SpiraTest | Migration and Integration Guide

Inflectra Corporation

Date: September 6th, 2008

Contents

1. Introduction

SpiraTest® provides an integrated, holistic Quality Assurance (QA) management solution that manages requirements, tests and incidents in one environment, with complete traceability from inception to completion.

This guide outlines both how to migrate data from other Test Management tools into SpiraTest, and also how to use SpiraTest in conjunction with certain Requirements Management and Issue Tracking systems.

This guide assumes that the reader is familiar with both SpiraTest and the appropriate tool being discussed. For information regarding how to use SpiraTest, please refer to the *SpiraTest User Manual*.

Each of the sections covers a different tool so we recommend using the table of contents on the left to locate the tool you're looking to either integrate or migrate from, then read the installation and usage instructions.

2. Migrating from HP QualityCenter

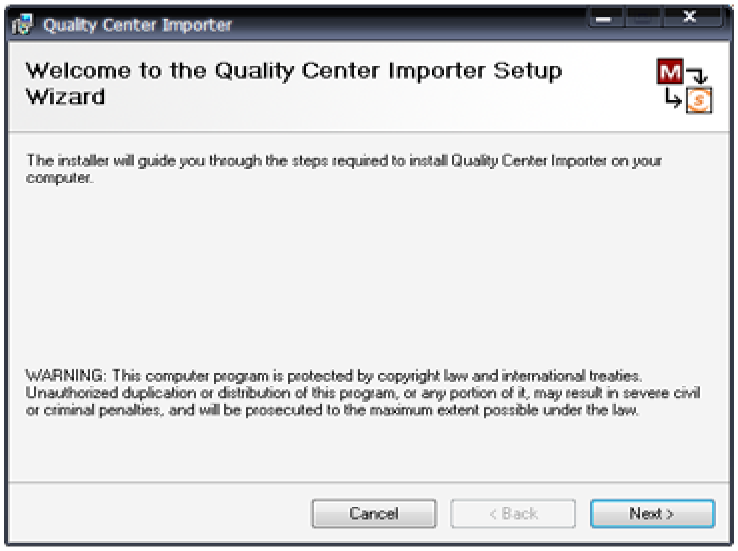
This section outlines how to use the included Migration Tool for importing Requirements, Test Cases, Test Runs and Incidents from HP QualityCenter (formerly known as Mercury TestDirector).

2.1. Installing the QualityCenter Migration Tool

This section outlines how to install the migration tool for QualityCenter onto a workstation so that you can then migrate whole projects from QualityCenter to SpiraTest. It assumes that you already have a working installation of SpiraTest v1.5 or later. If you have an earlier version of SpiraTest you will need to upgrade to at least v1.5 before trying to migrate projects.

To obtain the version of the migration tool that is compatible with your version of SpiraTest, you simply need to log-in as a project-level administrator to SpiraTest, go to the Administration home page and download the Migration Tool Windows Installer package (.msi). This process is described in the *SpiraTest Administration Guide* in more detail.

Once you have obtained the Windows Installer package, simply double-click on the package to begin the installation wizard which should display the following welcome page:

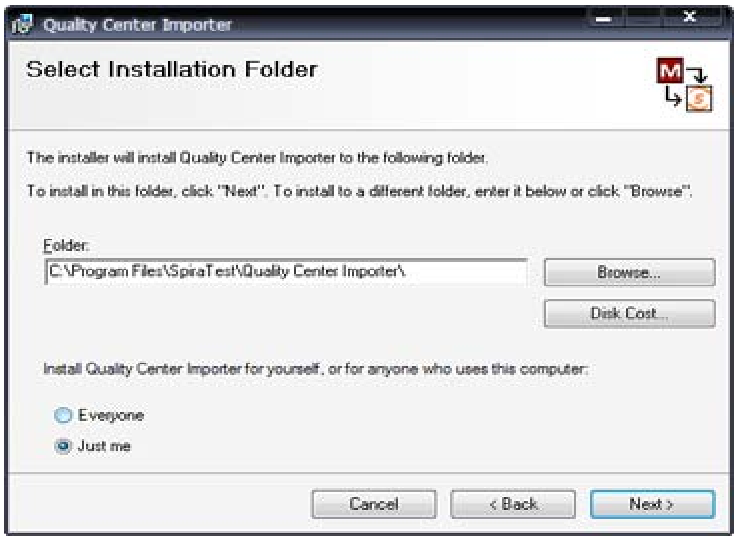


Welcome to the Quality Center Importer Setup Wizard

The installer will guide you through the steps required to install Quality Center Importer on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Click the <Next> button to choose the folder to install the migration tool to:


Quality Center Importer

Select Installation Folder

The installer will install Quality Center Importer to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder: C:\Program Files\SpiraTest\Quality Center Importer\

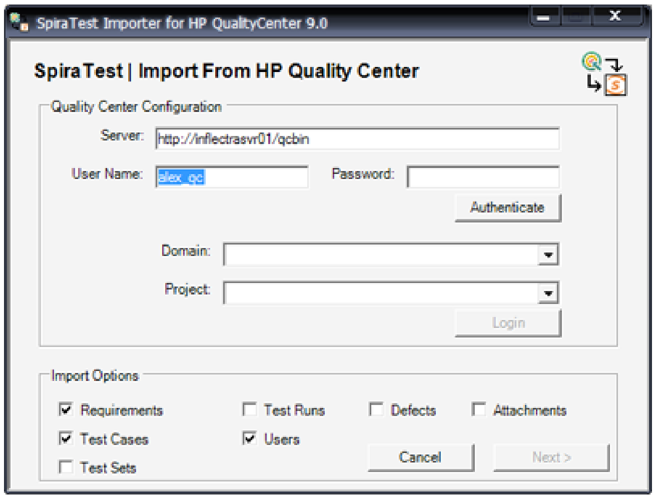
Install Quality Center Importer for yourself, or for anyone who uses this computer:

Everyone [✓ Just me]

Choose the folder to install to, and then decide whether the application should be accessible by all users on the workstation or just the current user. Then click the <Next> button to start the installation process. It will confirm if you want to proceed, click <Next> then wait for it to finish.

2.2. Using the HP QualityCenter Migration Tool

Now that you have installed the migration tool, you can launch it at any time by going to Start > Programs > SpiraTest > Tools > QualityCenter Importer. This will launch the migration tool application itself:


SpiraTest Importer for HP QualityCenter 9.0

SpiraTest | Import From HP Quality Center

Quality Center Configuration

Server: http://inflectrasvr01/qcbin

User Name: alex_qc

Password:

Domain:

Project:

Import Options

[✓ Requirements] Test Runs Defects Attachments

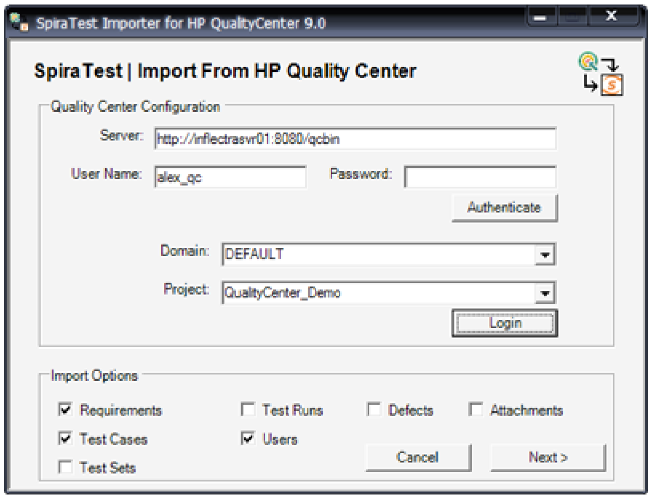
[✓ Test Cases] [✓ Users]

Test Sets

The first thing you need to do is to enter the URL for the instance of HP QualityCenter that you want to import the information from (typically of the form `http://<server name>/qcbin`) together with a valid username and password.

Note that the importer has only been tested against version 9.0 of Quality Center or later. It may not work correctly against previous versions. Once you have entered this information, click the <Authenticate> button and the list of possible domains and projects will be populated.

Select the QualityCenter domain and project that you want to ***import from*** and click the <Login> button:


SpiraTest Importer for HP QualityCenter 9.0

SpiraTest | Import From HP Quality Center

Quality Center Configuration

Server: http://inflectrasvr01:8080/qcbin

User Name: alex_qc

Password:

Domain: DEFAULT

Project: QualityCenter_Demo

Import Options

[✓] Requirements

[✓] Test Cases

[ ] Test Sets

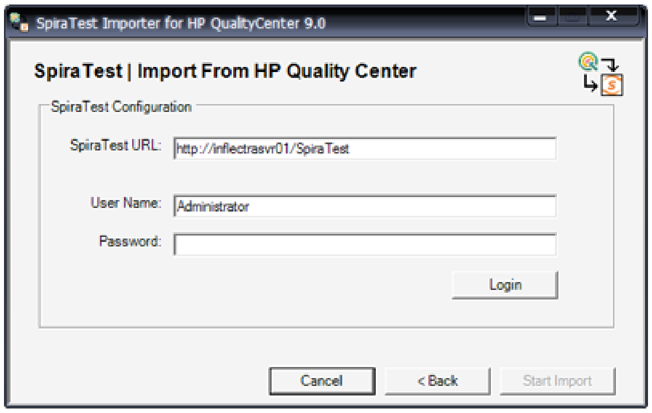
[ ] Test Runs

[✓] Users

[ ] Defects

[ ] Attachments

Assuming that the user name selected has permission to access this project, you will be prompted with a message box indicating that the login was successful. Now choose the types of artifact you want to import and then click the <Next> button to move to the next page in the import wizard:


SpiraTest Importer for HP QualityCenter 9.0

SpiraTest | Import From HP Quality Center

SpiraTest Configuration

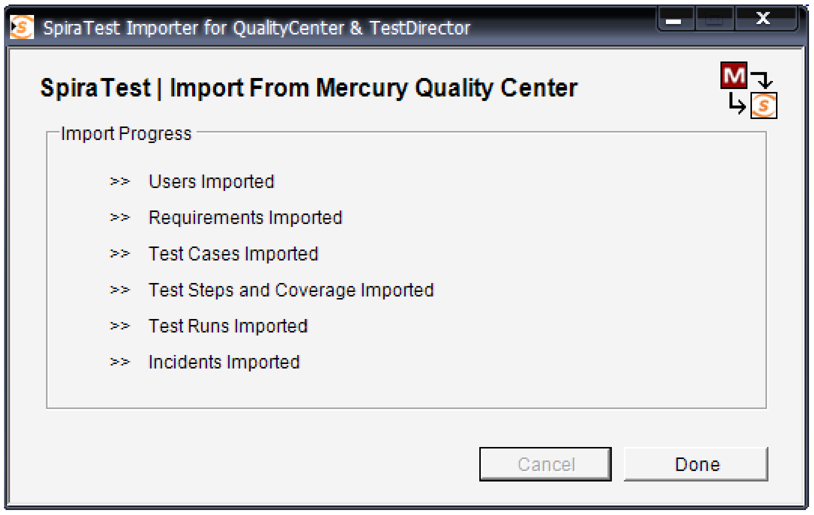
SpiraTest URL: http://inflectrasvr01/SpiraTest

User Name: Administrator

Password:

This page allows you to enter the URL, user name and password that you want to use to access the instance of SpiraTest that you want to ***import to*** and click <Login>. Typically the URL is of the form (`http://<server name>/SpiraTest`). The version of the importer being used must be compatible with the version of SpiraTest you're importing into; if not you will receive an error message.

Assuming that the login was successful, click the <Start Import> button to actually begin the process of importing the various artifacts from QualityCenter into SpiraTest. Note that the importer will automatically create a new project in SpiraTest to hold all the artifacts with the same name as that used in QualityCenter.



SpiraTest | Import From Mercury Quality Center

Import Progress

>> Users Imported
>> Requirements Imported
>> Test Cases Imported
>> Test Steps and Coverage Imported
>> Test Runs Imported
>> Incidents Imported

During the import process, as each of the types of artifact are imported, the progress display will change (as illustrated above). Once the import has finished, you will receive a message to that effect and the <Done> button will be enabled. Clicking this button closed the importer. You should now log into SpiraTest using the same user name and password that was used for the import to view the imported project.

The migration tool will import the following artifacts:

- Users (but not their roles and permissions)
- Requirements
- Test Cases and their associated manual design steps (but not any automated test scripts)
- Test Runs and their associated manual test steps
- Test Sets and the association with the test cases
- Defects, together with their associated lists of priorities and statuses
- The coverage relationship between requirements and test cases
- The linkages between any defects and test runs
- The first ten (10) user-defined fields on each of the above artifact types.
- Any attachments associated with the requirements, test cases, test sets or design steps.

Note: Once you have migrated a project into SpiraTest you will have the definitions of incident priorities and statuses from both SpiraTest and QualityCenter (this is because QualityCenter doesn't use the same codes as SpiraTest, so they will be imported). You should go back in to the Administration screen and make all the SpiraTest statuses and priorities inactive.

2. Using SpiraTest with RequisitePro

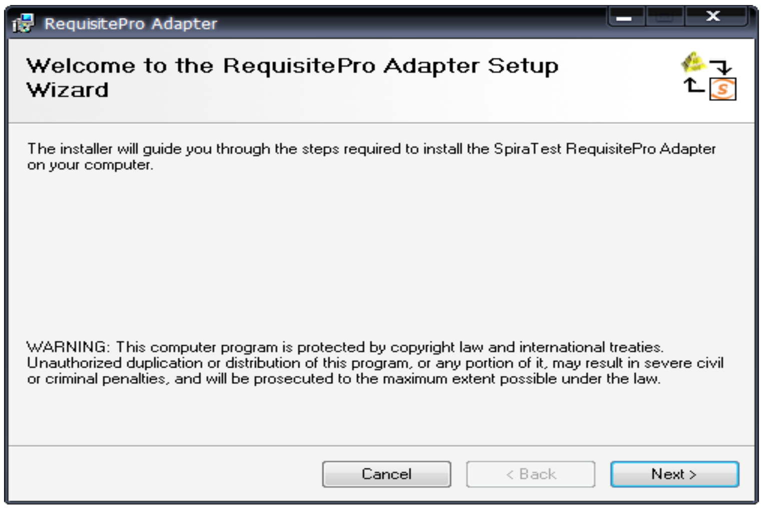
This section outlines how to use the included Integration Adapter for importing Requirements, and Use Cases from IBM Rational® RequisitePro® into SpiraTest®.

2.1. Installing the Integration Adapter

This section outlines how to install the integration adapter for RequisitePro onto a workstation so that you can then import requirements and use cases from RequisitePro into SpiraTest. It assumes that you already have a working installation of SpiraTest v1.0.4 or later. If you have an earlier version of SpiraTest you will need to upgrade to at least v1.0.4 before trying to import data.

To obtain the version of the migration tool that is compatible with your version of SpiraTest, you simply need to log-in as a project-level administrator to SpiraTest, go to the Administration home page and download the Integration Adapter Windows Installer package (.msi). This process is described in the *SpiraTest Administration Guide* in more detail.

Important: You must install the integration adapter on the same workstation that has the installed copy of RequisitePro. Once you have obtained the Windows Installer package, simply double-click on the package to begin the installation wizard which should display the following welcome page:

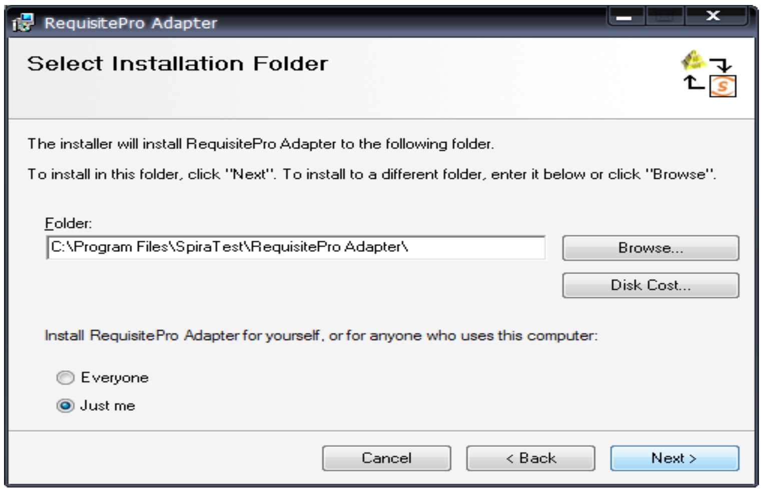


Welcome to the RequisitePro Adapter Setup Wizard

The installer will guide you through the steps required to install the SpiraTest RequisitePro Adapter on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Click the <Next> button to choose the folder to install the integration adapter to:



Select Installation Folder

The installer will install RequisitePro Adapter to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder: C:\Program Files\SpiraTest\RequisitePro Adapter\

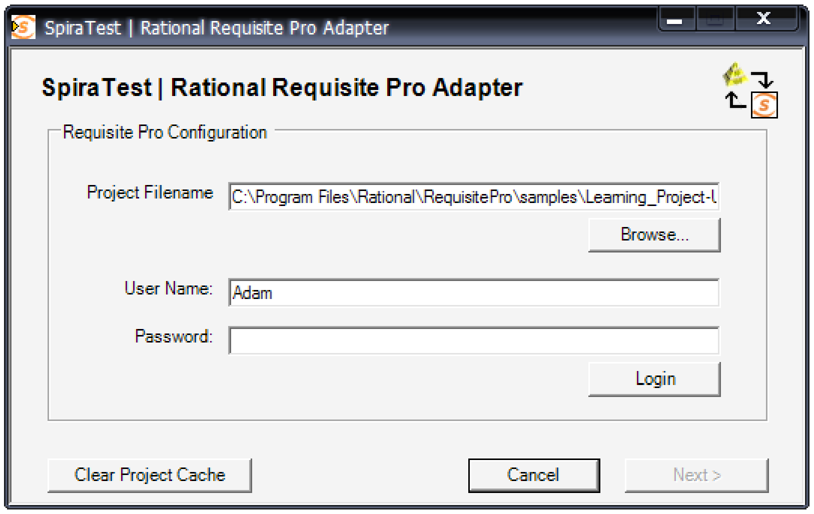
Install RequisitePro Adapter for yourself, or for anyone who uses this computer:

Everyone [✓ Just me]

Choose the folder to install to, and then decide whether the application should be accessible by all users on the workstation or just the current user. Then click the <Next> button to start the installation process. It will confirm if you want to proceed, click <Next> then wait for it to finish.

2.2. Importing From RequisitePro

Now that you have installed the integration adapter, you can launch it at any time by going to Start > Programs > SpiraTest > Tools > RequisitePro Adapter. This will launch the import application itself:



SpiraTest | Rational Requisite Pro Adapter

Requisite Pro Configuration

Project Filename: C:\Program Files\Rational\RequisitePro\samples\Learning_Project-L

User Name: Adam

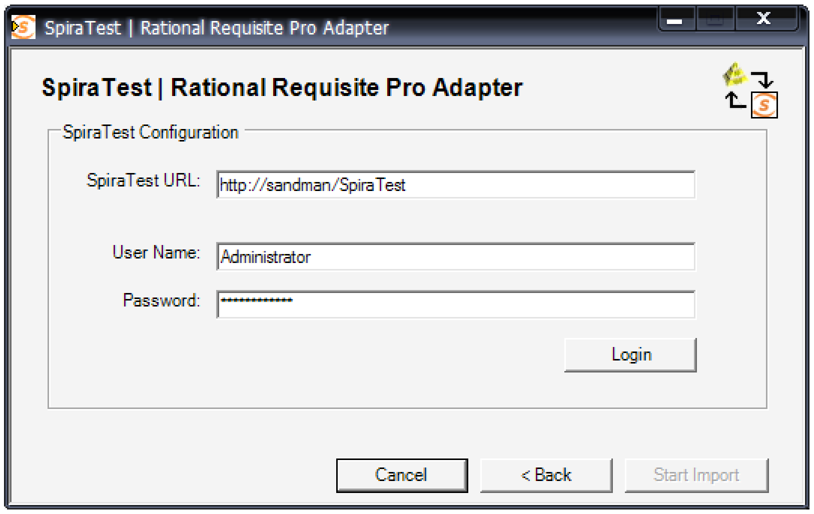
Password:

The first thing you need to do is to click the <Browse> button and select the Rational RequisitePro project file (.rqs) that you want to import from. You also need to select a valid username and password for that project. Once you have done this, click the <Login> button to verify that the project file can be opened.

Note that the importer has only been tested against version 7.0 of RequisitePro or later. It may not work correctly against previous versions.

The button marked <Clear Project Cache> will be explained later on in section 4.

Assuming that the user name selected has permission to access this project, you will be prompted with a message box indicating that the login was successful. Now click the <Next> button to move to the next page in the import wizard:



SpiraTest | Rational Requisite Pro Adapter

SpiraTest Configuration

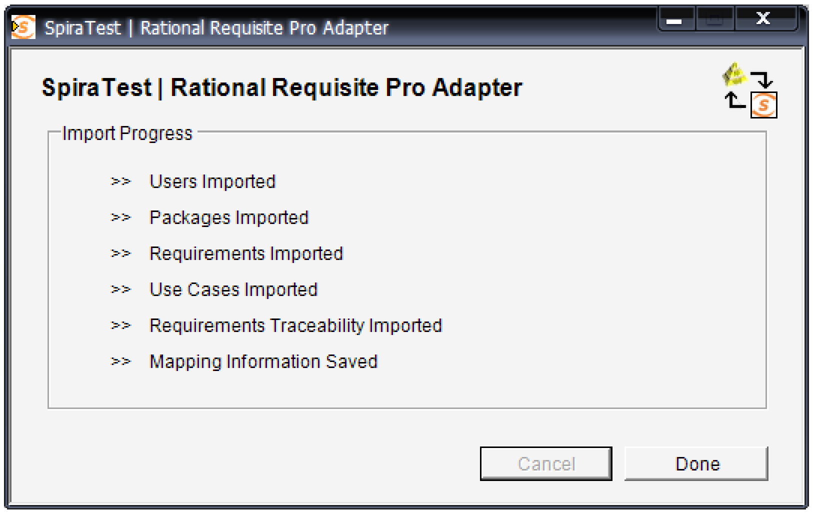
SpiraTest URL: http://sandman/SpiraTest

User Name: Administrator

Password: *************

This page allows you to enter the URL, user name and password that you want to use to access the instance of SpiraTest that you want to ***import to*** and click <Login>. Typically the URL is of the form (`http://<server name>/SpiraTest`). The version of the importer being used must be compatible with the version of SpiraTest you're importing into; if not you will receive an error message.

Assuming that the login was successful, click the <Start Import> button to actually begin the process of importing the various artifacts from RequisitePro into SpiraTest. Note that the first time the importer sees a particular project file, it will create a new project in SpiraTest to hold all the artifacts with the same name as that used in RequisitePro.


SpiraTest | Rational Requisite Pro Adapter
SpiraTest | Rational Requisite Pro Adapter
Import Progress
Cancel
Done

>> Users Imported
>> Packages Imported
>> Requirements Imported
>> Use Cases Imported
>> Requirements Traceability Imported
>> Mapping Information Saved

During the import process, as each of the types of artifact are imported, the progress display will change (as illustrated above). Once the import has finished, you will receive a message to that effect and the <Done> button will be enabled. Clicking this button closed the importer. You should now log into SpiraTest using the same user name and password that was used for the import to view the imported project.

2.3. Using RequisitePro w/ SpiraTest

Once you have completed this initial import, you will now have two systems that can be used together to manage your project's lifecycle. How they should be used together depends on which methodology you have been using in your RequisitePro project:

- **Traditional Mode** – In this mode, RequisitePro only contains product requirements and software requirements. These are both loaded into SpiraTest's requirements matrix and can be used as a starting point for developing the necessary test case coverage. In this mode, requirements are managed in RequisitePro and all other artifacts are managed in SpiraTest.
- **Use Cases Mode** – In this mode, RequisitePro contains features, supplementary requirements and use cases. The features and supplementary requirements are loaded into SpiraTest's requirements matrix, and the use cases are loaded into SpiraTest's test case list. Note that these use cases do not contain any test steps. In this mode, requirements and test cases are managed in RequisitePro, and test steps, test runs and incidents are managed in SpiraTest.

Regardless of the mode employed, you can manage the appropriate artifacts in RequisitePro and then periodically re-run the import application to update SpiraTest. The application will remember that the project was already used for an initial load and will simply update the requirements and/or test cases as appropriate as well as add any additional ones added.

Note that this update process does **not** delete any artifacts removed in RequisitePro and any changes to the requirement or use case hierarchies are not reflected. This allows you to change the organization of the artifacts in SpiraTest to make them easier to use without the changes being overwritten on the next import cycle.

Finally, should you want to start again and re-import a project from scratch that has already been imported once before, you should choose the <Clear Project Cache> button on the first screen which will remove all the stored history of all previously loaded projects. ***This option is irreversible and should be performed with care.***

3. Using SpiraTest with JIRA

This section outlines how to use SpiraTest in conjunction with the JIRA issue/bug tracking system. The built-in integration service allows the quality assurance team to manage their requirements and test cases in SpiraTest, execute test runs in SpiraTest, and then have the new incidents generated during the run be automatically loaded into JIRA. Once the incidents are loaded into JIRA as issues, the development team can then manage the lifecycle of these issues in JIRA, and have the status changes in JIRA be reflected back in SpiraTest.

3.1. Configuring the Integration Service

This section outlines how to configure the integration service to export incidents into JIRA and pick up subsequent status changes in JIRA and have them update SpiraTest. It assumes that you already have a working installation of SpiraTest v1.5.2 or later. If you have an earlier version of SpiraTest you will need to upgrade to at least v1.5.2 before trying to integrate with JIRA.

The steps that need to be performed to configure integration with JIRA are as follows:

- Download the latest JIRA Data-Sync plug-in for SpiraTest from our website
- Configure the service to point to the correct instance of JIRA
- Configure the data field mappings between SpiraTest and JIRA
- Start the service and verify data transfer

3.1.1. Download the JIRA Plug-In

Go to the Inflectra website and open up the page that lists the various downloads available for SpiraTest (http://www.inflectra.com/Products/Downloads.aspx). Listed on this page will be the JIRA Plug-In for SpiraTest. Right-click on this link and save the Zip compressed folder to the hard-drive of the server where SpiraTest is installed. Open up the compressed folder and extract the files to the following locations:

- Copy JiraDataSync.dll to C:\Program Files\SpiraTest\SpiraTestService
- Copy JiraDataSync.dll.config to C:\Program Files\SpiraTest\SpiraTestService
- Copy JiraDataSyncMappings.xml to C:\Program Files\SpiraTest\SpiraTestService

3.1.2. Configuring the Service

To configure the integration service, please open up the SpiraTestService.exe.config file located in C:\Program Files\SpiraTest\SpiraTestService with a text editor such as Notepad. Once open, it should look like:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
      <appSettings>
             <add key="SpiraTestService.PollingInterval" value="120000" />
             <add key="SpiraTestService.PlugIns" value="" />
      </appSettings>
</configuration>
```

In addition it may have a set of keys that are related to other data-synchronization plug-ins (for example the Bugzilla plug-in) – these can be ignored. The first thing you need to do is tell the SpiraTestService that it needs to use the JIRA plug-in. To do this you need to change the SpiraTestService.PlugIns key to point to the JIRA plug-In. To add the JIRA plug-in to the list of plug-ins that are executed you simply need to add the following information to the key – JiraDataSync|Inflectra.SpiraTest.PlugIns.JiraDataSync.DataSync. In the example above, to add the JIRA plug-in to the plug-in list you would change the file to:

```xml
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
       value="JiraDataSync|Inflectra.SpiraTest.PlugIns.JiraDataSync.DataSync"/>
  </appSettings>
</configuration>
```

Now that you have added the JIRA plug-in to the list of plug-ins that will be executed by the background service, you now need to copy the JIRA-specific configuration settings from JiraDataSync.dll.config to SpiraTestService.exe.config. Once you have done this, you should have a single SpiraTestService.exe.config configuration file that looks something like:

```xml
<?xml version="1.0" encoding="utf-8"?>
<configuration>
<configSections>
    <sectionGroup name="applicationSettings"
type="System.Configuration.ApplicationSettingsGroup, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" >
        <section name="Inflectra.SpiraTest.PlugIns.JiraDataSync.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
    </sectionGroup>
</configSections>
      <appSettings>
              <add key="SpiraTestService.PollingInterval" value="120000" />
              <add key="SpiraTestService.PlugIns"
                     value="JiraDataSync|Inflectra.SpiraTest.
                            PlugIns.JiraDataSync.DataSync" />
              <add key="JiraDataSync.StaticMappings.Path"
                     value="C:\Program Files\SpiraTest\SpiraTestService
                            \JiraDataSyncMappings.xml" />
              <add key="JiraDataSync.SpiraApi.Login" value="fredbloggs" />
              <add key="JiraDataSync.SpiraApi.Password" value="fredbloggs" />
              <add key="JiraDataSync.JiraApi.Login" value="fredbloggs" />
              <add key="JiraDataSync.JiraApi.Password" value="fredbloggs" />
              <add key="JiraDataSync.JiraApi.TimeBuffer" value="5" />
              <add key="JiraDataSync.JiraApi.TraceLogging" value="false" />
      </appSettings>
    <applicationSettings>
        <Inflectra.SpiraTest.PlugIns.JiraDataSync.Properties.Settings>
            <setting name="JiraDataSync_SpiraImport_Import" serializeAs="String">
                <value>http://localhost/SpiraTest/Services/Import.asmx</value>
            </setting>
            <setting name="JiraDataSync_JiraSoapService_JiraSoapServiceService"
                serializeAs="String">
                <value> http://[server-name]/rpc/soap/jirasoapservice-v2</value>
            </setting>
        </Inflectra.SpiraTest.PlugIns.JiraDataSync.Properties.Settings>
    </applicationSettings>
</configuration>
```

The sections that need to be changed are marked in yellow above. You need to enter the following information:

- The full URL to the instance of the JIRA SOAP API. It is typically of the form `http://<server name>/rpc/soap/jirasoapservice-v2`. Check the documentation that came with your installation of JIRA to confirm this.

- The full URL to your instance SpiraTest. It is typically of the form `http://<server name>/SpiraTest/Services/Import.asmx`. Make sure that you can browse to this URL on the server itself.
- The login name and password to your instance of SpiraTest. This user needs to have at least Manager permissions for project(s) you want to synchronize with JIRA.
- A valid login and password for the JIRA instance you want SpiraTest to synchronize with. This user needs to have permissions to create issues in the project.

Once you have made these changes, save the file and proceed to the next stage.

Important Note: Some of the settings have changed in SpiraTest v1.5. If you are upgrading from SpiraTest v1.4 or earlier, you will need to change the names of several of the keys from SpiraTestService to JiraDataSync and also add a new entry for SpiraTestService.PlugIns. These changes have been made to allow SpiraTest to support different plug-ins for integrating with different systems.

3.1.3. Configuring the Data Mapping

Next, you need to configure the data mapping between SpiraTest and JIRA. This allows the various incident types and status codes used in the two applications to be related to each other. This is important, as without a correct mapping, there is no way for the integration service to know that an “Enhancement” in SpiraTest is the same as a “New Feature” in JIRA (for example).

Please open up the `JiraDataSyncMappings.xml` file located in `C:\Program Files\SpiraTest\SpiraTestService` with a text editor such as Notepad. Once open, it should look something like:

```
<?xml version="1.0" encoding="utf-8" ?>
<staticmappings>
      <!-- This needs to be always a one-to-one mapping, no one-to-many -->
      <projectmappings>
             <mapping spiraid="1" externalid="IMP" annotation="Library Project" />
             <mapping spiraid="2" externalid="SMP" annotation="Sample Project One" />
      </projectmappings>

      <!-- This needs to be always a one-to-one mapping, no one-to-many -->
      <usermappings>
             <mapping spiraid="1" externalid="administrator" annotation="Administrator"
/>
             <mapping spiraid="2" externalid="fredbloggs" annotation="Fred Bloggs" />
             <mapping spiraid="3" externalid="joesmith" annotation="Joe Smith" />
      </usermappings>

      <!-- This needs to be always a one-to-one mapping, no one-to-many -->
      <releasemappings>
             <mapping spiraid="1" externalid="10000" annotation="LibSys Release 1" />
             <mapping spiraid="4" externalid="10001" annotation="LibSys Release 1.1" />
      </releasemappings>

      <!-- These map SpiraTest Incident Status to Jira Incident Status-->
      <incidentstatusmappings>
             <mapping spiraid="1" projectid="1" externalid="1" annotation="New" />
             <mapping spiraid="2" projectid="1" externalid="1" annotation="Open" />
             <mapping spiraid="3" projectid="1" externalid="3" annotation="Assigned" />
             <mapping spiraid="4" projectid="1" externalid="5" annotation="Fixed" />
             <mapping spiraid="5" projectid="1" externalid="6" annotation="Closed" />
             <mapping spiraid="6" projectid="1" externalid="1" annotation="Not
Reproducible"/>
             <mapping spiraid="7" projectid="1" externalid="1" annotation="Duplicate"/>
             <mapping spiraid="8" projectid="1" externalid="4" annotation="Reopen"/>
      </incidentstatusmappings>

      <!-- These map Jira Incident Status back to SpiraTest Incident Status-->
      <incidentstatusreversemappings>
              <mapping externalid="1" projectid="1" spiraid="2" annotation="Open" />
              <mapping externalid="3" projectid="1" spiraid="3" annotation="In Progress"
/>
              <mapping externalid="4" projectid="1" spiraid="8" annotation="Reopened"/>
              <mapping externalid="5" projectid="1" spiraid="4" annotation="Resolved" />
              <mapping externalid="6" projectid="1" spiraid="5" annotation="Closed" />
      </incidentstatusreversemappings>

      <!-- These map SpiraTest incident types to JIRA issue types -->
      <incidenttypemappings>
              <mapping spiraid="1" projectid="1" externalid="1" annotation="Incident" />
              <mapping spiraid="2" projectid="1" externalid="1" annotation="Bug" />
              <mapping spiraid="3" projectid="1" externalid="2" annotation="Enhancement"
/>
              <mapping spiraid="4" projectid="1" externalid="1" annotation="Issue" />
              <mapping spiraid="5" projectid="1" externalid="3" annotation="Training" />
              <mapping spiraid="6" projectid="1" externalid="3"
annotation="Limitation"/>
              <mapping spiraid="7" projectid="1" externalid="4" annotation="Change
Request"/>
              <mapping spiraid="8" projectid="1" externalid="3" annotation="Risk"/>
      </incidenttypemappings>

      <!-- These map JIRA issue types to SpiraTest incident types -->
      <incidenttypereversemappings>
              <mapping externalid="1" projectid="1" spiraid="2" annotation="Bug" />
              <mapping externalid="2" projectid="1" spiraid="3" annotation="New Feature"
/>
              <mapping externalid="3" projectid="1" spiraid="8" annotation="Task" />
              <mapping externalid="4" projectid="1" spiraid="7"
annotation="Improvement"/>
      </incidenttypereversemappings>

      <!-- These map SpiraTest Priorities to Jira Priorities-->
      <prioritymappings>
              <mapping spiraid="1" projectid="1" externalid="2" annotation="Critical" />
              <mapping spiraid="2" projectid="1" externalid="3" annotation="High" />
              <mapping spiraid="3" projectid="1" externalid="4" annotation="Medium" />
              <mapping spiraid="4" projectid="1" externalid="5" annotation="Low" />
      </prioritymappings>

      <!-- These map Jira Priorities to SpiraTest Priorities-->
      <priorityreversemappings>
              <mapping externalid="1" projectid="1" spiraid="1" annotation="Blocker" />
              <mapping externalid="2" projectid="1" spiraid="1" annotation="Critical" />
              <mapping externalid="3" projectid="1" spiraid="2" annotation="Major" />
              <mapping externalid="4" projectid="1" spiraid="3" annotation="Minor" />
              <mapping externalid="5" projectid="1" spiraid="4" annotation="Trivial" />
      </priorityreversemappings>

      <!-- These map SpiraTest custom property values to Jira component values-->
      <componentmappings>
              <mapping spiraid="20" externalid="10000" annotation="Libraries" />
              <mapping spiraid="21" externalid="10001" annotation="Authors" />
      </componentmappings>

      <!-- This maps a specific SpiraTest custom property to Jira's environment field-->
      <environmentmapping spiraid="1" />

      <!-- These map SpiraTest custom property values to Jira component values-->
      <componentmappings>
```

```
            <mapping spiraid="20" externalid="10000" annotation="Libraries" />
            <mapping spiraid="21" externalid="10001" annotation="Authors" />
      </componentmappings>

      <!-- These map SpiraTest list custom properties to Jira custom fields-->
      <custompropertymappings>
            <mapping spiraid="2" projectid="1" externalid="10000"
annotation="Operating System" />
      </custompropertymappings>

      <!-- These map SpiraTest custom property values to Jira custom field values-->
      <custompropertyvaluemappings>
            <mapping spiraid="9" externalid="Windows 2003" annotation="Windows 2003"
/>
            <mapping spiraid="10" externalid="Windows Vista" annotation="Windows
Vista" />
            <mapping spiraid="11" externalid="Windows XP" annotation="Windows XP" />
            <mapping spiraid="12" externalid="Windows 2000" annotation="Windows 2000"
/>
      </custompropertyvaluemappings>

      <!-- These map SpiraTest text custom properties to Jira custom fields-->
      <custompropertytextmappings>
            <mapping spiraid="2" projectid="1" externalid="10010" annotation="Text02 -
Notes" />
      </custompropertytextmappings>

</staticmappings>
```

Since SpiraTest v1.4 and later allows you to customize the various priorties, statuses and types on a per-project basis, you need to specify the projectid that the mapping relates to in each line of the mapping. This is a new requirement in v1.4, so if you have upgraded from a version of SpiraTest that predates this then you will need to add the projectid="x" attributes in to each of your mapping entries.

If you have upgraded to v1.5 from a previous version, the name of the file has changed from StaticMappings.xml to JiraDataSyncMappings.xml, and there is now an optional section for synchronizing the release numbers in the two systems.

The three sections of the file that have to be changed in any installation are the project, user and release/version mappings. These are described below:

- In the **project mappings** section you need to list a mapping entry for any project that you want to synchronize between SpiraTest and JIRA. The project must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the project in SpiraTest, and the **externalid** which should be the three-letter prefix you assigned to the project in JIRA.

  The **annotation** field is an optional entry that can be used in all of the mappings, it is not used by the integration service, but is instead used to help document what the mapping relates to. In this instance you might use it to list the name of the project being mapped.

- In the **user mappings** section you need to list a mapping entry for any user that can have issues assigned to him/her in SpiraTest and JIRA. The user must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the user in SpiraTest, and the **externalid** which should be the username you assigned to the user in JIRA.

- In the **release mappings** section you need to list a mapping entry for any release that can have issues assigned to it in SpiraTest and JIRA. The release must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the release in SpiraTest, and the **externalid** which should be the id of the equivalent “version” in JIRA. The ID can be found by looking at the URL inside JIRA which choosing to View/Edit the version name/description.

Next you need to synchronize the various incident types, incident priorities and incident statuses between SpiraTest and JIRA. Without establishing that a type of bug is ID 2 in SpiraTest and ID 1 in JIRA, there is no way for the synchronization service to be able to reconsile any changes made to the incident/issue. Since SpiraTest allows you to have different values for these entity for each project, you will need to have entries in the mapping sections for all the projects listed at the top of the XML document.

- **Incident Statuses**
  There are two mapping sections in the XML document, the first one – *incidentstatusmappings* – relates each SpiraTest incident status ID to the corresponding one in JIRA. The second one – *incidentstatusreversemappings* – relates each JIRA issue status ID back to the corresponding one in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of statuses in each SpiraTest project and match them against the list of statuses in each JIRA project. Then for each status, you need to ensure there is a mapping entry in the list that relates the SpiraTest incident status to the JIRA issue status, and one that relates the JIRA issue status back to the corresponding SpiraTest incident status.

- **Incident Types**
  There are two mapping sections in the XML document, the first one – *incidentypemappings* – relates each SpiraTest incident type ID to the corresponding one in JIRA. The second one – *incidenttypereversemappings* – relates each JIRA issue type ID back to the corresponding one in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of incident types in each SpiraTest project and match them against the list of issue types in each JIRA project. Then for each type, you need to ensure there is a mapping entry in the list that relates the SpiraTest incident type to the JIRA issue type, and one that relates the JIRA issue type back to the corresponding SpiraTest incident type.

- **Incident Priorities**
  There are two mapping sections in the XML document, the first one – *prioritymappings* – relates each SpiraTest incident priority ID to the corresponding one in JIRA. The second one – *priorityreversemappings* – relates each JIRA issue priority ID back to the corresponding one in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of incident priorities in each SpiraTest project and match them against the list of issue priorities in each JIRA project. Then for each priority, you need to ensure there is a mapping entry in the list that relates the SpiraTest incident priority to the JIRA issue priority, and one that relates the JIRA issue priority back to the corresponding SpiraTest incident priority.

- **JIRA Components**
  If your instance of JIRA requires that all new issues are submitted with a ‘Component’ then you will need to fill out this section. You first need to create an incident custom property in SpiraTest of type ‘LIST’ that contains the various component names that exist

inside JIRA. Once you have that list, this mapping section relates each Custom Property Value (e.g. PV00001) from SpiraTest to the equivalent component ID in JIRA. The spiraid of this mapping entry is the ID of the custom property value with the 'PV' prefix removed. The external ID can be found by looking at the URL inside JIRA which choosing to View/Edit the component name/description The name of the custom property inside SpiraTest doesn't matter as the synchronization service only looks for the custom property value id.
Note: In the mappings, you can specify the default JIRA component to use if the user doesn't provide one in SpiraTest. This is denoted with spiraid="-1".

- **JIRA Environment Description**
  If your instance of JIRA requires that all new issues are submitted with an 'Environment' description specified, then you will need to fill out this section. You first need to create an incident custom property in SpiraTest of type 'TEXT' that will be used to store the environment description within SpiraTest. Then you need to add the <environmentmapping> section to the mapping file, with a single value 'spiraid' that points to the number of the text property being used to hold the environment description. E.g. if you have assigned custom property Text03 to store environment, you would need an environment mapping with spiraid="3".

- **Custom List Properties**
  There are two mapping sections in the XML document that relate to custom list properties. The first one – *custompropertymappings* – maps each of the custom property list fields used in a particular SpiraTest project to the equivalent custom field in JIRA. The SpiraId should be the list field number of the custom property. E.g. if you have assigned Operating System as the alias for SpiraTest custom property List02 then the SpiraId should be set to "2". The ExternalId is the equivalent ID of the custom property in JIRA. It can be found by examining the URL inside JIRA. The second section – *custompropertyvaluemappings* – relates each custom property value in SpiraTest with the equivalent value in SpiraTest. The SpiraId can be found from the Administration > Edit Property Values screen, and the ExternalId is the full name of the equivalent value inside JIRA.

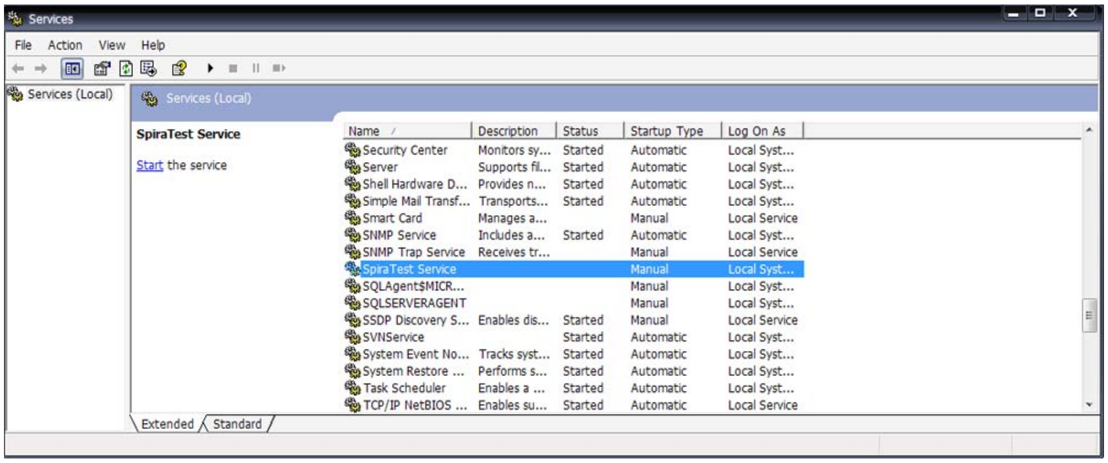
- **Custom Text Properties**
  If you need to have text custom properties in SpiraTest be submitted with a new incident and get synchronized over to JIRA, then you will need to fill out this section. The section – *custompropertytextmappings* - maps each of the custom property text fields used in a particular SpiraTest project to the equivalent custom field in JIRA. The SpiraId should be the text field number of the custom property. E.g. if you have assigned Notes as the alias for SpiraTest custom property Text02 then the SpiraId should be set to "2". The ExternalId is the equivalent ID of the custom property in JIRA. It can be found by examining the URL inside JIRA.

Once you have updated the various mapping sections, you are now ready to start the service.

3.1.4. Starting the Service

When SpiraTest is installed, a Windows Service – SpiraTestService – is installed along with the web application. However to avoid wasting system resources, this service is initially set to run manually. To ensure continued synchronization of SpiraTest with JIRA, we recommend starting the service and setting its startup-type to Automatic.

To make these changes, open up the Windows Control Panel, click on the “Administrative Tools” link, and then choose the Services option. This will bring up the Windows Service control panel:



SpiraTest Service

Start the service

Name	Description	Status	Startup Type	Log On As
Security Center	Monitors sy...	Started	Automatic	Local Syst...
Server	Supports fil...	Started	Automatic	Local Syst...
Shell Hardware D...	Provides n...	Started	Automatic	Local Syst...
Simple Mail Transf...	Transports...	Started	Automatic	Local Syst...
Smart Card	Manages a...		Manual	Local Service
SNMP Service	Includes a...	Started	Automatic	Local Syst...
SNMP Trap Service	Receives tr...		Manual	Local Service
SpiraTest Service			Manual	Local Syst...
SQLAgent$MICR...			Manual	Local Syst...
SQLSERVERAGENT			Manual	Local Syst...
SSDP Discovery S...	Enables dis...	Started	Manual	Local Service
SVNService		Started	Automatic	Local Syst...
System Event No...	Tracks syst...	Started	Automatic	Local Syst...
System Restore ...	Performs s...	Started	Automatic	Local Syst...
Task Scheduler	Enables a ...	Started	Automatic	Local Syst...
TCP/IP NetBIOS ...	Enables su...	Started	Automatic	Local Service

Click on the ‘SpiraTestService’ entry and click on the link to start the service. Then right-click the service entry and choose the option to set the startup type to ‘Automatic’. This will ensure that synchronization continues between SpiraTest and JIRA after a reboot of the server.

3.2. Using SpiraTest with JIRA

Now that the integration service has been configured and the service started, initially any incidents created in SpiraTest for the specified projects will be imported into JIRA. At this point we recommend opening the Windows Event Viewer and choosing the Application Log. In this log any error messages raised by the SpiraTest service will be displayed. If you see any error messages at this point, we recommend immediately stopping the SpiraTest service and checking the various mapping entries. If you cannot see any issues with the mapping information, we recommend sending a copy of the event log message together with your mapping XML file to Inflectra customer services (support@inflectra.com) who will help you troubleshoot the problem.

To use SpiraTest with JIRA on an ongoing basis, we recommend the following general processes be followed:

- When running tests in SpiraTest, defects found should be logged through the ‘Add Incident’ option as normal.
- Once an incident has been created during the running of the test, it will now be populated across into JIRA as an issue. It will be given the generic ‘Issue’ type.
- At this point, the incident should not be acted upon inside SpiraTest, and all data changes to the issue should be made inside JIRA. To enforce this, you should modify the workflows set up in SpiraTest so that the various fields are marked as inactive for all the incident statuses other than the “New” status. This will allow someone to submit an incident in SpiraTest, but will prevent them making changes in conflict with JIRA after that point.

- As the issue progresses through the customized JIRA workflow, changes to the type of issue, changes to its status, priority, description and resolution will be updated automatically in SpiraTest. In essence, SpiraTest acts as a read-only viewer of these incidents.
- You are now able to perform test coverage and incident reporting inside SpiraTest using the test cases managed by SpiraTest and the incidents managed on behalf of SpiraTest inside JIRA.

4. Using SpiraTest with Bugzilla

This section outlines how to use SpiraTest in conjunction with the open-source Bugzilla bug tracking system. The built-in integration service allows the quality assurance team to manage their requirements and test cases in SpiraTest, execute test runs in SpiraTest, and then have the new incidents generated during the run be automatically loaded into Bugzilla. Once the incidents are loaded into Bugzilla as bugs, the development team can then manage the lifecycle of these bugs in Bugzilla, and have the status changes in Bugzilla be reflected back in SpiraTest.

4.1. Configuring the Integration Service

This section outlines how to configure the integration service to export incidents into Bugzilla and pick up subsequent status changes in Bugzilla and have them update SpiraTest. It assumes that you already have a working installation of SpiraTest v1.5.2 or later and Bugzilla v3.0 or later. If you have earlier versions of SpiraTest or Bugzilla you will need to upgrade to at least v1.5.2 and v3.0 respectively.

The steps that need to be performed to configure integration with Bugzilla are as follows:

- Download the Bugzilla Data-Sync plug-in for SpiraTest from our website
- Configure the service to point to the correct instance of Bugzilla
- Configure the data field mappings between SpiraTest and Bugzilla
- Start the service and verify data transfer

4.1.1. Download the Bugzilla Plug-In

Go to the Inflectra website and open up the page that lists the various downloads available for SpiraTest (http://www.inflectra.com/Products/Downloads.aspx). Listed on this page will be the Bugzilla Plug-In for SpiraTest. Right-click on this link and save the Zip compressed folder to the hard-drive of the server where SpiraTest is installed. Open up the compressed folder and extract the files to the following locations:

- Copy BugzillaDataSync.dll to C:\Program Files\SpiraTest\SpiraTestService
- Copy BugzillaDataSync.dll.config to C:\Program Files\SpiraTest\SpiraTestService
- Copy BugzillaDataSyncMappings.xml to C:\Program Files\SpiraTest\SpiraTestService

4.1.2. Configuring the Service

To configure the integration service, please open up the SpiraTestService.exe.config file located in C:\Program Files\SpiraTest\SpiraTestService with a text editor such as Notepad. Once open, it should look like:

```xml
<?xml version="1.0" encoding="utf-8"?>
<configuration>
      <appSettings>
             <add key="SpiraTestService.PollingInterval" value="120000" />
             <add key="SpiraTestService.PlugIns"
                    value="JiraDataSync|Inflectra.SpiraTest.
                           PlugIns.JiraDataSync.DataSync" />
      </appSettings>
</configuration>
```

In addition it may have a set of keys that are related to other data-synchronization plug-ins (for example the JiraDataSync which is installed by default) – these can be ignored. The first thing you need to do is tell the SpiraTestService that it needs to use the Bugzilla plug-in. To do this you need to change the SpiraTestService.PlugIns key to point to the Bugzilla plug-In. To add the Bugzilla plug-in to the list of plug-ins that are executed you simply need to add the following information to the key – BugzillaDataSync|Inflectra.SpiraTest.PlugIns.BugzillaDataSync.DataSync. In the example above, to add the Bugzilla plug-in to the existing plug-in list you would change the file to:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
         value="JiraDataSync|Inflectra.SpiraTest.
                PlugIns.JiraDataSync.DataSync;BugzillaDataSync
               |Inflectra.SpiraTest.PlugIns.BugzillaDataSync.DataSync" />
  </appSettings>
</configuration>
```

However, if you actually want to remove the entry for other plug-ins (e.g. in this example you don't want to synchronize some projects with JIRA and some with Bugzilla, then you should remove the other entries:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
       value="BugzillaDataSync|Inflectra.SpiraTest.PlugIns.BugzillaDataSync.DataSync"/>
  </appSettings>
</configuration>
```

Now that you have added the Bugzilla plug-in to the list of plug-ins that will be executed by the background service, you now need to copy the Bugzilla-specific configuration settings from BugzillaDataSync.dll.config to SpiraTestService.exe.config. Once you have done this, you should have a single SpiraTestService.exe.config configuration file that looks something like:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <configSections>
     <sectionGroup name="applicationSettings"
type="System.Configuration.ApplicationSettingsGroup, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" >
        <section name="Inflectra.SpiraTest.PlugIns.BugzillaDataSync.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
     </sectionGroup>
  </configSections>
<applicationSettings>
  <Inflectra.SpiraTest.PlugIns.BugzillaDataSync.Properties.Settings>
   <setting name="BugzillaDataSync_SpiraImport_Import" serializeAs="String">
    <value>http://localhost/SpiraTest/Services/Import.asmx</value>
   </setting>
  </Inflectra.SpiraTest.PlugIns.BugzillaDataSync.Properties.Settings>
 </applicationSettings>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
       value="BugzillaDataSync|Inflectra.SpiraTest.PlugIns.BugzillaDataSync.DataSync"/>
    <add key="BugzillaDataSync.Bugzilla.Url"
         value="http://landfill.bugzilla.org/bugzilla-3.0-branch/xmlrpc.cgi" />
    <add key="BugzillaDataSync.StaticMappings.Path"
      value="C:\Program Files\SpiraTest\SpiraTestService\BugzillaDataSyncMappings.xml"/>
    <add key="BugzillaDataSync.SpiraApi.Login" value="fredbloggs" />
    <add key="BugzillaDataSync.SpiraApi.Password" value="fredbloggs" />
    <add key="BugzillaDataSync.BugzillaApi.Login" value="bugzillauser@spiratest.com" />
    <add key="BugzillaDataSync.BugzillaApi.Password" value="bugzillapassword" />
  </appSettings>
</configuration>
```

The sections that need to be changed are marked in yellow above. You need to enter the following information:

- The full URL to the instance of the Bugzilla XMP-RPC API. It is typically of the form http://MyServer/BugzillaPath/xmlrpc.cgi. Check the documentation that came with your installation of Bugzilla (or http://bugzilla.org) to confirm this.
- The full URL to your instance SpiraTest. It is typically of the form `http://<server name>/SpiraTest/Services/Import.asmx`. Make sure that you can browse to this URL on the server itself.
- The login name and password to your instance of SpiraTest. This user needs to have at least Manager permissions for project(s) you want to synchronize with Bugzilla.
- A valid login and password for the Bugzilla instance you want SpiraTest to synchronize with. This user needs to have permissions to create bugs in the project.

Once you have made these changes, save the file and proceed to the next stage.

4.1.3. Configuring the Data Mapping

Next, you need to configure the data mapping between SpiraTest and Bugzilla. This allows the various incident types and status codes used in the two applications to be related to each other. This is important, as without a correct mapping, there is no way for the integration service to know that a "P1 - Critical" incident in SpiraTest is the same as a "P1" bug in Bugzilla (for example).

Please open up the BugzillaDataSyncMappings.xml file located in C:\Program Files\SpiraTest\SpiraTestService with a text editor such as Notepad. Once open, it should look something like:

```
<?xml version="1.0" encoding="utf-8" ?>
<staticmappings>
       <!-- This needs to be always a one-to-one mapping, no one-to-many -->
       <projectmappings>
              <mapping spiraid="1" externalid="deleteable" annotation="Library Sample
Project" />
       </projectmappings>

       <!-- This needs to be always a one-to-one mapping, no one-to-many -->
       <usermappings>
              <mapping spiraid="1" externalid="administrator"
annotation="administrator@spiratest.com" />
              <mapping spiraid="2" externalid="fredbloggs"
annotation="fredbloggs@spiratest.com" />
              <mapping spiraid="3" externalid="joesmith"
annotation="joesmith@spiratest.com" />
       </usermappings>

       <!-- This needs to be always a one-to-one mapping, no one-to-many -->
       <releasemappings>
              <mapping spiraid="1" externalid="Version 1.0" annotation="Library System
Release 1" />
              <mapping spiraid="2" externalid="Version 1.1" annotation="Library System
Release 1.1" />
       </releasemappings>

       <!-- These map SpiraTest Incident Status to Bugzilla Bug Status-->
       <incidentstatusmappings>
              <mapping spiraid="1" projectid="1" externalid="NEW" annotation="New" />
              <mapping spiraid="2" projectid="1" externalid="NEW" annotation="Open" />
```

```
                <mapping spiraid="3" projectid="1" externalid="ASSIGNED"
annotation="Assigned" />
                <mapping spiraid="4" projectid="1" externalid="RESOLVED"
annotation="Resolved" />
                <mapping spiraid="5" projectid="1" externalid="CLOSED" annotation="Closed"
/>
                <mapping spiraid="6" projectid="1" externalid="UNCONFIRMED"
annotation="Not Reproducible"/>
                <mapping spiraid="7" projectid="1" externalid="UNCONFIRMED"
annotation="Duplicate"/>
                <mapping spiraid="8" projectid="1" externalid="REOPENED"
annotation="Reopen"/>
        </incidentstatusmappings>

        <!-- These map Bugzilla Bug Status back to SpiraTest Incident Status-->
        <incidentstatusreversemappings>
                <mapping externalid="NEW"                    projectid="1" spiraid="2"
annotation="New" />
                <mapping externalid="UNCONFIRMED"     projectid="1" spiraid="6"
annotation="Unconfirmed" />
                <mapping externalid="ASSIGNED"               projectid="1" spiraid="3"
annotation="Assigned"/>
                <mapping externalid="REOPENED"               projectid="1" spiraid="8"
annotation="Reopened" />
                <mapping externalid="RESOLVED"               projectid="1" spiraid="4"
annotation="Resolved" />
                <mapping externalid="VERIFIED"               projectid="1" spiraid="4"
annotation="Verified" />
                <mapping externalid="CLOSED"         projectid="1" spiraid="5"
annotation="Closed" />
        </incidentstatusreversemappings>

        <!-- These map SpiraTest Priorities to Bugzilla Priorities-->
        <prioritymappings>
                <mapping spiraid="-1" projectid="1" externalid="P3" annotation="Default"
/>
                <mapping spiraid="1" projectid="1" externalid="P1" annotation="Critical"
/>
                <mapping spiraid="2" projectid="1" externalid="P2" annotation="High" />
                <mapping spiraid="3" projectid="1" externalid="P3" annotation="Medium" />
                <mapping spiraid="4" projectid="1" externalid="P4" annotation="Low" />

        </prioritymappings>

        <!-- These map Bugzilla Priorities to SpiraTest Priorities-->
        <priorityreversemappings>
                <mapping externalid="P1" projectid="1" spiraid="1" annotation="P1" />
                <mapping externalid="P2" projectid="1" spiraid="2" annotation="P2" />
                <mapping externalid="P3" pæojectid="1" spiraid="3" annotation="P3" />
                <mapping externalid="P4" pæojectid="1" spiraid="4" annotation="P4" />
                <mapping externalid="P5" pæojectid="1" spiraid="5" annotation="P5" />
        </pæiorityreversemappings>

        <!-- These map SpiraTest Severities to Bugzilla Severities-->
        <severitymappings>
                <mapping spiraid="-1" pæojectid="1" externalid="normal"
annotation="Default" />
                <mapping spiraid="1" pæojectid="1" externalid="critical"
annotation="Critical" />
                <mapping spiraid="2" pæojectid="1" externalid="major" annotation="High" />
                <mapping spiraid="3" pæojectid="1" externalid="normal" annotation="Medium"
/>
                <mapping spiraid="4" pæojectid="1" externalid="minor" annotation="Low" />
        </severitymappings>

        <!-- These map Bugzilla Severities to SpiraTest Severities-->
        <severityreversemappings>
                <mapping externalid="blocker"        pæojectid="1" spiraid="1"
annotation="blocker" />
                <mapping externalid="critical"               pæojectid="1" spiraid="1"
annotation="critical" />
```

```
                <mapping externalid="major"                  pæojectid="1" spiraid="2"
annotation="major" />
                <mapping externalid="normal"        pæojectid="1" spiraid="3"
annotation="normal" />
                <mapping externalid="minor"                  pæojectid="1" spiraid="4"
annotation="minor" />
                <mapping externalid="trivial"       pæojectid="1" spiraid="4"
annotation="trivial" />
                <mapping externalid="enhancement"   pæojectid="1" spiraid="4"
annotation="enhancement" />
        </severityreversemappings>

        <!-- These map SpiraTest custom property values to Bugzilla components -->
        <componentmappings>
                <mapping spiraid="-1" externalid="Component 1" annotation="Default" />
                <mapping spiraid="26" externalid="Component 1" annotation="Component for
SpiraTest Project 1" />
        </componentmappings>

        <!-- These map SpiraTest custom property values to Bugzilla operating systems -->
        <operatingsystemmappings>
                <mapping spiraid="-1" externalid="Windows XP" annotation="Default" />
                <mapping spiraid="9" externalid="Windows Server 2003" annotation="Windows
2003" />
                <mapping spiraid="10" externalid="Windows Vista" annotation="Windows
Vista" />
                <mapping spiraid="11" externalid="Windows XP" annotation="Windows XP" />
                <mapping spiraid="12" externalid="Windows 2000" annotation="Windows 2000"
/>
                <mapping spiraid="13" externalid="Windows NT" annotation="Windows NT 4.0"
/>
        </operatingsystemmappings>

        <!-- These map SpiraTest custom property values to Bugzilla hardware values -->
        <hardwaremappings>
                <mapping spiraid="-1" externalid="HP" annotation="Default" />
                <mapping spiraid="27" externalid="HP" annotation="HP" />
                <mapping spiraid="28" externalid="Sun" annotation="Sun" />
        </hardwaremappings>

</staticmappings>
```

The three sections of the file that have to be changed in any installation are the project, user, and release/version mappings. These are described below:

- In the **project mappings** section you need to list a mapping entry for any project that you want to synchronize between SpiraTest and Bugzilla. The project must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the project in SpiraTest, and the **externalid** which should be the name of the **product** you assigned to the project in Bugzilla.

  The **annotation** field is an optional entry that can be used in all of the mappings, it is not used by the integration service, but is instead used to help document what the mapping relates to. In this instance you might use it to list the name of the project being mapped.

- In the **user mappings** section you need to list a mapping entry for any user that can have issues assigned to him/her in SpiraTest and Bugzilla. The user must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the user in SpiraTest, and the **externalid** which should be the username you assigned to the user in Bugzilla.

- In the **release mappings** section you need to list a mapping entry for any release that can have issues assigned to it in SpiraTest and Bugzilla. The release must be already created in both systems *before running the service*. The mapping entry consists of the

spiraid which should contain the ID of the release in SpiraTest, and the **externalid** which should be the name of the equivalent “version” in Bugzilla.

Next you need to synchronize the various incident severities, priorities and statuses between SpiraTest and Bugzilla. Without this there is no way for the synchronization service to be able to reconsile any changes made to the incident/bug. Since SpiraTest allows you to have different values for these entity for each project, you will need to have entries in the mapping sections for all the projects listed at the top of the XML document.

- **Incident Statuses**
  There are two mapping sections in the XML document, the first one – *incidentstatusmappings* – relates each SpiraTest incident status ID to the corresponding name in Bugzilla. The second one – *incidentstatusreversemappings* – relates each Bugzilla issue status name back to the corresponding ID in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of statuses in each SpiraTest project and match them against the list of statuses in each Bugzilla product. Then for each status, you need to ensure there is a mapping entry in the list that relates the SpiraTest incident status to the Bugzilla bug status, and one that relates the Bugzilla bug status back to the corresponding SpiraTest incident status.

- **Incident Severities**
  There are two mapping sections in the XML document, the first one – *severitymappings* – relates each SpiraTest incident severity ID to the corresponding name in Bugzilla. The second one – *severityreversemappings* – relates each Bugzilla severity name back to the corresponding ID in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of incident severities in each SpiraTest project and match them against the list of severities in each Bugzilla product. Then for each severity, you need to ensure there is a mapping entry in the list that relates the SpiraTest severity ID to the Bugzilla severity name, and one that relates the Bugzilla severity name back to the corresponding SpiraTest severity ID.
  Note: In the forward mappings, you can specify the default Bugzilla severity to use if the user doesn’t provide one in SpiraTest. This is denoted with spiraid=”-1”.

- **Incident Priorities**
  There are two mapping sections in the XML document, the first one – *prioritymappings* – relates each SpiraTest incident priority ID to the corresponding name in Bugzilla. The second one – *priorityreversemappings* – relates each Bugzilla priority name back to the corresponding ID in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of incident priorities in each SpiraTest project and match them against the list of bug priorities in each Bugzilla product. Then for each priority, you need to ensure there is a mapping entry in the list that relates the SpiraTest priority ID to the Bugzilla priority name, and one that relates the Bugzilla priority name back to the corresponding SpiraTest priority ID.
  Note: In the forward mappings, you can specify the default Bugzilla priority to use if the user doesn’t provide one in SpiraTest. This is denoted with spiraid=”-1”.

- **Bugzilla Components**
  If your instance of Bugzilla requires that all new bugs are submitted with a ‘Component’ then you will need to fill out this section. You first need to create an incident custom property in SpiraTest of type ‘LIST’ that contains the various component names that exist

inside Bugzilla. Once you have that list, this mapping section relates each Custom Property Value (e.g. PV00001) from SpiraTest to the equivalent component name in Bugzilla. The spiraid of this mapping entry is the ID of the custom property value with the 'PV' prefix removed. The name of the custom property inside SpiraTest doesn't matter as the synchronization service only looks for the custom property value id.
Note: In the mappings, you can specify the default Bugzilla component to use if the user doesn't provide one in SpiraTest. This is denoted with spiraid="-1".

- **Bugzilla Operating Systems**
  If your instance of Bugzilla requires that all new bugs are submitted with an 'Operating System' value then you will need to fill out this section. You first need to create an incident custom property in SpiraTest of type 'LIST' that contains the various operating system names that exist inside Bugzilla. Once you have that list, this mapping section relates each Custom Property Value (e.g. PV00001) from SpiraTest to the equivalent operating system name in Bugzilla. The spiraid of this mapping entry is the ID of the custom property value with the 'PV' prefix removed. The name of the custom property inside SpiraTest doesn't matter as the synchronization service only looks for the custom property value id.
  Note: In the mappings, you can specify the default Bugzilla operating system to use if the user doesn't provide one in SpiraTest. This is denoted with spiraid="-1".

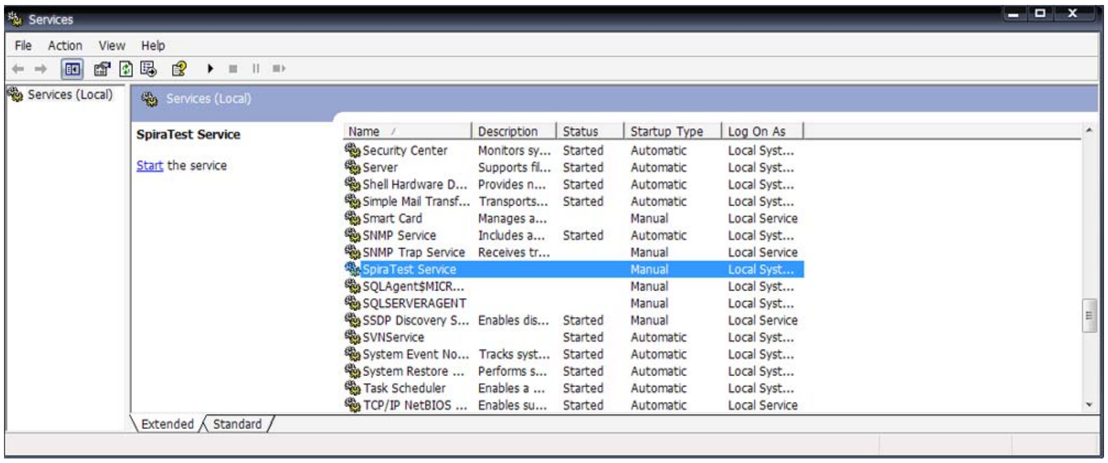
- **Bugzilla Hardware Platforms**.
  If your instance of Bugzilla requires that all new bugs are submitted with a 'Hardware Platform' value then you will need to fill out this section. You first need to create an incident custom property in SpiraTest of type 'LIST' that contains the various hardware platform names that exist inside Bugzilla. Once you have that list, this mapping section relates each Custom Property Value (e.g. PV00001) from SpiraTest to the equivalent hardware platform name in Bugzilla. The spiraid of this mapping entry is the ID of the custom property value with the 'PV' prefix removed. The name of the custom property inside SpiraTest doesn't matter as the synchronization service only looks for the custom property value id.
  Note: In the mappings, you can specify the default Bugzilla hardware to use if the user doesn't provide one in SpiraTest. This is denoted with spiraid="-1".

Once you have updated the various mapping sections, you are now ready to start the service.

4.1.4. Starting the Service

When SpiraTest is installed, a Windows Service – SpiraTestService – is installed along with the web application. However to avoid wasting system resources, this service is initially set to run manually. To ensure continued synchronization of SpiraTest with Bugzilla, we recommend starting the service and setting its startup-type to Automatic.

To make these changes, open up the Windows Control Panel, click on the “Administrative Tools” link, and then choose the Services option. This will bring up the Windows Service control panel:



Name	Description	Status	Startup Type	Log On As
Security Center	Monitors sy...	Started	Automatic	Local Syst...
Server	Supports fil...	Started	Automatic	Local Syst...
Shell Hardware D...	Provides n...	Started	Automatic	Local Syst...
Simple Mail Transf...	Transports...	Started	Automatic	Local Syst...
Smart Card	Manages a...		Manual	Local Service
SNMP Service	Includes a...	Started	Automatic	Local Syst...
SNMP Trap Service	Receives tr...		Manual	Local Service
SpiraTest Service			Manual	Local Syst...
SQLAgent$MICR...			Manual	Local Syst...
SQLSERVERAGENT			Manual	Local Syst...
SSDP Discovery S...	Enables dis...	Started	Manual	Local Service
SVNService		Started	Automatic	Local Syst...
System Event No...	Tracks syst...	Started	Automatic	Local Syst...
System Restore ...	Performs s...	Started	Automatic	Local Syst...
Task Scheduler	Enables a ...	Started	Automatic	Local Syst...
TCP/IP NetBIOS ...	Enables su...	Started	Automatic	Local Service

Click on the ‘SpiraTestService’ entry and click on the link to start the service. Then right-click the service entry and choose the option to set the startup type to ‘Automatic’. This will ensure that synchronization continues between SpiraTest and Bugzilla after a reboot of the server.

4.2. Using SpiraTest with Bugzilla

Now that the integration service has been configured and the service started, initially any incidents created in SpiraTest for the specified projects will be imported into Bugzilla. At this point we recommend opening the Windows Event Viewer and choosing the Application Log. In this log any error messages raised by the SpiraTest service will be displayed. If you see any error messages at this point, we recommend immediately stopping the SpiraTest service and checking the various mapping entries. If you cannot see any issues with the mapping information, we recommend sending a copy of the event log message together with your mapping XML file to Inflectra customer services (support@inflectra.com) who will help you troubleshoot the problem.

To use SpiraTest with Bugzilla on an ongoing basis, we recommend the following general processes be followed:

- When running tests in SpiraTest, defects found should be logged through the ‘Add Incident’ option as normal.
- Once an incident has been created during the running of the test, it will now be populated across into Bugzilla as a bug. It will be populated with the information captured in SpiraTest.
- At this point, the incident should not be acted upon inside SpiraTest, and all data changes to the issue should be made inside Bugzilla. To enforce this, you can modify the workflows set up in SpiraTest so that the various fields are marked as inactive for all the incident statuses other than the “New” status. This will allow someone to submit an incident in SpiraTest, but will prevent them making changes in conflict with Bugzilla after that point.

- As the issue progresses through the Bugzilla workflow, changes to the status, priority, severity, and resolution will be updated automatically in SpiraTest. In essence, SpiraTest acts as a read-only viewer of these incidents.
- You are now able to perform test coverage and incident reporting inside SpiraTest using the test cases managed by SpiraTest and the incidents managed on behalf of SpiraTest inside Bugzilla.

5. Using SpiraTest with Microsoft Team Foundation Server

This section outlines how to use SpiraTest in conjunction with the Microsoft's Visual Studio Team Foundation Server work item tracking system. The built-in integration service allows the quality assurance team to manage their requirements and test cases in SpiraTest, execute test runs in SpiraTest, and then have the new incidents generated during the run be automatically loaded into Microsoft Team Foundation Server (MSTFS) as new work items. Once the incidents are loaded into MSTFS as work items, the development team can then manage the lifecycle of these work items in MSTFS, and have the status changes in MSTFS be reflected back in SpiraTest.

5.1. Configuring the Integration Service

This section outlines how to configure the integration service to export incidents into MSTFS and pick up subsequent status changes in MSTFS and have them update SpiraTest. It assumes that you already have a working installation of SpiraTest v1.5.1 or later and MSTFS 2005 or later. If you have earlier versions of SpiraTest you will need to upgrade to at least v1.5.1.

The steps that need to be performed to configure integration with MSTFS are as follows:

- Download the MSTFS Data-Sync plug-in for SpiraTest from our website
- Configure the service to point to the correct instance of MSTFS
- Configure the data field mappings between SpiraTest and MSTFS
- Start the service and verify data transfer

5.1.1. Download the MSTFS Plug-In

Go to the Inflectra website and open up the page that lists the various downloads available for SpiraTest (http://www.inflectra.com/Products/Downloads.aspx). Listed on this page will be the MSTFS Plug-In for SpiraTest. Right-click on this link and save the Zip compressed folder to the hard-drive of the server where SpiraTest is installed. Open up the compressed folder and extract the files to the following locations:

- Copy MsTfsDataSync.dll to C:\Program Files\SpiraTest\SpiraTestService
- Copy all the Microsoft.TeamFoundation.[Name].dll files to C:\Program Files\SpiraTest\SpiraTestService
- Copy MsTfsDataSync.dll.config to C:\Program Files\SpiraTest\SpiraTestService
- Copy MsTfsDataSyncMappings.xml to C:\Program Files\SpiraTest\SpiraTestService

5.1.2. Configuring the Service

To configure the integration service, please open up the SpiraTestService.exe.config file located in C:\Program Files\SpiraTest\SpiraTestService with a text editor such as Notepad. Once open, it should look like:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
      <appSettings>
              <add key="SpiraTestService.PollingInterval" value="120000" />
              <add key="SpiraTestService.PlugIns"
                      value="JiraDataSync|Inflectra.SpiraTest.
                              PlugIns.JiraDataSync.DataSync" />
      </appSettings>
</configuration>
```

In addition it may have a set of keys that are related to other data-synchronization plug-ins (for example the JiraDataSync which is installed by default) – these can be ignored. The first thing you need to do is tell the SpiraTestService that it needs to use the MSTFS plug-in. To do this you need to change the SpiraTestService.PlugIns key to point to the MSTFS plug-In. To add the MSTFS plug-in to the list of plug-ins that are executed you simply need to add the following information to the key – MsTfsDataSync|Inflectra.SpiraTest.PlugIns.MsTfsDataSync.DataSync. In the example above, to add the MSTFS plug-in to the existing plug-in list you would change the file to:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
         value="JiraDataSync|Inflectra.SpiraTest.
                PlugIns.JiraDataSync.DataSync;MsTfsDataSync
               |Inflectra.SpiraTest.PlugIns.MsTfsDataSync.DataSync" />
  </appSettings>
</configuration>
```

However, if you actually want to remove the entry for other plug-ins (e.g. in this example you don't want to synchronize some projects with JIRA and some with MSTFS, then you should remove the other entries:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
       value="MsTfsDataSync|Inflectra.SpiraTest.PlugIns.MsTfsDataSync.DataSync"/>
  </appSettings>
</configuration>
```

Now that you have added the MSTFS plug-in to the list of plug-ins that will be executed by the background service, you now need to copy the MSTFS-specific configuration settings from MsTfsDataSync.dll.config to SpiraTestService.exe.config. Once you have done this, you should have a single SpiraTestService.exe.config configuration file that looks something like:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
   <configSections>
      <sectionGroup name="applicationSettings"
type="System.Configuration.ApplicationSettingsGroup, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" >
        <section name="Inflectra.SpiraTest.PlugIns.MsTfsDataSync.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
      </sectionGroup>
  </configSections>
<applicationSettings>
  <Inflectra.SpiraTest.PlugIns.MsTfsDataSync.Properties.Settings>
   <setting name="MsTfsDataSync_SpiraImport_Import" serializeAs="String">
    <value>http://localhost/SpiraTest/Services/Import.asmx</value>
   </setting>
  </Inflectra.SpiraTest.PlugIns.MsTfsDataSync.Properties.Settings>
 </applicationSettings>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
       value="MsTfsDataSync|Inflectra.SpiraTest.PlugIns.MsTfsDataSync.DataSync"/>
    <add key="MsTfsDataSync.StaticMappings.Path" value="C:\Program
Files\SpiraTest\SpiraTestService\MsTfsDataSyncMappings.xml" />
    <add key="MsTfsDataSync.SpiraApi.Login" value="fredbloggs" />
    <add key="MsTfsDataSync.SpiraApi.Password" value="fredbloggs" />
    <add key="MsTfsDataSync.MsTfsApi.Login" value="Administrator" />
    <add key="MsTfsDataSync.MsTfsApi.Password" value="Password" />
    <add key="MsTfsDataSync.MsTfsApi.Domain" value="INFLECTRASVR03" />
    <add key="MsTfsDataSync.MsTfsApi.ServerUrl" value="http://inflectrasvr03:8080" />
  </appSettings>
</configuration>
```

The sections that need to be changed are marked in yellow above. You need to enter the following information:

- The *host name* and *port number* being used to access the MSTFS web services API. The host name should be the name of the server that MSTFS was installed on, and the port number should be 8080 unless you have explicitly changed it.
- The full URL to your instance SpiraTest. It is typically of the form `http://<server name>/SpiraTest/Services/Import.asmx`. Make sure that you can browse to this URL on the server itself.
- The *login name* and *password* to your instance of SpiraTest. This user needs to have at least Manager permissions for project(s) you want to synchronize with MSTFS.
- A valid *Windows login, password* and *domain name* for a user that has the permissions to create, modify and view work items inside your instance of MSTFS. Typically the user would need to be a member of the [Project Name] Contributors group.

Once you have made these changes, save the file and proceed to the next stage.

5.1.3. Configuring the Data Mapping

Next, you need to configure the data mapping between SpiraTest and MSTFS. This allows the various incident types and status codes used in the two applications to be related to each other. This is important, as without a correct mapping, there is no way for the integration service to know that a "P1 - Critical" incident in SpiraTest is the same as a "Priority 1" bug work item in MSTFS (for example).

Please open up the MsTfsDataSyncMappings.xml file located in C:\Program Files\SpiraTest\SpiraTestService with a text editor such as Notepad. Once open, it should look something like:

```
<?xml version="1.0" encoding="utf-8" ?>
<staticmappings>
      <!-- This needs to be a one-to-one mapping from SpiraTest Project to/from MSTFS
Area -->
      <projectmappings>
              <mapping spiraid="1" externalid="Library Information System"
annotation="Library Sample Project" />
      </projectmappings>

      <!-- This needs to be always a one-to-one mapping, no one-to-many -->
      <usermappings>
              <mapping spiraid="1" externalid="Administrator" annotation="Administrator"
/>
              <mapping spiraid="2" externalid="Fred Bloggs" annotation="Fred Bloggs" />
              <mapping spiraid="3" externalid="Joe Smith" annotation="Joe Smith" />
      </usermappings>

      <!-- This needs to be a one-to-one mapping from SpiraTest Project to/from MSTFS
Iteration -->
      <releasemappings>
              <mapping spiraid="1" externalid="Library Information System"
annotation="Library System Release 1" />
              <mapping spiraid="2" externalid="Library Information System\Iteration 0"
annotation="Library System Release 1 SP1" />
```

```
            <mapping spiraid="3" externalid="Library Information System\Iteration 1"
annotation="Library System Release 1 SP2" />
      </releasemappings>

      <!-- These map SpiraTest Incident Status to MSTFS Work Item State+Reason-->
      <incidentstatusmappings>
            <mapping spiraid="1" projectid="1" externalid="Active+New"
annotation="New" />
            <mapping spiraid="2" projectid="1" externalid="Active+New"
annotation="Open" />
            <mapping spiraid="3" projectid="1" externalid="Active+New"
annotation="Assigned" />
            <mapping spiraid="4" projectid="1" externalid="Resolved+Fixed"
annotation="Resolved" />
            <mapping spiraid="5" projectid="1" externalid="Closed+Fixed"
annotation="Closed" />
            <mapping spiraid="6" projectid="1" externalid="Closed+Cannot Reproduce"
annotation="Not Reproducible"/>
            <mapping spiraid="7" projectid="1" externalid="Closed+Duplicate"
annotation="Duplicate"/>
            <mapping spiraid="8" projectid="1" externalid="Active+Not Fixed"
annotation="Reopen"/>
      </incidentstatusmappings>

      <!-- These map MSTFS Work Item State+Reason back to SpiraTest Incident Status-->
      <incidentstatusreversemappings>
            <mapping externalid="Active+New" projectid="1" spiraid="2"
annotation="State=Active,Reason=New" />
            <mapping externalid="Active+Wrong Fix" projectid="1" spiraid="8"
annotation="State=Active,Reason=Wrong Fix"/>
            <mapping externalid="Active+Build Failure" projectid="1" spiraid="8"
annotation="State=Active,Reason=Build Failure"/>
            <mapping externalid="Active+Regression" projectid="1" spiraid="8"
annotation="State=Active,Reason=Regression"/>
            <mapping externalid="Active+Reactivated" projectid="1" spiraid="8"
annotation="State=Active,Reason=Reactivated"/>
            <mapping externalid="Active+Closed in Error" projectid="1" spiraid="8"
annotation="State=Active,Reason=Closed in Error"/>
            <mapping externalid="Active+Resolution Denied" projectid="1" spiraid="8"
annotation="State=Active,Reason=Resolution Denied"/>
            <mapping externalid="Resolved+Fixed" projectid="1" spiraid="4"
annotation="State=Resolved,Reason=Fixed" />
            <mapping externalid="Resolved+As Designed" projectid="1" spiraid="4"
annotation="State=Resolved,Reason=As Designed" />
            <mapping externalid="Resolved+Unable to Reproduce" projectid="1"
spiraid="6" annotation="State=Resolved,Reason=Unable to Reproduce" />
            <mapping externalid="Resolved+Duplicate" projectid="1" spiraid="7"
annotation="State=Resolved,Reason=Duplicate" />
            <mapping externalid="Resolved+Deferred" projectid="1" spiraid="4"
annotation="State=Resolved,Reason=Deferred" />
            <mapping externalid="Resolved+Obsolete" projectid="1" spiraid="4"
annotation="State=Resolved,Reason=Obsolete" />
            <mapping externalid="Closed+Fixed" projectid="1" spiraid="5"
annotation="State=Closed,Reason=Fixed" />
            <mapping externalid="Closed+Unable to Reproduce" projectid="1" spiraid="6"
annotation="State=Closed,Reason=Unable to Reproduce" />
            <mapping externalid="Closed+Duplicate" projectid="1" spiraid="7"
annotation="State=Closed,Reason=Duplicate" />
            <mapping externalid="Closed+Deferred" projectid="1" spiraid="5"
annotation="State=Closed,Reason=Deferred" />
            <mapping externalid="Closed+Not a Bug" projectid="1" spiraid="5"
annotation="State=Closed,Reason=Not a Bug" />
            <mapping externalid="Closed+Obsolete" projectid="1" spiraid="5"
annotation="State=Closed,Reason=Obsolete" />
      </incidentstatusreversemappings>

      <!-- These map SpiraTest incident types to MSTFS work item types -->
      <incidenttypemappings>
            <mapping spiraid="1" projectid="1" externalid="Bug" annotation="Incident"
/>
            <mapping spiraid="2" projectid="1" externalid="Bug" annotation="Bug" />
```

```
              <mapping spiraid="3" projectid="1" externalid="Bug"
annotation="Enhancement" />
              <mapping spiraid="4" projectid="1" externalid="Bug" annotation="Issue" />
              <mapping spiraid="5" projectid="1" externalid="Task" annotation="Training"
/>
              <mapping spiraid="6" projectid="1" externalid="Task"
annotation="Limitation"/>
              <mapping spiraid="7" projectid="1" externalid="Bug" annotation="Change
Request"/>
              <mapping spiraid="8" projectid="1" externalid="Risk" annotation="Risk"/>
      </incidenttypemappings>

      <!-- These map MSTFS work item types to SpiraTest incident types -->
      <incidenttypereversemappings>
              <mapping externalid="Bug" projectid="1" spiraid="2" annotation="Bug" />
              <mapping externalid="Task" projectid="1" spiraid="5" annotation="Task" />
              <mapping externalid="Quality of Service Requirement" projectid="1"
spiraid="6" annotation="Quality of Service Requirement" />
              <mapping externalid="Risk" projectid="1" spiraid="8" annotation="Risk"/>

      </incidenttypereversemappings>

      <!-- These map SpiraTest Priorities to MSTFS Priorities-->
      <prioritymappings>
              <mapping spiraid="1" projectid="1" externalid="1" annotation="Critical" />
              <mapping spiraid="2" projectid="1" externalid="2" annotation="High" />
              <mapping spiraid="3" projectid="1" externalid="3" annotation="Medium" />
              <mapping spiraid="4" projectid="1" externalid="3" annotation="Low" />
      </prioritymappings>

      <!-- These map MSTFS Priorities to SpiraTest Priorities-->
      <priorityreversemappings>
              <mapping externalid="1" projectid="1" spiraid="1" annotation="Priority 1"
/>
              <mapping externalid="2" projectid="1" spiraid="2" annotation="Priority 2"
/>
              <mapping externalid="3" projectid="1" spiraid="3" annotation="Priority 3"
/>
      </priorityreversemappings>

      <!-- These map SpiraTest list custom properties to MSTFS configurable fields-->
      <custompropertymappings>
              <mapping spiraid="2" projectid="1"
externalid="Microsoft.VSTS.Common.Triage" annotation="Triage" />
      </custompropertymappings>

      <!-- These map SpiraTest custom property values to MSTFS configurable fields-->
      <custompropertyvaluemappings>
              <mapping spiraid="10" externalid="Approved" annotation="Approved" />
              <mapping spiraid="11" externalid="Investigate" annotation="Investigate" />
      </custompropertyvaluemappings>

</staticmappings>
```

The three sections of the file that have to be changed in any installation are the project, user, and release/version mappings. These are described below:

- In the **project mappings** section you need to list a mapping entry for any project that you want to synchronize between SpiraTest and MSTFS. The project must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the project id of the project in SpiraTest, and the **externalid** which should be the full name of the project in MSTFS.
  The **annotation** field is an optional entry that can be used in all of the mappings, it is not used by the integration service, but is instead used to help document what the mapping relates to. In this instance you might use it to list the name of the project being mapped.

- In the **user mappings** section you need to list a mapping entry for any user that can have issues assigned to him/her in SpiraTest and MSTFS. The user must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the user in SpiraTest, and the **externalid** which should be the full name of the user inside MSTFS (typically taken from Windows).
- In the **release mappings** section you need to list a mapping entry for any release/iteration that can have issues assigned to it in SpiraTest and MSTFS. The release/iteration must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the release in SpiraTest, and the **externalid** which should be the Iteration Path of the equivalent "iteration" in MSTFS.
- Next you need to synchronize the various incident types, priorities and statuses between SpiraTest and MSTFS. Without this there is no way for the synchronization service to be able to reconsile any changes made to the incident/bug. Since SpiraTest allows you to have different values for these entity for each project, you will need to have entries in the mapping sections for all the projects listed at the top of the XML document.
- **Incident Statuses**
  There are two mapping sections in the XML document, the first one – *incidentstatusmappings* – relates each SpiraTest incident status ID to the matching combination of State and Reason in MSTFS. The second one – *incidentstatusreversemappings* – relates each combination of MSTFS State and Reason back to the corresponding incident status in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of statuses in each SpiraTest project and match them against the list of allowed states and reasons in each MSTFS project. Then for each status, you need to ensure there is a mapping entry in the list that relates the SpiraTest incident status to a valid combination of MSTFS State+Reason, and one that relates all allowed MSTFS State+Reason combinations back to a corresponding SpiraTest incident status.
- **Incident Types**
  There are two mapping sections in the XML document, the first one – *incidentypemappings* – relates each SpiraTest incident type ID to the corresponding work item type in MSTFS. The second one – *incidenttypereversemappings* – relates each MSTFS work item type back to the corresponding incident type ID in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of incident types in each SpiraTest project and match them against the list of work item types in each MSTFS project. Then for each type, you need to ensure there is a mapping entry in the list that relates the SpiraTest incident type to the MSTFS work item type, and one that relates the MSTFS work item type back to the corresponding SpiraTest incident type.
- **Incident Priorities**
  There are two mapping sections in the XML document, the first one – *prioritymappings* – relates each SpiraTest incident priority ID to the corresponding priority number in MSTFS. The second one – *priorityreversemappings* – relates each MSTFS work item priority number back to the corresponding priority ID in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of incident priorities in each SpiraTest project and match them

against the list of work item priorities in each MSTFS project. Then for each priority, you need to ensure there is a mapping entry in the list that relates the SpiraTest incident priority to the MSTFS work item priority, and one that relates the MSTFS work item priority back to the corresponding SpiraTest incident priority.

- **Custom Properties**
  There are two mapping sections in the XML document that relate to custom properties. The first one – *custompropertymappings* – maps each of the custom property list fields used in a particular SpiraTest project to one of the configurable MSTFS fields. The SpiraId should be the list field number of the custom property. E.g. if you have assigned Triage as the alias for SpiraTest custom property List02 then the SpiraId should be set to "2". The ExternalId is the fully-qualified name of the filed inside MSTFS. We have provided a list of MSTFS fields in the table below.
  The second section – *custompropertyvaluemappings* – relates each custom property value in SpiraTest with the equivalent value in MSTFS. The SpiraId can be found from the Administration > Edit Property Values screen, and the ExternalId is the full name of the equivalent value inside MSTFS. The example above shows the two values for the MSTFS triage field.

The following table lists all the possible fields in MSTFS that you may want to synchronize with SpiraTest using a custom property to capture the information within SpiraTest:

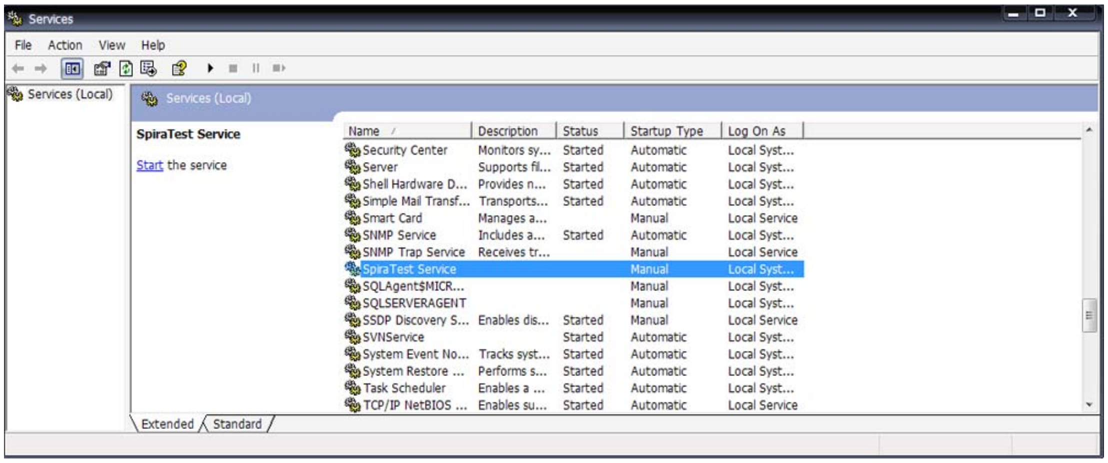
Field Caption	**Field Name**
Issue	Microsoft.VSTS.Common.Issue
State Change Date	Microsoft.VSTS.Common.StateChangeDate
Activated Date	Microsoft.VSTS.Common.ActivatedDate
Activated By	Microsoft.VSTS.Common.ActivatedBy
Resolved Date	Microsoft.VSTS.Common.ResolvedDate
Resolved By	Microsoft.VSTS.Common.ResolvedBy
Resolved Reason	Microsoft.VSTS.Common.ResolvedReason
Closed Date	Microsoft.VSTS.Common.ClosedDate
Closed By	Microsoft.VSTS.Common.ClosedBy
Priority	Microsoft.VSTS.Common.Priority
Triage	Microsoft.VSTS.Common.Triage
Rank	Microsoft.VSTS.Common.Rank
Test Name	Microsoft.VSTS.Test.TestName
Test Id	Microsoft.VSTS.Test.TestId
Test Path	Microsoft.VSTS.Test.TestPath
Found In	Microsoft.VSTS.Build.FoundIn
Integration Build	Microsoft.VSTS.Build.IntegrationBuild
Exit Criteria	Microsoft.VSTS.Common.ExitCriteria
Discipline	Microsoft.VSTS.Common.Discipline
Remaining Work	Microsoft.VSTS.Scheduling.RemainingWork
Completed Work	Microsoft.VSTS.Scheduling.CompletedWork
Baseline Work	Microsoft.VSTS.Scheduling.BaselineWork
Start Date	Microsoft.VSTS.Scheduling.StartDate
Finish Date	Microsoft.VSTS.Scheduling.FinishDate
Task Hierarchy	Microsoft.VSTS.Scheduling.TaskHierarchy
Quality of Service Type	Microsoft.VSTS.Common.QualityOfServiceType
Rough Order of Magnitude	Microsoft.VSTS.Common.RoughOrderOfMagnitude
Severity	Microsoft.VSTS.Common.Severity

Once you have updated the various mapping sections, you are now ready to start the service.

5.1.4. Starting the Service

When SpiraTest is installed, a Windows Service – SpiraTestService – is installed along with the web application. However to avoid wasting system resources, this service is initially set to run manually. To ensure continued synchronization of SpiraTest with MSTFS, we recommend starting the service and setting its startup-type to Automatic.

To make these changes, open up the Windows Control Panel, click on the “Administrative Tools” link, and then choose the Services option. This will bring up the Windows Service control panel:



SpiraTest Service

Start the service

Name	Description	Status	Startup Type	Log On As
Security Center	Monitors sy...	Started	Automatic	Local Syst...
Server	Supports fil...	Started	Automatic	Local Syst...
Shell Hardware D...	Provides n...	Started	Automatic	Local Syst...
Simple Mail Transf...	Transports...	Started	Automatic	Local Syst...
Smart Card	Manages a...		Manual	Local Service
SNMP Service	Includes a...	Started	Automatic	Local Syst...
SNMP Trap Service	Receives tr...		Manual	Local Service
SpiraTest Service			Manual	Local Syst...
SQLAgent$MICR...			Manual	Local Syst...
SQLSERVERAGENT			Manual	Local Syst...
SSDP Discovery S...	Enables dis...	Started	Manual	Local Service
SVNService		Started	Automatic	Local Syst...
System Event No...	Tracks syst...	Started	Automatic	Local Syst...
System Restore ...	Performs s...	Started	Automatic	Local Syst...
Task Scheduler	Enables a ...	Started	Automatic	Local Syst...
TCP/IP NetBIOS ...	Enables su...	Started	Automatic	Local Service

Click on the ‘SpiraTestService’ entry and click on the link to start the service. Then right-click the service entry and choose the option to set the startup type to ‘Automatic’. This will ensure that synchronization continues between SpiraTest and MSTFS after a reboot of the server.

5.2. Using SpiraTest with MSTFS

Now that the integration service has been configured and the service started, initially any incidents created in SpiraTest for the specified projects will be imported into MSTFS. At this point we recommend opening the Windows Event Viewer and choosing the Application Log. In this log any error messages raised by the SpiraTest service will be displayed. If you see any error messages at this point, we recommend immediately stopping the SpiraTest service and checking the various mapping entries. If you cannot see any issues with the mapping information, we recommend sending a copy of the event log message together with your mapping XML file to Inflectra customer services (support@inflectra.com) who will help you troubleshoot the problem.

To use SpiraTest with MSTFS on an ongoing basis, we recommend the following general processes be followed:

- When running tests in SpiraTest, defects found should be logged through the ‘Add Incident’ option as normal.
- Once an incident has been created during the running of the test, it will now be populated across into MSTFS as a work item. It will be populated with the information captured in SpiraTest.
- At this point, the incident should not be acted upon inside SpiraTest, and all data changes to the work item should be made inside MSTFS. To enforce this, you can modify the workflows set up in SpiraTest so that the various fields are marked as inactive for all the incident statuses other than the “New” status. This will allow someone to submit an

incident in SpiraTest, but will prevent them making changes in conflict with MSTFS after that point.

- As the work item progresses through the MSTFS workflow, changes to the state, reason, assigned owner, priority, and resolution will be updated automatically in SpiraTest. In essence, SpiraTest acts as a read-only viewer of these incidents.
- You are now able to perform test coverage and incident reporting inside SpiraTest using the test cases managed by SpiraTest and the incidents managed on behalf of SpiraTest inside MSTFS.

6. Using SpiraTest with FogBugz

This section outlines how to use SpiraTest in conjunction with the popular FogBugz bug tracking system. The built-in integration service allows the quality assurance team to manage their requirements and test cases in SpiraTest, execute test runs in SpiraTest, and then have the new incidents generated during the run be automatically loaded into FogBugz. Once the incidents are loaded into FogBugz as cases, the development team can then manage the lifecycle of these cases in FogBugz, and have the status changes in FogBugz be reflected back in SpiraTest.

4.1. Configuring the Integration Service

This section outlines how to configure the integration service to export incidents into FogBugz and pick up subsequent status changes in FogBugz and have them update SpiraTest. It assumes that you already have a working installation of SpiraTest v1.5.2 or later and FogBugz v6.0 or later. If you have earlier versions of SpiraTest or FogBugz you will need to upgrade to at least v1.5.2 and v6.0 respectively.

The steps that need to be performed to configure integration with FogBugz are as follows:

- Download the FogBugz Data-Sync plug-in for SpiraTest from our website
- Configure the service to point to the correct instance of FogBugz
- Configure the data field mappings between SpiraTest and FogBugz
- Start the service and verify data transfer

6.1.1. Download the FogBugz Plug-In

Go to the Inflectra website and open up the page that lists the various downloads available for SpiraTest (http://www.inflectra.com/Products/Downloads.aspx). Listed on this page will be the FogBugz Plug-In for SpiraTest. Right-click on this link and save the Zip compressed folder to the hard-drive of the server where SpiraTest is installed. Open up the compressed folder and extract the files to the following locations:

- Copy FogBugzDataSync.dll to C:\Program Files\SpiraTest\SpiraTestService
- Copy FogBugzDataSync.dll.config to C:\Program Files\SpiraTest\SpiraTestService
- Copy FogBugzDataSyncMappings.xml to C:\Program Files\SpiraTest\SpiraTestService

6.1.2. Configuring the Service

To configure the integration service, please open up the SpiraTestService.exe.config file located in C:\Program Files\SpiraTest\SpiraTestService with a text editor such as Notepad. Once open, it should look like:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
      <appSettings>
              <add key="SpiraTestService.PollingInterval" value="120000" />
              <add key="SpiraTestService.PlugIns"
                      value="JiraDataSync|Inflectra.SpiraTest.
                              PlugIns.JiraDataSync.DataSync" />
      </appSettings>
</configuration>
```

In addition it may have a set of keys that are related to other data-synchronization plug-ins (for example the JiraDataSync which may be installed by default) – these can be ignored. The first thing you need to do is tell the SpiraTestService that it needs to use the FogBugz plug-in. To do this you need to change the SpiraTestService.PlugIns key to point to the FogBugz plug-In. To add the FogBugz plug-in to the list of plug-ins that are executed you simply need to add the following information to the key – FogBugzDataSync|Inflectra.SpiraTest.PlugIns.FogBugzDataSync.DataSync. In the example above, to add the FogBugz plug-in to the existing plug-in list you would change the file to:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
         value="JiraDataSync|Inflectra.SpiraTest.
                PlugIns.JiraDataSync.DataSync;FogBugzDataSync
               |Inflectra.SpiraTest.PlugIns.FogBugzDataSync.DataSync" />
  </appSettings>
</configuration>
```

However, if you actually want to remove the entry for other plug-ins (e.g. in this example you don't want to synchronize some projects with JIRA and others with FogBugz, then you should remove the other entries:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="SpiraTestService.PollingInterval" value="120000" />
    <add key="SpiraTestService.PlugIns"
       value="FogBugzDataSync|Inflectra.SpiraTest.PlugIns.FogBugzDataSync.DataSync"/>
  </appSettings>
</configuration>
```

Now that you have added the FogBugz plug-in to the list of plug-ins that will be executed by the background service, you now need to copy the FogBugz-specific configuration settings from FogBugzDataSync.dll.config to SpiraTestService.exe.config. Once you have done this, you should have a single SpiraTestService.exe.config configuration file that looks something like:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <configSections>
        <sectionGroup name="applicationSettings"
type="System.Configuration.ApplicationSettingsGroup, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" >
            <section
name="Inflectra.SpiraTest.PlugIns.FogBugzDataSync.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
        </sectionGroup>
    </configSections>
    <applicationSettings>
        <Inflectra.SpiraTest.PlugIns.FogBugzDataSync.Properties.Settings>
            <setting name="FogBugzDataSync_SpiraImport_Import" serializeAs="String">
                <value>http://localhost/SpiraTest/Services/Import.asmx</value>
            </setting>
        </Inflectra.SpiraTest.PlugIns.FogBugzDataSync.Properties.Settings>
    </applicationSettings>
  <appSettings>
    <add key="FogBugzDataSync.FogBugz.Url" value="https://myserver.fogbugz.com" />
    <add key="FogBugzDataSync.StaticMappings.Path" value="C:\Program
Files\SpiraTest\SpiraTestService\FogBugzDataSyncMappings.xml" />
    <add key="FogBugzDataSync.SpiraApi.Login" value="fredbloggs" />
    <add key="FogBugzDataSync.SpiraApi.Password" value="fredbloggs" />
    <add key="FogBugzDataSync.FogBugzApi.Email" value="myemailaddress@mycompany.com" />
    <add key="FogBugzDataSync.FogBugzApi.Password" value="password" />
    <add key="FogBugzDataSync.FogBugzApi.VerifyCertificate" value="false" />
  </appSettings>
</configuration>
```

The sections that need to be changed are marked in yellow above. You need to enter the following information:

- The base URL of the instance of FogBugz that you're connecting to. It is typically of the form https://myserver.fogbugz.com. If you are using the externally hosted FogBugz on demand then it will typically use the HTTPS protocol, whereas if you're using a locally hosted instance, it may simply use the HTTP protocol. Check the URL that you use in your browser when logging in to FogBugz.
- The full URL to your instance SpiraTest. It is typically of the form `http://<server name>/SpiraTest/Services/Import.asmx`. Make sure that you can browse to this URL on the server itself.
- The login name and password to your instance of SpiraTest. This user needs to have at least Manager permissions for project(s) you want to synchronize with FogBugz.
- A valid email address and password for the FogBugz instance you want SpiraTest to synchronize with. This user needs to have permissions to create cases in the project.

Once you have made these changes, save the file and proceed to the next stage.

6.1.3. Configuring the Data Mapping

Next, you need to configure the data mapping between SpiraTest and FogBugz. This allows the various incident types and status codes used in the two applications to be related to each other. This is important, as without a correct mapping, there is no way for the integration service to know that a "P1 - Critical" incident in SpiraTest is the same as a "Must Fix" case in FogBugz (for example).

Please open up the FogBugzDataSyncMappings.xml file located in C:\Program Files\SpiraTest\SpiraTestService with a text editor such as Notepad. Once open, it should look something like:

```
<?xml version="1.0" encoding="utf-8" ?>
<staticmappings>
      <!-- This needs to be always a one-to-one mapping, no one-to-many -->
      <projectmappings>
              <mapping spiraid="1" externalid="1" annotation="Library Sample Project" />
      </projectmappings>

      <!-- This needs to be always a one-to-one mapping, no one-to-many -->
      <usermappings>
              <mapping spiraid="1" externalid="2" annotation="administrator" />
              <mapping spiraid="2" externalid="3" annotation="fredbloggs" />
              <mapping spiraid="3" externalid="4" annotation="joesmith" />
      </usermappings>

      <!-- These map SpiraTest releases to FogBugz FixFor's (always a one-to-one
mapping, not one-to-many) -->
      <releasemappings>
              <mapping spiraid="1" externalid="2" annotation="Library System Release
1.0" />
              <mapping spiraid="2" externalid="3" annotation="Library System Release
1.1" />
      </releasemappings>

  <!-- These map SpiraTest incident types to FogBugz categories -->
  <incidenttypemappings>
    <mapping spiraid="1" projectid="1" externalid="1" annotation="Incident" />
    <mapping spiraid="2" projectid="1" externalid="1" annotation="Bug" />
    <mapping spiraid="3" projectid="1" externalid="2" annotation="Enhancement" />
    <mapping spiraid="4" projectid="1" externalid="3" annotation="Issue" />
    <mapping spiraid="5" projectid="1" externalid="3" annotation="Training" />
    <mapping spiraid="6" projectid="1" externalid="3" annotation="Limitation"/>
    <mapping spiraid="7" projectid="1" externalid="3" annotation="Change Request"/>
    <mapping spiraid="8" projectid="1" externalid="3" annotation="Risk"/>
  </incidenttypemappings>

  <!-- These map FogBugz categories to SpiraTest incident types -->
  <incidenttypereversemappings>
    <mapping externalid="1" projectid="1" spiraid="2" annotation="Bug" />
    <mapping externalid="2" projectid="1" spiraid="3" annotation="Feature" />
    <mapping externalid="3" projectid="1" spiraid="4" annotation="Enquiry" />
  </incidenttypereversemappings>

  <!-- These map FogBugz Case Status back to SpiraTest Incident Status-->
      <incidentstatusreversemappings>
              <mapping externalid="1" projectid="1" spiraid="2" annotation="Active" />
              <mapping externalid="2" projectid="1" spiraid="4" annotation="Resolved
(Fixed)" />
              <mapping externalid="3"       projectid="1" spiraid="6"
annotation="Resolved (Not Reproducible)"/>
              <mapping externalid="4" projectid="1" spiraid="7" annotation="Resolved
(Duplicate)" />
              <mapping externalid="5" projectid="1" spiraid="8" annotation="Resolved
(Postponed)" />
              <mapping externalid="6" projectid="1" spiraid="5" annotation="Resolved
(Won't Fix)" />
              <mapping externalid="8" projectid="1" spiraid="6" annotation="Resolved (By
Design)" />
              <mapping externalid="9" projectid="1" spiraid="4" annotation="Resolved
(Implemented)" />
              <mapping externalid="10" projectid="1" spiraid="5" annotation="Resolved
(Won't Implement)" />
              <mapping externalid="11" projectid="1" spiraid="7" annotation="Resolved
(Already Exists)" />
              <mapping externalid="12" projectid="1" spiraid="4" annotation="Resolved
(Responded)" />
              <mapping externalid="13" projectid="1" spiraid="6" annotation="Resolved
(SPAM)" />
              <mapping externalid="14" projectid="1" spiraid="8" annotation="Resolved
(Waiting For Info)" />
              <mapping externalid="15" projectid="1" spiraid="5" annotation="Resolved
(Completed)" />
              <mapping externalid="16" projectid="1" spiraid="5" annotation="Resolved
(Canceled)" />
      </incidentstatusreversemappings>

      <!-- These map SpiraTest Priorities to FogBugz Priorities-->
      <prioritymappings>
              <mapping spiraid="-1" projectid="1" externalid="3" annotation="Default" />
              <mapping spiraid="1" projectid="1" externalid="1" annotation="Critical" />
              <mapping spiraid="2" projectid="1" externalid="2" annotation="High" />
              <mapping spiraid="2" projectid="1" externalid="3" annotation="High" />
              <mapping spiraid="3" projectid="1" externalid="4" annotation="Medium" />
              <mapping spiraid="3" projectid="1" externalid="5" annotation="Medium" />
              <mapping spiraid="3" projectid="1" externalid="6" annotation="Medium" />
              <mapping spiraid="4" projectid="1" externalid="7" annotation="Low" />
      </prioritymappings>

      <!-- These map FogBugz Priorities to SpiraTest Priorities-->
      <priorityreversemappings>
              <mapping externalid="1" projectid="1" spiraid="1" annotation="Must Fix" />
              <mapping externalid="2" projectid="1" spiraid="2" annotation="Must Fix" />
              <mapping externalid="3" projectid="1" spiraid="2" annotation="Must Fix" />
              <mapping externalid="4" projectid="1" spiraid="3" annotation="Fix If Time"
/>
              <mapping externalid="5" projectid="1" spiraid="3" annotation="Fix If Time"
/>
              <mapping externalid="6" projectid="1" spiraid="3" annotation="Fix If Time"
/>
              <mapping externalid="7" projectid="1" spiraid="4" annotation="Don't Fix"
/>
      </priorityreversemappings>
```

```
        <!-- These map SpiraTest custom property values to FogBugz areas -->
        <areamappings>
                <mapping spiraid="-1" externalid="1" annotation="Default" />
                <mapping spiraid="1" externalid="1" annotation="Code" />
                <mapping spiraid="2" externalid="2" annotation="User Interface" />
                <mapping spiraid="3" externalid="3" annotation="Documentation" />
                <mapping spiraid="4" externalid="4" annotation="Miscellaneous" />
         </areamappings>
</staticmappings>
```

The three sections of the file that have to be changed in any installation are the project, user, and release/version mappings. These are described below:

- In the **project mappings** section you need to list a mapping entry for any project that you want to synchronize between SpiraTest and FogBugz. The project must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the project in SpiraTest, and the **externalid** which should be the id of the project in FogBugz.

  To determine the id of the project in FogBugz, go to Settings > Projects and hover the mouse over the hyperlink for the project you're synchronizing and the URL will contain ixProject=X where X is an integer. This is the id of the project.

  The **annotation** field is an optional entry that can be used in all of the mappings, it is not used by the integration service, but is instead used to help document what the mapping relates to. In this instance you might use it to list the name of the project being mapped.

- In the **user mappings** section you need to list a mapping entry for any user that can have issues assigned to him/her in SpiraTest and FogBugz. The user must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the user in SpiraTest, and the **externalid** which should be the id of the user in FogBugz. The user's id can be found in FogBugz by hovering over the user's name in Settings > Users and looking for the URL element ixPerson=X.

- In the **release mappings** section you need to list a mapping entry for any release that can have issues assigned to it in SpiraTest and FogBugz. The release must be already created in both systems *before running the service*. The mapping entry consists of the **spiraid** which should contain the ID of the release in SpiraTest, and the **externalid** which should be the name of the equivalent "fix-for" release in FogBugz.

Next you need to synchronize the various incident types, statuses, priorities and areas between SpiraTest and FogBugz. Without this there is no way for the synchronization service to be able to reconcile any changes made to the incident/bug. Since SpiraTest allows you to have different values for these entity for each project, you will need to have entries in the mapping sections for all the projects listed at the top of the XML document.

- **Incident Types**
  There are two mapping sections in the XML document, the first one – *incidenttypemappings* – relates each SpiraTest incident type ID to the corresponding case category in FogBugz. The second one – *incidenttypereversemappings* – relates each FogBugz category ID back to the corresponding ID in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of incident types in each SpiraTest project and match them

against the list of categories in each FogBugz project. Then for each incident type, you need to ensure there is a mapping entry in the list that relates the SpiraTest incident type ID to the FogBugz category ID, and one that relates the FogBugz category ID back to the corresponding SpiraTest incident type ID.

- **Incident Statuses**
  The incident status reverse mappings – *incidentstatusreversemappings* – relate each FogBugz case status ID back to the corresponding ID in SpiraTest. You need to review the list of statuses in each SpiraTest project and match them against the list of statuses in each FogBugz project. Then for each status, you need to ensure there is a mapping entry in the list that relates the FogBugz case status back to the corresponding SpiraTest incident status.

- **Incident Priorities**
  There are two mapping sections in the XML document, the first one – *prioritymappings* – relates each SpiraTest incident priority ID to the corresponding priority ID in FogBugz. The second one – *priorityreversemappings* – relates each FogBugz priority ID back to the corresponding ID in SpiraTest. The reason for the two separate lists is that there may be a one-to-many relationship between the different values in either direction.
  You need to review the list of incident priorities in each SpiraTest project and match them against the list of case priorities in each FogBugz project. Then for each priority, you need to ensure there is a mapping entry in the list that relates the SpiraTest priority ID to the FogBugz priority ID, and one that relates the FogBugz priority ID back to the corresponding SpiraTest priority ID.
  Note: In the forward mappings, you can specify the default FogBugz priority to use if the user doesn't provide one in SpiraTest. This is denoted with spiraid="-1".

- **FogBugz Areas**
  If your instance of FogBugz requires that all new bugs are submitted with an 'Area' then you will need to fill out this section. You first need to create an incident custom property in SpiraTest of type 'LIST' that contains the various area names that exist inside FogBugz. Once you have that list, this mapping section relates each Custom Property Value (e.g. PV00001) from SpiraTest to the equivalent area ID in FogBugz. The spiraid of this mapping entry is the ID of the custom property value with the 'PV' prefix removed. The name of the custom property inside SpiraTest doesn't matter as the synchronization service only looks for the custom property value id.
  Note: In the mappings, you can specify the default FogBugz area to use if the user doesn't provide one in SpiraTest. This is denoted with spiraid="-1".

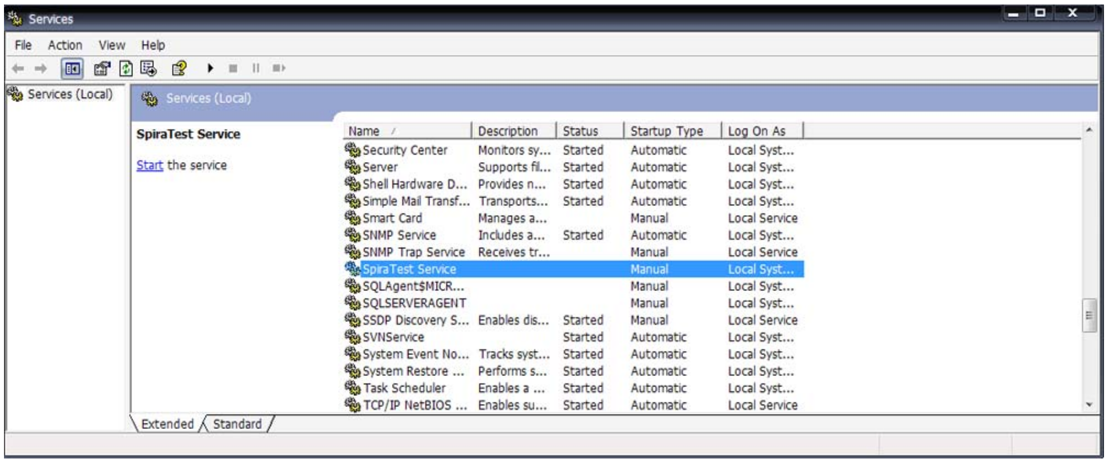
- **FogBugz Computer and Version fields**
  These are two optional text fields that the user can fill out in FogBugz. If you would like users inside SpiraTest to be able to populate these when creating new incidents, all you need to do is create two custom properties in the project - Text01 and Text02 – give them the custom property aliases of 'Computer' and 'Version' respectively, and finally enable the fields for newly created incidents in the workflow editor. Once you have done this, users will be able to enter textual values for these properties when logging an incident in SpiraTest and have the values be synchronized across into FogBugz.

Once you have updated the various mapping sections, you are now ready to start the service.

6.1.4. Starting the Service

When SpiraTest is installed, a Windows Service – SpiraTestService – is installed along with the web application. However to avoid wasting system resources, this service is initially set to run manually. To ensure continued synchronization of SpiraTest with FogBugz, we recommend starting the service and setting its startup-type to Automatic.

To make these changes, open up the Windows Control Panel, click on the “Administrative Tools” link, and then choose the Services option. This will bring up the Windows Service control panel:



SpiraTest Service

Start the service

Name	Description	Status	Startup Type	Log On As
Security Center	Monitors sy...	Started	Automatic	Local Syst...
Server	Supports fil...	Started	Automatic	Local Syst...
Shell Hardware D...	Provides n...	Started	Automatic	Local Syst...
Simple Mail Transf...	Transports...	Started	Automatic	Local Syst...
Smart Card	Manages a...		Manual	Local Service
SNMP Service	Includes a...	Started	Automatic	Local Syst...
SNMP Trap Service	Receives tr...		Manual	Local Service
SpiraTest Service			Manual	Local Syst...
SQLAgent$MICR...			Manual	Local Syst...
SQLSERVERAGENT			Manual	Local Syst...
SSDP Discovery S...	Enables dis...	Started	Manual	Local Service
SVNService		Started	Automatic	Local Syst...
System Event No...	Tracks syst...	Started	Automatic	Local Syst...
System Restore ...	Performs s...	Started	Automatic	Local Syst...
Task Scheduler	Enables a ...	Started	Automatic	Local Syst...
TCP/IP NetBIOS ...	Enables su...	Started	Automatic	Local Service

Click on the ‘SpiraTestService’ entry and click on the link to start the service. Then right-click the service entry and choose the option to set the startup type to ‘Automatic’. This will ensure that synchronization continues between SpiraTest and FogBugz after a reboot of the server.

6.2. Using SpiraTest with FogBugz

Now that the integration service has been configured and the service started, initially any incidents created in SpiraTest for the specified projects will be imported into FogBugz. At this point we recommend opening the Windows Event Viewer and choosing the Application Log. In this log any error messages raised by the SpiraTest service will be displayed. If you see any error messages at this point, we recommend immediately stopping the SpiraTest service and checking the various mapping entries. If you cannot see any issues with the mapping information, we recommend sending a copy of the event log message together with your mapping XML file to Inflectra customer services (support@inflectra.com) who will help you troubleshoot the problem.

To use SpiraTest with FogBugz on an ongoing basis, we recommend the following general processes be followed:

- When running tests in SpiraTest, defects found should be logged through the ‘Add Incident’ option as normal.
- Once an incident has been created during the running of the test, it will now be populated across into FogBugz as an active case. It will be populated with the information captured in SpiraTest.
- At this point, the incident should not be acted upon inside SpiraTest, and all data changes to the issue should be made inside FogBugz. To enforce this, you can modify the workflows set up in SpiraTest so that the various fields are marked as inactive for all the incident statuses other than the “New” status. This will allow someone to submit an incident in SpiraTest, but will prevent them making changes in conflict with FogBugz after that point.

- As the issue progresses through the FogBugz workflow, changes to the status, priority, assignee, estimated effort, due-date, closed-date, fix-for release and resolution will be updated automatically in SpiraTest. In essence, SpiraTest acts as a read-only viewer of these incidents.
- You are now able to perform test coverage and incident reporting inside SpiraTest using the test cases managed by SpiraTest and the incidents managed on behalf of SpiraTest inside FogBugz.

Legal Notices

Please send comments and questions to:

Technical Publications

Inflectra Corporation

10301 Julep Avenue

Silver Spring, MD 20902

U.S.A.

support@inflectra.com