treat *https://clm.process.ws/niftyprocess1* as the final release location, so no additional updates are required.

__1. Deploy the new process template:

__a. In the browser, navigate to *https://clm.process.ws/ccm/admin*

__b. Click *templates.*

__c. Click *import templates*

__d. Browse to *root/PEW/Workspaces/Lab1/Acme Templates* and select acme.nifty.processtemplate.zip*.* Then select OK.

__e. Select OK in the Import Process Template dialog to import the template.

__2. Create a Project Area to customize the process:

__*a. Click Project Areas > Create > Project Area*

__b. Enter the following values and select **Next**:

__i. Name: Nifty Application Project

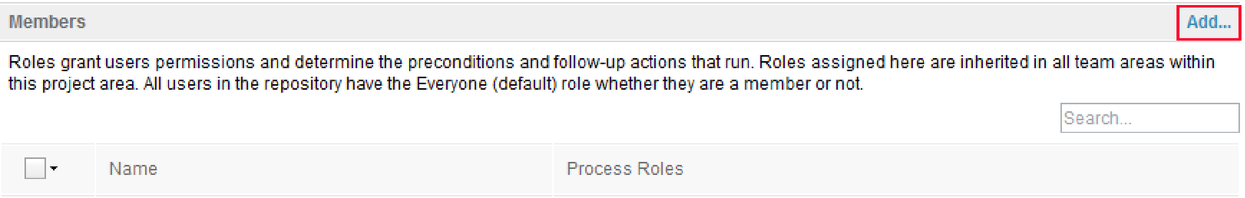
__ii. Summary: The project for the Nifty application.

__c. Select the *Nifty* process template

__d. Click **Save***.* The project area is created and the project area editor is opened.

__3. Add yourself (Jim) to the project

__a. In the *Overview* tab, in the **Members** area of the *Acme Process Development* window, Select **Add**.



Members Add...

Roles grant users permissions and determine the preconditions and follow-up actions that run. Roles assigned here are inherited in all team areas within this project area. All users in the repository have the Everyone (default) role whether they are a member or not.

Search...

□	Name	Process Roles

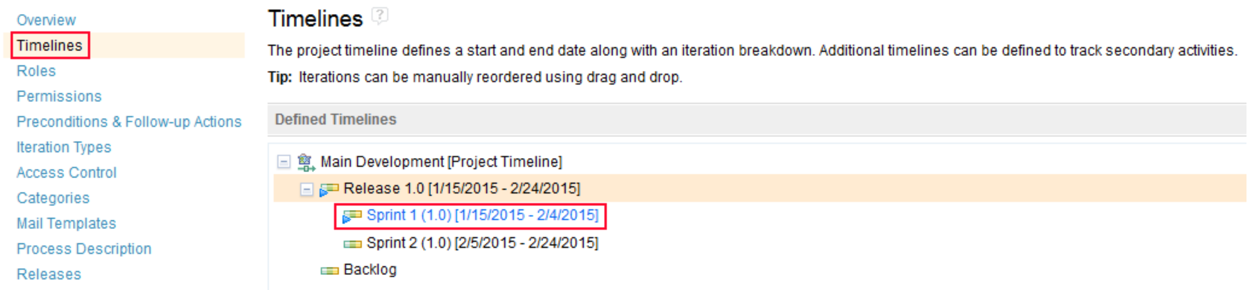
__b. Enter *jim* in the search bar.

__c. *jim* appears as a matching user. Click to select. Click the arrow (Add) to add to the selected users list. Click **Next**..

__d. Add *Scrum Master* and *Chief Architect* to the selected roles. Click **Finish**. Click **Save**.

__4. Tailor the new project. You will have a construction iteration and a short but important architecture iteration.

__a. Select *Sprint 1* in the *Timelines* tab and select **Edit Properities**. Create an architecture iteration with the following attributes, then select **OK**:



Overview
Timelines
Roles
Permissions
Preconditions & Follow-up Actions
Iteration Types
Access Control
Categories
Mail Templates
Process Description
Releases

Timelines

The project timeline defines a start and end date along with an iteration breakdown. Additional timelines can be defined to track secondary activities.

Tip: Iterations can be manually reordered using drag and drop.

Defined Timelines

- Main Development [Project Timeline]
  - Release 1.0 [1/15/2015 - 2/24/2015]
    - Sprint 1 (1.0) [1/15/2015 - 2/4/2015]
    - Sprint 2 (1.0) [2/5/2015 - 2/24/2015]
  - Backlog

__i. Iteration Type: <None>

__ii. Display Name: Architecture Iteration

__iii. Unselect *A release is scheduled for this iteration*.

__iv. Select **OK**.

__b. Select *Sprint 2* and select **Edit Properties**. Create a construction iteration with the following attributes, then select **OK**.

__i. Iteration Type: <None>

__ii. Display Name: Construction Iteration C1

__iii. Leave *A release is scheduled for this iteration* selected.

__iv. Select **OK**.

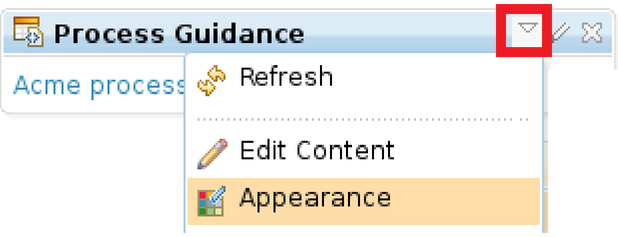
__c. *Save the project.*

__5. Update the project dashboard

__a. Click the *Explore project* link.

__b. Click *Add Widget*. Under *Select category*, click *General.* Under *HTML* click *Add Widget.* Click *save.* Close the widget selection dialog (x at the top right of the widget selection pane).
You have added a Dashboard widget named HTML that you can customize.

__c. Rename the widget. Click the dropdown (top right of widget). Click *Appearance*. Change the title to *Process Guidance*. Click *OK*.



__d. Click the pencil (edit) icon in the process guidance widget.

__e. Delete the text that is there, then click the Link icon to add the following link:

__i. URL: *https://clm.process.ws/niftyprocess1*

__ii. Description: *Nifty Project Process.*

__iii. Target*: New window*

__iv. Click *Set.*

__f. Click *OK*. Click *Save* (at top right).

__6. Click the *Nifty Project Process* link to confirm it works.

__a. The browser will open to your Nifty process website.

__b. Notice that the new role *Chief Architect* is in the list of roles.

2.4 Harvest the Process

Process changes are inevitable after the team has started working. Unanticipated tasks need to be performed, more efficient ways of doing things are discovered, new procedures are required from inside or outside the organization. These changes need to be communicated to and between team members.

You started this lab by making changes to the team process in RMC. In a real situation, you'd continue to make changes to your team process description as you modify the process in RTC. Now you'll harvest those changes so other teams can take advantage of new best practices that your team has discovered.

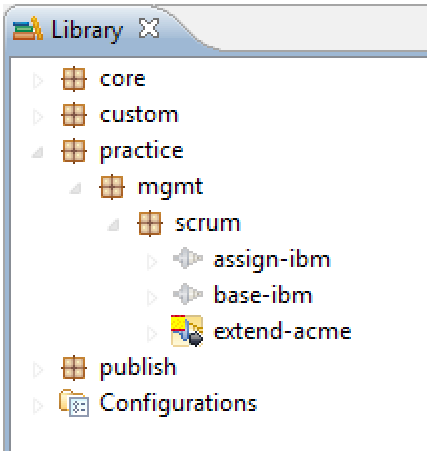
Harvesting a team's changes into the organization's library so it can be reused is generally something the Process Engineer does. The Process Engineer understands the underlying process architecture and can integrate the changes into the existing (or a new) process.

2.4.1 Move Team Changes to the Process Library

__1. Move the team's changes into the existing Scrum practice.

__a. Switch back to the RMC application and select the *Authoring* perspective.

__b. In the *Library* view expand *practice* > *mgmt* > *scrum*



Plug-in Naming Convention

The "." in the name of an RMC plugin defines the location where the plug-in appears in the tree. This Unified Method Framework (UMF) standard places your new plugin alongside other related plugins.

__c. Right-click on the *extend-acme* plug-in in the *Library* view and select **Show in Resource Navigator**. This displays the actual files that make up the plugin.

__d. Right-click the *extend-acme* plugin folder and select **Team>Share Project...**

__e. Select *Jazz Source Control*, then **Next**.

__f. Under *Acme Process Development*, select the *Process Extensions* component. Then select **Finish**.

> **Library Plug-in Locations**
>
> You have added your first plug-in to the Process Development component. Your library now seamlessly contains plug-ins from an RTC component as well as plug-ins that only live in the local file system. The plug-ins in the RTC component are the only ones that are controlled by RTC.

__2. Add the team's changes to the Acme plug-in.

__a. In the *Library* view, right click on *custom* > *acmeprocess* > *Nifty Team* > *Method Content* > *Content Packages* > *Content Elements* > *Roles* > *chief_architect*. Select *Copy*.

> **Team Process perspective customizations**
>
> The “custom” node is where customizations made in the Team Process perspective are kept.

__b. Right click on *practice* > *mgmt* > *scrum* > *extend-acme* > *Method Content* > *Content Packages* > *acme.* Click *paste.*

> **Library structure**
>
> In large libraries, you may want to place roles in a separate plug-in where they can be shared across practices and processes. We are using a single plug-in to keep things simple.

__c. In the *acme* content package, right-click, then click *new* > *task*

__d. Set the name to *share_product_vision.acme.*

__e. Scroll down and change *variability type* to *contributes*.

__f. For *base,* click *select* and then click *share_product_vision* __g. Select the *Roles* tab. Select **Add..** next to *Additional performers*.

__h. Select the *chief_architect* role.

__i. **Save** everything.

2.4.2 Publish the New Acme Process

> **A Shared Process**
> The process you published earlier is a team process you created by customizing the ACME process. You could publish that same process to other team areas, but it's not easily reusable by other teams because they can't do further customization on the published process.
>
> Now you are creating a process configuration that other teams can use to customize and publish their own process, based on the customizations you created for your own team. You harvested those customizations into the process library in the previous section.

__1. Validate and Publish the Process

__a. Preview the process.

__i. Switch to the *Browsing* perspective.

__ii. Select the *acmeprocess* configuration from the drop-down in the toolbar.

__iii. Select the *Navigation* view option on the *Configuration* view toolbar.

__iv. Open the *Roles* node and verity the *Chief Architect* role is there.

__v. In the treeview, select **Tasks > Share Product Vision**. Confirm that Chief Architect is listed under “Additional performers”.

__b. Publish the process

__i. Select **Configuration > Publish** from the main menu.

__ii. Select *Use publish options stored in the method configuration* and *Don't review publish options*, then select **Next**.

__iii. Select *acmeprocess*, then select **Next**.

__iv. In the *Directory* field, specify that you want to publish to *root/PEW/Downloads/acmeprocess*.

__v. Make sure the option *Java EE web application* is selected with the following options, then select **Finish** (it's OK to overwrite the destination folder).

__a. Include search capability: yes

__b. Servlet based RSS feed: yes

__c. Web application name: *acmeprocess2* __d. Package format: WAR

__vi. Make sure the *Publishing Report* shows no errors then close the window.

__vii. Move the *acmeprocess2.war* file from *root/PEW/Downloads/acmeprocess* to *root/PEW/IBM/JazzTeamServer/server/tomcat/webapps* . The WAR file will deploy.

__viii. Navigate to https://clm.process.ws/acmeprocess2 to view the published process.

Broken Links

We intentionally published a new website, acmeprocess2, rather than replace acmeprocess1.
We also did not remove niftyprocess1, since there are work items that still point to this process.

If you do decide to replace a process website with a newer one, be aware that any changes to the name or plug-in location of an element will change it's URL, which could result in broken links.
Changes made via variability elements, such as “contributes” elements do not change the URL, since it is the base element that defines the URL.

Follow-on steps

After harvesting a team's changes and publishing and testing the changes, you would typically also update any process template, do some testing, and add the new process templates and war file into source control, and deploy and advertise the new assets. Since we have already done similar steps, we do not repeat those in this lab.

Congratulations! You have completed one process development cycle. You customized an existing process configuration and published it for your team so they have a description of the process they're following. Then you customized an RTC process template so it matches the changes you made to the description, and deployed it to your team.

Finally, you harvested your process description changes into your process library so other teams can take advantage of your process improvements.