



Release Control

CA Agile Central (Rally) Quick Reference

Copyright © 2019 Micro Focus or one of its affiliates.

The only warranties for products and services of Micro Focus and its affiliates and licensors ("Micro Focus") are as may be set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Except as specifically indicated otherwise, this document contains confidential information and a valid license is required for possession, use or copying. If this work is provided to the U.S. Government, consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Part number: 1.0

Publication date: 2019-07-12

Table of Contents

Preparing the Plugins	5
Release Control Administration Overview	5
Plugins	7
Installing Plugins	7
Adding Plugins	8
Upgrading Plugins	8
Configurations	9
Adding Base Configurations	10
Adding Action and Item Configurations	11
Updating Plugin Configuration Details.....	13
Tags	13
Adding Plugin Configuration Tags	14
Updating Collections to Use Different Tags	15
Using Different Tags for Linking Objects to Tasks	15
Auxiliary Table Level Configuration Overrides	16
Making Fields Eligible for Override	19
Adding Configuration Overrides	19
Applying Configuration Overrides	21
Configuring Environment-Specific Field Overrides	21
Limiting Task Actions by Environment	26
Adding Global Task Action Rules	27
Filtering Rules for Specific Release Items	28
Custom Columns	29
Updating Custom Column Table Entries	29
Adding Custom Column Entries	30
Applying Custom Column Entries	31
General Custom Column Fields	31
Provider-Specific Custom and Extended Fields	32
Adding Custom Columns to Collections.....	36

Rally Plugin Configuration Overview	36
Rally Base Configuration Details	37
Rally Request Items Configuration Properties	39
Custom Column Fields for Rally	39
Using the Rally Plugin	39
Adding Rally Request Items	39
Plugins Glossary	41

Preparing the Plugins

The integration between each integrating product and Release Control is implemented through the corresponding plugin.

Documentation References

- Information on installing, configuring, and using this plugin is included in this document.
- For overview and getting started information on Release Control, see the *Release Control Getting Started Guide*.

For information on preparing plugins for use, see the following:

- [Release Control Administration Overview \[page 5\]](#)
- [Plugins \[page 7\]](#)
- [Configurations \[page 9\]](#)
- [Tags \[page 13\]](#)
- [Auxiliary Table Level Configuration Overrides \[page 16\]](#)
- [Limiting Task Actions by Environment \[page 26\]](#)

Release Control Administration Overview

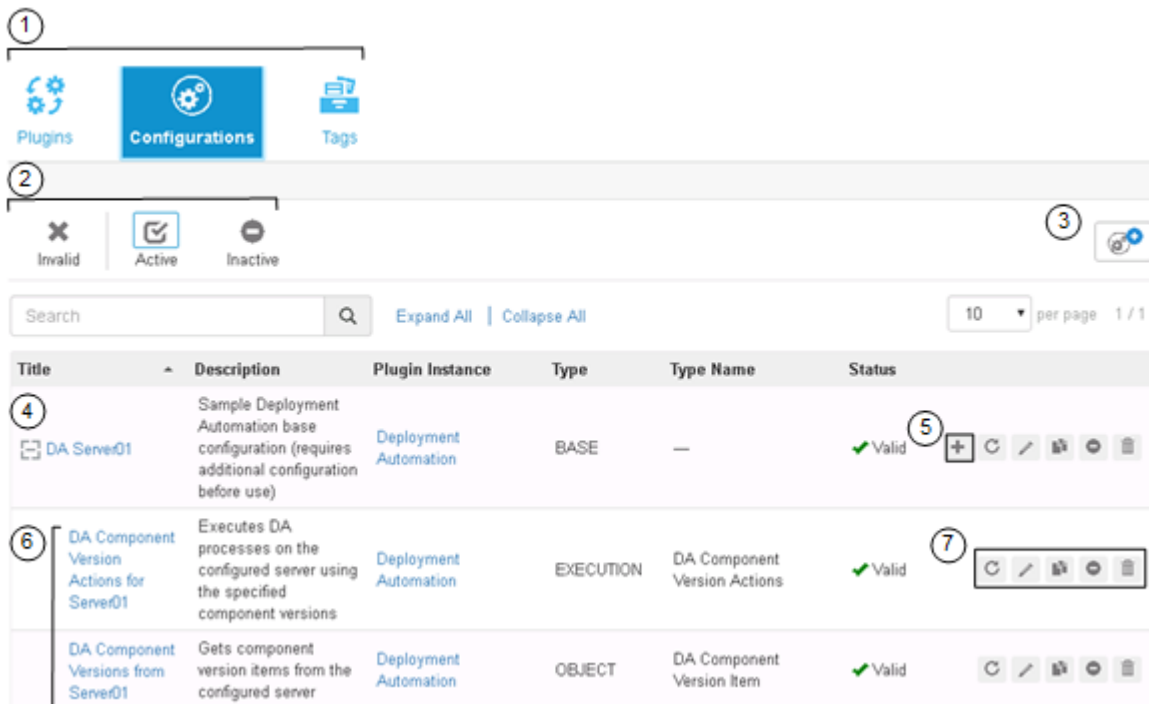
Use Release Control Administration to prepare your plugins for use.


If your plugin is ready to use, continue with

To access Release Control Administration, in your Web browser, enter the URL for Release Control Administration. For example:

```
http://serverName/workcenter/tmtrack.dll?StdPage&Template=rlc/admin
```

The following figure shows the Release Control Administration **Configurations** page with example plugin configurations that have been updated with valid connection information.



1. Release Control Administration includes the following menu options:
 - **Plugins:** Update and add plugin instances. You can add plugin instances from multiple versions of a plugin. See [Plugins \[page 7\]](#).
 - **Configurations:** Update and add plugin configurations as needed. See [Configurations \[page 9\]](#).
 - **Tags:** Optionally add custom tags if you need categories of objects other than the defaults provided. See [Tags \[page 13\]](#).
2. To filter the information shown in the content area, click the filter buttons to select or toggle filter criteria. The filter Search box searches various columns and further filters the list by showing only items that match the search string. In this example, the list is filtered by all active configurations.
3. Click the **Add** button to add the object in context of the page you are on, **Plugins**, **Configurations**, or **Tags**. In this example, you would click () to add a base configuration. See [Adding Base Configurations \[page 10\]](#).
4. The content area for each page lists any existing plugins, configurations, or tags.
5. In the **Configurations** page, click the **+** button beside a selected base to add configurations derived from that base. See [Adding Action and Item Configurations \[page 11\]](#).
6. View an overview of the configuration in the content area. Click the configuration title to view configuration details.
7. Select options in context, such as those shown for configurations. See [Updating Plugin Configuration Details \[page 13\]](#).

Related Topics

[Adding Plugins \[page 8\]](#)

[Upgrading Plugins \[page 8\]](#)

[Adding Base Configurations \[page 10\]](#)

[Adding Action and Item Configurations \[page 11\]](#)

[Adding Plugin Configuration Tags \[page 14\]](#)

Plugins

Plugins implement integrations between external products and Release Control. Plugins enable you to relate external requests to your release items, execute actions through external tools through Release Control tasks, and install deployment units from diverse sources onto target environments.

Default Plugins

When you apply the Release Control configuration in SBM Configurator, all default plugins are installed for you. You can then use Release Control Administration to add plugin instances and one or more unique configurations for each plugin.

Provider Types

Plugins implement object and execution provider types. In the default plugins, object provider type is used for requests and deployment units and execution provider type is used for tasks. Each plugin can implement any number of variations of provider types as needed, so only one plugin is needed per integrating product.



Note: In the examples, configurations using object provider types are called "item configurations" and those using execution provider types are called "action configurations".

For information on creating custom plugins, see the *Release Control SDK Reference*.

For details, see the following:

- [Installing Plugins \[page 7\]](#)
- [Adding Plugins \[page 8\]](#)
- [Upgrading Plugins \[page 8\]](#)

Installing Plugins

Default plugins are installed by the Release Control installer.

If you have created a custom plugin, are using a plugin version that is not installed by default, or are restoring a plugin, you will need to install it.

If the plugin (or version of the plugin) you want to use is already installed, continue with [Adding Plugins \[page 8\]](#).

For information on upgrading plugins, see the latest Plugins Release Notes on [Documentation Center](#).

To install a plugin:

1. Download the plugin jar files that you want to install from the [Support website](#), under the latest version of Release Control.

The latest plugin file name is as follows:

2. For Release Control version 6.2 or later only, copy the jar files to the Release Control plugins installation directory. For example:

C:\Program Files\Micro Focus\Release Control\Plugins



Important: Plugins version 3.0 and later are supported only for Release Control version 6.2 and later. For installation instructions for earlier plugin versions with earlier versions of Release Control, see the plugin Quick References for those versions.

Adding Plugins

Several of the default plugins have been added for you. If you don't see the default plugin you want to use, you can add it. If you have just installed a non-default or custom plugin, you will need to add it.

If the plugin (or version of the plugin) you want to use is already added, continue with [Adding Action and Item Configurations \[page 11\]](#).



Important: The Release Control Administration options can be performed only by users with Managed Administrator privileges.

CAUTION:



If you are using Internet Explorer as your web browser, ensure Compatibility View is not set for the Release Control site. Otherwise, you will receive the error "Your browser is not supported."

To add a plugin:

1. In your Web browser, enter the URL for Release Control Administration. For example:

`http://serverName/workcenter/tmtrack.dll?StdPage&Template=rlc/admin`



Tip: The link for your installation is in SBM Configurator on the **Release Control** page.

2. Select **Administration > Plugins**.
3. Click the **Add Plugin** icon.
4. In the **Available Plugins** list, select the



Note: If you have multiple versions of the same plugin installed, you can choose an the version to add. By default, the latest one is selected.

Upgrading Plugins

When editing a plugin instance, you can change the version of the plugin used by that plugin instance and therefore, the configurations based on that plugin instance.

When you edit a plugin instance that has multiple versions installed:

- The **Plugin Version** field appears. This appears only if you have multiple versions of the same plugin installed.
- When a plugin version later than the one in use is installed, the **Upgrade** action and icon appears when you view the plugin.
- When an upgrade or downgrade is performed, all configurations created based on the plugin instance are migrated to the new version.

CAUTION:



If the plugin to which you migrate has a different set of fields or supported configuration types, you must open existing configurations and deployment tasks and save them with the proper data.




Important: The Release Control Administration options can be performed only by users with Managed Administrator privileges.

1. In your Web browser, enter the URL for Release Control Administration. For example:

`http://serverName/workcenter/tmtrack.dll?StdPage&Template=rlc/admin`



Tip: The link for your installation is in SBM Configurator on the **Release Control** page.

2. Select **Administration > Plugins**.
3. Optionally click the **Upgrades** filter to show just the plugins that have upgrades available.
4. Beside the plugin that you want to upgrade, click the **Upgrade** option ().

Configurations

Base configurations and their related item and action configurations define the properties needed to connect to and use integrating products or process apps.

Base configurations are a set of shared properties that enable you to quickly create and update configurations of different types using that information.

- A base must be created for each set of configurations.
- Multiple versions of configurations can be added for the same base configuration.
- You can add multiple execution and object type configurations (called actions and item configurations in the examples) within one base configuration.

Item and action configurations are a specific set of properties for associating objects or executing actions. Release Managers and Release Engineers select these configurations when they are adding objects in Release Control, such as requests, deployment units, and deployment tasks.

- You can create as many item and action configurations as you need from each base.

- You can override base fields in item and action configurations if needed. If the base is updated and fields are overridden, the updates in the base are ignored.



Note: You can also override configuration settings through an auxiliary table. See [Auxiliary Table Level Configuration Overrides \[page 16\]](#).

Several example configurations for these are included in the Release Control installation.


For details, see the following:

- [Adding Base Configurations \[page 10\]](#)
- [Adding Action and Item Configurations \[page 11\]](#)
- [Updating Plugin Configuration Details \[page 13\]](#)

Adding Base Configurations

A base plugin configuration must be created for each set of configurations. It stores connection information and other shared properties and enables you to quickly create configurations of different types using that information.



Tip: You can clone existing base configurations to add new ones. Just select the base you want to clone and click the **Clone** option (). During the clone you can change settings as needed for the new base.

Any password or other secure fields are included, but will remain shown as asterisks in the clone configuration.



Important: The Release Control Administration options can be performed only by users with Managed Administrator privileges.


To add a base configuration:

1. In your Web browser, enter the URL for Release Control Administration. For example:

```
http://serverName/workcenter/tmtrack.dll?StdPage&Template=rlc/admin
```



Tip: The link for your installation is in SBM Configurator on the **Release Control** page.

2. Select **Administration > Configurations**.
3. Click the **Add Configuration** button ().
4. Give a unique title for the base that encompasses the scope of the configurations that will inherit its settings.



Tip: Consider including the server name if you have configurations for multiple servers, integrating product version, plugin version, or any other distinguishing information.

5. In the **Plugin Instance** field, select the
6. In the **Type Name** field, select **BASE**.

-
7. Fill out the remaining fields. .
 8. If you do not want to make the configuration visible for selection by end users immediately upon creation, click the **Make Configuration Active** button to turn it off. By default it is on.



Tip: When this option is off, it is white. You can also view the tooltip to see the current setting.

9. Click **Validate** to test the connection.

An invalid status will not prohibit use of the plugin if you later use the plugin with valid connection values, such as a valid user ID through Single Sign-On (SSO) or through a configuration override.



Note: If you use SSO, be aware of the following:

- The connection validation is attempted using the currently logged in SBM user. This may not be the user you want to use for the connection when you use the plugin.
- Since the user ID used during configuration must be an SBM managed administrator, if you want to validate the connection using the user ID that you will use for SSO, you must make that user ID a managed administrator.


Adding Action and Item Configurations

Plugin configurations provide connection information, product-specific details, and filters telling the plugin what data to obtain from the integrating product. You may use multiple configurations to point to different systems or projects, to use different versions of an integrating product or plugin, or to use different filters.

Before you can use a plugin, you must at minimum update it with your specific connection information for the configuration. Each of the plugin configurations in the default implementation have example values, but you'll need to update these according to your own implementation.

If you are using one of the plugins with example configurations, you can update the example configurations with your connection information and other details or you can add new configurations.



Tip: You can clone existing configurations to add new ones. Just select the configuration you want to clone and click the **Clone** option (). During the clone you can change settings and select a different existing base from the same plugin.

Any password or other secure fields are included, but will remain shown as asterisks in the clone configuration.



Important: The Release Control Administration options can be performed only by users with Managed Administrator privileges.

To add a plugin configuration:

1. In your Web browser, enter the URL for Release Control Administration. For example:

<http://serverName/workcenter/tmtrack.dll?StdPage&Template=rlc/admin>



Tip: The link for your installation is in SBM Configurator on the **Release Control** page.

2. Select **Administration > Configurations**.
3. Click the + beside the base that you want to use to create the configuration.
4. Give a unique title and description for the configuration.



Tip: The configuration title is the name that appears in the drop-down selection when adding objects such as requests, deployment units, and deployment tasks. Consider including the server name if you have configurations for multiple servers. You may also want to include integrating product version, plugin version, or other distinguishing information.

5. In the **Plugin Instance** field, select the
6. In the **Type Name** field, select the specific implementation of the provider type that you want to use, such as an item or actions implementation.
7. Ensure that the **Base Configuration** field has the correct base selected.



Note: When creating directly from an existing base using the + option, you cannot change the base configuration name. When creating a configuration through the **Add Configuration** button, you can select a base or select **New** to create a new one.

8. Fill out the remaining fields.
9. To enter different information in any grayed out fields, select **Override**.

The following figure shows a plugin configuration with overrides for the fields **Application Filter** and **Environment Filter**.

DA Application Request Process Link:	<input type="text" value="/da/#applicationProcessRequest/"/>	<input type="checkbox"/> Override
DA Post Deploy URL:	<input type="text" value="http://orbmpddv-sold02.serena.com:8"/>	<input type="checkbox"/> Override
DA Post Deploy Message Variables:	<input type="text" value="\${p.request.id}:\${p.finalStatus}"/>	<input type="checkbox"/> Override
User Name:	<input type="text" value="admin"/>	<input type="checkbox"/> Override
Password:	<input type="password" value="....."/> <input type="button" value="Clear"/>	<input type="checkbox"/> Override
Use Single Sign-On (SSO):	<input type="checkbox"/>	<input type="checkbox"/> Override
Application Filter:	<input type="text" value="Qlarius"/>	<input checked="" type="checkbox"/> Override
Environment Filter:	<input type="text" value="DEV,QA,PROD"/>	<input checked="" type="checkbox"/> Override

These overrides:

-
- Limit field value selections when adding objects in collections, such as requests, deployment units, and deployment tasks.
 - Are used by the system when it validates and executes tasks.

If you need to override at a broader level, such as for an entire release package or deployable release train, see [Auxiliary Table Level Configuration Overrides \[page 16\]](#).

Updating Plugin Configuration Details

Some settings can be changed after you have created a configuration.

To update a plugin configuration:

- Edit the base configuration to change shared information for all its subordinate configurations.
- Edit the configurations to change non-base settings.
- Override base settings at the configuration level.
 - You can change a field that is inherited from the base, shown in gray, only if there is an override option beside it. Select **Override** to make the field editable. If there is no override option, you must update it at the base level, or create a new base configuration with alternate values for that field.
 - If you override at the configuration level and then update the parent base configuration, the configuration retains the override rather than inheriting the new base configuration setting.



Note: You can also override fields at the item or project level to filter selections available in collections. See [Auxiliary Table Level Configuration Overrides \[page 16\]](#).

For field descriptions, see

Tags

Tags are unique names you can assign to be used with plugin configurations in addition to the plugin-defined Type Name.

- These are referenced in collection widgets to categorize the objects, such as requests, deployment units, and deployment tasks.
- Several default tags are assigned in the default implementation of Release Control, such as Request, Deployment Unit, and Deployment Task tag.
- Multiple configuration types may have the same tag and each configuration type may have more than one tag. For example, for the Dimensions CM plugin, Baseline Item and Request Item configurations both have the default tag Deployment Unit, and Request Item has two tags, Deployment Unit and Request.

For details, see the following:

- [Adding Plugin Configuration Tags \[page 14\]](#)

- [Updating Collections to Use Different Tags \[page 15\]](#)
- [Using Different Tags for Linking Objects to Tasks \[page 15\]](#)

Adding Plugin Configuration Tags

Tags are unique names you can assign to plugin configurations in addition to the system-assigned Type Name. These tags are used in collection widgets to show or hide the objects, such as requests and deployment units.

Tags are provided for the default plugin configurations and are pre-selected in the example settings. However, you may add any type of object needed for your custom implementations of Release Control and the plugins.

- Tags added at any level are added at the global level as well. All global tags are available for selection at other levels.
- Tags added in the **Default Tags** tab are the tags that are assigned when a new item or action configuration is added. Changing these does not impact existing configurations, it only impacts future ones.
- Tags defined for a item or action configuration are the current tags available, and can be edited at any time. At least one tag must be assigned to a configuration.

CAUTION:



Do not remove all of the default tags or configuration tags unless you are customizing the system. Object configurations must have tags assigned to them; otherwise, items cannot be linked to deployment tasks. If tags are missing, the item selection says "No configurations available".



Note: When you create tags or assign tags to a configuration, those tags are not used by existing items such as requests, deployment units, or deployment tasks. The tags will be used only for items added after the tags are created or assigned.



Important: The Release Control Administration options can be performed only by users with Managed Administrator privileges.

To add tags to a specific configuration:

1. In your Web browser, enter the URL for the Release Control Administration. For example:

`http://serverName/workcenter/tmtrack.dll?StdPage&Template=rlc/admin`



Tip: The link for your installation is in SBM Configurator on the **Release Control** page.

2. In **Administration > Configurations**, select a plugin configuration.



Note: If you add a tag at the configuration level, that tag is available to be selected by other configurations and is also added to the global **Tags** list.

3. Click the **Tags** tab.

-
4. Click **Edit**.
 5. To use existing tags, select one or more tags, use the arrows to move them to the **Selected Tags** box.
 6. To create a new tag, click **Create New** below the selection box and give the tag a name and description. This creates the tag and automatically adds it to the **Selected Tags** box.



Tip: You can view tags for all configurations in the global **Tags** tab. From there you can create tags, edit existing tag names and descriptions, and copy tags.

Updating Collections to Use Different Tags

Tags defined in Release Control Administration are used in collections to determine what plugin information to show.

If you are using the default implementation, you do not need to make any changes. However, if you've added tags to use for custom collections or to combine types of information shown in the default collections, you must modify the collections to use the added tags.

To update the collections to use different tags:

1. In SBM Composer, open the process app that you want to update, such as *RLC - Release Package* or *RLC - Release Train*.
2. Select **Visual Design** in the App Explorer.
3. Select the form to update, such as **RT Deployment State Form**.
4. In the form, select the tab for the collection you want to update.
5. Select the collection widget.
6. If you are updating an object collection, such as **Requests** or **Deployment Units**, in the Property Editor Parameters **Tags** field, update the list of tags. In execution collections, such as Deployment Tasks, update the **View Tags** field.
7. If you are changing the name of the collection, you should update the tab name to match.
8. Deploy the process app.

For details on updating and deploying process apps, see the SBM Composer documentation.

Using Different Tags for Linking Objects to Tasks

By default, the tags defined for objects that can be linked to deployment tasks are Deployment Unit and Request. If you want to link a different type of object to deployment tasks than these defaults, you must define the tag in the project and in the corresponding plugin object configuration.

For example, if you want to link requirements to a deployment task, you can add a Requirements tag.

To add and use default tags:

1. Add a new tag in the project.
 - a. Add a new tag in the project. In SBM Application Administrator, open the project in which you want to override the tag.
For example, navigate to All projects > Base Project > Release Packages Projects > Release Packages.
 - b. In the navigation pane, select **Default Fields**.
 - c. Search for and select the **Entity Tags** field.
 - d. In the **Attributes** section, add a comma and a new tag to the **Default Value** field. For example: ,REQUIREMENT
 - e. Save changes.
2. Add the tag to the object configuration.
 - a. In Release Control Administration, view the object configuration that you want to use. For example, the DA Component Versions configuration.
 - b. Open the **Tags** tab, click Edit and create a new tag that matches the one you added in the Entity Tags field, such as REQUIREMENT. If you are changing a configuration that already has tags you no longer want, remove the old tags. For example, for the DA Component Versions configuration, delete the Deployment Unit tag from the right column and leave only the REQUIREMENT tag.
 - c. Save changes.
3. Test as follows:
 - a. Create a new item in the project you modified, such as a release package.
 - b. Create a deployment task using the plugin configuration in which you added the new tag, such as the DA Component Version Actions configuration.
 - c. Click the link icon to add a component version, and the object configuration that you modified to use the new tag should be listed, such as the DA Component Versions configuration.



Important: Existing project items are not affected, just future items, so if you add a deployment task in an existing release package, for example, and you have only the new tag assigned to the object configuration, no configurations will be listed when you click the link icon.

Auxiliary Table Level Configuration Overrides

Using entries in the *RLC Configuration Overrides* table, you can pre-define sets of field values for specific purposes, such as for a project, an application, or a release.

Auxiliary table level configuration overrides are used to:

- Limit value selections when adding objects in collections, such as requests, deployment units, and deployment tasks.
- Define sets of overrides for all configurations (global) or limited by a configuration UUID or tag.
- Define environment-specific overrides to enable matching Release Control environments to provider environments.



Note: These overrides are not used when validating and executing tasks. Overrides available at the configuration level, which override base configuration settings, are used when validating and executing tasks. See [Adding Action and Item Configurations \[page 11\]](#).

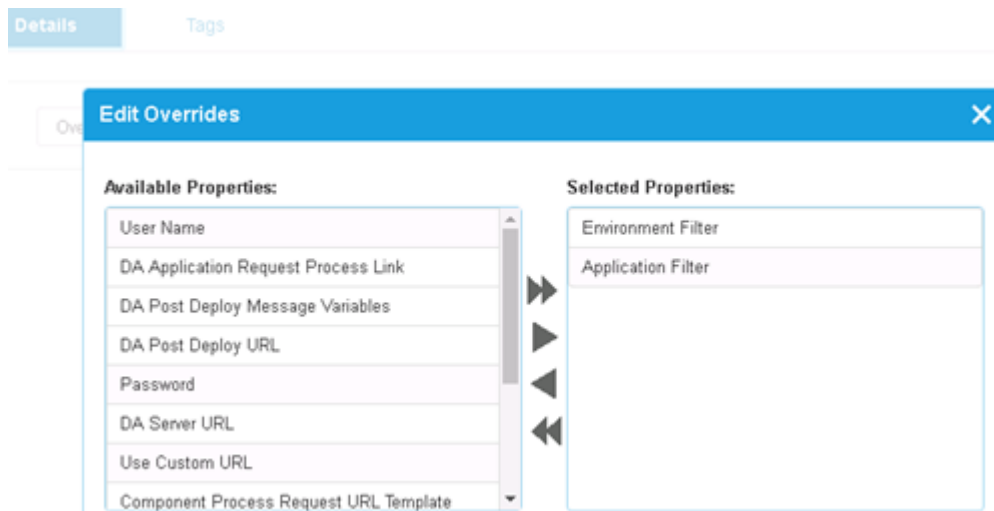
CAUTION:



It is not recommended to override connection information this way unless you are doing it only to restrict the selections for objects such as requests. If a user ID specified in an override has permission to make selections but the user ID specified in the configuration or its internal override does not have enough permissions to execute the deployment, the deployment may fail. The user ID in the configuration or its internal override must have the permissions necessary for validation and execution.

To override configuration settings:

1. Select the fields for which overriding is allowed. See [Making Fields Eligible for Override \[page 19\]](#).



2. Add entries to the *Configuration Overrides* table. See [Adding Configuration Overrides \[page 19\]](#).

You can use dynamic values in the *Configuration Overrides* table to access the exact values for fields set within an item, such as a release package or deployable release train. To dynamically override the field value, the field must be used on a form.

The following figure shows property value overrides for Environment Filter. For information on using environment-specific overrides, see [Configuring Environment-Specific Field Overrides \[page 21\]](#).

Override config field settings

First Item | Previous Item | Next Item | Last Item

[Update](#) [Delete](#)

Overview

Title: Override config field settings

Description: Example of overriding plugin configuration settings

Details

Property Name	Property Value / Env DB Field Name	Env Scope
environmentFilter	DEV,QA	Global

3. Apply the configuration override entry in release items. See [Applying Configuration Overrides \[page 21\]](#).

Update 000246: RP2 for DA

Options

Bypass Deployed State: Yes

Deployment Unit Widget View:

Default DU Custom Columns

Request Widget View:

Default Requests Custom Columns

Configuration Override:

Override config field settings

For more details, see the following:

- [Making Fields Eligible for Override \[page 19\]](#)
- [Adding Configuration Overrides \[page 19\]](#)
- [Applying Configuration Overrides \[page 21\]](#)

-
- [Configuring Environment-Specific Field Overrides \[page 21\]](#)

Making Fields Eligible for Override

To override plugin configuration field values in an item or project using a *Configuration Override* auxiliary table entry, the fields must be eligible for override.



Important: The Release Control Administration options can be performed only by users with Managed Administrator privileges.

To select overrides for a configuration:

1. In your Web browser, enter the URL for Release Control Administration. For example:

`http://serverName/workcenter/tmtrack.dll?StdPage&Template=rlc/admin`



Tip: The link for your installation is in SBM Configurator on the **Release Control** page.

2. In the **Configurations** tab, select a plugin configuration at the level you want to allow overrides, either base or configuration level.

If you allow configuration overrides on the base level, the child action or item configurations inherit these settings. You can also allow overrides on the child level even if they were already set in the base configuration. If you remove an override selection from the base in this case, it will still be allowed on the child level.

3. In the **Details** tab, select **Overrides** from the drop-down.
4. Click **Edit**.
5. Select the fields you want to make eligible for override and click **Save**.



Note: If you allow overrides at the configuration for an actions configuration but not at the item configuration level, there may be situations where your deployment task cannot be overridden completely due to the associations of items. In this case, it is better to override at the base level so that it encompasses the related actions and item configurations.

Adding Configuration Overrides

Configuration override auxiliary table entries can be set up to enable overriding plugin configuration field values at the item or project level.

Each configuration override entry can have multiple property value rows, so that a set of field overrides is contained in a single table entry. Once created, the set of configuration overrides can be selected in options for deployable release trains and release packages.

To create a configuration override entry:

1. In Work Center, click the user icon in the upper right of the page, and then select **Manage Data**.
2. In the **Table** field, select *Configuration Override*.

3. Click **Submit**.



Tip: If you want to see if an override already exists, select any filters needed to limit results and click **Search** to search for existing rules.

4. Enter a title and description.

5. Click **Add Row**.

6. Enter **Property Name**, which is the internal name of the field you want to override.




Tip: In the plugin configuration **Overrides** view, click the copy icon next to the field name to copy the field's internal name to the clipboard.

7. Enter **Property Value**, which is the value for the field that you want to use instead of the one defined in Release Control Administration.



Note:

- For selection values, use *true* for selected and *false* for deselected.
- The characters " and ' in a URL must be encoded correctly, see: https://www.w3schools.com/tags/ref_urlencode.asp
- To set up environment-specific configuration overrides, some process app customization is required. See [Configuring Environment-Specific Field Overrides \[page 21\]](#). After the customization is completed, you must:
 - a. Enter the configuration field internal name in **Property Name** and the environment database field you want to use to override it in **Property Value**. For example:
 - To limit selection of a ChangeMan ZMF environment to its matching Release Control environment, enter `promotionSiteFilter` for **Property Name** and `ZMF_SITE` for **Property Value**.
 - To limit selection of a Deployment Automation environment to its matching Release Control environment, enter `environmentFilter` for **Property Name** and `SRA_ENVIRONMENT_UUID` for **Property Value**.
 -  **Tip:** You can copy the field's internal filter name to the clipboard from the configuration **Overrides** page using the copy button next to the field.
 - b. Select the **Env** checkbox so that it will use the environment database field rather than the property values.

8. Select **Scope** as follows:

- **Global:** The override is applied to all same-named properties for all configurations.
- **By UUID:** The override is applied only to a specific configuration. For this, you must specify the UUID of the configuration.

-
- **By Tags:** The override is applied only to configurations with a specified tag. You must specify the internal name of the tag.



Tip: In Release Control Administration in a **Tags** page, click the copy icon beside the tag definition to copy the field internal name to the clipboard.

9. Enter **Scope Value**. For example, if **UUID** is selected for **Scope**, enter the UUID from the configuration to which you want this override to apply.



Tip: In Release Control Administration in the configuration view, click the copy icon beside the UUID field to copy the UUID to the clipboard. The UUID is not shown on the configuration edit page.

10. Click **Submit**.

Applying Configuration Overrides

After you have defined the configuration overrides, including environment-specific field overrides, you can apply them at the deployable release train, release package, or project levels.

Apply an override entry in one of the following ways:

- **Release Package Level:** In a release package, select an entry in the **Options** section **Configuration Override** field. See *Release Control Getting Started Guide* or online Help, "Creating Release Packages".
- **Deployable Release Train Level:** In a deployable release train, select an entry in the **Options** section **Configuration Override** field. See *Release Control Getting Started Guide* or online Help, "Creating Deployable Release Trains".
- **Project Level:** In SBM Application Administrator, you can select the override entry for the project and set the **Configuration Override** field to read only so that end users cannot change it. This way the overrides are set automatically for each new release package and deployable release train in that project.

In SBM Composer, you can hide the **Configuration Override** field in the process app if you do not want end users to see it.

Configuring Environment-Specific Field Overrides

You can override fields per environment. This is an extension of the configuration overrides feature, and requires some additional setup.

These overrides can be used to ensure that logical environments in Release Control match the intended target deployment environments, such as a Release Control development environment matching the development environment in Deployment Automation, QA matching the QA environment, and so on. After the override is set up and applied, only the matching environments can be selected in deployment tasks.

The following procedure includes an example for ChangeMan ZMF `ZMF_SITE`. For additional examples, see Knowledgebase items [S142319](#) (ChangeMan ZMF) and [S142326](#) (Deployment Automation).

To prepare to use a configuration override for a specific environment:

1. If it does not already exist, add the field that identifies the environment for your plugin, such as Deployment Automation `SRA_ENVIRONMENT_UUID` or ChangeMan ZMF `ZMF_SITE`. The field you use to hold the environment information may be as simple as a text field or you can add selection fields and add auxiliary tables to hold the environment values. This gives Release Control a way to hold the information about matching sites. To do this, in SBM Composer open the *RLC - Environment* process app and in **Data Design**, update the *Environments* table. The following figure shows `ZMF_SITE` defined as a text field.

Environments (primary Table)
The Environments table.

Field name	Type	Database field name	Section	Dependent fields
Deployment Approval Re...	Binary/Trinary	RLM_DEPLOYMENT_APPRO...	Standard	N/A
Group Member Requires ...	Binary/Trinary	RLM_GROUP_MEMBER_APP...	Standard	N/A
Owner	User	OWNER	Manager	N/A
Secondary Owner	Multi-User	SECONDARYOWNER	Manager	N/A
Last Modifier	User	LASTMODIFIER	System	N/A
Last State Changer	User	LASTSTATECHANGER	System	N/A
ZMF_SITE	Text	ZMF_SITE	Standard	N/A
SRA Application UUID	Text	SRA_APPLICATION_UUID	Standard	N/A
Temp Application	Text	RLM_TEMP_APPLICATION	Deleted	N/A
SRA Environment	Single Relational	SRA_ENVIRONMENT	Deleted	N/A
Approval Required	Binary/Trinary	RLM_APPROVAL_REQUIRED	Deleted	N/A



Tip: You must add any fields that do not already exist in the Environments table. If you use an existing field that is in Deleted status, you must restore it and move it to a valid section.

2. In SBM Composer, update the *RLC - Environment* process app's Visual Design Environment **Master** and **Update** forms to add the field you just added in the database to the form. The **ZMF_SITE** field has been added to the Update Form in the following figure.

▼ Overview

Title:

Title

Description:

Description

Accessibility: Deployment Approval Required:

Environment Group: **ZMF_SITE: ZMF_SITE**

Select from the following environments for this group. Note that the environments must have the s

Filter environments:

After you deploy the updated process app, this gives you a way to add the matching name of the environment while creating and updating an environment item. If the new field is a selection field to be populated by an auxiliary table, the names can be selected. If it is a simple text field, you type them in the field. See Step 6.

3. In SBM Composer, save and check in as needed and deploy the updated *RLC - Environment* process app.
4. In the plugin configuration, allow override for the environment fields.

ChangeMan ZMF Server03

Details | Configurations | Tags

Overrides

Not allowed for override:	Allowed for override:
Promotion Environment Filter ❌	Promotion Site Filter ✅
Use Single Sign-On (SSO) ❌	

See [Making Fields Eligible for Override \[page 19\]](#).

5. Add the configuration override auxiliary table entry.

Update 000230: RP1 for ZMF

Bypass Deployed State:
Yes ▾

Deployment Unit Widget View:
Enter value to find here
Default DU Custom Columns ▾

Deployment Unit Widget Add:
Enter value to find here
Default DU Add Custom Columns ▾

Request Widget View:
Enter value to find here
Default Requests Custom Columns ▾

Request Widget Add:
Enter value to find here
Default Request Add Custom Columns ▾

Configuration Override:
Enter value to find here
Override fields by environment ▾

Task Action Rules:
Enter value to find here

See [Applying Configuration Overrides](#) [page 21].

8. To test, create a deployment task using the plugin configuration for which you have added the override.

Your task environments should automatically be populated with the environment names that you entered when you created or updated the environment items.

Edit Task for ChangeMan ZMF Change Package Actions for Server03 Configuration

***Promotion Site:** UATSITE ▼

***Promotion Environment:** UATSITE/UATAREA/20 ▼

Schedule Options: (None) ▼ **i**

Environment: PATSITE

***Promotion Site:** PATSITE ▼

***Promotion Environment:** PATSITE/PATAREA/30 ▼

Schedule Options: (None) ▼ **i**

Environment: BKUPSITE

***Promotion Site:** BKUPSITE ▼ **i**

***Promotion Environment:** -- Select -- BKUPSITE **i**

Schedule Options: (None) ▼ **i**

See "Adding Deployment Tasks" in the *Release Control Getting Started Guide* or online Help.



Note: Actions can also be allowed or disallowed (filtered) in plugin action configurations. If you want to prohibit an action for all environments, you should use the configuration action filter to do so. See [Adding Global Task Action Rules \[page 27\]](#).

Limiting Task Actions by Environment

Task action rules set global limits on the available deployment and failure task actions for selected environments and environment groups.

One example where action rules for environments are important is when you execute deployment tasks that use the ChangeMan ZMF plugin. The final approval of change packages should be limited to only production environments, because upon this approval, the installation is immediately started. Therefore, the Approve (PROD) action should be allowed only on production environments.

Another example is when you execute deployment tasks that use the Deployment Automation plugin. You may want to disallow running component processes on production environments.

All task action rules apply to all future release items. The main purpose for this feature is to set rules that should apply across the system. However, if necessary, administrative users can select just specific rules for a release item if they don't want all of the rules to apply.

Viewing the Rules for a Release Item

To view the task action rules that are set for a release package or deployable release train, view the release package and in the **History** tab, expand the **System** section. The list of rules appears in the **Task Action Rules** field. If no rules are listed, all rules that are defined apply.

Rules and Task Templates

Task action rules are not enforced in task templates at the time their tasks are created. Instead the rules are enforced for tasks created from task templates, based on the rules set for the release package or deployable release train. Even if the rule is not visually apparent at the time the deployment or failure task is created from the task template by grayed out environments, it will be enforced at the time of deployment.



Important: If rules are added after a release item has already been created, the rule doesn't apply, so you should set the rules you want before beginning to create release items.

For more details, see the following:

- [Adding Global Task Action Rules \[page 27\]](#)
- [Filtering Rules for Specific Release Items \[page 28\]](#)

Adding Global Task Action Rules

When you add task action rules, they are applied globally by default.

To add a task action rule:

1. In Work Center, click the user icon in the upper right of the page, and then select **Manage Data**.
2. In the **Table** field, select **RLC Task Action Rules**.
3. Click **Submit**.



Tip: If you want to see if a particular rule already exists, select any filters needed to limit results and click **Search** to search for existing rules.

4. Enter a title and description. If you leave the title blank, it is automatically generated based on the plugin configuration and action name.
5. In **Action Name**, enter the fully-qualified internal action name for the action that you want to limit by environment. This is the fully-qualified name with `<database name>.<internal action name>`.

For example:

- For the Deployment Automation Run Component Process action, enter: `rlc-provider-sda.runComponentProcess`

- For the Deployment Automation Run Application Process action, enter: `rlc-provider-sda.runApplicationProcess`

For a list of internal action names by plugin, see Knowledgebase item [S142281](#).



Tip: To copy the internal action name from a deployment task, edit the task and click the copy icon next to the action field you want to copy. This copies the fully-qualified internal name to your clipboard.



Note: Only one rule can be set for each unique action name.

6. Select the **Rule Type** you want to apply to a set of environment as follows:
 - **Allow:** Select this if most environments should not be allowed this action.
 - **Disallow:** Select this if most of the environments should be allowed this action.

If you select **Allow** for a set of environments, any others are disallowed. If you select **Disallow** for a set of environments, any others are allowed.

7. Find and select the environments and environment groups for which you want to allow or disallow the specified action.



Note: After you have added the rule, any future environments or environment groups that are created have the action either implicitly allowed or disallowed when this rule is applied, depending on which of these you selected. If you set Allow, all new ones will be disallowed; if you set Disallow, all new ones will be allowed. You must update the rule if you want to explicitly allow or disallow the action in additional environments or environment groups.

8. Click **Submit**.

After you have added the rules, they are applied to all future release items by default. If you want to override them at the release item level, you must select just the ones you want at that level.



Note: Actions can also be allowed or disallowed (filtered) in plugin action configurations. If you want to prohibit an action for all environments, you should use the configuration action filter to do so.

Filtering Rules for Specific Release Items

After you have defined the task rules in the auxiliary table, they are automatically applied to all future release items. If you do not want all rules applied for a specific release item, you can select just the ones you want at that level, including release package, deployable release train, or project level. If no task actions are selected at the release item level, all apply.

Filter task action rules by selecting just the ones you want as follows:

- **Release Package Level:** In a release package, find and select one or more entries in the **Options** section **Task Action Rules** field. See *Release Control Getting Started Guide* or online Help, "Creating Release Packages".

-
- **Deployable Release Train Level:** In a deployable release train, find and select one or more entries in the **Options** section **Task Action Rules** field. See *Release Control Getting Started Guide* or online Help, "Creating Deployable Release Trains".
 - **Project Level:** In SBM Application Administrator, you can select the task action rules entries for the project and set them the **Task Action Rules** field to read only so that end users cannot change them. This way the rules are set automatically for each new release package and deployable release train in that project.

In SBM Composer, you can hide the **Task Action Rules** field in the process app if you do not want end users to see it.



Important: If you select a subset of the global action rules, any existing tasks with other rules already applied are not impacted. You must apply the rule before setting up the tasks for the rule to take affect.

Custom Columns

Custom columns enable you to specify the fields from integrating products that you want to reference in Release Control. These are typically used to display information in collection lists, such as requests and deployment units.

Default collection columns and their related custom column table entries are pre-configured in the default installation, so you do not need to do any extra configuration to have a working implementation. However, you can customize them for your implementation.

For information on configuring custom column fields, see the following:

- [Updating Custom Column Table Entries \[page 29\]](#)
- [Adding Custom Column Entries \[page 30\]](#)
- [Applying Custom Column Entries \[page 31\]](#)
- [General Custom Column Fields \[page 31\]](#)
- [Provider-Specific Custom and Extended Fields \[page 32\]](#)
- [Adding Custom Columns to Collections \[page 36\]](#)

Updating Custom Column Table Entries

To update custom column auxiliary table entries:

1. In SBM Application Administrator, click **Auxiliary Data**.
2. Select the *RLC Custom Columns* auxiliary table.
3. Select a row and click **Details**.



Note: Default rows include View and Add entries for requests and deployment units. The View entries are used for lists showing existing requests and deployment units, and the Add entries are used in the selection lists shown when adding requests and deployment units.

4. Click **Update**.
5. Update existing or add new field information in **Display Names** and matching **Internal Names** using the built-in fields.
 - Use the general fields for columns that are not provider-specific. See [General Custom Column Fields \[page 31\]](#).
 - Use the provided aliases if you are using more than one provider for deployment units or requests and they use similar column data, such as create date. See [Aliases for Similar Provider-Specific Fields \[page 33\]](#).
 - Use extended or custom fields in the plugin configurations to specify any other fields you may want to reference. See [Provider-Specific Custom and Extended Fields \[page 32\]](#).
 - Specify a width for a column by appending a colon and the width to the Internal name. For example, description:300.



Tip: Use the Help on the right side of the update form for more information.

Adding Custom Column Entries

To add custom column entries:

1. In SBM Application Administrator, click **Auxiliary Data**.
2. Select the *RLC Custom Columns* auxiliary table.
3. Click **New**.
4. Enter a title and description.
5. Enter **Display Names** and matching **Internal Names** using the built-in fields.
 - Use the general fields for columns that are not provider-specific. See [General Custom Column Fields \[page 31\]](#).
 - Use the provided aliases if you are using more than one provider for deployment units or requests and they use similar column data, such as create date. See [Aliases for Similar Provider-Specific Fields \[page 33\]](#).
 - Use extended or custom fields in the plugin configurations to specify any other fields you may want to reference. See [Provider-Specific Custom and Extended Fields \[page 32\]](#).
 - Specify a width for a column by appending a colon and the width to the Internal name. For example, description:300.



Tip: Use the Help on the right side of the update form for more information.

Applying Custom Column Entries

To use custom column entries for release packages or deployable release trains, you must apply the entry.

An administrator should set a default value for this field in the Release Packages project, and typically, non-administrative users should not be able to change the selection. However, if you need to select the custom column entries to use for a particular release package or deployable release train, use the following procedure.



Note: The *RLC Custom Column* auxiliary table implementation is provided so that you don't have to manually update columns on the forms. However, if you don't want to use this implementation and instead choose to manually enter custom columns as parameters in the widget, you must update all occurrences of the widget on state forms and transition forms.

To select custom column entries:

1. In Work Center in Release Control, create or update a release package or other release item that includes the Deployment Unit or Request Collection widget.
2. In the **Options** section of the submit or edit form you are filling out, find and select the custom column entry you want to use for each of the following fields:
 - Deployment Unit Widget View
 - Deployment Unit Widget Add
 - Request Widget View
 - Request Widget Add

The default selections are shown in the following figure:

Options	
Bypass Deployed State:	Bypass Deployment Complete State:
Yes	No
Deployment Unit Widget View:	Deployment Unit Widget Add:
Enter value to find here	Enter value to find here
Default DU Custom Columns	Default DU Custom Columns
Request Widget View:	Request Widget Add:
Enter value to find here	Enter value to find here
Default Requests Custom Columns	Default Requests Custom Columns

General Custom Column Fields

Following are the general case-sensitive custom column fields, which are common across providers:

```
id
artifact_id
request_id
description
name
provider_instance_UUID
provider_instance_name
syncattempteddatetimestring
syncsucceededatetimestring
syncmessage
syncstatus
title
type
url
```

General Fields for View Widgets Only

Some of the general fields can be shown only in the request and deployment unit *view* widgets, because they are populated on the Release Control side rather than on the provider side. If you use these fields in the request or deployment unit *add* widget, you should create a separate *RLC Custom Column* table entry for the add widget rather than sharing the column entry for the view and add widgets; otherwise, these fields will be empty. These case-sensitive fields are as follows:

```
artifact_id           request_id
provider_instance_UUID  provider_instance_name
syncattempteddatetimestring  syncsucceededatetimestring
syncmessage           syncstatus
```

Synchronization Fields for Reload Data Transitions

Synchronization data appears in the columns for a request or deployment unit widget only if **Reload Request Data** or **Reload Deployment Unit Data** transitions are executed. (By default, this is done in release packages on the **Requests** or **Deployment Unit** tabs, respectively.) The sync fields are populated as follows:

- `syncattempteddatetimestring`: shows the last time a reload was done
- `syncmessage`: if the reload failed, shows the error received from the provider
- `syncstatus`: either **FAILED** or **SUCCEEDED**
- `syncsucceededatetimestring`: shows the last time a reload was successful

For information on the reload transitions, see "Reloading Request Data" and "Reloading Deployment Unit Data" in the *Release Control Getting Started Guide*.

Provider-Specific Custom and Extended Fields

Some plugins include custom fields, extended fields, or custom attribute fields that enable you to specify fields that you want to reference. You can specify them in custom column entries in addition to all of the other types of fields available. This enables them to appear in the collection columns in the associated widgets, such as **Requests** and **Deployment Units**.

Provider-Specific Custom Column Fields

All providers that associate requests or deployment units have provider-specific custom column fields pre-defined so that you can use those for column data in **Requests** and **Deployment Units** collections.

Some providers use extended fields and custom attributes in conjunction with custom column fields to populate and format the collection data. Examples of these include SBM and ServiceNow.

For the list of custom column fields for this plugin, see:

[Custom Column Fields for Rally \[page 39\]](#)

Aliases for Similar Provider-Specific Fields

Aliases are a type of general field that should be used if you are using multiple providers for deployment units or requests. Alias fields should be used instead of the related provider-specific field to avoid having the same type of data appear in separate columns.

Alias Field Table

Following are the aliases and the provider-specific fields that they include.

Alias Display Name	Alias Internal Name	Provider	Provider Display Name	Provider Internal Name
Created By	createdBy	ZMF	Creator	creator
		DA	Creator	creator
		Dimensions CM	Originator	originator
		CA Agile Central (Rally)	Created By	createdBy
		SBM	Created By	createdBy
		JIRA	Reporter	reporter
		ALM	Created By - For Defects	BG_DETECTED_BY
		ALM	Created By - For Requirements	RQ_REQ_AUTHOR
		ALM Octane	Created By	createdBy
		ServiceNow	Created By	sys_created_by

Alias Display Name	Alias Internal Name	Provider	Provider Display Name	Provider Internal Name
Created On	createdOn	ZMF	Create Date	createDate
		DA	Created	created
		Dimensions CM	Creation Date	creationDate
		CA Agile Central (Rally)	Created On	createdOn
		SBM	Create Date	createdDate
		JIRA	Created On	createdOn
		ALM	Created On - For Defects	BG_DETECTION_DATE
		ALM	Created On - For Requirements	RQ_REQ_DATE
		ALM Octane	Created On	createdOn
		ServiceNow	Created On	sys_created_on
Owner	owner	SBM	Owner	owner
		JIRA	Assignee	assignee
		ServiceNow	Assigned To	assigned_to
Updated By	updatedBy	Dimensions CM	Last Updated By	lastUpdatedBy
		SBM	Modified By	modifiedBy
		ServiceNow	Updated By	sys_updated_by

Alias Display Name	Alias Internal Name	Provider	Provider Display Name	Provider Internal Name
Updated On	updatedOn	Dimensions CM	Update Date	updateDate
		SBM	Modified Date	modifiedDate
		ServiceNow	Updated On	sys_updated_on
type	Type	SBM	Active/Inactive	activeinactive
		JIRA	Type	issuetype

Adding Custom Columns to Collections

After you have added custom column entries to the auxiliary table and have applied the entries to projects or individual release items, they are now available to be used in collections, such as request and deployment unit lists.

To add custom columns to collections:

1. Add a request or deployment unit.
2. The columns defined in the custom column table entries for view and add will be available for the corresponding list.



Note: Existing requests and deployment units are not automatically updated with the new columns. Click **Reload Request Data** or **Reload Deployment Unit Data** to display the updated columns and their data.

Rally Plugin Configuration Overview

The CA Agile Central (Rally) plugin enables you to add Rally request items to release packages.

Before you can use the Rally plugin, administrators must do the following:

- Add the plugin. See [Preparing the Plugins \[page 5\]](#).
- Add a base configuration.
- Add one or more item configurations of the plugin.
- Configure custom column fields to display appropriate data. See [Custom Column Fields for Rally \[page 39\]](#).

See the following for more details.

- [Rally Base Configuration Details \[page 37\]](#)

- [Rally Request Items Configuration Properties \[page 39\]](#)
- [Custom Column Fields for Rally \[page 39\]](#)

Rally Base Configuration Details

Add a base configuration to define shared properties for a set of Rally request item configurations.

Field Descriptions

The field descriptions are included in the UI. While creating or editing a configuration, point to a field name to view its description.

The base configuration fields are described in the following tables.

Configuration Properties

Comparison	Description	Field Types	Example
Rally URL	<ul style="list-style-type: none"> • Specify a valid URL for a Rally API server. • Cannot be overridden. 	TEXT	<code>https://rally1.rallydev.com</code>
Rally API Key	<ul style="list-style-type: none"> • Valid Rally API key. 	TEXT	
Object Types	<p>Specify one or more of the following Rally object types to be displayed:</p> <ul style="list-style-type: none"> • Defect • Task • Story <p>Format:</p> <p><code>defect:<label>, task:<label>, hierarchicalrequirement:<label></code></p> <p>where:</p> <p><code><label></code> specifies the name of the object type that is displayed.</p>	TEXT	<code>defect:Defect1,defect:Defect2, task:Task1,task:Task2, hierarchicalrequirement:Story</code>

Comparison	Description	Field Types	Example
Show advanced search options	Select this option to enable users to widen a search to include Rally workspaces and projects.		
Query filter	<p>Specify a query filter using <code>\${search_text}</code> as a placeholder. You can search for names, descriptions, or IDs in Rally. Each filter clause consists of a tuple {field, operator, value} in this format:</p> <pre>((Name contains "\${search_text}") OR ((Description contains "\${search_text}") OR (FormattedID = "\${search_text}")))</pre> <ul style="list-style-type: none"> Name and description return any text that matches. FormattedID only returns an exact match. <p>Use AND and OR to connect several filter clauses. If you have more than two clauses, use AND and OR to only connect two of them, see the Example Values column.</p> <ul style="list-style-type: none"> You must preserve the query structure, particularly the opening and closing parentheses. Check available fields and their internal names at https://rally1.rallydev.com/slm/doc/webservice Returns a maximum of 200 results. If 200 results are returned there might be additional results, so refine your search to return less. 	TEXT	<pre>((Field1 > "value1") OR ((Field2 < "value2") OR ((Field3 = "value3") AND (Field4 contains "value4")))))</pre>

Validate Configuration

Field	Description
Validate	Checks the connection to the Rally server.

Rally Request Items Configuration Properties

The fields and options are the same as [Rally Base Configuration Details \[page 37\]](#).

If you create a new base configuration when you add a request item configuration, enter a title and description.

Related Topics

[Adding Action and Item Configurations \[page 11\]](#)

[Updating Plugin Configuration Details \[page 13\]](#)

Custom Column Fields for Rally

If you are using the CA Agile Central (Rally) plugin for requests, these are the custom column fields specific to this provider. For information on configuring them, see [Custom Columns \[page 29\]](#).

Display Names	Internal Names
Workspace	workspace
Project	project

Using the Rally Plugin

After configuration is complete, you are ready to use the functionality the plugin provides within Release Control.

For more information, see the following:

- [Adding Rally Request Items \[page 39\]](#)

Adding Rally Request Items

You can add Rally request items to a deployable release package or train.

To add Rally request items:

1. In a deployable release train or release package, select the **Requests** tab.
2. Click **Edit Requests**.
3. Click the + icon to add a request item and select **Rally**.
4. Select a Rally object type.
5. To search for Rally requests enter a value in **Search Text**.

6. (Optional) To search Rally projects do the following:
 - a. Click the **Project** field.
 - b. (Optional) Select an option from the **Workspace** list.
 - c. Enter a value in the **Project** field.
 - d. Click **Find Items**.
 - e. Select a Rally project from the list of results and click **Add**.
7. (Optional) To include other Rally projects in the project tree do the following:
 - a. To include projects *above* the selected project, select **Include Parent Projects**.
 - b. To include projects *below* the selected project, select **Include Child Projects**.
8. Click **Find Items**.
9. Select one or more Rally request items from the list of results.
10. Click **Add**.

Plugins Glossary

Actions

Actions are defined in execution plugin configurations and are executed in integrating products through deployment and failure tasks.

Artifacts

Files or other software elements. In Deployment Automation, these are represented by component versions.

Base Configurations

A base plugin configuration must be created for each set of configurations. It stores connection information and other shared properties and enables you to quickly create configurations of different types using that information. Some fields can be overridden at the derivative configuration level so that you can create configurations for different organizations or for slightly different purposes.

Baselines

In Dimensions CM, containers for release-ready components or artifacts. For more details, refer to the Dimensions CM documentation.

Change Packages

In ChangeMan ZMF, containers for release-ready components. For more details, see the ChangeMan ZMF documentation.

Collections

Collections are the lists of objects that you work with as part of Release Control, such as requests, deployment units, and deployment tasks. These are populated by collection widgets.

Component Versions

Instances of a Deployment Automation component, which contain the artifacts to be deployed or otherwise configured.

Configuration Overrides

You can override field values when creating or updating a configuration. You can also override the configuration settings at the SBM item level using auxiliary table override values. See [Auxiliary Table Level Configuration Overrides \[page 16\]](#).

Configurations

Provide connection information, product-specific details, and filters telling the plugin what data to obtain from the integrating product.

Deployment Packages

Instances of a Deployment Automation system. Deployment packages enable you to deploy artifacts for multiple applications. They may also include component processes where components are shared among multiple applications and associated versions are to be deployed as part of the larger package.

Plugin Instances

Variations of a plugin that may represent different versions of the plugin or instances owned by different business units who want to maintain an independent maintenance schedule.

Plugins

Plugins are used to implement the integrations between Release Control and other products.

Projects

In SBM, containers for release items, such as application releases, release trains, and environments; SBM projects.

Providers

Providers implement the plugin functionality. Plugins may have more than one provider implemented, such as an object provider and an execution provider.

Release Items

A general term used to refer to Release Control items that are submitted as SBM items, such as release packages and deployable release trains.

Single Sign-on (SSO)

Serena-installed software that enables a user to log in to a Web-based component of SBM and be recognized on subsequent accesses to that component or other Web-based components of SBM. This software also provides the ability for security tokens to be used in an orchestration, allowing Web services to be called without requiring the user to provide credentials at inconvenient times.

Snapshots

A Deployment Automation snapshot captures an application's current state, and as the application moves through different environments, the snapshot ensures that proper component versions are used.

Tags

Tags are unique names you can assign to plugin configurations in addition to the system-assigned Type Name. These are referenced in collection widgets to categorize the objects, such as requests, deployment units, and deployment tasks.

Tasks

Release Control tasks include deployment tasks, failure tasks, and tasks defined in task templates. In some cases, the term deployment tasks may be used in general to refer to all of these types of tasks.

Type Name

Type indicates the type of configuration based whether the configuration is a base configuration or one of the provider types, object (item) or execution (action). Type Name gives the specific instance name of the base configuration or provider type.