RC2010 to Blueprint 6.0

Migration Guide

Contents

Requirements Center 2010 to Blueprint Migration Guide

Migration Overview

The migration tool provides a way for you to export your Requirements Center 2010 (RC2010) projects to a format that can be imported into Blueprint. It is a staged migration, and does not require direct communication between RC2010 and Blueprint. Each RC2010 project must be migrated one at a time. Users cannot access newly migrated projects in Blueprint until an instance administrator grants access to the project.

The migration process involves the following high level steps:

1. Open RC2010 and fully reconcile the project that you want to migrate.
2. Launch the migration tool and export the fully reconciled project to Blueprint format.
3. Open Blueprint and import the project.
4. Create a project administrator role and assign the role to a user or group.
5. Verify that the project was migrated as expected, and cleanup custom properties and artifact types. When the project verification is complete, create additional roles and role assignments to grant access to users.

Refer to the detailed migration steps to learn more about the migration process. You may also be interested in reading about migration tips, tricks, and workarounds before beginning your migration.

What can be migrated from RC2010 to Blueprint?

The following data can be migrated from RC2010 to Blueprint:

- Project structure (models and packages in RC2010 become folders in Blueprint)
- Textual requirements
- Use cases
- Use case diagrams
- UI mockups (screens)
- Business process diagrams
- Glossaries
- External source data
- Files
- Traces
- Custom fields and properties
- Hyperlinks
    - Cross-model hyperlinks are migrated as manual traces
    - Hyperlinks to files are migrated as document references
    - Hyperlinks to URLs are migrated as rich text hyperlinks
- Comments
- Rich text formatting
    - Rich text tables can be migrated.
    - You can choose to disable the migration of font size/type formatting. If desired, you can also disable the migration of all rich text formatting.

What cannot be migrated?

The following data CANNOT be migrated:

- Access rights
- QC to RC2010 mappings
- Generated tests (you can re-generate your tests using Blueprint)
- Historical versions (only the most recent version is migrated)
- Document templates (Blueprint offers a new, and more powerful, document template solution)
- Stepwise settings
- Some RC2010 data is not migrated to Blueprint if a RC2010 feature is not supported in Blueprint (example: data operations)

What will my project look like after it is migrated?

Keep in mind that RC2010 models and packages become folders in Blueprint. Here's what the BP Air model looks like in Blueprint after the migration:
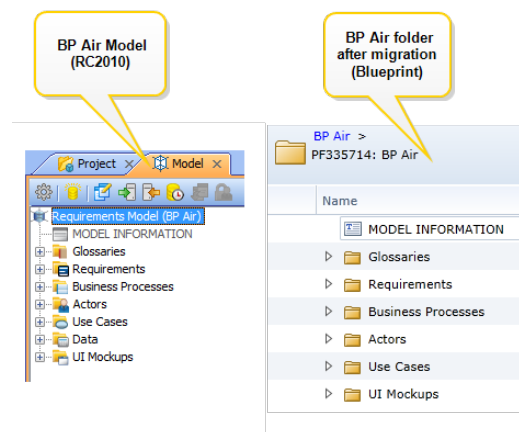
After you import your RC2010 project into Blueprint, a new textual requirement called **MODEL INFORMATION** is created for each model in the RC2010 project.

> **Note**: As you can see in the example, the Data object in RC2010 does not appear in Blueprint. At the current time, data objects cannot be migrated.

Migration Conflicts

In RC2010, custom properties and custom artifact types exist at the model level. In Blueprint, custom properties and artifact types exist at the project level. Therefore, during the migration, there is the risk of conflicting custom properties and artifact types.

The migration process follows a number of rules to determine whether or not to merge custom properties and artifact types. Below are some conflict scenarios and the result after migration. You may also be interested in learning more about cleaning up custom properties and artifact types after the migration.

Artifact type conflicts

- if an artifact type with the **same prefix and the same name** exists in multiple models in RC2010, *a single artifact* with that prefix and name appears in Blueprint after the migration.

    > **Note**: In Blueprint, the artifact type includes the properties from all of the artifact types (across all the models).

- if an artifact with the **same prefix but different name** exists in multiple models in RC2010, multiple artifact types appear in Blueprint.

    > **Note**: The prefix of each type is made unique by appending a number to the end of the prefix (example: **BR** and **BR2**). This can be modified, if desired.

- if an artifact with a **different prefix but the same name** exists in multiple models in RC2010, multiple artifact types appear in Blueprint.

    > **Note**: The name of each type is made unique by appending a number to the end of the artifact type (example: **Business Requirement** and **Business Requirement (2)**). This can be modified, if desired.

Custom property conflicts

- if a custom property with the **same name and same values** exists in multiple models in RC2010, a single custom property appears in Blueprint after the migration.
- if a custom property with the **same name but different values** exists in multiple models in RC2010, multiple custom properties appear in Blueprint after the migration.

    > **Note**: In Blueprint, the name of each property is preserved and appended with a number (example: **Type** and **Type (2)**)

Migration Prerequisites

- The migration tool must be run on any computer that is capable of running RC2010.
- The project that you want to migrate must be fully reconciled on the computer that you will use to run the migration tool.
- The project that you want to migrate must be from RC2010 version SR4 or later.

Migration Steps

> **Note**: Projects must be migrated one at a time. Repeat the process below to migrate additional projects.

Step 1: Fully reconcile the project that you want to migrate

> **Note**: This step is only required if your project is *published* and made available to other users using the Blueprint Team Repository (BTR). If your project is *published*, you must fully reconcile the project with the repository prior to beginning the migration steps.

1. Open RC2010.
2. Open the project that you want to migrate.
3. Reconcile each model in the project:

> **Important**: Repeat the following four steps for each model in the project. The models must be reconciled one at a time.

1. Double-click the model. The *Model* tab is displayed.
2. Click the **Reconciliation** 🔒 button.
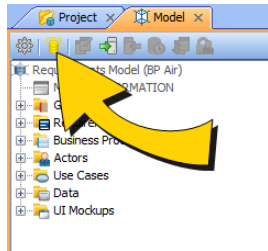
The *Reconciliation* dialog appears to warn you that the reconciliation cannot be undone.

3. Click **Yes** to continue.
   Another *Reconciliation* dialog appears to warn you that the reconciliation requires a lock on the entire model.
4. Click **Yes** to proceed.

> **Warning**: If a lock could not be obtained on the entire model, a message is displayed that identifies the lock owner. In this case, ask the lock owner to publish their changes, and then try to reconcile the model again.

After you have fully reconciled all models in the project, all models and packages on the *Project* tab are displayed with a green checkmark ✅ indicator:
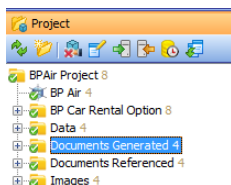
Step 2: Run the migration tool

1. Launch the migration tool from the computer that contains the fully reconciled project. The migration tool can be launched using the following files:

   - **`startup.bat`**: This version of the migration tool utilizes more memory and is intended for machines with more than 2GB of memory.
   - **`startuplowmemory.bat`**: This version of the migration tool is intended for machines with low memory (example: 2GB).
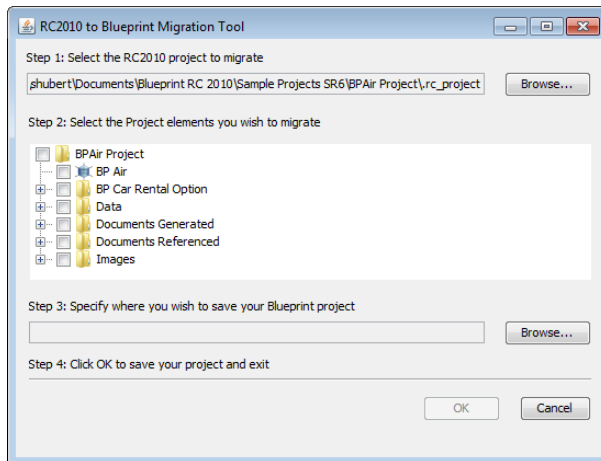
   The migration tool looks like this:
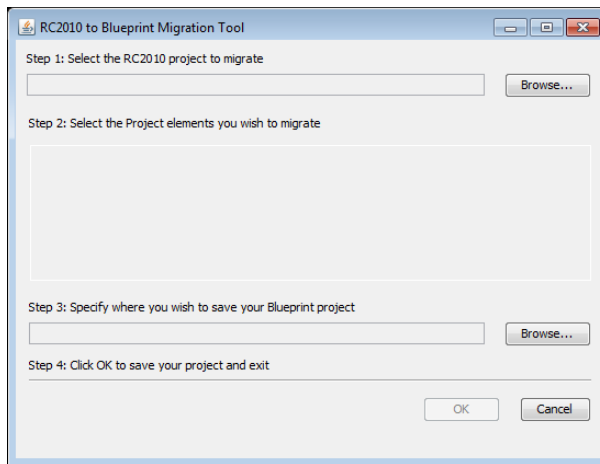
2. **Select the RC2010 project to migrate**:

   1. Click the **Browse…** button. The Select RC2010 Project File dialog is displayed.
   2. Locate and select the RC2010 project that you want to migrate. The project must be in **`.rc_project`** format.
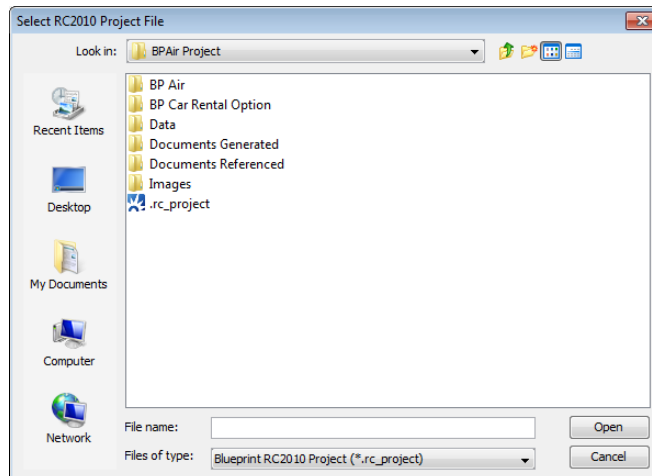

   3. Click **Open**.

The path and filename of the RC2010 project is displayed under the Step 1 heading:

3. Select the project elements that you wish to migrate:

> **Note**: You are not required to migrate the entire project. You can select individual items to migrate.

   1. Place a checkmark beside all items in your RC2010 project that you want to migrate to Blueprint. When you select an item, all children of that item are automatically selected.
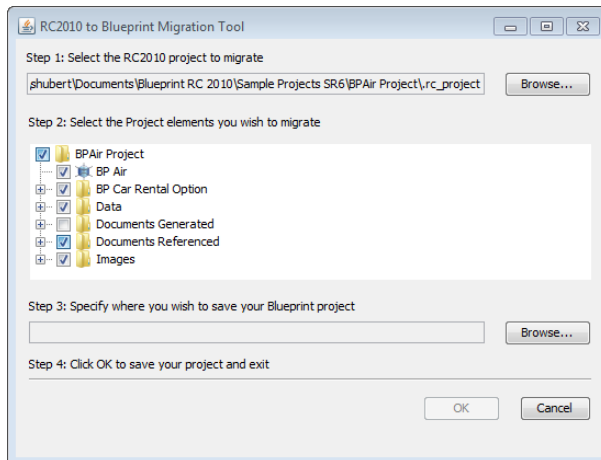
4. Specify where you wish to save your Blueprint project:

   1. Click the **Browse...** button.
   2. Navigate to the folder where you want to save the exported project.
   3. Type a filename for the exported project. The file must be a `.zip` file.
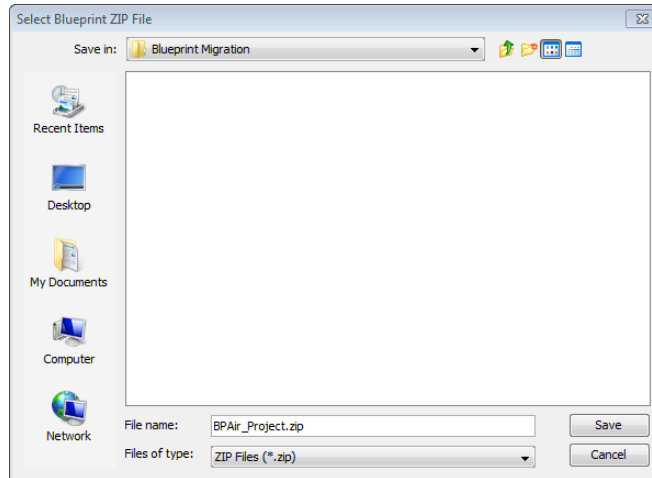
4. Click **Save**.

   The path and filename that you selected is displayed under the Step 3 heading.
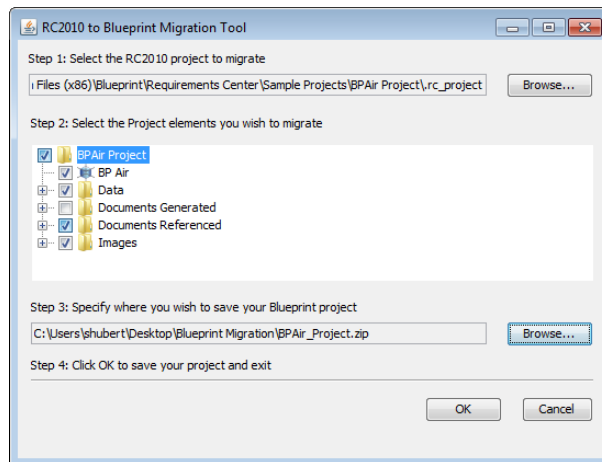
5. Click **OK** to export your RC2010 project to Blueprint format.

> **Note**: Please be patient. The export process can take an extended period of time, depending on the size of your project. If there is a problem with the export process, an error message is displayed.

The following dialog is displayed while the export is in progress:
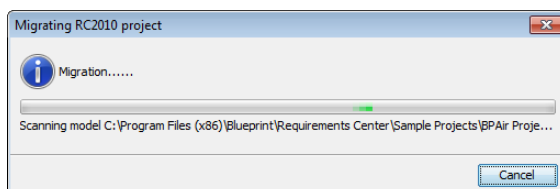
The following dialog is displayed when the export is complete:

The migration tool automatically closes after you click **OK**.

You can find your exported project in `.zip` format at the location you specified.

> **Warning**: If the Project.Xml file (contained in the `.zip` file) is larger than 250MB, you may experience problems importing your project into Blueprint. Learn more about the maximum project import size and applicable workarounds.

Step 3: Import the project into Blueprint

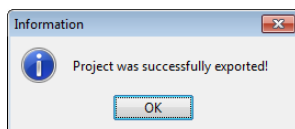1. Login to Blueprint with an account that has *Instance Admin* privileges.
2. Open the *Instance Administration Console* by clicking the application menu `Menu` > **Manage** > **Instance Administration**.
3. Click the **Projects** button on the ribbon.
4. right-click the folder that you want to hold your migrated RC2010 project, and then click **Import Project**. The *Import Project* wizard appears.
5. Click the `...` button and select the `.zip` file that you exported from RC2010 using the migration tool.
6. Click **Next**.
7. Specify a name for the project.
8. Click **Import**.

> **Note**: Please be patient. The import process can take an extended period of time, depending on the size of your project. If there is a problem with the migration, an error message is displayed.

9. When the import is complete, click **Close**. The project now appears in the tree on the left side of the screen.
10. Select the new project, and then click the **Launch Project Administration** button located in the lower right corner of the screen to open the *Project Administration Console*.

Step 4: Grant access to a project administrator

> **Note**: Until you create roles and role assignments, only the instance administrators can access the project.

1. Create a **Project Admin** role with **Project Admin** privileges:

   ### To create a new project role:

   1. Open the *Project Roles* tab.

      1. Open the *Project Administration Console*.
      2. Click the **Project Roles** link.
         The *Project Roles* tab is displayed.

   2. Click the **New** button.

      A new role appears in the Project Role list.

   3. Provide a name for the project role.

   4. Provide a description for the project role.

5. Select the privileges you want to grant to the project role.

Place a checkmark beside the privileges that are applicable to the project role:

Privilege	Description
Read	Provides the ability to view artifacts.
Edit	Provides the ability to modify and publish artifacts. If the Edit privilege is granted, the Read privilege is automatically granted.
Delete	Provides the ability to delete artifacts.
Trace	Provides the ability to create trace relationships between artifacts. If the Trace privilege is granted, the Read privilege is automatically granted. Note: In order to create a trace, the user must also have Edit permissions on at least one of the artifacts.
Comment	Provides the ability to add new comments and replies. If the Comment privilege is granted, the Read privilege is automatically granted.
Steal Lock	Provides the ability to steal the lock from other users. If the Steal Lock privilege is granted, the Read and Edit privileges are automatically granted. Warning: Stealing a lock discards all unpublished changes of the user that previously held the lock.
Can Report	Provides the ability to produce Blueprint Analytics reports. If the Can Report privilege is granted, the user can produce Blueprint Analytics reports using the project data in PowerPivot. Note: Blueprint Analytics reporting requires a Blueprint Analytics license.
Share via Home Page	Provides the ability to share projects and artifacts to the Home page, allowing all other users with author or collaborate licenses to view the item under Shared Items.

6. Select the project administrator role with the administrative privileges that you want the project role to have.

7. Click **Save**.

2. Create a new project role assignment so one of your users (or groups) can administer the project:

To add a new project role assignment:

1. Open the *Project Role Assignments* tab.

    1. Open the *Project Administration Console*.
    2. Click the **Project Role Assignments** link.
       The *Project Role Assignments* tab is displayed.

2. Click the **New** button on the ribbon.

    The *Select Identity* dialog is displayed.

3. Select the user or group for the new role assignment and click **OK**.

    Click the *Users* and *Group* tabs to toggle between the list of users and groups. You can only select one user or group for each role assignment. If you want to assign multiple users to the role, consider creating a new group with the users that require access.

After you click **OK**, the new project role assignment is created and the details of the new role assignment are displayed in the *Project Role Assignment Details* panel on the right side of the page.

4. Change the identity, if desired.

5. Select the role that you want to assign.

   All project roles are displayed in the drop-down list. You must create at least one role before you can create a role assignment.

6. Set the project role assignment scope.

   The project role assignment scope allows you to control whether the project role assignment applies to the entire project, or to a specific folder or artifact.

7. Click **Save**.

Step 5: Cleanup the project and create additional roles and role assignments

> **Tip**: The project administrator may also want to consider cleaning up custom properties and custom artifact types that were migrated from RC2010.

Now that the project has been migrated and you have created a role and role assignment, your project administrator is able to login. The project administrator can open the project and verify that the migration has occurred as expected. After the verification is complete, the project administrator can create additional roles and role assignments so other users can access the project.

For more information about managing artifact types, custom properties, roles, and role assignments, click the help ⑦ button located in the upper right corner of Blueprint and refer to the Project Administration guide.

Tips, Tricks, and Workarounds

Cleaning up custom properties

After your RC2010 project is imported into Blueprint, you may want to consider cleaning up some of the custom properties.

If your RC2010 project consisted of models that contained the same custom property name, but with different values, some cleanup may be required.

> **Example**
>
> A project, called **Hotel Booking**, contains a model called **Breakfast Add-On** and **Newspaper Add-On**. Both of these models contain a custom property called **Type** but the options are different for each model.
>
> After the project is imported into Blueprint, there will be 2 custom properties called **Type** and **Type (2)**.
>
> You may want to consider renaming these custom properties (example: **Breakfast Type** and **Newspaper Type**).

blueprint

> ## Example
> You may wish to delete this custom property if it is not required.

Read below to learn how to modify and delete custom properties.

To modify an existing custom property:

1. Open the *Properties* tab.

    1. Open the *Project Administration Console.*
    2. Click the **Properties** link.
       The *Properties* tab is displayed.

2. Select the custom property you want to modify.

    Select a property by clicking a row in the table. The property details are displayed on the right side of the page.

3. Update the custom property details.

    - `Name`: Indicates the name of the custom property.
    - `Type`: The custom property can be one of the following types: *text*, *number*, *date/time*, *choice*, or *user/group*.
    - `Settings`: The settings options are different depending on the selected `Type`. Here are the associated settings for each `Type`:
        - Text:
            - `Required`: Defines whether or not the property is required. If the property is required, artifacts cannot be saved unless a value for this property is specified.
            - `Rich Text`: Defines whether or not the field supports rich text.
            - `Multi Line`: Defines whether or not the field supports multi lines of text.
            - `Has Default Value`: Defines whether or not the property has a default value. If enabled, specify the default value into the space below.
        - Number
            - `Required`: Defines whether or not the property is required. If the property is required, artifacts cannot be saved unless a value for this property is specified.

- **Validated**: Defines whether or not the value specified for this property is validated.
- **Number of decimal places**: Defines the number of decimal places to save.
- **Max Value**: Defines the maximum acceptable number. This option is only applicable if the **Validated** option is enabled.
- **Min Value**: Defines the minimum acceptable number. This option is only applicable if the **Validated** option is enabled.
- **Has Default Value**: Defines whether or not the property has a default value. If enabled, specify the default value into the space below.

- Date/Time
  - **Required**: Defines whether or not the property is required. If the property is required, artifacts cannot be saved unless a value for this property is specified.
  - **Validated**: Defines whether or not the value specified for this property is validated.
  - **Max Value**: Defines the latest acceptable date. This option is only applicable if the **Validated** option is enabled.
  - **Min Value**: Defines the earliest acceptable date. This option is only applicable if the **Validated** option is enabled.
  - **Has Default Value**: Defines whether or not the property has a default value. If enabled, specify the default value into the space below.

- Choice
  - **Required**: Defines whether or not the property is required. If the property is required, artifacts cannot be saved unless a value for this property is specified.
  - **Allow Custom Value**: Defines whether or not users can specify a custom value for this property.
  - **Allow Multiple Choices**: Defines whether or not users can select more than one choice for this property.
  - **Set Valid Values**: Click this button to add, delete, and reorder the valid choices for this property.

- User/Group
  - **Required**: Defines whether or not the property is required. If the property is required, artifacts cannot be saved unless a value for this property is specified.
  - **Has Default Value**: Defines whether or not the property has a default value. If enabled, specify the default value into the space below.

- **Applies To Artifact Types**: Place a checkmark beside the artifact types that should contain this property. Click the **Manage Artifact Types** link to add or manage artifact types.

4. Click **Save**.

To delete a custom property:

> **Warning**: Deleting a custom property results in data lost. The custom property data is lost for all artifacts if the custom property is deleted.

1. Open the *Properties* tab.

   1. Open the *Project Administration Console*.
   2. Click the **Properties** link.
      The *Properties* tab is displayed.

2. Click the **Delete** button on the ribbon.

   The confirmation dialog appears.

3. Click **OK** to confirm the deletion.

For more information about custom properties, click the help ⑦ button located in the upper right corner of Blueprint and refer to the Project Administration guide.

Cleaning up custom artifact types

If an artifact with the **same basetype, same prefix, but different name** exists in multiple models in RC2010, multiple artifact types appear in Blueprint.

> **Note**: The prefix of each type is made unique by appending a number to the end of the prefix (example: **BR** and **BR2**). This can be modified, if desired

> **Example**
>
> A project, called **Hotel Booking**, contains a model called **Breakfast Add-On** and **Newspaper Add-On**. Both of these models contain a custom artifact type with the same base type, same prefix (**PR**), but different names (**Product Requirement** and **Process Requirement**).
>
> After the project is migrated to Blueprint, there are 2 custom artifact types. The names of the artifact types are preserved but the prefixes are changed to **PR** and **PR2** so they are unique.
>
> You may want to consider changing the prefixes of these custom properties.

> **Example**
>
> A project, called **Hotel Booking**, contains a model called **Breakfast Add-On** and **Newspaper Add-On**. Both of these models contain a custom artifact type with the same base type, different prefix (**BR** and **BUS**), and the same name(**Business Requirement**).
>
> After the project is migrated to Blueprint, there are 2 custom artifact types called **Business Requirement** and **Business Requirement (2)**. The prefix of each artifact type is preserved.
>
> You may want to consider changing the name of these custom properties.

Read below to learn how to modify artifact types.

To modify an existing custom artifact type:

> **Warning**: Modifying an existing custom artifact type can result in data loss! For example, if you change the `Base Type` of an artifact type from *GenericDiagram* to *TextualRequirement*, the existing artifacts of that type will no longer contain diagrams.

1. Open the *Custom Artifact Types* tab.

   1. Open the *Project Administration Console*.
   2. Click the **Custom Artifact Types**link.
      The *Custom Artifact Types* tab is displayed.

2. Select the artifact type you want to modify.

   Select an artifact type by clicking a row in the table. The artifact type details are displayed on the right side of the page.
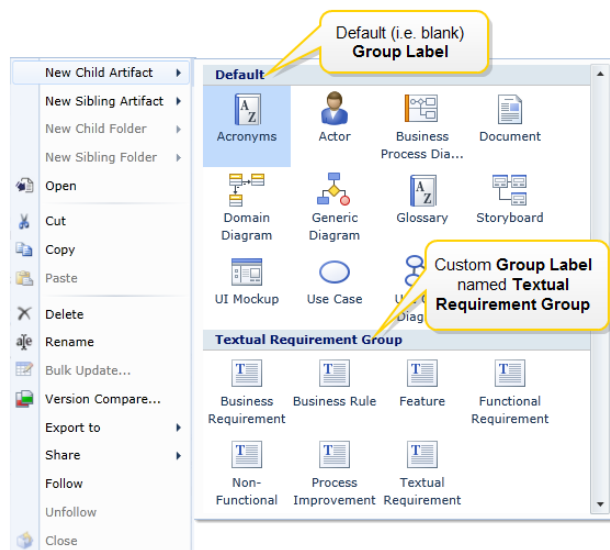
3. Update the artifact type details.

- `Name`: Indicates the name of the artifact type. This name appears in the list of options when users are selecting a new artifact to create. It also appears in the *Artifact Type* column when users are viewing the artifact list.

- `Icon`: If a new icon is added, the new icon appears in the list of options when users are selecting a new artifact to create. It also appears in the *Artifact Type* column when users are viewing the artifact list. Any uploaded icon should be a PNG or JPG file that is 32x32 pixels.

- `Prefix`: Indicates the ID prefix of the artifact type. Prefixes are unique across artifact types. All artifacts have a unique ID that begins with this prefix.

> **Note:** The following base type prefixes are already in use within Blueprint: **AC**, **BP**, **DOC**, **DD**, **GD**, **GL**, **PF**, **PR**, **RQ**, **SB**, **UC**, **UCD**, **UM**.

- `Group Label`: If specified, this artifact type appears under the specified label when users are selecting a new artifact to create. To set the group label, simply use the drop-down to select an existing group label, or type the name of a new group label into the field. If no value is specified, the artifact types appear under the *Default* label.

  In the example below, all artifact types are displayed under the *Default* label, except for the artifacts under the label called **Textual Requirement Group**:

- `Base Type`: All artifact types must have a base type. The base type determines the type of editor that is used when a user opens an artifact. The available base types are pre-configured.

- `Description`: Provides a description of the artifact type, such as the purpose or intended usage.

- `Properties`: Indicates any custom properties that are applied to this artifact type. Place a checkmark beside the custom properties that are applicable to this artifact type. Click the **Manage Properties** link to add or manage custom properties.

> **Tip**: You can customize the display order and layout of properties.

4. Click **Save**.

For more information about artifact types, click the help ⓘ button located in the upper right corner of Blueprint and refer to the Project Administration guide.

Determining the RC2010 ID of artifacts

For your convenience, Blueprint automatically creates a custom property, **RC2010 ID**, that provides you with the RC2010 ID of each artifact. Learn more about cleaning up custom properties if you do not need this custom property, or any other custom properties that were migrated from your RC2010 project.

Workaround for the maximum project import size issue

Under most circumstances, the migration tool will successfully migrate RC2010 projects that contain one model. The migration tool can also handle projects with multiple models, as long as the size of the exported project is not too large.

To ensure that your project is migrated properly, it is highly recommended that you import your project using a single `.zip` file, even if it means that you do not migrate all files and data from RC2010.

> **Tip**: You may want to consider adding large files (example: 50MB) to Blueprint directly, rather than migrating them using the migration tool.

If, however, the `Project.Xml` file (contained in the exported `.zip` file) is still greater than 250MB and will not successfully import into Blueprint, you can migrate the project in phases rather than in a single export/import process. In other words, you can use the migration tool to export segments of the project rather than exporting the data all at one time. For example, you may want to migrate one model at a time. If you choose to migrate using this method, you will have multiple `.zip` files that you must import into Blueprint.

> **Warning**: Traces between artifacts are only retained if they are exported into the same `.zip` file.