

Worksoft Certify Technical Note

Integrating Worksoft Certify with Atlassian JIRA

Worksoft Certify® integration with Atlassian® JIRA® enables a user to submit issues into JIRA from the Certify Result Viewer. JIRA is an IT issue management system that allows users to capture and organize IT issues, assign work, and follow team activities.

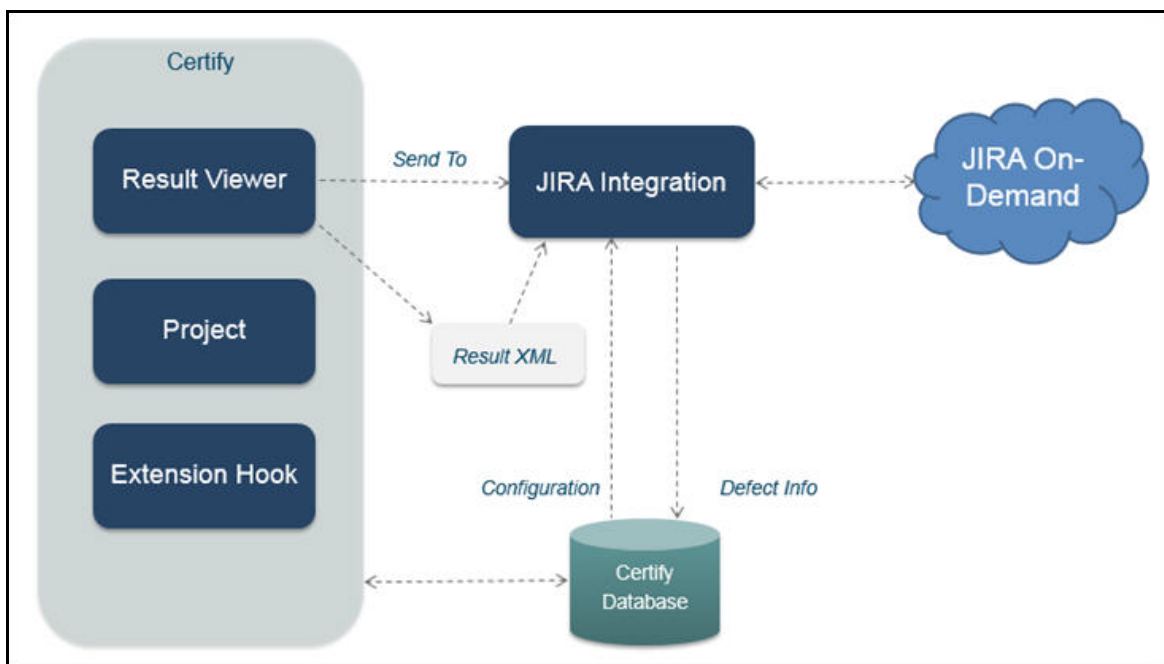
With the integration software, you can create JIRA issues and populate all of the issue's fields by using information from Certify results and processes. The issue is then imported into a JIRA system where you are able to edit and refine the issue. After the issue is submitted to JIRA, you are unable to see the details of the issue in Certify.

In order to integrate Certify with JIRA, you will need to do the following:

- ◆ Create an Extension hook in Certify
- ◆ Configure JIRA
- ◆ Map JIRA fields

Integration Architecture

The following diagram shows an overview of how this integration works.



In the Certify Result Viewer, a user right-clicks on a failed step and selects **Send To > JIRA**. Certify generates a result XML file and an image file that is sent to the JIRA Integration tool.

The Integration tool completes the following tasks:

- ◆ Reads the result file and image file
- ◆ Queries the Certify database for details of the target JIRA system, including the URI, user name, password, and field mapping
- ◆ Generates a JIRA issue
- ◆ Populates fields for the issue in JIRA by using Certify process and results data
- ◆ Submits the issue to JIRA which provides back an ID for a new issue
- ◆ Records the issue ID into the Certify database
- ◆ Launches JIRA to allow users to do additional editing on the new issue

JIRA Integration Command Line Modes

The JIRA Integration tool is implemented using a standalone command line executable. This executable has five modes of operation:

- ◆ Help
- ◆ GET_METADATA
- ◆ SET_CONFIG
- ◆ GET_CONFIG
- ◆ PROJECT

When you start the JIRA Integration executable without any arguments, you will see a summary of all the modes.

```
C:\Program Files (x86)\Worksoft\Certify\Client>jiraintegration.exe
JiraIntegration.exe can be called 5 ways:

* JiraIntegration.exe
  - Called with no arguments in order to see this help message

* JiraIntegration.exe GET_METADATA filename
  - Extract metadata from JIRA and write to the file specified

* JiraIntegration.exe SET_CONFIG filename
  - Save configuration data from the file specified to Certify database

* JiraIntegration.exe GET_CONFIG filename
  - Get configuration data from Certify database to the file specified

* JiraIntegration.exe PROJECT projectname filename
  - Extract result details from file then populate and submit an issue to JIRA for the specified proje

Press enter key to continue.
```

Help Mode

JIRA Integration executable, JiraIntegration.exe, starts in the Help mode. In this mode, JIRA configuration stored in Certify is retrieved, and an attempt will be made to connect with the JIRA system. Any conditions resulting in an error will be shown to the user.

GET_METADATA Mode

Use the GET_METADATA mode to extract metadata from JIRA and write to a specified file. This mode is useful to retrieve details about fields from the back-end system, and it will most likely be used by someone editing fields in configuration data.

You must have saved JIRA URL, user ID, and password in Certify through option SET_CONFIG before performing this operation. Certify uses these JIRA Credentials to establish connection with the JIRA System to retrieve meta data.

SET_CONFIG Mode

The SET_CONFIG mode saves configuration data from a specified file and stores the data to a Certify database.

Example 1:

```
<DETAILS>
  <USERNAME>myuserid</USERNAME>
  <PASSWORD>letmein</PASSWORD>
  <URI>    https://worksoftdefect.atlassian.net/rest/api/2/issue/</URI>
</DETAILS>
```

This configuration saves JIRA credentials into Certify. You must have at least this configuration saved in Certify before performing the GET_METADATA operation.

Example 2:

```
<DETAILS>
  <MyProjectName>
    <FIELD>
      <FIELDSTRING>"project":{"key":"CER"}</FIELDSTRING>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"summary":{"0"}</FIELDSTRING>
      <VALUE_PATH>CertifyResults/LogTestStep/LogTestStepDetails/Narrative</VALUE_PATH>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"issuetype":{"name":{"0"}}</FIELDSTRING>
      <DEFAULT_VALUE>Bug</DEFAULT_VALUE>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"reporter":{"name":{"0"}}</FIELDSTRING>
      <DEFAULT_VALUE>CertifyTester1</DEFAULT_VALUE>
      <FIELD_NAME>Reporter</FIELD_NAME>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"description":{"0"}</FIELDSTRING>
      <VALUE_PATH>CertifyResults/LogTestStep/LogTestStepDetails/ErrorMsg</VALUE_PATH>
      <FIELD_NAME>Description</FIELD_NAME>
    </FIELD>
  </MyProjectName>
</DETAILS>
```

```

</FIELD>
<FIELD>
  <FIELDSTRING>"assignee":{"name":"{0}"}/>FIELDSTRING>
  <DEFAULT_VALUE>CertifyTester1</DEFAULT_VALUE>
  <FIELD_NAME>Assignee</FIELD_NAME>
</FIELD>
</MyProjectName>
<USERNAME> myuserid</USERNAME>
<URI> https://worksoftdefect.atlassian.net/rest/api/2/issue/</URI>
<PASSWORD>letmein</PASSWORD>
</DETAILS>

```

Example 3:

```

<DETAILS>
  <MyProjectName>
    <FIELD>
      <FIELDSTRING>"project":{"key":"CER"}/>FIELDSTRING>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"summary":"{0}"</FIELDSTRING>
      <PROCESSATTRIBUTE>AttributeName</PROCESSATTRIBUTE>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"issuetype":{"name":"{0}"}/>FIELDSTRING>
      <DEFAULT_VALUE>Bug</DEFAULT_VALUE>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"reporter":{"name":"{0}"}/>FIELDSTRING>
      <VALUE_PATH>CertifyResults/LogTestStep/LogTestStepParams/Param[@Name='Returned
      Value']/>VALUE_PATH>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"description":"{0}"</FIELDSTRING>
      <PROCESSDESCRIPTION></PROCESSDESCRIPTION>
    </FIELD>
    <FIELD>
      <FIELDSTRING>"assignee":{"name":"{0}"}/>FIELDSTRING>
      <DEFAULT_VALUE>CertifyTester1</DEFAULT_VALUE>
    </FIELD>
  </MyProjectName>
  <USERNAME> myuserid</USERNAME>
  <URI> https://worksoftdefect.atlassian.net/rest/api/2/issue/</URI>
  <PASSWORD>letmein</PASSWORD>
</DETAILS>

```

The password field is specified in clear text, but it is encrypted when it is stored in the Certify database. Fields specify which JIRA fields to submit as part of the ticket.

The project name is a child of the root node field <DETAILS>. This project name should reflect the same name that you would use in Certify when invoking JIRA integration execution. Each configuration is identified by the project name that surround all <FIELD> tags. Certify can hold multiple configurations where each configuration is identified by unique project name.

- ◆ Value can be hard-coded in the case of the project field

Example: <FIELDSTRING>"project": {"key": "JI"}</FIELDSTRING>

- ◆ Value can be extracted from a Certify result

Example: <FIELDSTRING>"summary": "{0}"</FIELDSTRING>

where {0} is the token to be replaced by a value

Example: <VALUEPATH>CertifyResults/LogTestStep/LogTestStepDetails/Narrative</VALUEPATH>

where value path is the XPath expression for extracting a value from a Certify result xml

- ◆ Value can be extracted from a process attribute

Example: <FIELDSTRING>>"description2": "Temp {0} description"</FIELDSTRING>

where {0} is the token to be replaced by a value

Example: <PROCESSATTRIBUTE>AttributeName</PROCESSATTRIBUTE>

where attribute name is the name of the process attribute from which to select the value

- ◆ Value can be extracted from a process description

Example: <FIELDSTRING>>"description4": "Temp {0} description"</FIELDSTRING>

where {0} is the token to be replaced by a value

<PROCESSDESCRIPTION></PROCESSDESCRIPTION>

tells the integration to use the description of the top level process

- ◆ Value can be extracted from a linked requirement name

<FIELDSTRING>>"description3": "Temp {0} description"</FIELDSTRING>

where {0} is the token to be replaced by a value

<FIRSTLINKEDREQUIREMENT></FIRSTLINKEDREQUIREMENT>

tells the integration to find the first linked requirement and to extract its name

GET_CONFIG Mode

The GET_CONFIG mode extracts configuration details from the Certify database and stores the details in a specified file.

The password will not be decrypted to prevent anyone from finding out the JIRA password. If the mapping file is updated and needs to be saved back to Certify, you are able to do one of the following to protect your password:

- ◆ Omit the password tag.
Password field in the database will not be updated.
- ◆ Provide the correct clear-text value for the password.

PROJECT Mode

The PROJECT mode extracts result details from the Certify Result file, then populates and submits an issue to JIRA for a specified project.

You will need to provide the following two parameters for the PROJECT mode:

- ◆ Projectname - Refers to project name in the JIRA system where the JIRA issue is created. The project name corresponds to same project name that is defined in the configuration.
- ◆ Filename - Refers to the Certify Result file that is supplied by Certify when the PROJECT mode is invoked from Certify.

The Project option is intended to be used only from Certify where the file name is automatically provided by Certify during the execution.

PROJECT Issue Mode

The PROJECT Issue mode extracts result details from the result file, queries the Certify database for details of the target JIRA system, and generates a JIRA issue. If the integration was successful, then details of the JIRA issue will be associated with the result in the Certify database.

An entry will be created in a table called External Issue, and the entity ID field will be the ID of the result log header. Details will be added to a child table called External Issue Details, and these details will include the ID and key of the JIRA issue. If it is not successful, you will receive an error message. If you have an image attachment for the result step, it will be attached to the JIRA issue.

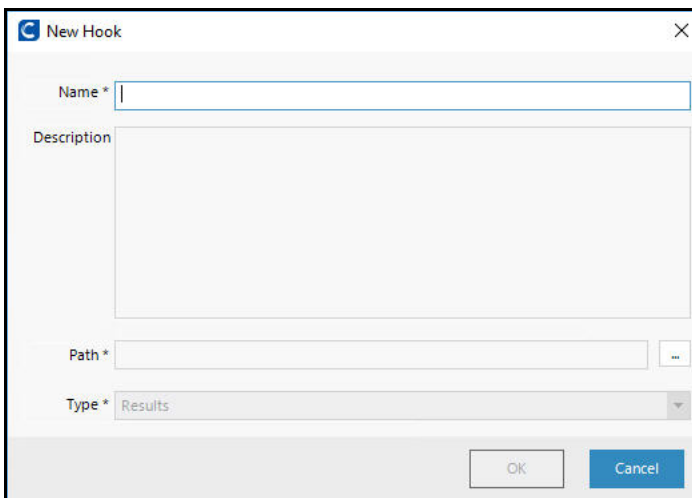
Creating an Extension Hook in Certify

The JIRA integration is implemented by creating an Extension hook in Certify and associating the hook to a Certify project.

► *To create a hook:*

- 1 In the Certify Navigation pane, click **Extensions**.
- 2 In the Navigation tree, select **Extensions > Hooks > Result**.
- 3 Right-click in the Summary pane and select **New Hook**.

The New Hook dialog box appears.



- 4 In the Name field, type **JIRA**.
- 5 In the Description field, type **JIRA Hook**.
- 6 In the Path field, type the path where the **JIRAIntegration.exe** file is found in the Worksoft Certify client folder:

```
. . .Worksoft\Certify\Client\JiraIntegration.exe PROJECT myprojectName
```

The key word PROJECT must be followed by the same project name that was used in the configuration earlier. Certify can not create a JIRA ticket if it is unable to find the project name.
- 7 Click **OK**.

► ***To add a hook to a project:***

- 1 In the Certify Navigation pane, click **Projects**.
The Projects window appears.
- 2 In the Summary pane, select a project.
- 3 Click the **Hooks** tab.
- 4 Right-click in the Hooks tab and select **Add Hook**.
The Select Hooks dialog box appears.
- 5 Select a hook in the Summary pane.
- 6 Click **OK**.
The hook is added to the project and appears in the Hooks tab. Also, the JIRA option appears in the right-click menu option **Send To** in the Results Viewer Summary pane.

Configuring the JIRA Connection

In order for you to integrate with JIRA, you will need to do the following tasks:

- ◆ Create a configuration file for the JIRA Integration executable. The file requires a URI, user name, and password. This information gets stored in the Certify database along with details of which fields to populate for a JIRA issue.
- ◆ Save the configuration file in the Certify database.

► ***To create a configuration file:***

- 1 In a text editor, create a JIRA Integration configuration file with the following code:

```
<DETAILS>  
  <USERNAME>myusername</USERNAME>  
  <PASSWORD>worksoft</PASSWORD>  
  <URI>https://worksoftdefect.atlassian.net/rest/api/2/issue/</URI>  
</DETAILS>
```
- 2 Add the following JIRA information:
 - User name
 - Password
 - URI

- 3 Save the updated configuration file as an XML file.

Example:

```
. . . C:\Temp\JiraTemplate.xml
```

You now need to save the configuration file to the Certify database.

► *To save the configuration file to the Certify database:*

- 1 Open the Command Prompt window and navigate to the JIRA Integration executable.
- 2 Execute the JIRA Integration executable with the **SET_CONFIG** mode to upload the configuration file to the Certify database.

Example:

```
...Worksoft\Certify\Client\JiraIntegration.exe SET_CONFIG c:\temp\JiraTemplate.xml
```

If there are errors, then the Command Prompt window will indicate the source of the problem.

- 3 Execute the JIRA Integration executable again with the **GET_METADATA** mode to extract metadata from JIRA and write to a specified file.

Example:

```
...Worksoft\Certify\Client\JiraIntegration.exe GET_METADATA c:\temp\JiraTemplate.xml
```

The remote JIRA system is contacted, extracts metadata, and stores the data to the specified file.

- 4 After you have added the metadata, you will execute the JIRA Integration executable with the **SET_CONFIG** mode again to upload the updated configuration file to the Certify database.

Example:

```
...Worksoft\Certify\Client\JiraIntegration.exe SET_CONFIG c:\temp\JiraTemplate.xml
```

- 5 If you want to retrieve the JIRA configuration that is now stored in Certify, run the executable with the **GET_CONFIG** mode to retrieve the configuration. This step is optional.

Example:

```
...Worksoft\Certify\Client\JiraIntegration.exe GET_CONFIG c:\temp\JiraTemplate.xml
```

Mapping JIRA Fields

When an issue is created and submitted into JIRA, it must have values in the required fields. You will need to identify required fields, optional fields, and values by consulting your JIRA administrator.

You can also run the JIRA Integration executable with the GET_METADATA mode to identify needed fields. The output file includes details about projects, fields, and values from your JIRA instance.

Building Mapping Configuration

Mapping information is specified in a XML file that is subsequently loaded into the Certify database. This file has the following structure:

```
<DETAILS>
  <FIELDS>
    <FIELD>Field 1 details</FIELD>
    <FIELD>Field 2 details</FIELD>
    <FIELD>Field 3 details</FIELD>
  </FIELDS>
  <USERNAME>myuserid</USERNAME>
  <PASSWORD>worksoft</PASSWORD>
  <URI>https://worksoftdefect.atlassian.net/rest/api/2/issue/</URI>
</DETAILS>
```

Username and password fields are optional. These fields need to be loaded into Certify one time, and they do not need to be loaded with field definitions.

Where to Get Data for JIRA Fields

As part of your implementation, you need to determine where to get data for the fields in your JIRA issue. Data can be gathered from the following places:

- ◆ **Hard-coded value** - A value is hard-coded in the configuration

```
<FIELD>
<FIELDSTRING>"project": {"key": "JI"}</FIELDSTRING>
</FIELD>
```

In this case the project will be identified using a key and will always get a value of JI.

- ◆ **Result XPATH** - A value is retrieved from the result using an XPATH expression.

```
<FIELD>
<FIELDSTRING>"summary": "{0}"</FIELDSTRING>  <VALUEPATH>CertifyResults/
LogTestStep/LogTestStepDetails/Narrative</VALUEPATH>
</FIELD>
```

In this case, the Summary field has a token {0} which will get replaced by the value extracted by the XPATH expression on the result XML file.

- ◆ **Process Description** - A value is retrieved from the description property of the top-level process in the execution.

```
<FIELD>
<FIELDSTRING>"description": "{0}"</FIELDSTRING>
<PROCESSDESCRIPTION></PROCESSDESCRIPTION>
</FIELD>
```

In this case, the Description field has a token {0} which will get replaced by the description of the top-level process in the execution.

- ◆ **Process Attribute** - A value is retrieved from a process attribute for the top-level process in the execution.

```
<FIELD>
<FIELDSTRING>"integrationtype": "{0}"</FIELDSTRING>
<PROCESSATTRIBUTE>Scope</PROCESSATTRIBUTE>
</FIELD>
```

In this case, the Integrationtype field has a token {0} which will get replaced by the value of the Scope attribute of the top level process in the execution.

- ◆ **First Linked Requirement** - A value is the name of a first linked requirement from a top-level process.

```
<FIELD>
<FIELDSTRING>"description3": "Temp {0} description"</FIELDSTRING>
<FIRSTLINKEDREQUIREMENT></FIRSTLINKEDREQUIREMENT>
</FIELD>
```

In this case, the Description3 field has some fixed text and a token {0} which will get replaced by the name of the first linked requirement.

Save Field Mapping File to Certify

In the Command Prompt window, execute JIRAIntegration.exe to load your mapping file into the Certify database.

Next, execute JIRA integration executable and tell it to load your mapping file into the Certify database with the SET_CONFIG mode.

Example:

```
JiraIntegration.exe C:\temp\configuration.xml SET_CONFIG
```

If there are errors, then the Command Prompt window will indicate the source of the problem.

Submitting an Issue to JIRA

After you have completed configuring the JIRA configuration, you can submit an issue to JIRA. You will use the Certify Result Viewer to submit your issues into JIRA.

► ***To submit an issue:***

- 1 In the Certify Result Viewer, select a process in the Navigation pane.

The Summary pane lists the steps of the process.

- 2 Right-click on a step and select **Send To > JIRA**.

Certify generates a result XML file and an image file that is sent to the JIRA Integration tool.

The JIRA Integration tool completes the following tasks:

- Reads the result file.
- Reads the image file if one exists.
- Queries the Certify database for details of the target JIRA system, including the destination URI, user name, password, and field mapping.
- Generates a JIRA issue.
- Populates fields for the issue in JIRA by using Certify process and results data.
- Submits the issue to JIRA which provides back an ID for a new issue.
- Records the issue ID into the Certify database.
- Launches JIRA to allow users to do additional editing on the new issue.

While the issue is being constructed and submitted, the Command Prompt window shows the details of the progression.