

# Agile Requirements Designer Integrations Configuration

These instructions are designed to help with the configuration of the following Integrations

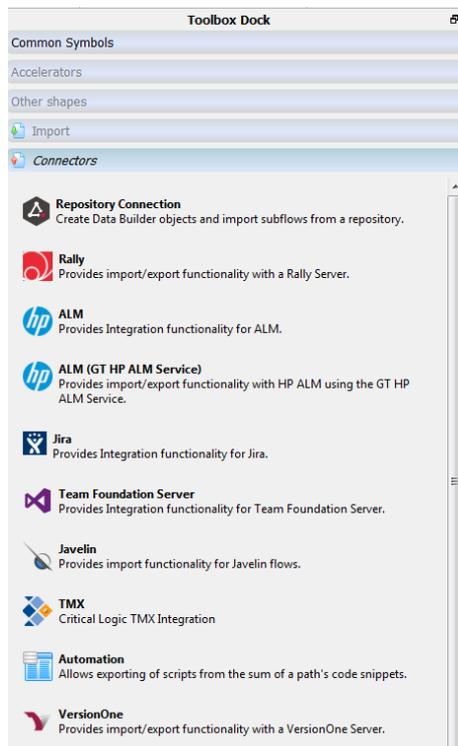
- HP Application Lifecycle Management.
- Microsoft Team Foundation Sever.
- Atlassian Jira.
- CA Agile Central.

## Connector

Each Integration within Agile Requirements Designer has its own connector option.

These can be found in the 'Connectors' panel within the 'Toolbox'

By clicking the desired Integration, a panel will appear, giving you access to the Integration's features.



## Configured Projects

Agile Requirements Designer is capable of connecting to multiple projects within a single session. This is done by querying the service for every project the user has access to.

As we know the user may not wish to interact with each project, Agile Requirement Designer only displays projects which have been configured correctly.

Each project has a list of 'Entities', an example would be a test case.

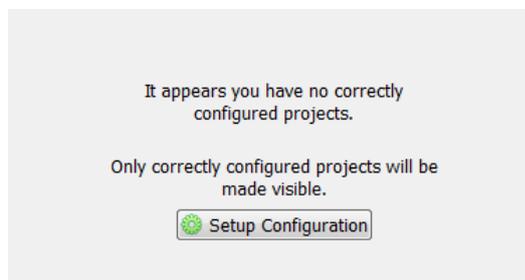
A project is defined as 'Configured' if at least one entity is configured correctly.

An entity is defined as configured correctly if

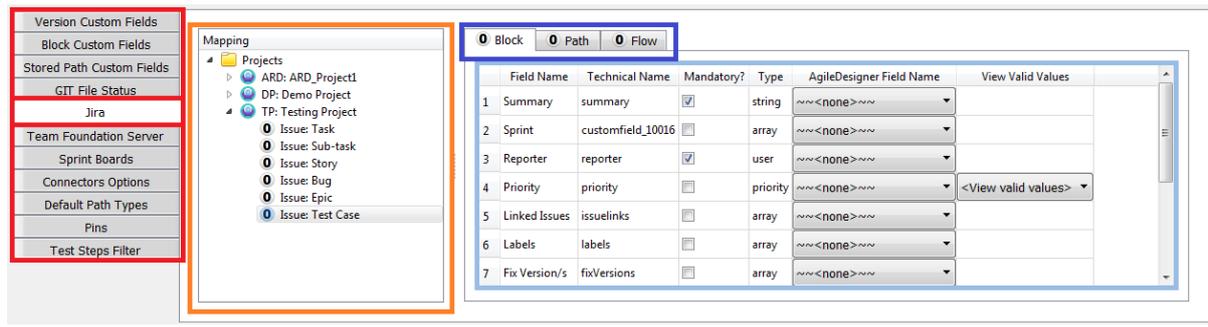
- At least one variable is mapped.
- All 'Mandatory' Variables are mapped.

These mappings are the minimum variables that Agile Requirements Designer must handle when creating these entities, as defined by the server.

Upon first logging into an integration, or if the user has no projects configured correctly, the following error message is presented and no projects are made visible to the user. Configuring any project is completed in the 'General Configuration' Dialog, which is made easily accessible from this error message.



## Configuring an Entity



All configuration for entities are handled within the 'General Configuration' dialog.

When the dialog first launches, make sure you have the correct integration **selected**.

The next step is to identify the project you wish to configure. To begin configuring a project, expand the menu where you will see a list of Entities you are able to **configure**.

The first decision to make when configuring an entity is to decide what object this is equivalent to within Agile Requirements Designer. We have three object types to choose from

- Block
- Path
- Flow

In certain Integrations Agile Requirements designer is able to make this decision for you and limit the configuration to a single object type. In the case of others, you will be presented with all the options to **choose**.

In our example, we can see that we want to configure a 'Jira Test Case' in the Project 'Testing Project'. A test case makes most sense to configure to a 'Path' within Agile Requirements Designer. Thus, in this example we will only configure within the 'Path' configuration tab.

## Entity Attribute Mapping

Configuring an Entity means to map each **required** attribute from the server to an attribute within Agile Requirements Designer.

The screenshot shows the configuration interface for an entity. On the left, a tree view shows the project structure: ARD: ARD\_Project1, DP: Demo Project, and TP: Testing Project. Under TP: Testing Project, several issue types are listed, with 'Issue: Test Case' selected. The main area displays a table for mapping attributes for the 'Path' entity.

| Field Name      | Technical Name    | Mandatory?                          | Type     | AgileDesigner Field Name | View Valid Values       |
|-----------------|-------------------|-------------------------------------|----------|--------------------------|-------------------------|
| 1 Summary       | summary           | <input checked="" type="checkbox"/> | string   | Path Name                |                         |
| 2 Sprint        | customfield_10016 | <input type="checkbox"/>            | array    | ~>>>none~>>>             |                         |
| 3 Reporter      | reporter          | <input checked="" type="checkbox"/> | user     | Jira Username            |                         |
| 4 Priority      | priority          | <input type="checkbox"/>            | priority | ~>>>none~>>>             | <<<View valid values>>> |
| 5 Linked Issues | issuelinks        | <input type="checkbox"/>            | array    | ~>>>none~>>>             | <<<View valid values>>> |
| 6 Labels        | labels            | <input type="checkbox"/>            | array    | ~>>>none~>>>             | Highest                 |
| 7 Fix Version/s | fixVersions       | <input type="checkbox"/>            | array    | ~>>>none~>>>             | High                    |
| 8 Epic Link     | customfield_10017 | <input type="checkbox"/>            | any      | ~>>>none~>>>             | Medium                  |
| 9 Description   | description       | <input type="checkbox"/>            | string   | Description of Blocks    | Low                     |
| 10 Component/s  | components        | <input type="checkbox"/>            | array    | ~>>>none~>>>             | Lowest                  |
| 11 Assignee     | assignee          | <input type="checkbox"/>            | user     | ~>>>none~>>>             |                         |

For each object type with Agile Requirements Designer has its own list of attributes to choose from. For example, when configuring an Agile Requirements Designer 'Path' we have the attribute named 'Path Name', however if we were configuring a 'Block' we would not have the same option, but 'Block Name'.

As we can see, to correctly configure a 'Jira Test Case' I need to map at least the two **Mandatory** fields, "Summary" and "Reporter".

We have mapped our "Test Case: Summary" to our "Path Name".

We have mapped our "Test Case: Reporter" to our "Jira Username".

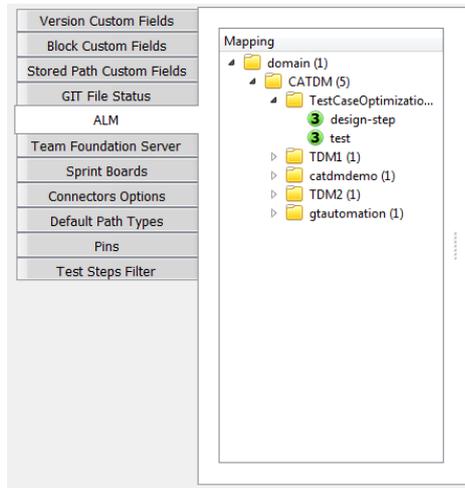
We have mapped our "Test Case: Description" to our "Description of Blocks".

Once we have correctly mapped at least one entity within the project, the project becomes visible to us within or Connector.

This screenshot shows the configuration interface with the project structure on the left. The 'Issue: Test Case' is selected. The main area displays the same mapping table as in the previous screenshot, but with the 'Issue: Test Case' selected in the tree view.

## Sub Entities

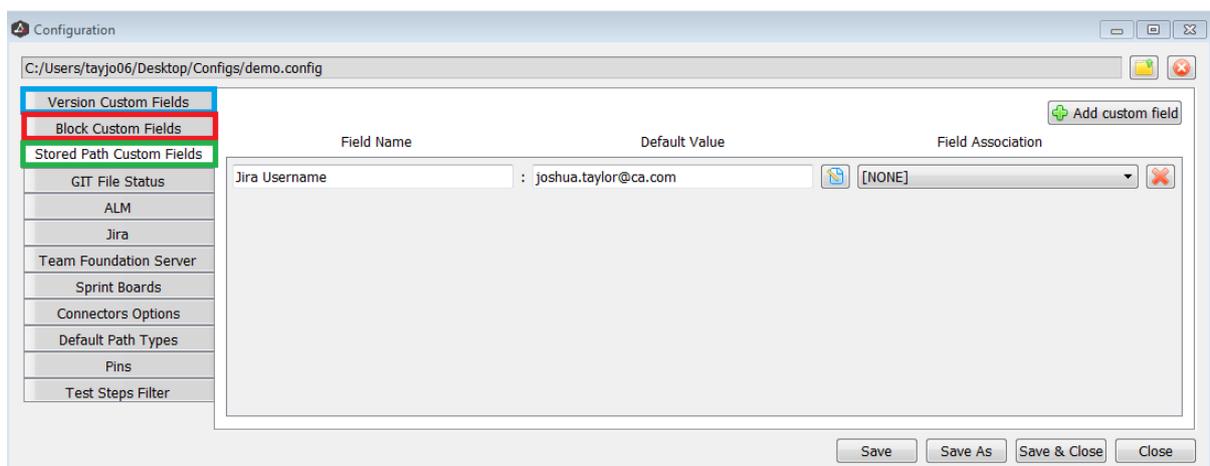
In some examples just configuring a single entity may not give you the desired results when using the Integrations. For example, in some instances a “Test Case” is actually made up of multiple “Steps”. So to export a “Test Case” with multiple “Steps” both Entities must be correctly configured. The integration with ALM is a good example of this. We have the parent entity “Test” and the child entity “Design Step”.



## Creating Attributes within Agile Requirements Designer

In certain situations, there are attributes for an Entity which has no relevant connection to a variable in Agile Requirements Designer. A good example of this is a username. I am trying to configure a “Jira Test Case” and one of the attributes is “Reporter”. We need to provide a person who reported the test, but Agile Designer Requirement has no appropriate field in the list of options to choose from.

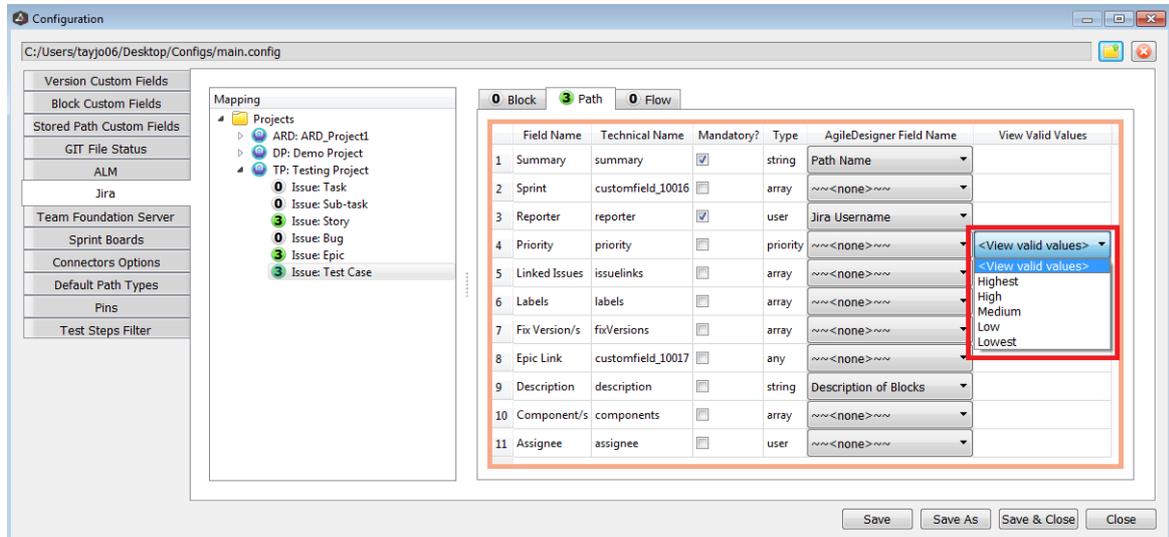
In this situation we are able to create a variable per ‘Object Type’, Block, Path and Flow.



For our purposes we are configuring a “Jira Test Case” as a “Path” Object, thus I need to create a “Stored Path Custom Field”, once saved the new “Custom Field” is available in the drop down.

## Valid Values

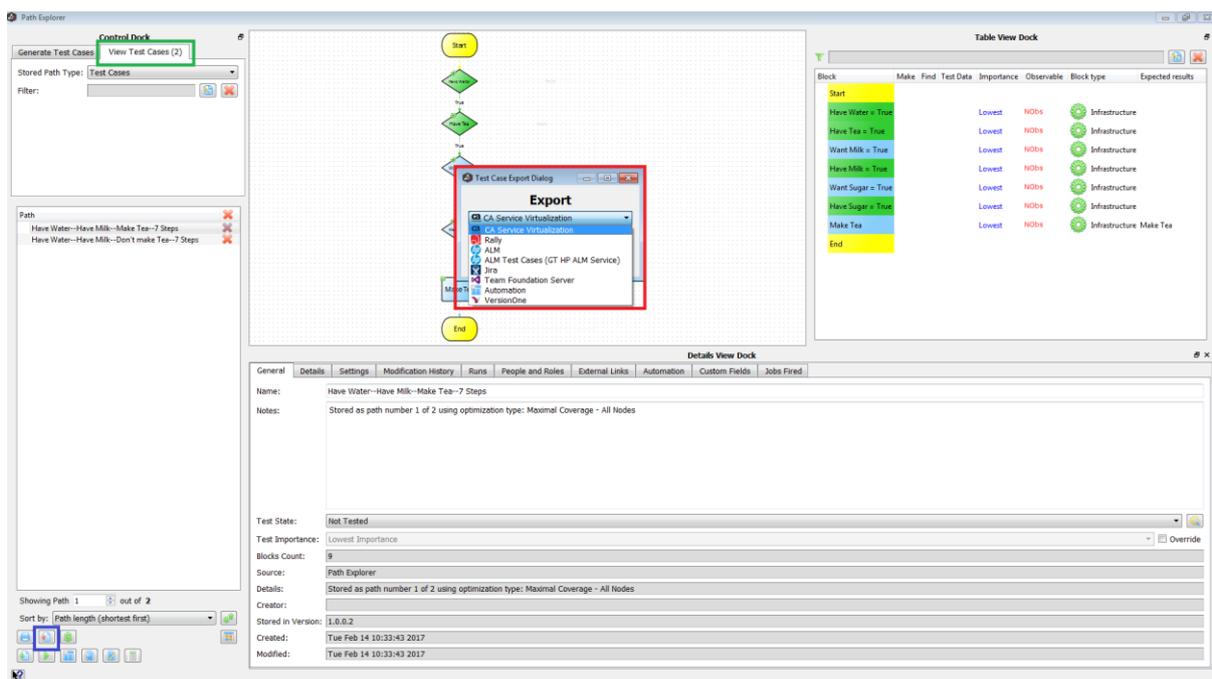
If an attribute from the server needs to be chosen from a list. Agile Requirements Designer will display the **list**.



## Export Types

Each integration is capable of exporting items from Agile Requirements Designer to a selected place on the server side. The type of object Agile Requirements Designer is able to export changes per Integration. The object Types are 'Block', 'Path' and 'Flow'.

If the integration you are using is capable of exporting 'Paths', you will be able to choose this export option from 'Path Explorer', when clicking the 'Export Paths' button you will see a **list** of applicable integrations which can export 'Paths'. The 'Paths' must be saved and you must be within the 'View Stored Paths' Screen.



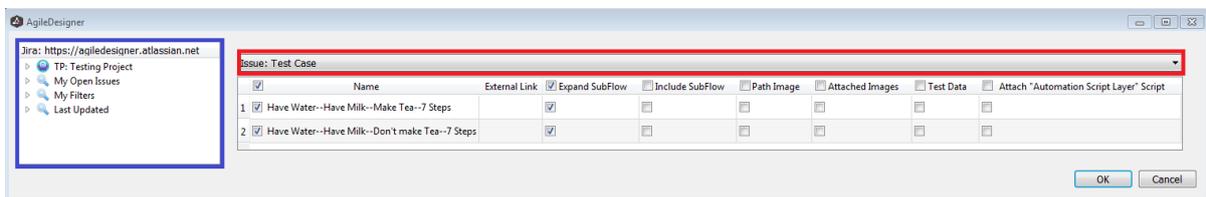
If the integration you are using is capable of exporting 'Paths' or 'Blocks' you will be able to choose this export option from the connector panel once you are logged in. If you hover over each button it will describe which object type you are exporting. The **first** is to export a 'Flow', and the **second** is used to export a 'Block'. Please note, not all integrations may offer these export options.



## Exporting

When exporting an object from Agile Requirements Designer there are two main decisions to make.

- 1) What Entity Type you want to export the Agile Requirements Designer object as.
- 2) The location where the Entity should be created on the server.



The Entity Type selection is drop by the **dropdown options** and the location is handled by clicking a location within the **servers' tree structure**. Agile Requirements Designer will handle validation once the user clicks the 'OK' button, various error messages can be presented to the user such as 'Incorrect Export Location'.

The dropdown options to select an Entity Type will only show correctly configured Entity Types for the Agile Requirements Designer object type the user is currently exporting. For example, if I am exporting the object type "Paths" from within "Path Explorer" I will only be presented with Entity types correctly mapped as "Paths".

## Export Options

The user also has a number of extra options when exporting. Most of these are to control if any attachments should be sent to the server. These can be found as optional tick boxes when exporting.

### Block

- 1) Attached Images – Constructs and attaches images which are attached to the block itself.

### Path

- 1) Path Image – Constructs and attaches the image of the test case in the flow.
- 2) Expand SubFlow – If the Stored Path utilises subflows, it will use every block from the subflows.
- 3) Include SubFlow – If a stored path utilises subflows, it will not use blocks from the subflow, rather the test case will represent a subflow as a single step.
- 4) Test Data – Constructs and attaches a CSV of test data for this path.
- 5) Attach Automation – Constructs and attaches an automation script for a given layer.

### Flow

- 1) Flow Image – Constructs and attaches an image of the flow.

## Default Export Options

For each possible export option, except attaching automation scripts, it is possible set defaults. If the user wishes to always attach an image when exporting, this can be set in the 'General Configuration' dialog under 'Connectors Options'.

