

## **Integrity 10.4** *Integrations User Guide*

June, 2013

---

## Integrity 10.4 Integrations User Guide

**Copyright © 2013 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.**

User and training guides and related documentation from PTC Inc. and its subsidiary companies (collectively "PTC") are subject to the copyright laws of the United States and other countries and are provided under a license agreement that restricts copying, disclosure, and use of such documentation. PTC hereby grants to the licensed software user the right to make copies in printed form of this documentation if provided on software media, but only for internal/personal use and in accordance with the license agreement under which the applicable software is licensed. Any copy made shall include the PTC copyright notice and any other proprietary notice provided by PTC. Training materials may not be copied without the express written consent of PTC. This documentation may not be disclosed, transferred, modified, or reduced to any form, including electronic media, or transmitted or made publicly available by any means without the prior written consent of PTC and no authorization is granted to make copies for such purposes.

Information described herein is furnished for general information only, is subject to change without notice, and should not be construed as a warranty or commitment by PTC. PTC assumes no responsibility or liability for any errors or inaccuracies that may appear in this document.

The software described in this document is provided under written license agreement, contains valuable trade secrets and proprietary information, and is protected by the copyright laws of the United States and other countries. It may not be copied or distributed in any form or medium, disclosed to third parties, or used in any manner not provided for in the software licenses agreement except with written prior approval from PTC.

**UNAUTHORIZED USE OF SOFTWARE OR ITS DOCUMENTATION CAN RESULT IN CIVIL DAMAGES AND CRIMINAL PROSECUTION.** PTC regards software piracy as the crime it is, and we view offenders accordingly. We do not tolerate the piracy of PTC software products, and we pursue (both civilly and criminally) those who do so using all legal means available, including public and private surveillance resources. As part of these efforts, PTC uses data monitoring and scouring technologies to obtain and transmit data on users of illegal copies of our software. This data collection is not performed on users of legally licensed software from PTC and its authorized distributors. If you are using an illegal copy of our software and do not consent to the collection and transmission of such data (including to the United States), cease using the illegal version, and contact PTC to obtain a legally licensed copy.

**Important Copyright, Trademark, Patent, and Licensing Information:** See the About Box, or copyright notice, of your PTC software.

### **UNITED STATES GOVERNMENT RESTRICTED RIGHTS LEGEND**

This document and the software described herein are Commercial Computer Documentation and Software, pursuant to FAR 12.212(a)-(b) (OCT'95) or DFARS 227.7202-1(a) and 227.7202-3(a) (JUN'95), and are provided to the US Government under a limited commercial license only. For procurements predating the above clauses, use, duplication, or disclosure by the Government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 (OCT'88) or Commercial Computer Software-Restricted Rights at FAR 52.227-19(c)(1)-(2) (JUN'87), as applicable. 01282013

**PTC Inc., 140 Kendrick Street, Needham, MA 02494 USA**

---

# Table of Contents

<b>1</b>	<b>Introduction</b> .....	<b>1</b>
	About This Guide .....	2
	Roles .....	2
	Assumptions .....	2
	Integrity Integrations .....	3
	Before You Begin .....	3
	Configuring Integrations .....	3
	Before Using an Integration .....	5
	Where To Go From Here .....	6

## **PART I: CODEGEAR**

<b>2</b>	<b>CodeGear Delphi Architect</b> .....	<b>8</b>
	Key Considerations .....	9
	Configuring the CodeGear Delphi Integration .....	9
	Using the CodeGear Delphi Integration .....	9
	Delphi Architect Commit Browser .....	10
	Creating an Integrity Project .....	11
	Creating a Sandbox .....	11
	Adding a Member to an Integrity Project .....	12
	Checking Out Members .....	12
	Checking In Members .....	12
<b>3</b>	<b>CodeGear JBuilder</b> .....	<b>14</b>

## **PART II: IBM**

<b>4</b>	<b>IBM Rational Eclipse Platform</b> .....	<b>17</b>
	Overview .....	18
	Supported Versions .....	18
	Before You Start .....	18
	Configuring the IBM Rational Eclipse Integration .....	19
	Enabling Access to Java API Libraries (Linux Only) .....	19
	Enabling the Integration .....	19
	Setting Preferences .....	21
	Deactivating the Integration .....	23
	Using the Integration .....	23
	Online and Offline Mode .....	24
	Setting Up an Integrated Workspace .....	25
	Working in an Integrated Workspace .....	30
	Integrity Commands .....	36
	Refactoring .....	40
	Comparing Revisions .....	41
	Team Synchronizing .....	42
	Best Practices .....	46
	Limitations .....	47
<b>5</b>	<b>IBM Rational Rose</b> .....	<b>49</b>
	Configuring the Rational Rose Integration .....	50
	Using the Rational Rose Integration .....	50

Creating an Integrity Project .....	51
Creating a Sandbox .....	51
Adding Members to an Integrity Project .....	51
Checking Out Members .....	52
Checking In Members .....	52

## PART III: MICROSOFT

<b>6 Microsoft Visual Studio (SDK) .....</b>	<b>54</b>
Before You Start .....	55
Setting Up and Configuring the Integration .....	56
Installing the Visual Studio Integration .....	56
Enabling the Integrity Plug-In in Microsoft Visual Studio .....	57
Toggling the Integrity Toolbar .....	57
Setting Preferences .....	58
Working With Keywords .....	58
Configuring the Location of VS Solutions and Projects in the Integrity Repository .....	59
Online and Offline Mode .....	60
Working With Active Change Packages .....	61
Integrity Glyphs in Visual Studio .....	62
Managing Work In Progress .....	63
Managing Assigned Work .....	66
Placing Visual Studio Solutions Under Integrity Source Control .....	67
Sharing a Visual Studio Solution .....	67
Importing a Visual Studio Solution .....	70
Adding a Visual Studio Project to a Shared Solution .....	70
Dropping a Visual Studio Project From a Shared Solution .....	71
Importing a Visual Studio Project .....	72
Migrating a Visual Studio Solution from the MKS SCC VS Integration .....	73
Ignoring Visual Studio Entities From Integrity Source Control .....	74
Branching a Visual Studio Solution .....	75
Resynchronizing a Visual Studio Solution .....	76
Reverting a Visual Studio Solution .....	76
Checkpointing a Visual Studio Solution .....	77
Viewing a Sandbox for a Visual Studio Solution .....	77
Working With Visual Studio Files .....	77
Adding Members to an Integrity Project .....	78
Dropping Members From an Integrity Project .....	78
Checking Out Members .....	79
Checking In Members .....	79
Renaming Members .....	79
Moving Members .....	79
Advanced Integrity Commands .....	80
Best Practices .....	81
Limitations .....	83
Troubleshooting .....	84
<b>7 Microsoft Visual Studio .NET .....</b>	<b>86</b>
Implementing the Integration .....	87
Assumptions .....	87
Required Permissions .....	87
Implementing the New Project Structure .....	87

Setting Up Projects to Reuse Code . . . . .	88
Setting Preferences . . . . .	90
Parallel Development Considerations . . . . .	91
Optimistic Locking . . . . .	91
Development Paths . . . . .	92
Building Projects . . . . .	93
Command Functionality . . . . .	94
Adding a Solution to Integrity . . . . .	96
Adding a New Project to Source Control . . . . .	98
Joining Development of a Solution . . . . .	98
Adding Members to an Integrity Project . . . . .	99
Checking Out Members . . . . .	100
Checking In Members . . . . .	100
Troubleshooting . . . . .	101
<b>8 Microsoft Visual Basic . . . . .</b>	<b>102</b>
Using the Microsoft Visual Basic Integration . . . . .	103
Creating an Integrity Project . . . . .	104
Creating a Sandbox . . . . .	104
Adding Members to an Integrity Project . . . . .	104
Checking Out Members . . . . .	105
Checking In Members . . . . .	105
<b>9 Microsoft Visual C++ . . . . .</b>	<b>106</b>
Configuring the Visual C++ Integration . . . . .	107
Using the Microsoft Visual C++ Integration . . . . .	107
Creating an Integrity Project . . . . .	108
Creating a Sandbox . . . . .	108
Adding Members to an Integrity Project . . . . .	108
Checking Out Members . . . . .	108
Checking In Members . . . . .	109
<b>10 Microsoft Project . . . . .</b>	<b>110</b>
Overview . . . . .	111
Before You Start . . . . .	111
Integrity Server Requirements . . . . .	112
Client Requirements . . . . .	112
Integration Components . . . . .	112
Assumptions . . . . .	113
Key Considerations . . . . .	113
Pre-installation Administration . . . . .	114
Configuring the Integrity Server . . . . .	114
Configuring Integrity . . . . .	115
Setting Permissions . . . . .	116
Modifying the Integration XML Mapping Template . . . . .	116
Installing the Microsoft Project Integration . . . . .	117
Repairing or Removing the Integration . . . . .	119
Microsoft Project Configuration . . . . .	119
Integrity Custom Project Properties . . . . .	119
Integrity Custom Project Fields . . . . .	120
Enabling SSL Communication . . . . .	121
Using Microsoft Project Server and Enterprise Resources . . . . .	121
Logging Configuration . . . . .	122
Using the Microsoft Project Integration . . . . .	123

Specifying Task Types . . . . .	124
Setting Task Relationships . . . . .	125
Synchronizing Resources . . . . .	126
Synchronizing All Tasks . . . . .	126
Synchronizing Tasks By Query . . . . .	126
Synchronizing Linked Tasks . . . . .	127
Synchronizing Selected Tasks . . . . .	127
Detecting Conflicts . . . . .	128
<b>11 Microsoft Word . . . . .</b>	<b>130</b>
Assumptions . . . . .	131
Overview of the Integration . . . . .	131
System Requirements . . . . .	133
Configuring the Integrity Server . . . . .	133
Installing and Uninstalling the Integration . . . . .	134
Customizing the Word Integration . . . . .	135
Customizing the DSD . . . . .	136
Importing the DSD . . . . .	138
Importing a Local DSD . . . . .	139
Understanding the Static TXT File . . . . .	139
Understanding the XML Mapping File . . . . .	140
Understanding the Process Items XLST File . . . . .	141
Exporting Word Documents to Integrity . . . . .	141
Troubleshooting . . . . .	143
<b>12 Microsoft Excel . . . . .</b>	<b>144</b>
Overview . . . . .	145
Before You Start . . . . .	145
Integrity Server Requirements . . . . .	145
Client Requirements . . . . .	146
Integration Components . . . . .	146
Assumptions . . . . .	146
Key Considerations . . . . .	146
Setting the Integrity Server Connection Policy . . . . .	147
Installing the Microsoft Excel Integration . . . . .	147
Repairing or Removing the Integration . . . . .	148
Customizing the Microsoft Excel Integration . . . . .	149
XML Mapping Template . . . . .	149
Integrity Custom Properties . . . . .	151
Enabling SSL Communication . . . . .	151
Logging . . . . .	152
Using the Microsoft Excel Integration . . . . .	152
Importing the XML Mapping Template . . . . .	153
Creating a New List . . . . .	154
Retrieving Items From Integrity . . . . .	154
Synchronizing Data . . . . .	155
Synchronizing Using Queries . . . . .	156
Working With Requirement Documents . . . . .	157
Detecting Conflicts . . . . .	159
Working With Special Fields . . . . .	159

---

## PART IV: OTHER

<b>13 Sybase PowerBuilder</b> .....	<b>163</b>
Configuring the PowerBuilder Integration .....	164
Using the PowerBuilder Integration .....	164
Creating an Integrity Project .....	166
Creating a Sandbox .....	166
Adding Members to an Integrity Project .....	166
Checking Out Members .....	167
Checking In Members .....	167
<b>14 HP Quality Center Requirements and Defect Modules</b> .....	<b>169</b>
Integration Overview .....	170
Assumptions .....	171
System Requirements .....	171
Installation and Configuration Overview .....	171
Step 1: Configuring the Integrity Server .....	172
Step 2: Configuring Quality Center .....	173
Creating and Editing Projects .....	174
Creating MKS ID Field for Requirements and Defects .....	174
Creating Additional Requirements User Fields .....	175
Setting the MKS Type in Quality Center .....	176
Step 3: Configuring Integrity .....	176
Step 4: Installing the Integration .....	177
Step 5: Configuring the Integration .....	178
Modifying Process Configuration and Mapping Files .....	179
Setting Up E-mail Notification .....	181
Setting Up Attachment Handling .....	181
Step 6: Running the Integration .....	182
Troubleshooting .....	183
<b>15 CA Endeavor</b> .....	<b>185</b>
Understanding the CA Endeavor Integration .....	186
Integrity Commands .....	186

## PART V: APPENDIXES

<b>A Configuring a Mapping Template</b> .....	<b>188</b>
Overview .....	189
Mapping Template XML Elements .....	190
Mapping Template Example .....	199
Index .....	202





For configuration management, Integrity provides a number of integrations for industry leading Integrated Development Environments (IDEs), such as IBM Rational Software Development/ Eclipse Platform and Microsoft Visual Studio. For workflows and documents, Integrity integrates into HP Quality Center and Microsoft® Office applications (Microsoft Project, Word, and Excel). Integrity blends transparently into your tools, providing you with access to Integrity core functionality from within your host environment.

When using Integrity for configuration management, commands such as creating an Integrity project or checking out a member are available from within your IDE. More advanced commands (such as setting the member revision) that cannot be performed within the integrations, can be performed in the Integrity Client graphical user interface (GUI). Typically, the Integrity Client GUI can be opened from within the tool you are working in.

When using Integrity for workflows and documents, you can create new items from within your chosen tool with a one-to-one mapping of fields, configured specifically for your workflow.

This chapter provides information on the following topics:

- “About This Guide” on page 2
- “Roles” on page 2
- “Assumptions” on page 2
- “Integrity Integrations” on page 3
- “Before You Begin” on page 3
- “Configuring Integrations” on page 3
- “Before Using an Integration” on page 5
- “Where To Go From Here” on page 6

---

## About This Guide

For content that is essential and applicable to all guides, PTC provides a single location for you to access information. See the *Integrity Getting Started Guide* for information on the following topics:

- detailed descriptions for base concepts used in Integrity
- installing and configuring the Integrity Client
- an overview of the interfaces available for the Integrity Client

Most procedures in this guide are documented using menu-based commands; however, toolbar buttons, shortcut menus, and shortcut keys exist for most procedures. For more information, refer to descriptive tooltips and menu items in the interface.

For detailed information on wizards, views, and dialog box options, see the online help.

---

**NOTE** For this release, instances of the term “issue” appearing on the interface and in configuration files, refer to Integrity items, and therefore have the same meaning.

---

## Roles

There are two primary roles to consider when using Integrity – the administrator and the user.

The *administrator* installs the Integrity database on a network, defines and customizes item types and workflow, manages groups, creates projects and assigns groups to them, manages e-mail notification for users and groups, creates additional user accounts, and assigns permissions that allow users access to specific Integrity operations.

The *user* is anyone who needs to work with Integrity. Users are assigned *user permissions* by the administrator. Users are also assigned to *groups* that have specific Integrity *group permissions* assigned by the administrator.

## Assumptions

Before using Integrity, PTC assumes that you have experience with the following:

- the operating systems used in your work environment (whether Windows, Solaris, Linux, or IBM-AIX)
- your chosen application (for example, Microsoft Project if you are using the Microsoft Project integration or Microsoft Visual Studio if you are using an Microsoft IDE application)

---

**NOTE** At this time, only Windows-based IDEs are supported

---

- for workflow management, the search operators and syntax for the database used by Integrity (that is, the MS SQL Server, DB2, or Oracle database)
- HTML (if you are creating and editing report templates)
- XML (if you are modifying integration templates for the Microsoft Project integration)

---

## Integrity Integrations

For current information on supported integrations for Integrity, go to the Integrity Support Center:

<http://www.ptc.com/support/integrity.htm>

## Before You Begin

Some integrations have specific requirements that must be completed before enabling the integration itself. The following is a list of pre-conditions required for certain integrations:

- If you are using Change Package Reviews in Integrity, some integrations may require you to use Integrity directly to complete tasks. For more information, see the *Integrity User Guide*.
- If you are using a pre-2006 Integrity Client, you cannot view subproject change package entries.
- To use the HP Quality Center integration, the Integrity Client and Quality Center server must reside on the same machine.
- To use Worktray for Visual Studio .NET 2003, it must be enabled as an integration from the Integrity Client. In addition, you must also run the `VSIP Interop Assembly Redist.msi` file located in the following directory:

```
<Integrity Client install directory>/integrations/Microsoft/  
Visual Studio Worktray
```

## Configuring Integrations

The Integrity Client includes the **File > Integrations** menu action for enabling and disabling most supported integrations. You can select from a list of available integrations and enable or disable them, as required to work with your preferred application. There is no requirement to re-install the Integrity Client.

---

**NOTE** Using the **Enable/Disable Integrations** dialog box, you can always restore an available IDE integration at a later date.

---

Certain supported integrations cannot be enabled using the **File > Integrations** menu action. Included in this category are certain Microsoft integrations (Visual Studio SDK, Word, Excel, and Project) and the Integrity integrations with IBM/WebSphere/Rational/Eclipse. For more information, refer to the respective chapters for these integrations.

### ***Integrity Client Default Installation Directory***

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier.

---

The new default installation directory for the Integrity Client is:

On Windows: `C:\Program Files\Integrity\IntegrityClient10`

On UNIX: `$HOME/Integrity/IntegrityClient10`

---

which is changed from the previous directory (`.../MKS/IntegrityClient`).

If you had a previous version of the Integrity Client, PTC recommends that you first disable any existing integrations and then uninstall the previous client *before* installing Integrity Client 10 (or greater).

If you have installed Integrity Client 10.x and uninstalled the previous client, but did not disable previous integrations, those integrations remain enabled but refer to a location that no longer contains supporting files. In this scenario, SCC integrations are removed due to the missing DLLs. The Visual Studio SDK integration will not operate due to the missing DLL. Eclipse-based integrations will fail after a clean re-start of Eclipse.

If you have installed Integrity Client 10.x without first uninstalling the previous version of the client, the enabled integrations continue to reference supporting files from the old location but open with the new Integrity Client 10.x. Integrations continue to function in this scenario; however, any integration-specific HotFixes applied to the Integrity Client 10.x will not update the supporting files in the old client location.

In both of these scenarios, the integrations can be repaired. For more information on repairing an integration, contact PTC - Integrity Support.

### **Enabling and Disabling Integrations**

The following sections provide instructions on how to enable and disable the available IDE integrations using the Integrity Client graphical user interface. You can also enable or disable integrations using the `si integrations` command in the CLI. For more information on the `si integrations` command, see the *Integrity CLI Reference Guide for Configuration Management*.

#### **To enable integrations using the graphical user interface**



- 1 Open the Integrity Client and select **File > Integrations**.

---

**IMPORTANT** If the command is not visible, you must enable it in the active viewset. For information on viewsets, see the *Integrity Getting Started Guide*.

---

The **Enable/Disable Integrations** dialog box displays.

- 2 In the **Disabled Integrations** list, select the available integration(s) that you want to enable.
- 3 To move the selected integrations to the **Enabled Integrations** list, click . To move all disabled integrations to the **Enabled Integrations** list, click .

---

**TIP** You can select multiple integrations by using CTRL + click.

---

The selected integration is moved to the **Enabled Integrations** list and displays in bold type.

- 4 To activate the selected integration(s), click **OK**. The selected integration is activated.
- 5 To complete the activation, restart your computer.

#### **To disable integrations using the graphical user interface**

- 1 Open the Integrity Client and select **File > Integrations**.

---

**IMPORTANT** If the command is not visible, you must enable it in the active viewset. For information on viewsets, see the *Integrity Getting Started Guide*.

---



The **Enable/Disable Integrations** dialog box displays.

- 
- 2 In the **Enabled Integrations** list, select the integration(s) that you want to disable.

---

**TIP** You can select multiple integrations by using CTRL + click.

---

- 3 To move the selected integration(s) to the **Disabled Integrations** list, click . To move all enabled integrations to the **Disabled Integrations** list, click . The selected integration is moved to the **Disabled Integrations** list.
- 4 To disable the selected integration, click **OK**. The selected integration is disabled.
- 5 To complete the process, restart your computer.

## Before Using an Integration

Note the following when using Integrity integrations for configuration management:

- This guide assumes you know how to use your chosen IDE. For more information on using your tool and its version control features, refer to the product documentation for that IDE.
- When you select a version control command in your IDE, you may be prompted to provide the name and port number of the Integrity Server you want to connect to.
- Most integrations provide version control features and indicators. For example, in Sybase PowerBuilder, a green check mark beside a file indicates that the file is currently checked out. In addition, most integrated tools support version control commands through shortcut menus when you right click a file.
- Integrity integrations only work in the graphical user interface.
- At this time, only Windows-based IDEs for Integrity are supported.
- Do not create more than one Integrity Sandbox in a directory. The integrations use implicit Sandbox detection, so version control commands may fail if there is more than one Sandbox in a directory.
- When you create an Integrity configuration management project and Sandbox, the project can reside anywhere on the Integrity Server. Any files operated on by the integration must be in a directory with a Sandbox, or the IDE project file must be in the tree that is visible to the Sandbox, for example, `c:/projects/vc/project.dsp` and `c:/projects/project.pj`.
- When you add members to an Integrity configuration management project, the integration may prompt you to add a description for each member. Adding a description is optional, since the description appears in the **Create Archive** dialog box.
- Do not create or use sparse Sandboxes in Integrity when using an integration.
- To access advanced configuration management functionality, such as checkpointing a project, open the Sandbox in the Integrity graphical user interface.
- When you perform a version control operation in an IDE, the focus shifts to an Integrity dialog box or window; however, you may have to manually shift the focus back to the IDE.
- Subprojects are not automatically created by integrations. To use subprojects, create your project structure in Integrity, without adding any members, and use existing Sandboxes for the project.

---

## IDE Integration Tips

### To manually force the storage of archives in binary format

- 1 Locate your `IntegrityClient.rc` file in the following Windows directory:

```
C:/Documents and Settings/UserDirectory
```

- 2 Add the following line to the file:

```
integrations.DefaultBinaryFileExtensionList=value
```

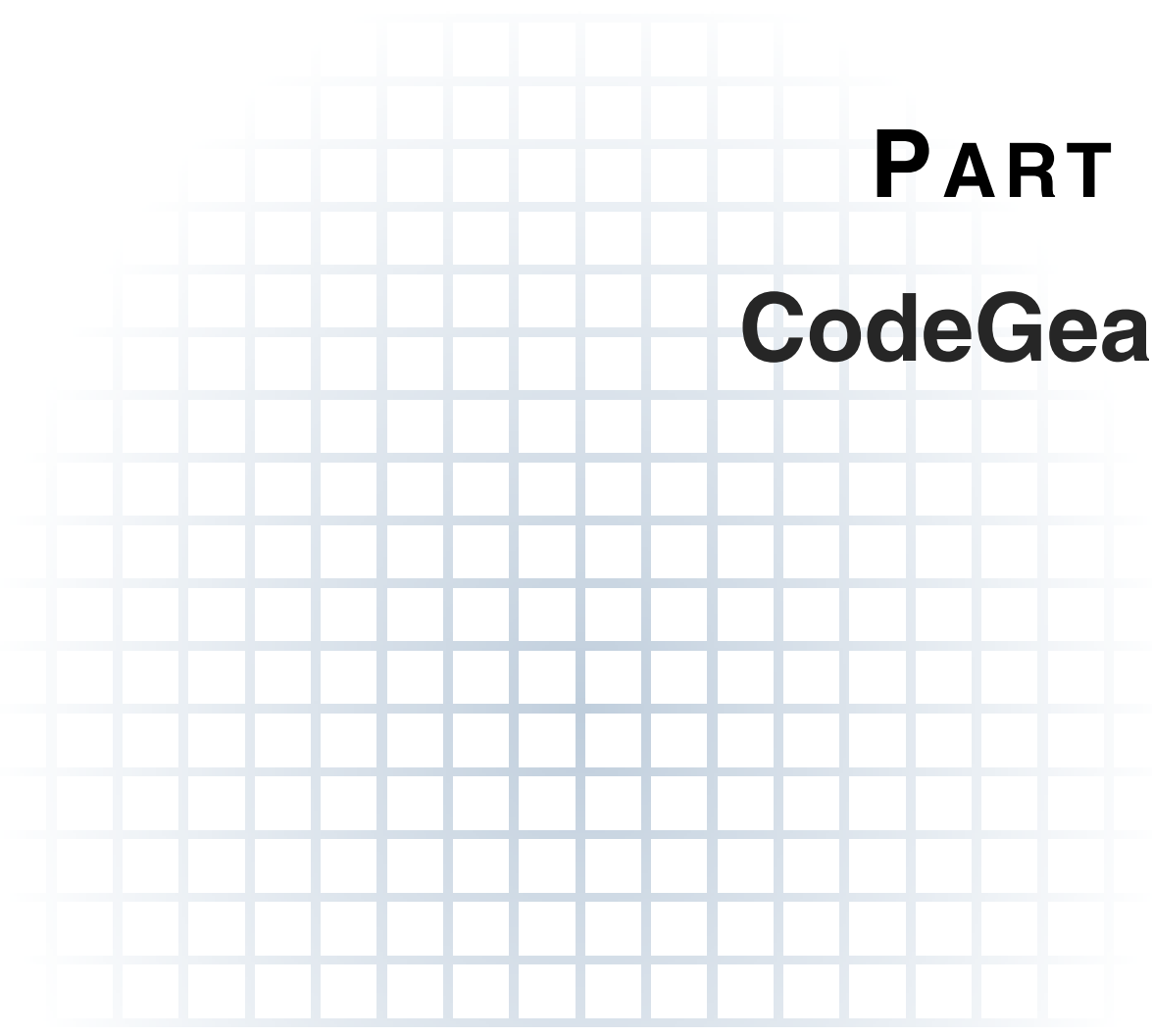
where *value* specifies the file extensions you want stored in binary format, separated by the “|” symbol. For example:

```
integrations.DefaultBinaryFileExtensionList=  
exe|ocx|frx|doc|bmp|jpg|gif|wri|apf
```

- 3 Save the `IntegrityClient.rc` file and restart the Integrity Client.

## Where To Go From Here

To Use the Integration With ...	See ...
CodeGear Delphi Architect	“CodeGear Delphi Architect” on page 8
CodeGear JBuilder	“CodeGear JBuilder” on page 14
IBM Rational - Eclipse Platform	“IBM Rational Eclipse Platform” on page 17
IBM Rational Rose	“IBM Rational Rose” on page 49
Microsoft Visual Studio (SDK)	“Microsoft Visual Studio (SDK)” on page 54
Microsoft Visual Studio .NET	“Microsoft Visual Studio .NET” on page 86
Microsoft Visual Basic	“Microsoft Visual Basic” on page 102
Microsoft Visual C++	“Microsoft Visual C++” on page 106
Microsoft Project	“Microsoft Project” on page 110
Microsoft Word	“Microsoft Word” on page 130
Microsoft Excel	“Microsoft Excel” on page 144
Sybase PowerBuilder	“Sybase PowerBuilder” on page 163
HP Quality Center—Defect Module	“HP Quality Center Requirements and Defect Modules” on page 169
HP Quality Center—Requirements Module	“HP Quality Center Requirements and Defect Modules” on page 169
CA Endeavor	“CA Endeavor” on page 185



# **PART I**

# **CodeGear**

This chapter provides information on using the Integrity integration with CodeGear Delphi Architect 2005 (Delphi), including key considerations and the various Integrity operations and actions you can perform using the Delphi Commit Browser.

This chapter provides information on following:

- “Key Considerations” on page 9
- “Configuring the CodeGear Delphi Integration” on page 9
- “Using the CodeGear Delphi Integration” on page 9
- “Delphi Architect Commit Browser” on page 10
- “Creating an Integrity Project” on page 11
- “Creating a Sandbox” on page 11
- “Adding a Member to an Integrity Project” on page 12
- “Checking Out Members” on page 12
- “Checking In Members” on page 12

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---



---

## Key Considerations

Note the following when using the Delphi integration:

- When you add a project to source control, the Sandbox location you specify in Integrity must be the original path the project was saved to in Delphi. You must not edit this location.
- Do not rename files in Delphi. Instead, in Integrity, use the **Rename** command, then use the **Resynchronize Member** command in Delphi.
- When performing a check out, check in, add member, drop member, resynchronize, or revert operation, if the file is a form or its associated **.pas** file, the operation is performed on both the **DFM** and **PAS** files. Similarly, when performing an Integrity operation on a Delphi project file (**DPR**), its associated **RES** files are included in the operation.
- You cannot double click a Delphi member revision in the Integrity **Member History** view to open it. Instead, open the revision from within a Delphi session.
- If the Integrity GUI is already running when you run a command from the integration that launches an Integrity GUI dialog box, the dialog box is not active in the application window. You must press **ALT + TAB** to make the dialog box active.

## Configuring the CodeGear Delphi Integration

The Integrity Client includes the **File > Integrations** menu action for enabling and disabling the integration with CodeGear Delphi Architect. You can select from a list of available integrations and enable or disable them, as required to work with your preferred application. For more information, see “Configuring Integrations” on page 3.

## Using the CodeGear Delphi Integration

When you select a source control command in your IDE, you may be prompted to provide the name and port number of the Integrity Server you want to connect to. You can access basic Integrity configuration management functionality from within Delphi by selecting one or more files in the **Project Manager** window and opening the files for viewing/editing. Then select **Tools > Team** from the menu to access the available commands. For detailed information on using Integrity configuration management commands, see the *Integrity User Guide*.

The following table summarizes the available commands:

Command	Function
<b>Check Out Files</b>	Equivalent to the Integrity <b>Member &gt; Check Out</b> command. Checks out the selected file. The Integrity <b>Check Out</b> dialog box displays.
<b>Check In Files</b>	Equivalent to the Integrity <b>Member &gt; Check In</b> command. Checks in the selected file. The Integrity <b>Check In</b> dialog box displays.
<b>Add Files</b>	Equivalent to the Integrity <b>Member &gt; Add</b> command. Adds the selected file to the Integrity configuration management project. The Integrity <b>Create Archive</b> dialog box displays.

Command	Function
<b>Remove Files</b>	Equivalent to the Integrity <b>Member &gt; Drop</b> command. Drops the selected file from the Integrity configuration management project. The Integrity <b>Drop Member</b> dialog box displays.
<b>Undo Check Out Files</b>	Equivalent to the Integrity <b>Member &gt; Revert</b> command. Replaces the selected working file with the revision that was checked out, as it appeared prior to modification, and unlocks the file. The Integrity <b>Overwrite</b> dialog box may appear.
<b>Commit Browser</b>	Launches the <b>Commit Browser</b> dialog box for the open Delphi project that is under source control with Integrity. For available command operations, see "Delphi Architect Commit Browser" on page 10.
<b>Place Project into Source Control</b>	Equivalent to the Integrity <b>Project &gt; Create</b> and <b>Sandbox &gt; Create</b> command. Creates an Integrity configuration management project and Sandbox from the selected Delphi project. <b>Note:</b> This command can be performed only once.
<b>Pull Project from Source Control</b>	Equivalent to the Integrity <b>Sandbox &gt; Create</b> command. Creates a Sandbox from a project on the server.
<b>Run SCC Integration</b>	Equivalent to launching the Integrity graphical user interface if there is no project open that is currently under source control. If a project under source control is open, the <b>Run SCC Integration</b> command displays the <b>Sandbox</b> view.

## Delphi Architect Commit Browser

The integration also provides Integrity functionality from the Delphi Commit Browser, available from **Tools > Team > Commit Browser**. For detailed information on the Commit Browser, consult the CodeGear Delphi product documentation. For detailed information on the Integrity configuration management commands, see the *Integrity User Guide*.

The following table summarizes the available actions you can perform on members through the Commit Browser:

Action	Function
<b>No Activity</b>	No Integrity command performed on the member.
<b>Commit</b>	Equivalent to the Integrity <b>Member &gt; Check In</b> command. Checks in the member, updating the head revision.
<b>Check Out</b>	Equivalent to the Integrity <b>Member &gt; Check Out</b> command. Checks out the head revision of the member.
<b>Undo Check Out</b>	Equivalent to the Integrity <b>Member &gt; Revert</b> command. Reverts deferred operations.
<b>Add</b>	Equivalent to the Integrity <b>Member &gt; Add</b> command. Adds members to a project.
<b>Remove</b>	Equivalent to the Integrity <b>Member &gt; Drop</b> command. Drops members from a project.
<b>Get Latest</b>	Equivalent to the Integrity <b>Member &gt; Resynchronize</b> command. Gets the latest version of the selected file and puts it in your working directory. The Integrity <b>Overwrite</b> dialog box may appear.

---

Integrity information is also available from the following Commit Browser tabs:

- **Summary Content** specifies revision description for all members.
- **Individual Comment** specifies the revision description for the selected member.
- **Local Source** displays the working file contents of the selected member.
- **Diff and History**
  - **Show Differences** launches the difference tool specified in the Integrity Client preferences.
  - **Show History** displays the member history in Integrity.

## Creating an Integrity Project

### To create an Integrity project in Delphi

- 1 In Delphi, select the project you want to put under version control.
- 2 Open the project file.
- 3 Select **Tools > Team > Place Project Into Source Control**. The **Specify the Project to Create** dialog box displays.

---

**NOTE** If you are using an existing project, click **Cancel** and select that project from the **Create Sandbox Wizard** in step 5.

---

- 4 Create a project as described in the *Integrity User Guide*. The **Create Sandbox Wizard** displays.
- 5 Create a Sandbox as described in the *Integrity User Guide*. The **Create Archive** dialog box displays.
- 6 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.

## Creating a Sandbox

### To create a Sandbox in Delphi

- 1 In Delphi, select the project under version control that you want to create a Sandbox from.

---

**IMPORTANT** When you add a project to source control, the Sandbox location you specify in Integrity must be the original path the project was saved to in Delphi. You must not edit this location.

---

- 2 Open the project file.

---

**TIP** You can also create a Sandbox directly from Integrity.

---

- 3 Select **Tools > Team > Pull Project From Source Control**. The **Create Sandbox Wizard** displays.
- 4 Create a Sandbox as described in the *Integrity User Guide*.

---

## Adding a Member to an Integrity Project

### To add a member to an Integrity project in Delphi

---

**NOTE** You can only add one member at a time to an Integrity project in Delphi.

---

Select **Tools > Team > Add Files**. The **Add File(s)** dialog box displays. Select the files you want to add, then click **OK**. The **Create Archive** dialog box displays. Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.

## Checking Out Members

### To check out members in Delphi

Select **Tools > Team > Check Out Files**. The **Check Out File(s)** dialog box displays. Select the files you want to check out, then click **OK**. The **Check Out** dialog box displays. Check out each member as described in the *Integrity User Guide*.

## Checking In Members

### To check in members in Delphi

To check in one file, select **Tools > Team > Check In Files**. The **Check In File(s)** dialog box displays. Select the files you want to check in, then click **OK**. The **Check In** dialog box displays. Check in each member as described in the *Integrity User Guide*.



---

## CHAPTER THREE

# CodeGear JBuilder

---

# 3

The Integrity integration with CodeGear JBuilder Enterprise 2007 and 2008 allows you to access the configuration management functionality of Integrity through your JBuilder development environment.

CodeGear JBuilder Enterprise 2007/2008 is based on the Eclipse Platform. For detailed information on configuring and using an Eclipse-based integration, see “IBM Rational Eclipse Platform” on page 17.

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---





# **PART II**

# **IBM**



The Integrity integration with Eclipse Platform and the IBM® Rational® Software Development Platform series allows you to access Integrity version control commands through several IBM products that use the IBM Rational Platform technology, based on the Eclipse open source initiative.

---

**IMPORTANT** Integrations with the IBM Rational Eclipse platform work only on Windows and Linux operating systems supported for the Integrity Client. For more information on supported operating systems, go to the Integrity Support Center:

<http://www.ptc.com/support/integrity.htm>

---

Eclipse Team Support provides flexibility in designing and implementing support of the repository on the workbench. In turn, the Integrity integration is easier to use and provides greater control.

To assist you in using the integration, the following topics are discussed:

- “Overview” on page 18
- “Supported Versions” on page 18
- “Before You Start” on page 18
- “Configuring the IBM Rational Eclipse Integration” on page 19
- “Using the Integration” on page 23
- “Best Practices” on page 46
- “Limitations” on page 47

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Overview

The Integrity integration with Eclipse Platform and the IBM Rational® Software Development Platform allows you to access Integrity version control commands through several IBM products that use the IBM Rational Platform technology, based on the Eclipse open source initiative.

The integration also allows you to access Integrity version control commands through several open source and commercially available development products, built as Eclipse Platform plug-ins.

The integration includes software developed by the Eclipse Project. For more information on the Eclipse Project, browse to:

<http://www.eclipse.org>

## Supported Versions

The integration with Eclipse Platform and IBM Rational® Software Development Platform is designed to work with the following:

- A supported and licensed IBM Rational Software Development Platform product
- A product built on the Open Source Eclipse Platform
- Any commercial development tool, certified as Ready for IBM Rational, built on the Eclipse Platform

---

**IMPORTANT** For more information on supported versions of Eclipse Platform and IBM Rational Software Development Platform, go to the Integrity Support Center:

<http://www.ptc.com/support/integrity.htm>

---

## Before You Start

Before you use the integration, note the following:

- This guide assumes you know how to use Eclipse/IBM Rational Software Development Platform products, Integrity, and Implementer (if working with Implementer change packages). For more information about using a product, refer to the appropriate documentation from the product vendor.
- Java Development Kit (JDK) 1.5 or higher must be installed.
- Read “Best Practices” on page 46 for more detailed information about using the Eclipse integration.
- If the Integrity Client is shut down and you run a command that requires Integrity, the Integrity Client automatically initializes.

---

# Configuring the IBM Rational Eclipse Integration

For integrations based on Eclipse Platform, including IBM's Rational Architecture Management and Construction solutions such as RAD and RSA, you enable the integration using the **Add Site** function. For more information, see "To enable the Integrity integration with Eclipse 3.4" on page 19 and "To enable the Integrity integration with Eclipse 3.5+" on page 20.

---

**IMPORTANT** To make use of a new version of the Eclipse integration, you must either configure Eclipse to perform updates automatically, or force a manual update to pick up the new version of the integration. In general, you can configure updates by selecting **Help > Check for Updates**. For more information on updating software, see the Eclipse product documentation.

With the release of Integrity 10.0, the default location of the Integrity integration installation has changed and you must point Eclipse to the new location as the update site. For example on Windows, the new default location is:

```
Program Files\Integrity\IntegrityClient10\integrations\  
IBM\eclipse_3.4\eclipse
```

When uninstalling a previous version of the Integrity Client, you must also remove the previous update site within Eclipse or an error occurs (**No repository found**) after installing the new integration.

---

If you are working with Implementer, the Integrity Server must also be configured to allow remote API connections to the server. For more information, see the *Integrity Integrations Builder API Guide*.

After enabling the integration, you can also set general preferences and additional options related to change packages and file annotations.

This section discusses the following topics:

- "Enabling Access to Java API Libraries (Linux Only)" on page 19
- "Enabling the Integration" on page 19
- "Setting Preferences" on page 21
- "Deactivating the Integration" on page 23

## Enabling Access to Java API Libraries (Linux Only)

On the Linux platform, the Eclipse integration requires access to the Java API libraries.

Set the `LD_LIBRARY_PATH` environment variable to the following:

```
export LD_LIBRARY_PATH=<Integrity Client installdir>/lib/linux
```

where `<Integrity Client installdir>` is the path to the Integrity Client installation location.

## Enabling the Integration

### To enable the Integrity integration with Eclipse 3.4

- 1 In Eclipse, select **Help > Software Updates**. The **Software Updates and Add-ons** dialog box displays.
- 2 Under the **Available Software** tab, click **Add Site**. The **Add Site** dialog box displays.

- 
- 3 To select the directory where you placed your plug-ins as an extension location, click **Local** and navigate to the following directory:

`<Integrity Client installdir>/integrations/IBM/eclipse_3.4/eclipse`

where `<Integrity Client installdir>` is the path to the directory where you installed the Integrity Client.

- 4 To enable the integration, check the box next to the newly-added integration and then click **Install**. Eclipse copies the selected plug-in to the **Eclipse Features and Plugins** directory.
- 5 Restart the Eclipse workbench to enable the integration.

---

**NOTE** You can also use Eclipse drop-ins to add a plug-in. For more information on drop-ins, consult the Eclipse product documentation.

---

### To enable the Integrity integration with Eclipse 3.5+

---

**NOTE** The Classic Update method is not recommended for installing this version of the Eclipse integration. Before installing the integration, you can confirm the setting for the Classic Update option by selecting **Window > Preferences** and expanding the **General** node. Click the **Capabilities** subnode and confirm that the **Classic Update** option is cleared.

---

- 1 In Eclipse, select **Help > Install New Software**. The **Install** dialog displays.
- 2 Click **Add**. The **Add Site** dialog box displays.
- 3 Click **Local**. The **Browse For Folder** dialog box displays.
- 4 Browse to the following folder on the Integrity Client machine:  
`<Integrity Client installdir>/integrations/IBM/eclipse_3.4/eclipse`  
where `<Integrity Client installdir>` is the path to the directory where you installed the Integrity Client.  
Click **OK** and the selected folder displays in the **Location** field.
- 5 In the **Name** field, enter a name for the integration (for example, *Integrity*) and click **OK**.
- 6 In the **Work with** list, select the Integrity integration location.

---

**TIP** If the Integrity integration location does not automatically appear in the **Work with** list, click the link for **Available Software Sites** and select the site you just added. The Integrity integration then becomes available for selection in the **Work with** list.

---

- 7 From the list, check **collaboration** check box. The check box for **Integrity Eclipse Integration** is automatically selected.
- 8 Click **Next**. The **Install Details** dialog box displays with the Integrity integration in the list of items to be installed.
- 9 To install the integration, click **Finish**.
- 10 Follow the remaining steps of the installation wizard to accept content and the license agreement.

In the **Software Updates** dialog box, click **Yes** to restart the Eclipse workbench and complete the installation. When Eclipse restarts, the **Integrity** menu is available for selection.

---

## Setting Preferences

Integration specific preferences allow you to select settings for Integrity commands. The Integrity preferences become the default.

In addition, the File Content and Ignored Resources preferences affect functions in your workspace related to the integration.

File Content preferences allow you to specify a file type and the type of content in that particular file (binary or ASCII content).

Ignored Resources preferences allow you to specify resource name patterns that you do not want to add to Integrity version control. In addition, any file that is marked **derived** is ignored by Integrity. To see if a file is marked **derived**, check the file's properties.

Click **Restore Defaults** at any time to clear the changes you made.

### To set Integrity preferences

- 1 In your workspace, select **Window > Preferences**. The **Preferences** dialog box displays.
- 2 Under **Team**, select **Integrity Source**. The **Integrity Source** panel displays.
- 3 Make the necessary changes to the preferences:

---

**NOTE** Any modifications to change package preferences will require that you restart your Eclipse workspace.

---

#### ■ Updating non-locked files

- **Lock file** performs the Integrity **Lock** command, allowing you to edit the file. Enable this option to implicitly check out a file while editing it. This option is enabled by default.

---

**NOTE** If you are using non-exclusive locks, you can lock a file; however, other users are not prevented from locking the same file.

---

- **Make file writeable** performs the Integrity **Make Working File Writable** command, allowing you to edit the working file, but not preventing other users from locking the file. Enable this option to edit a file locked by another user when you have no intention of checking the file back in. If you intend to check the file in later on, you can lock the file or associate the modified working file with a change package, and then submit your changes.

#### ■ Annotation of Subprojects

The Eclipse integration provides specific annotations for shared subprojects, including development path names for variant projects and checkpoint revision numbers for build projects. When changes are made to a project or subproject configuration from outside the Eclipse integration, those changes are dynamically displayed within the integration.

Only projects and subprojects under source control are annotated. Ordinary folders/subfolders are not annotated. In addition, to correspond with behavior in the Integrity Client, children of shared subprojects are not marked as shared.

The integration also provides annotation for the configuration details of shared and configured subprojects. For variant subprojects, the annotation shows development path names. For build subprojects, the annotation shows checkpoint revision numbers. Normal subprojects configured within a build or variant subproject do not show any annotation.

---

To set the type of annotation required, click to select from the following options under **Annotation of Subprojects**:

- **Display configuration at project level only.**
- **Annotate shared subprojects. (Default)**
- **Annotate all subprojects with configuration information.**

---

**NOTE** To avoid displaying more information than may be generally required, the default setting provides annotation for shared subprojects only. After setting your preference options, you must restart Eclipse to have the new annotations displayed in the integration.

---

- **Use Change Package**

The **Use Change Package** option specifies the use of a default change package when performing Integrity commands that use a change package. This option is enabled by default.

When this option is enabled, all changes must be submitted using a change package and you no longer have the option of submitting individual changes using the **Submit Changes** command. To ensure the correct behavior, the **Use Change Package** option should be used only if the policy for `ChangePackagesEnabled` is set to true on the Integrity Server.

- **When performing Integrity commands that require a change package, prompt for an active change package** provides for prompting when performing any software configuration management operation that requires a change package, if there is no active change package context. You can only enable this option if the **Use Change Package** option is enabled. The default setting for this option is false (disabled).

When this property is enabled, you are prompted to choose or create a change package if active change package tracking is enabled and there is no active change package set. If you do not choose or create a change package when first prompted, you are re-prompted when you attempt further file operations that normally require a change package (such as additional edits or saving the file). You are also prompted when creating or dropping subprojects, when moving or renaming files, when adding or dropping files, and when running the Sharing Wizard to add files.

---

**IMPORTANT** To return the expected prompt messages when obtaining locks and change packages, ensure that change package options are configured consistently between the Integrity Server and Eclipse (that is, if change packages are mandatory in Eclipse, then they are also configured as mandatory on the Integrity Server). If the change package policies are not consistent, prompting may not occur as expected when locking and changing members.

---

- **Drop Integrity Sandbox when Eclipse project is deleted** specifies that if your Eclipse project is deleted, the associated Sandbox is also dropped. The default setting for this option is true (enabled).

4 To set additional preferences, click **Integrity Preferences**. The **Preferences Configuration** dialog box displays.

5 Click **Apply** to save the changes.

6 Click **OK** to close the **Preferences** dialog box.

### To set File Content preferences

1 In your workspace, select **Window > Preferences**. The **Preferences** dialog box displays.

2 Under **Team**, select **File Content**. The **File Content** preferences display.

- 
- 3 To add a file type, click **Add Extension**. To add a file type via a file name, click **Add Name**. To delete a file type from the list, click **Remove**. To modify the file type content, click **Change**.
  - 4 To save the changes, click **Apply**.
  - 5 To close the **Preferences** dialog box, click **OK**.

#### To set Ignored Resources preferences

- 1 In your workspace, select **Window > Preferences**. The **Preferences** dialog box displays.
- 2 Under **Team**, select **Ignored Resources**. The **Ignored Resources** preferences display.
- 3 To add a pattern, click **Add Pattern**. To delete a pattern from the list, click **Remove**.
- 4 To save the changes, click **Apply**.
- 5 To close the **Preferences** dialog box, click **OK**.

## Deactivating the Integration

To deactivate the Integrity integration with Eclipse IBM platform, you remove the installed Integrity software plugin.

#### To deactivate the Eclipse integration

- 1 Select **Help > About Eclipse SDK**.
- 2 In the **About Eclipse SDK** dialog box, click the **Installation Details** button. The **Eclipse SDK Installation Details** dialog box displays.
- 3 Under the **Installed Software** tab, highlight **Integrity Eclipse Integration** and click **Uninstall**. The **Uninstall** dialog box displays with the Integrity integration in the list of items to be uninstalled.
- 4 To uninstall the integration, click **Finish**.

---

**IMPORTANT** As part of uninstalling the integration, you should also remove the Integrity integration update site. To remove the Integrity update site, select **Window > Preferences** and under the **Install/Update** node, choose **Available Software Sites**. In the **Available Software Sites** panel, highlight the Integrity plugin location, and then click **Remove**.

---

## Using the Integration

This section provides information on how to use the Integrity integration with Eclipse. The following topics are discussed:

- “Online and Offline Mode” on page 24
- “Setting Up an Integrated Workspace” on page 25
- “Working in an Integrated Workspace” on page 30
- “Integrity Commands” on page 36
- “Refactoring” on page 40
- “Comparing Revisions” on page 41
- “Team Synchronizing” on page 42

---

## Online and Offline Mode

When you start Eclipse, the Integrity Client automatically attempts to establish a connection with the Integrity Server. If the client cannot establish a connection with the server, you can still work in Eclipse.

When working in Eclipse in online mode without a server connection, all Integrity views and commands that affect the Integrity repository stay enabled. If you perform a command that requires a server connection, the Integrity Client attempts to reconnect, prompting you to enter your credentials if required.



If you know that you will not have access to the server for long periods of time, for example, if you are working remotely without an Internet connection, you can switch to working in offline mode. In this mode, you can perform any commands that do not require a server connection, such as editing a file (the file is made writable, but is not locked). Once you switch to online mode, all disabled Integrity commands and views become active again, allowing you to resynchronize your changes with the Integrity repository.

---

**NOTE** You will be automatically switched to offline mode in the following situations:

- You cancel your server connection dialog without entering any credentials.
  - The Integrity Client becomes unavailable.
- 

### Switching Between Online and Offline Modes

To toggle online and offline mode, select **Integrity > Work Online/Offline**, or click the online () or offline () icons in the Integrity trim.

---

**NOTE** The Integrity trim displays in the bottom-right corner of the workbench by default; however, you can drag the trim anywhere in the workbench.

---

When switching from offline to online mode, note the following:

- Switching from offline to online mode refreshes the status of the workspace, retrieving the latest decorators. Depending on how many files or Eclipse projects are visible, this process may take a long time; however, it occurs in the background and does not prevent you from working in Eclipse.
- After switching from offline to online mode, you cannot submit changes in the Synchronize view while the Integrity status cache refreshes. Before you submit changes, check the status of the Eclipse progress bar and ensure there is no Integrity command activity.

### Working in Offline Mode

When working in offline mode, note the following:

- Depending on the information available in your resource cache before you switch to offline mode, file decorators may not update correctly, or at all, when you select files or make changes to them. Locks made by other users, new revisions, and incoming changes do not update file decorators.
- You cannot move outgoing changes to change packages.
- You cannot rename or move files that are under source control. You cannot rename or move files that are not under Integrity version control, but exist in an Eclipse project that is under Integrity version control. You can refactor (add, drop, rename, or move) ignored resources.



- 
- Integrity information does not display in **Properties** dialog boxes.
  - If you delete a subproject, the Sandbox icon is removed from the subproject. While in offline mode, do not perform resource operations that affect the repository, such as deleting or renaming resources.

## Setting Up an Integrated Workspace

This section discusses the specific details of setting up projects in your integrated workspace, including placing projects under Integrity version control, and working with Project Sets.

### Placing Eclipse Projects Under Integrity Version Control

Projects must be placed under Integrity version control, which requires an Integrity project and Sandbox. Using the Team Support approach, you can work through a wizard that creates the Integrity projects and Sandboxes, and adds existing project files as members.

---

**IMPORTANT** When you create an Integrity project and Sandbox, the project can reside anywhere on the Integrity Server, but the Sandbox must reside in the same directory as the Eclipse project.

---

Once an Eclipse project is under Integrity version control, you can perform Integrity operations, such as checking in files and checkpointing projects.

### To create an Integrity configuration management project and Sandbox in your workspace

- 1 In your workspace, select the Eclipse project you want to place under Integrity version control.
- 2 Right click and select **Team > Share Project**. The **Share Project** panel displays.
- 3 Select **Integrity** as the repository type.
- 4 Click **Next**. The **Integrity Sharing Wizard** displays.
- 5 Select one of the following options:
  - **Activate the integration for an existing Integrity Source Sandbox** initiates Integrity and opens the Sandbox you previously created for the project in your workspace. This option is selected by default. If a Sandbox does not exist, this option is disabled.
  - **Create a new Sandbox for an existing Integrity Source project** creates a new Sandbox for an existing Integrity project under Integrity version control. If you select this option, you are prompted to choose an Integrity project and, optionally, a development path.
  - **Create a new Integrity Source project and Sandbox** creates a new Integrity project accessible to all users and a Sandbox in your Eclipse workspace. You must be granted the correct project permissions for this functionality to work properly.
- 6 By default, the option for **Add all files when creating the new Integrity Source project** is enabled. You can disable this option if you want to manually add each file in your project to Integrity version control.

This option is only available if you selected the **Create a new Integrity Source project and Sandbox** option in step 5.

- 7 Click **Next**. The options for creating an Integrity project display.

---

8 For the option you selected in step 5, choose one of the available options:

- **Create a new top-level Integrity Source project** and click **Finish**. The **Specify Project** dialog box displays. In the **File name** field, specify the name of the project you want to create on the Integrity Server.
- **Create a new subproject of an existing Integrity Source project** and click **Next**. Choose the project to create the subproject against. Optionally, you can select one of the project's development path. To create the subproject, click **Finish**.

---

**NOTE** By default, if you create a chain of nested directories, all subdirectories in the directory are also created as subprojects. If a subdirectory does not include a `.pj` file, `project.pj` is also added to the subproject string.

---

9 If your project contains files and you are using change packages, you are prompted to specify a change package to associate with the files. Select an existing change package, or click **Create** to create a new change package, then click **OK**.

If you are not using change packages, proceed to the next step. The **Create Archive** dialog box displays.

10 Modify the **Create Archive** options.

---

**NOTE** If Integrity finds an existing archive, the **Existing archive detected** dialog box displays.

---

Integrity automatically creates a Sandbox with the project name in your Eclipse workspace.

11 If your project contains files and you associated them with a change package, you must commit them to the Integrity repository by submitting the change package associated with the files via the Synchronize view. For more information, see “Team Synchronizing” on page 42. If your project contains file and you are not using change packages, submit the changes by selecting **Integrity > Submit Changes**. For more information, see “Integrity Commands” on page 36.

### To import a project from Integrity

If you want to start using an existing Integrity project, you can import it to create a Sandbox in your workspace.

---

**NOTE** This is the only method of sharing a project that allows you to rename the project in your workspace. This enables you to import multiple versions of the same project.

---

- 1 In your workspace, select **File > Import**. The **Select** panel of the **Import** wizard displays.
- 2 Under **Integrity**, select **Projects from Integrity**.
- 3 Click **Next**. The **Choose the Integrity project to import** panel displays.
- 4 Select the project to import. All Eclipse projects currently under Integrity version control are listed. You can use the **Filter** field to filter the projects by name.

---

**NOTE** If an Eclipse project is missing from this list, it means that the project description file (`.project` file) has not been put under Integrity version control.

---

- 
- 5 Specify the configuration of the project to import. If importing a variant project, specify the **Development Path Name** where the project is located. If importing a build project, specify the **Revision** number or **Label** applied to the project.

---

**NOTE** If you are importing a variant or build configuration of a project that you already have in your workspace, you must edit the project name in the **Project Name** field since you cannot have two projects with the same name in the same workspace.

---

- 6 Specify the location to import the project to. This can either be the default workspace location or a location specified by you.

In either case, the projects are placed in:

*selectedLocation/projectName*

where *projectName* is the name of the project as specified in the **Project Name** field.

---

**NOTE** The relative paths of imported projects are not maintained.

---

- 7 Edit the **Project Name** if required.

---

**NOTE**

- If there are dependencies between projects, renaming the project will result in compile errors.
  - The name in the `.project` file is not changed when you edit the project name in this field.
- 

- 8 Click **Finish**.

A directory is created for the project in the specified location, a Sandbox is created within that directory and the Eclipse project files are added.

Importing an Integrity project results in an identical Eclipse project structure and information within your workspace.

---

**NOTE** The Import wizard is only available if the Eclipse integration is connected to an Integrity Server that uses the database repository option. If the integration is connected to a server that uses the RCS-style repository option, the Import wizard is not available and an error message displays when attempting to the run the wizard.

---

## Unassociating Eclipse Projects from Integrity Version Control

If you place Eclipse projects under Integrity version control on a temporary basis, for example, as part of a pilot or proof of concept, you can unassociate the projects from Integrity once you are done. This removes the association between the Eclipse project and the Integrity project and Sandbox.

### To unassociate an Eclipse project from Integrity version control

- 1 In your workspace, select the Eclipse project that you want to unassociate from Integrity version control.
- 2 Right click and select **Team > Unshare Project**.

You can no longer perform Integrity operations on the project.

---

## Sharing Projects With Project Sets

The Team Support approach also allows for collaboration on projects. In an integrated workspace, you can share groups of Integrity projects with other users by creating a Project Set or by exporting groups of projects in your workspace. Other users can then import the Project Set File (`.psf`) and the projects contained in the Project Set are automatically created for them.

This includes the creation and population of the necessary Sandboxes, based on the Integrity projects referenced in the Project Set. Project sets provide a simple method for team members to share their workspaces.

### Key Considerations

- The Eclipse project must be under Integrity version control before you create a Project Set.
- Once the `.psf` is created, do not attempt to edit this file unless errors occur when importing using the file.
- You can export subSandboxes with Team Project Sets. Developers can share workspaces in their entirety. Importing a Team Project Set results in identical Eclipse project structure and information within the new workspace, and creates corresponding common root Sandboxes. For example, if a top level Sandbox and a subsandbox are exported using the Team Project Set feature, they are recreated in the same hierarchy when the Team Project Set is imported.
- If a `.project` file is not under Integrity version control and you attempt to export a Project Set, an error message displays (the `.psf` file is created, but is mostly empty). If the `.project` file is under Integrity version control, but has been modified since it was last checked in, exporting the Project Set displays a warning message; however, the `.psf` file still exports. As long as the `.project` file exists, the export should complete successfully.
- If an Eclipse project referenced by the Team Project Set does not have its `.project` file under version control and you attempt to import the Project Set, an error message displays. A Sandbox is created for the project in the Integrity Client, even though the project does not exist in your workspace. To successfully import the Project Set, you must complete the following steps:
  - Drop the Sandbox in the Integrity Client
  - If you still want to import the project, add the `.project` file as a member in the Integrity Client
  - If you do not want to import the project, remove the appropriate entry from the `.psf` file. Re-distribute the updated `.psf` file to all users of the Team Project Set.
  - Re-import the Project Set
- If the `.classpath` file is not under Integrity version control, any imported Java projects fail to compile because the build path does not include the JRE System Library. Once the `.classpath` file is in the project, subsequent imports include the new file.

### To create a Project Set in your workspace

- 1 In your workspace, select **File > Export**. The **Export** wizard displays.
- 2 Under **Team**, select **Team Project Set**.
- 3 Click **Next**. The **Team Project Set** dialog box displays.
- 4 From the list, select the projects you want to include in the Project Set.

- 
- 5 In the **File name** field, provide the path and file name for the `.psf`, or click **Browse**, to browse to a location.
  - 6 Click **Finish**. The Project Set is created. The `.psf` is ready for distribution to other team members who can import it into their workspaces.

### To import a Project Set

- 1 In your workspace, select **File > Import**. The **Import** wizard displays.
- 2 Under **Team**, select **Team Project Set**.
- 3 Click **Next**. The **Import a Team Project Set** dialog box displays.
- 4 In the **File name** field, type the path and file name for the `.psf` you want to import, or click **Browse** to browse to the `.psf`.
- 5 If you do not want the Integrity progress dialog box to display while the Sandboxes are being created, select the **Run the import in the background** option. This option is useful when importing a large `.psf` file that requires many Sandboxes to be created.
- 6 Click **Finish**. The **Integrity Import Team Project Set Wizard** displays, allowing you to create Sandboxes in a common root location.
- 7 Click **OK** to create the Sandboxes in a common root location. By default, the workspace is specified as the common root location. You may also specify or browse to a location for the new Sandbox, where it is created automatically in the directory you specified.

Under this root location, directories are created for each project in the project team set. Sandboxes are then created within those directories and the associated Eclipse project files are added.

---

**NOTE** If you specify an invalid location as the common root, Eclipse deciphers that invalid entry to find a valid location.

---

If the description file for a project in the Team Project Set is not under Integrity version control, the import fails with the following error message:

```
There is no project description (.project) file in the repository
for <Eclipse project name> referenced by the Team Project Set file. Please use the
Integrity client to drop the Sandbox for <Eclipse project name>, resolve the
inconsistencies with the Team Project Set, and re-try your import.
```

To resolve inconsistencies with the Team Project Set, do one of the following:

- If you still want to import the project, add its `.project` file as a member in the Integrity Client
- If you do not want to import the project, remove the appropriate entry from the `.psf` file

### Specifying Directories or Files as Team Ignored

In addition to specifying ignored resource name patterns in your preferences, you can specify a particular directory or file as ignored. This is useful when you need a directory or file to be under Integrity version control in one situation but ignored in another.

For example, a Java resource bundle file (`*.properties`) is typically added to version control in source directories, but is also copied over to build directories, where it does not need to be under version control. You can specify the build directory as ignored.

---

Note the following:

- The first time you specify a directory or file as team ignored, an `.mkignore` file is created in the root directory of the Eclipse project. Once you submit changes to the project, the created `.mkignore` file is added to Integrity version control.
- Use the **Team** shortcut menu to specify directories or files as team ignored.
- Once under Integrity version control, the `.mkignore` file is editable only through Eclipse or after it is checked out in Integrity.
- Ignored files do not display icon decorators or annotations.
- With the exception of the **Remove from Ignore List** command, all Integrity commands are disabled when you select a team ignored resource.
- Glob patterns are unsupported in the `.mkignore` file.

### To specify a directory or file as team ignored

Right click the directory or file you want to ignore and select **Team > Add to Ignore List**. Icon decorators and annotations for the resources disappear.

---

**NOTE** You can only select one file at a time to specify as team ignored.

---

To remove a directory or file from the team ignored list, right click the directory or file and select **Team > Remove from Ignore List**. Icon decorators and annotations for the resources appear.

## Working in an Integrated Workspace

This section discusses the specific details of working with projects in your integrated workspace, including:

- “Integrity Label Decorations” on page 30
- “Displaying Integrity Information” on page 32
- “Working With Active Change Packages” on page 33
- “Managing Items With Integrity Worktray” on page 34

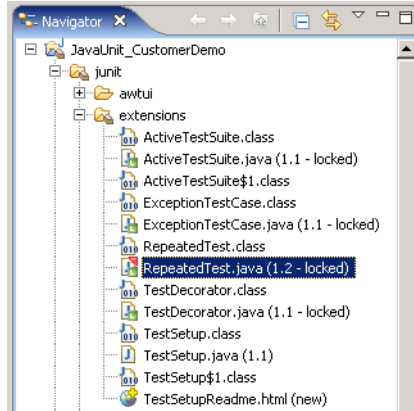
### Integrity Label Decorations

By default, label decorations (in the form of icons and annotations) appear on Eclipse projects and files, updating dynamically to display the latest Integrity version control status.

Note the following:

- If decorator rollups are enabled and a container (package, subproject, directory) contains modified files, the container displays a **Working file changed** decorator. Modified files include files with changes, moved or renamed files, and newly added files. In addition, files that are locked, but not modified are included. This is known as decorator rollup. By default, decorator rollup is disabled.
- Some label decorations display in certain views only. For example, when you delete a file, the Dropped Member decorator displays in the Synchronize view only.
- Label annotations include revision numbers, development path names for variant projects, and lock status. Integrity labels applied to revisions do not display as label annotations. For detailed Integrity information about a file, view the file’s properties (see “Displaying Integrity Information” on page 32).

- The Eclipse Platform integration provides specific annotations for shared subprojects, including development path names for variant projects and checkpoint revision numbers for build projects. When changes are made to a project or subproject configuration from outside the Eclipse integration, those changes are dynamically displayed within the integration. For detailed information on the available preferences for annotations, see “Annotation of Subprojects” on page 21.



### To toggle label decorations




In your workspace, select **Window > Preferences**. The **Preferences** dialog box displays. Under **General > Appearance**, select **Label Decorations**. From the list of available label decorations, toggle the check box for **Integrity Decorators** and click **OK**.








### To toggle decorator rollup

In your workspace, select **Window > Preferences**. The **Preferences** dialog box displays. Under **General > Appearance**, select **Label Decorations**. From the list of available label decorations, toggle the check box for **Integrity Decorators Rollup** and click **OK**.

### Icon Decorations

Icon decorations are appended to directory and file icons:

Decorator	Function
	<p>The <b>Sandbox</b> decorator indicates that the Eclipse project is under Integrity version control or the directory is a subproject under Integrity version control.</p> <p>A variant or build project also displays the corresponding development path or revision number.</p> <p>A file belonging to a project or subproject under Integrity version control displays the revision number, and, if applicable, status and change package (Synchronize view only), for example, <code>status.java (1.8 - locked [1:45])</code>.</p> <p>Packages and directories created or converted to subprojects also display the Sandbox decorator.</p>
	<p>The <b>Added Member</b> decorator and the annotation (<code>new</code>) indicates that the file is not one of the following:</p> <ul style="list-style-type: none"> <li>■ linked resource</li> <li>■ team ignored</li> <li>■ member of the Integrity repository (If a file is a deferred add, this icon displays. If a file is a pending add, this icon does not display.)</li> </ul> <p>When a file is committed to the Integrity repository, the revision number annotation displays and the Added Member icon disappears.</p>
	<p>The <b>Dropped Member</b> decorator indicates that the file is a candidate for dropping as a member from an Integrity project.</p> <p><b>NOTE:</b> Dropping the member deletes the file locally; however, it still exists in the Integrity repository. From the Synchronize view, the <b>Submit Changes</b> command drops the file from the Integrity repository.</p>

Decorator	Function
	The <b>Moved/Renamed Member</b> decorator indicates that the file is a moved and/or renamed member, and the move and/or rename is not yet committed to the Integrity repository.
	The blue lock decorator indicates that another user has a lock on the member revision and you do not have a lock on the member or working revision.
	The green lock decorator indicates that you have a lock on the member revision and no other users have locks on the same revision.
	The red lock decorator indicates that you have a lock on the working or member revision and another user has an exclusive lock on the member revision.
	The yellow lock decorator indicates one of the following: <ul style="list-style-type: none"> <li>■ you have an exclusive or non-exclusive lock on the member revision and another user has a non-exclusive lock on the member revision</li> <li>■ you have an exclusive or non-exclusive lock on the working revision, and your working revision is not the same as the member revision</li> </ul>
	The <b>Working file changed</b> decorator indicates that the working file has been modified. A directory containing modified working files also displays this decorator.
	The <b>Revision out of sync</b> decorator indicates that the working revision does not match the member revision. In the Packages and Navigator views, this decorator indicates the following: <ul style="list-style-type: none"> <li>■ a new revision that does not exist on the current development path. More specifically, the member revision is the working revision; however, a new revision is available.</li> <li>■ an uncommitted update where a user submitted a change in a transactional change package, or a change package has been submitted, but not reviewed (if change package reviews are enabled). For example, the user's working revision is 1.4 but the member revision is 1.3.</li> </ul>

## Displaying Integrity Information

To display detailed Integrity information about a resource that is under Integrity version control, right-click the resource (project or file) and select **Properties**. From the **Properties** page, select the **Integrity Source** node. If the resource is under Integrity version control, detailed information displays in the Integrity page. If the object is not under Integrity version control, **This resource is not under Integrity Control** displays.

Integrity information displays for the following resources:

- **Projects/Sandboxes**

The **Properties** page for a Sandbox displays the corresponding project type (normal, variant, build), Sandbox path and name, server name and port number the project resides on, corresponding project path and name, and the last checkpoint, description, development path, or build revision information.

For more detailed information about the project or Sandbox, click **Integrity Sandbox Info**. The **Sandbox Information** dialog box displays.

- **Members**

The **Properties** page for a member displays project and Sandbox information, the path and name of the member, and the revision number. *rename?*

If the member is locked, the locker, lock type, revision number the member was locked at, and change package ID associated with the revision also display.

For more detailed information about the member, click **Integrity Member Info**. The **Member Information** dialog box displays.



---

## Upgrading and Downgrading Locks

To downgrade an exclusive lock to a non-exclusive lock, in the **Properties** page for a member click **Downgrade Lock**.

To upgrade a non-exclusive lock to an exclusive lock, click **Upgrade Lock**. The new lock type displays.

## Working With Active Change Packages

If you have enabled the **Use Change Package** preference, all changes must be submitted using a change package.

The active change package is a change package that has been set as the default change package for Integrity member operations in the workspace. This also allows other developers to identify what you are currently working on, because they can see which files you have locked and the associated change package.

For Implementer users, this integration includes the active item feature. The active item is the default item used for an Implementer check out operation.

### Active Change Package Display

When you open your workspace, the last change package used in the workspace is displayed as the active change package in the Integrity trim.



By default, the active change package label displays the change package ID and change package summary. Text that exceeds the size of the label is truncated; however, you can hover your mouse over the label to display the change package ID, server and port that the change package resides on, and change package summary in a tooltip.

If the last change package used in the workspace has been closed, or if the workspace is new, `<no active Change Package>` displays in the Integrity trim.

If you disable the **Use Change Package** preference, all elements in the Integrity trim are disabled until Eclipse is restarted, at which point the change package components are removed. No Integrity commands explicitly set the change package.

The Integrity trim displays in the bottom-right corner of the workbench by default; however, you can drag the trim anywhere in the workbench. The Integrity trim always displays the Integrity logo and the online/offline mode button (for more information, see “Online and Offline Mode” on page 24). If the **Use Change Package** preference is enabled, it also displays the active change package label, a list allowing you to select the active change package, and a button to create new change packages.

### Specifying an Active Change Package

To specify an active change package or select a different one, click the drop-down arrow button next to the change package label in the Integrity trim. The list of available change packages is displayed immediately, without having to wait for the command to complete on the Integrity Client.

To create a new change package, click . The **Create Change Package** dialog box displays.


---

## Managing Items With Integrity Worktray

The Integrity Worktray provides support for Integrity items and Implementer change packages within Eclipse. Although the Integrity integration must be installed for Integrity Worktray to function, you can configure it to use only Implementer features.

The Integrity Worktray consists of views that display item and change package data. As with any other Eclipse element, the location and size of each view can be customized.

---

**IMPORTANT** Integrity Worktray view data is not dynamically refreshed. To display changes made since the last time the view was opened (or changed based on a link to another view), on the view toolbar click .

---


### Configuring Integrity Worktray Preferences

Integration specific preferences allow you to select settings for the Integrity Worktray views. Click **Restore Defaults** at any time to clear the changes you made.

#### To configure Integrity Worktray preferences

- 1 In your workspace, select **Window > Preferences**. The **Preferences** dialog box displays.
- 2 Under **Team**, select **Integrity Worktray**. The **Integrity Worktray Preferences** display.
- 3 Make the necessary changes to the following preferences for **Integrity Applications**:
  - **Enable Integrity Source actions only** enables Integrity configuration management commands in the views. This option is enabled by default.
  - **Enable Integrity Implementer actions only** enables Implementer commands in the views.
  - **Enable Integrity Source and Implementer** actions enables all commands in the views.

---

**IMPORTANT** If the Integrity Worktray view is open when making changes to the Integrity Worktray preferences, you must refresh the view (click ) to show the changes.

---

### Integrity Worktray View

#### To display the Integrity Worktray view

- 1 Select **Window > Show View > Other**. The **Show View** dialog box displays.
- 2 Open the **Integrity Solution** directory, and then select **Integrity Worktray**. The Integrity Worktray view displays.

The Integrity Worktray view displays Integrity items based on the selected query. By default, items are displayed based on your Quick Query criteria. Query criteria must be defined and made visible from Integrity before that query can be used from the Integrity Worktray.

---

**NOTE** To make new queries available in the list, you must refresh the view.

---

The Integrity Worktray view displays the column set associated with the selected Integrity query, rather than only displaying a defined, default column set. Therefore, the column set selection is no longer available under **Integrity Worktray Preferences**.


---

To change the column set displayed in the Integrity Worktray view, you modify the column set referenced in the underlying query. For more information on working with queries and column sets, see the *Integrity User Guide*.

---

**IMPORTANT** Depending on your Integrity Worktray preferences settings, not all commands or toolbars documented in this section may be displayed.

---


The following operations are available from the  menu in the Integrity Worktray view:


Command	Operation
<b>Create Item</b>	Creates an Integrity item. <b>TIP:</b> The <b>Create Item</b> operation is also available from the shortcut menu when you right click in the Integrity Worktray view.
<b>Edit Item</b>	Edits the selected item.
<b>View Item</b>	For the selected item, displays its details in the Integrity Client GUI.
<b>Create Related Item</b>	Creates an item related to the selected item.
<b>Refresh</b>	Refreshes the Integrity Worktray view.

### ***Integrity Implementer Change Package View***

- 1 Select **Window > Show View > Other**. The **Show View** dialog box displays.
- 2 Open the **Integrity Solution** directory, and then select **Integrity Implementer Change Package**. The Integrity Implementer Change Package view displays.

The Integrity Implementer Change Package view displays information for Implementer change packages. The view only displays the information for a single change package at a time, but you can change the displayed change package by using the arrows on the toolbar.

The  button links the Integrity Implementer Change Package view with the Integrity Worktray view so that when an item is selected, the corresponding change package displays. Toggle the button to remove the linking. For detailed information on Implementer change packages, see the *Implementer 10.2 Installation and Administration Guide*.

The following operations are available from the  menu in the Integrity Implementer Change Package view:

Command	Operation
<b>Previous Change Package</b>	If more than one change package is associated with the selected item, displays the previous change package by order of change package ID.
<b>Next Change Package</b>	If more than one change package is associated with the selected items, displays the next change package by order of change package ID.
<b>Open in Integrity</b>	Displays the change package details in the Integrity Client GUI.
<b>Refresh</b>	Refreshes the Integrity Implementer Change Package view.

---

## Integrity Commands

To provide a more seamless Integrity experience within Eclipse, basic Integrity commands, such as adding members and checking out files occur implicitly in the Integrity repository when you perform the equivalent Eclipse commands. File status is displayed immediately to other Eclipse users working in the project. For more information on performing operations, refer to the following sections:

- “Adding Members to an Integrity Project” on page 36
- “Dropping Members From an Integrity Project” on page 36
- “Checking Out Members” on page 37
- “Checking In Members” on page 37

To access advanced Integrity version control functionality, such as submitting changes or checkpointing a project, select an Eclipse project or one or more files, and then select the **Integrity** menu, or right click the selected item and then choose the **Team** menu. Note that not all commands are available in both menus, and that the available commands depend on your selection. For example, the **View Member Differences** command is available only when you select a file under Integrity version control. The Eclipse status bar indicates when an Integrity command is complete. For a list of Integrity commands, see “Advanced Integrity Commands” on page 37.

### Adding Members to an Integrity Project

#### To add members to an Integrity configuration management project in your workspace

- 1 In your workspace, create one or more new files.

---

**NOTE** The integration also adds members if you refresh Eclipse and it discovers one or more new files that are unknown to the Integrity repository. For example, if you copy a file to your Sandbox controlled in Eclipse, and then refresh Eclipse, the file is added to the Integrity project.

---

The files are added as members and decorators display immediately for each member. If change packages are enabled and an active change package exists, the new files are associated with the active change package and the files can be committed to the Integrity repository.

---

**NOTE** If you are using change packages and have not set an active change package when a file is created, use the **Move to Change Package** command to associate the file with a change package.

---

- 2 Commit the changes to the Integrity repository by doing one of the following:
  - If you are using change packages, submit the change package associated with the new files. For more information, see “Team Synchronizing” on page 42.
  - If you are not using change packages, select the new files, and then select **Integrity > Submit Changes**. For more information, see “Advanced Integrity Commands” on page 37.

### Dropping Members From an Integrity Project

#### To drop members from an Integrity configuration management project in your workspace

- 1 In your workspace, select one or more files to drop from the Integrity project.
- 2 Select **Edit > Delete**.

---

3 Commit the changes to the Integrity repository by doing one of the following:

- If you are using change packages, submit the change package associated with the deleted files. For more information, see “Team Synchronizing” on page 42.

If you are using change packages and have not set an active change package when a file is deleted, use the **Move to Change Package** command to associate the file with a change package.

- If you are not using change packages, select the files to delete in the Synchronize view, and then select **Integrity > Submit Changes**. For more information, see “Advanced Integrity Commands” on page 37.

## Checking Out Members

### To check out members in your workspace

In your workspace, open a file and begin editing. If you have the **Lock file** option selected in the preferences, the file is locked and the appropriate lock decorator displays.

---

**NOTE** If the **Lock file** option is not selected in the Integrity options, the working file is made writable when you edit the file and you must manually lock the file later, or associate the modified working file with a change package. For more information, see “Setting Preferences” on page 21.

---

## Checking In Members

### To check in members in your workspace

Commit the changes to the Integrity repository by doing one of the following:

- If you are using change packages, submit the change package associated with the modified files. For more information, see “Team Synchronizing” on page 42.
- If you are not using change packages, select the modified files, and then select **Integrity > Submit Changes**. For more information, see “Advanced Integrity Commands” on page 37.


## Advanced Integrity Commands

The following advanced Integrity commands are available from the **Integrity** menu and/or **Team** shortcut menu:

Integrity Command	Function
Move to Change Package	<p>Moves the selected files to an existing or new change package. The change package containing the associated changes can then be submitted to the Integrity repository.</p> <p>From the list, select a change package or click <b>Create Change Package</b>. The status bar indicates when the command is complete.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"><li>■ You cannot move a rename or move change package entry out of a change package. You can only move these type of entries to another change package.</li><li>■ You cannot move a rename or move operation to a change package if it is not already associated with a change package.</li></ul>

---

Integrity Command	Function
Submit Changes	<p>Equivalent to the Integrity <b>Submit</b> command. Submits uncommitted changes on individual files.</p> <p>The status bar indicates when the command is complete.</p> <p>If the selection is from the Packages view, a submit does not include dropped files; however, a submit includes dropped files displayed in the Synchronize view.</p> <p><b>IMPORTANT:</b> The <b>Submit Changes</b> command is not available if you have selected the option for <b>Use active Change Package</b>. For more information, see “Use Change Package” on page 22.</p>
Resynchronize	<p>Equivalent to the Integrity <b>Resynchronize</b> command.</p> <p>Gets the latest version of the selected file and puts it in your working directory.</p> <p>The <b>Overwrite</b> dialog box may display.</p> <p>The status bar indicates when the command is complete.</p>
Resynchronize by Change Package	<p>Equivalent to the Integrity <b>Resynchronize Member by Change Package</b> command.</p> <p>Processes the change packages associated with the member you are resynchronizing, and brings the changes from the project to your Sandbox.</p> <p>Depending on the preferences you have set for the <b>Resynchronize</b> command, the <b>Confirm Overwrite Working File</b> dialog box displays.</p>
Revert	<p>Equivalent to the Integrity <b>Revert</b> command.</p> <p>Replaces the working file with the revision that was checked out, as it appeared prior to modification, and unlocks the file (and removes it from the associated change package).</p> <p><b>NOTE:</b> You can revert a file in a change package that is not the active change package. Reverting the file removes it from the change package.</p> <p>The <b>Overwrite</b> dialog box may display.</p> <p>The status bar indicates when the command is complete.</p>
Lock Member	<p>Equivalent to the Integrity <b>Lock</b> command.</p> <p>Right-click the selected file in the Package Explorer or Project Explorer view, and select <b>Lock</b>.</p>
View Member Differences	<p>Equivalent to the Integrity <b>Differences</b> command.</p> <p>Compares the selected working file with the member revision.</p> <p>Visual Difference automatically launches and displays the two files.</p>
View Annotated Revision	<p>Equivalent to the Integrity <b>View Annotated</b> command.</p> <p>Displays the annotated revision history of the selected file.</p> <p>The Annotated Revision view displays.</p>
View Member History	<p>Equivalent to the Integrity <b>View Member History</b> command.</p> <p>Displays the revision history of the selected file.</p> <p>The Member History view displays.</p>
Create Change Package	<p>Equivalent to the Integrity <b>Create Change Package</b> command.</p> <p>Creates a change package.</p> <p>The <b>Create Change Package</b> dialog box displays.</p>
View Active Change Package	<p>Equivalent to the Integrity <b>View Change Package</b> command.</p> <p>Displays the active change package associated with the selected file. If there is no active change package, this command is disabled.</p> <p>The Change Package view displays.</p> <p>For more information on active change packages, see “Working With Active Change Packages” on page 33.</p>

Integrity Command	Function
Resynchronize Change Packages	<p>Equivalent to the Integrity <b>Resynchronize Change Package</b> command.</p> <p>Previews the changes listed in change packages in the context of a Sandbox before propagating them to the project.</p> <p>The <b>Resynchronize Change Packages</b> dialog box displays.</p> <p>After you resynchronize the change package containing the changes, the changes appear as operations not yet committed to the Integrity repository.</p> <p>Once you are satisfied with the changes and want to commit them to the Integrity repository, submit the change package, and then resynchronize the members.</p> <p><b>IMPORTANT:</b> After resynchronizing by change package, the changes may appear as incoming changes in the Synchronize view. Do not resynchronize the changes; otherwise, they will be lost. Update the revisions by submitting the changes.</p> <p>For more information on submitting changes, see “Team Synchronizing” on page 42.</p>
Submit Active Change Package	<p>The <b>Submit Active Change Package</b> command allows you to submit an active change package directly from the <b>Integrity</b> menu within Eclipse.</p> <p>The <b>Submit Active Change Package</b> command operates in the same way as the <b>Submit Change Package</b> command on the Integrity Client, and follows most client preferences as set for that command.</p>
Create Subproject	<p>Equivalent to the Integrity <b>Create Subproject</b> command.</p> <p>Creates a subproject in the selected directory and adds it to the Integrity project.</p> <p>After you type a name for the subproject, the <b>Create Subproject</b> dialog box displays.</p> <p><b>NOTE:</b> If a conflict or error occurs when attempting to name the subproject, the name entry dialog box displays, prompting you to type a new name for the subproject.</p>
Convert to Subproject	<p>Converts an empty directory or a directory containing files that are not under Integrity version control to a subproject. If the directory contains files, they must be added to Integrity version control after the subproject is created.</p> <p>This command is useful for defining a directory structure for a build project.</p>
Drop Subproject	<p>Equivalent to the Integrity <b>Drop Subproject</b> command.</p> <p>Drops the selected subproject(s) from the Integrity project, deleting all files and directories under the subproject(s).</p> <p>The <b>Drop Subproject</b> dialog box displays.</p>
View Sandbox	<p>Equivalent to the Integrity <b>View Sandbox</b> command.</p> <p>Displays a Sandbox view.</p>
Merge child Development Path	<p>Equivalent to the Integrity <b>Merge Child Development Path</b> command.</p> <p>Merges a development path into its parent development path. The parent development path may be a mainline, or it may be a development path itself. The merge destination must be the parent of the child being merged into it. The command creates a propagation change package containing the changes necessary to perform the development path merge. For more information, see the <i>Integrity User Guide</i>.</p>
View Project Differences	<p>Equivalent to the Integrity <b>Project Differences</b> command.</p> <p>Displays the differences between project checkpoints.</p> <p>The Project Differences view displays.</p>
Checkpoint	<p>Equivalent to the Integrity <b>Checkpoint</b> command.</p> <p>Checkpoints one or more Eclipse projects under Integrity version control. If the project is a subproject, the checkpoint recursively checkpoints everything under it; parent projects in the hierarchy are not checkpointed.</p> <p>A dialog box displays, prompting you to choose one or more Eclipse projects, and to, optionally, add a label and description.</p> <p><b>NOTE:</b> Labels are applied to the Eclipse project, but not project members.</p> <p>A confirmation dialog box indicates which projects were successfully or unsuccessfully checkpointed. A maximum of 15 projects appear in the confirmation dialog box.</p>
Work Offline/Online	<p>Toggles offline/online mode.</p> <p>For more information, see “Online and Offline Mode” on page 24.</p>
Open Integrity Client	<p>Equivalent to launching the Integrity Client GUI.</p> <p><b>TIP:</b> You can also launch the Integrity Client by clicking  in the Integrity trim.</p>

---

Note the following:

- You can make the **Integrity** menu always available from the perspective you have open by selecting **Window > Customize Perspective**. In the **Commands** panel, enable **Integrity Source Menu**.
- You can use Eclipse's key binding functionality to assign key sequences to commands in the **Integrity** and **Team** menus. Note that Integrity commands are only visible when **Include unbound commands** is enabled on the **Preference > Keys** page. For more information, refer to the Eclipse documentation.
- Although Eclipse supports linked resources, known as out-of-tree members in Integrity, they cannot be placed under Integrity version control. As a result, Integrity commands are disabled and decorators do not appear when you select linked resources.
- If you select one or more files and then perform a revert or resynchronize, only the selected files are reverted or resynchronized. If you select one or more containers (Eclipse projects, directories, packages) and then perform a revert or resynchronize, the integration examines the first Sandbox or subsandbox of each container and performs a revert or resynchronize on the entire Sandbox or subsandbox. If you select an Eclipse container that maps to a directory in Integrity, the Sandbox containing the directory is resynchronized or reverted.

## Refactoring

The Eclipse integration allows you to refactor your source code and preview the changes before you commit them to the Integrity repository. Refactoring activities (adds, drops, moves, and renames) are handled as member operations not yet committed to the Integrity repository, and are recorded in change packages that you can either submit or discard.

Submitting a change package commits your changes to the Integrity repository. This is done by clicking one of the Submit buttons in the Synchronize view (for more information, see "Team Synchronizing" on page 42), or the **Submit Changes/Submit Change Package** commands from the **Integrity** menu (for more information, see "Advanced Integrity Commands" on page 37).

To discard your changes, discard the change package or change package entry using the **Integrity > Revert** command, or undo the change through Eclipse. For example, if you want to undo a move operation, move the file back to its original location.

Note the following:

- Where possible, use Eclipse's **Undo** command to undo changes. If it is not possible to use the **Undo** command, use the Integrity **Revert** command.
- If you create an Integrity configuration management project in the Integrity Client and then import it into Eclipse, renaming a package in the project modifies the subproject organization in Integrity. More specifically, the directory is created on the file system (in the Sandbox directory), and the files are moved via move member commands to make the changes happen in Integrity.
- Renaming an Eclipse project under Integrity version control is not supported.
- When you rename a package that corresponds to a subproject in Eclipse, the operation is recorded as move operations for the members contained in that package. When these changes are submitted, the old subproject folder is not removed until the corresponding subproject is dropped. To drop a subproject, the **Drop Subproject** operation must be submitted in a separate change package.



If you rename a folder in the context of the Eclipse workspace, the old folder is removed as a result of submitting the changes to the files in that folder.



- 
- If you delete a project from your workspace, the associated Sandbox is automatically dropped if the preference for **Drop Integrity Sandbox when Eclipse project is deleted** is enabled. For more information, see “Setting Preferences” on page 21.
  - When you use the Eclipse **Edit > Delete** command (or press the Delete key) on a directory, the integration drops files in the directory. To remove the entire subproject, use the **Integrity > Drop Subproject** command.

## Comparing Revisions

The integration with Integrity supports functionality for comparing and merging revisions. When conflicts are detected during a synchronization, you are automatically prompted to resolve those conflicts by differencing the files. The integration with Integrity supports both two and three-way differencing. For three-way differencing, you are prompted to perform a merge on the branch and then a second merge against the selected project root.

After synchronizing a project, you can navigate all revisions that show differences using **Go to Next Difference**  and **Go to Previous Difference**  in the Synchronize view. All differences in the file are visited before opening the next file in the view.

Differencing is supported for both plain text, binary files, and models, provided the required tools are available on your system. The differencing tool presented is based on the tools you use in your IDE environment and on the file type, for example, **UML**, **JAVA**, or **TEXT**.

### To compare differences in a revision

In the Synchronize view, right click the target file and select **Open in Compare Editor**. The local and remote revisions of the target file display in the **Compare** window for the available differencing tool. Perform the required merge operations to resolve the differences. Save your changes.

---

**TIP** To compare differences between the working file and member revision, select the file in the Navigator or Packages view, then select **Integrity > Member Differences**. If differences exist, the **Integrity Visual Difference** window displays.

---

## History View

The History view displays the Integrity revision history of a file.

---

**TIP** You can also view a file’s revision history by selecting the file, then selecting **Integrity > View Member History**. The Member History view displays.

---

To display the History view, select **Window > Show View > Other**. The **Show View** dialog box displays. Open the **Team** directory, and then select **History**.

The History view displays the file name, revision number, modification date, author, change package, the name of any user with a locked revision and corresponding change package, and state information. A \* beside a revision number indicates the member revision, and bold indicates the working revision.

To refresh the History view, click .

The  button links the History view with the Editor view such that when a file is opened in the Editor view, the corresponding Integrity revision history displays. Toggle the button to remove the linking and display the local history.

To pin the History view, click .

---

## Team Synchronizing

Eclipse synchronization support provides an interface for managing dynamic collections of information. Synchronizing is the operation that displays the changes between resources in different locations and optionally allows you to modify the synchronization state by performing an action.

The Team Synchronizing perspective allows you to see all project resources that require synchronization in your workspace. Essentially, the Team Synchronizing perspective summarizes all changes present in the selected resources. The integration supports the Eclipse Team Synchronizing perspective through the Integrity synchronizer.

### Synchronizing an Integrity Project

Synchronizing provides dynamic updates of changes to shared resources. The Integrity synchronizer provides a view that allows you to see all project resources that show comparative changes from your local workspace to the remote Integrity configuration management project on the Integrity Server.





The Integrity synchronizer provides filters to control whether you view incoming changes, outgoing changes, conflicts, or all changes. Incoming changes come from the Integrity project. If accepted, they update your local revision to the latest version currently committed in the project. Outgoing changes come from your local workspace. If committed, they change the project member to match the revision present in your Sandbox.









Integrity provides several options for filtering the Synchronize view. You can filter projects by specific resources or change packages containing uncommitted member operations. Once the Synchronize view is populated, you can further filter changes in the view according to the type of change, whether incoming, outgoing, or a conflict.

Whichever mode (or filter) you select, the Integrity synchronizer shows you any conflicts for your locally modified revision when a more recent version is available. In this situation you can choose to do one of three things: update the revision from the Integrity project, commit your version of the resource to the Integrity project, or merge your work with the changes into the Integrity project. Generally you merge your changes to avoid losing modifications. For more information on resolving conflicts, see “Comparing Revisions” on page 41.

The **Collapse All** button allows you to instantly collapse the project directory structure. Expanding a project tree expands according to the selected filter, if any. For other native Eclipse options, see the Eclipse product documentation.









The following commands are available from the menu bar in the Synchronize view:

Button Icons	Function
	<b>Synchronize Integrity</b> allows you to work with the Synchronize Wizard and display the state of your workspace as it relates to the Integrity project.
	<b>Pin Current Synchronization</b> prevents dynamic updates of the view, allowing you to hold your workspace synchronization. <b>NOTE:</b> You cannot pin a Synchronize view that is filtered on the active change package; however, you can on an open change package or a set of resources.
	<b>Go to Next Difference</b> navigates through the project hierarchy to the next revision containing differences. Automatically launches the available differencing tool.
	<b>Go to Previous Difference</b> navigates through the project hierarchy to the previous revision containing differences. Automatically launches the available differencing tool.

Button Icons	Function
	<b>Collapse All</b> collapses all expanded entries in the view.
	<b>Incoming Mode</b> filters the view to show only those working files with associated incoming changes or incoming add operations.
	<b>Outgoing Mode</b> filters the view to show only those working files with associated outgoing changes or outgoing add operations.
	<b>Incoming/Outgoing Mode</b> filters the view to show all working files with incoming and outgoing changes.
	<b>Conflicts Mode</b> filters the view to show only those working files with incoming and outgoing changes containing conflicts.
	<b>Submit this Change Package</b> submits the selected or active change package containing your changes to the Integrity repository. This button is only visible when you filter the Synchronize view by change packages.
	<b>Submit All Outgoing Changes</b> submits the outgoing changes to the Integrity repository. This button is only visible when you are not using active change package tracking and you filter the Synchronize view by resources.
	<b>Resync all incoming changes</b> resynchronizes your workspace with all incoming changes. This button is only visible when you filter the Synchronize view by resources.

## Team Synchronizing Icons

The following team synchronizing decorators are appended to file icons:

Decorator	Function
	The <b>Incoming changes</b> decorator indicates that there are incoming changes available for the member.
	The <b>Incoming add</b> decorator indicates the member was added to the Eclipse project under Integrity version control, but does not exist in your workspace.
	The <b>Incoming drop</b> decorator indicates that the member was dropped from the Eclipse project under Integrity version control; however, the file still exists in your workspace.
	The <b>Outgoing changes</b> decorator indicates that there are outgoing changes available for the member.
	The <b>Outgoing add</b> decorator indicates that the member was added to your workspace, but is not currently a member of the Eclipse project under Integrity version control.
	The <b>Outgoing drop</b> decorator indicates that the member was deleted from your workspace, but has not been dropped from the Eclipse project under Integrity version control.
	The <b>Conflict changes</b> decorator indicates that there are conflicting changes between incoming and outgoing changes for a member. This can include changes to an earlier revision of the member.
	The <b>Conflicting drop</b> decorator indicates that you modified a member that was dropped from the Eclipse project.

---


## To perform a Team Synchronization

- 1 From your Eclipse workspace, open the **Team Synchronizing** perspective by selecting **Window > Open Perspective > Other** and choosing **Team Synchronizing** from the list.

---

**TIP** You can also open the Synchronize view by selecting **Window > Show View > Other > Team > Synchronize**. When you finish running the wizard and Eclipse prompts you to switch perspectives, click **No**.

---

- 2 Click **OK**. The Synchronize view displays.
- 3 To run a synchronization, click . The **Synchronize** wizard displays.

---

**TIP** The **F5** function always refreshes the view and operations within Eclipse trigger an update to the Team Synchronizing perspective.

---

- 4 From the list, choose **Integrity**, then click **Next**.
- 5 From the list, choose an option for populating the Synchronize view:
  - **Filter by the active Change Package** populates the Synchronize view with project resources that have uncommitted changes in the active change package.
  - **Filter by one of my open Change Packages** populates the Synchronize view with project resources that have uncommitted changes in an open change package.

Click **Next**, then select an open change package from the list.

- **Filter by a selection of resources** populates the Synchronize view with specific project resources that may have changed. You can optionally choose to **Show only changes not associated with a change package**.

Click **Next**, then select specific resources to synchronize, or select a scope to automatically select a group of resources:

- **Workspace** synchronizes all available resources in the workspace.
- **Selected Resources** synchronizes the resources you select in the **Available resources to Synchronize** list.
- **Working Set** synchronizes the resources in the working sets you select from the **Select Working Sets** dialog box. To choose a working set, click **Choose**.

---

### NOTE

The Synchronize view filters on the entire workspace if you select the following options:

- **Selected Resources** with no projects selected in the **Available resources to Synchronize** list.
  - **Working Set** with one of the following options selected in the **Select Working Sets** dialog box: **Window Working Set**, **No Working Set**, or **Selected Working Set** with no projects selected in the list.
- 

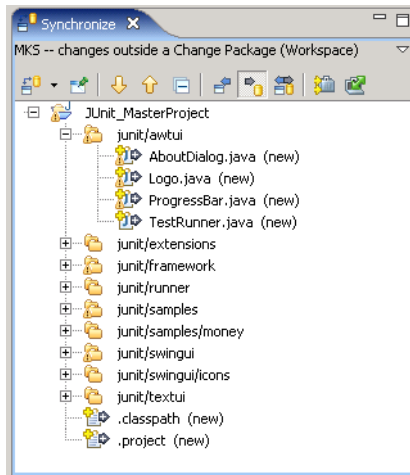
- 6 To populate the Synchronize view, click **Finish**. If there are changes, they display in the Synchronize view. If there are no changes, the Synchronize view is empty.

---

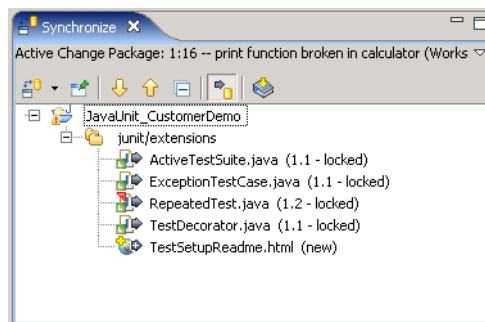
**IMPORTANT** Ignored or derived resources do not display in the Synchronize view.




---


If you filtered the Synchronize view by specific resources, the Synchronize view appears similar to the following:



If you filtered the Synchronize view by change package, the Synchronize view appears (similar to the following):



- 7 To review the changes in a member in the Integrity view, right click the member and choose **View Member Differences**. Visual Differences launches, displaying the differences between the working file and the member revision. To launch the Eclipse Compare editor, use the **Open in Compare Editor** command or double click the Synchronize view entry.
- 8 If you filtered the Synchronize view by specific resources, do one of the following:
  - To submit all outgoing changes when not using a change package, click  .  
To submit all outgoing changes when using a change package, click  .
  - To resync all incoming changes, click  .
  - To resynchronize a working file with its corresponding change package, select the incoming member(s), and then right click and select **Resynchronize by Change Package**.

If you filtered the Synchronize view by change package, submit the change package by clicking  .

---

## Best Practices

This section describes best practices for using the Eclipse integration. It also points out efficiencies that can be gained by using the integration in a certain way, as well as identifying any risks, constraints, or other limits within the integration or within a particular implementation of the integration.

- If you are sharing an Eclipse project containing a large number of subdirectories and files, do one of the following:
  - Place the project under version control in the Integrity Client, and then share the project in Eclipse.
  - Disable the **Add all files when creating the new Integrity Source project** option when placing an Eclipse project under Integrity version control. After the Integrity project and Sandbox are created, add the project files in small batches using a separate change package for each batch.
- For ease of distribution, control, and versioning, create a specific Integrity project and Sandbox for `.psf` files. Users can then access the required `.psf` file from a central location.
- Mark `.class` files as team ignored. For more information, see “To set Ignored Resources preferences” on page 23.
- To optimize the resynchronization of project resources and submission of changes, the integration is designed to use change packages and the Synchronize view. When you begin using the integration, open the Synchronize view and filter it by the resources you will be working on.
- A change package is a container for distinct tasks, not a generic container for all tasks performed during a project. A change package containing too many entries may impact the integration’s performance. Before you perform a task that requires a change package, create a change package and enable it as the active change package.
- Resynchronize project resources frequently and incrementally. Resynchronizing a large number of resources in a single operation may take a long time.
- Allocate memory to Eclipse and the Integrity Client according to the size of your Eclipse projects. Use the following as a guideline to help you determine memory allocations:
  - **50 projects with 100,000 members**
    - Set the Integrity Client maximum heap size to 512 MB
    - Set the Integrity Client cache size to 100 MB
    - Allocate at least 512 MB of memory to Eclipse
  - **100 projects with 200,000 members**
    - Set the Integrity Client maximum heap size to at least 1 GB
    - Set the Integrity Client cache size to 200 MB
    - Allocate at least 1 GB of memory to Eclipse

- 
- **200 projects with 400,000 members**
    - Set the Integrity Client maximum heap size to the highest possible number for your platform (most commonly this is 1.5 GB)
    - Set the Integrity Client cache size to 400 MB
    - Allocate at least 2 GB of memory to Eclipse. Use the 64 bit version of Eclipse if you need more than 2 GB.

---

**NOTE**

To set the Integrity Client cache size, edit the `si.cache.default.size` setting in:

```
<installdir>/IntegrityClientSite.rc
```

where `<installdir>` is the path to the directory where you installed the Integrity Client.

---

- To improve performance with large projects, use working sets to display only necessary resources in navigation views. For more information, see the Eclipse documentation.
- The Eclipse integration is tightly coupled with the Integrity Client. When using Eclipse, focus issues may occur if you open the Integrity Client GUI. If this occurs, close the Integrity Client interface but do not shut down the Integrity Client process. If Eclipse shuts down, shut down the Integrity Client, and then restart the client.
- Working in offline mode for long periods of time is not recommended. When you switch to online mode after an extended period of time, performance issues may arise from rebuilding the cache.
- To ensure the expected prompts when obtaining locks and change packages, make sure that change package options are consistently configured between the Integrity Server and Eclipse. If change packages are mandatory on the Integrity Server, set the integration option for **Use active Change Package** and enable prompting for the active change package. If change package policies are not consistent, changes you make to the files in your workspace may not be recorded as expected in Integrity. For more information on the available preferences, see “Setting Preferences” on page 21.

## Limitations

- After switching from offline to online mode, you cannot submit changes in the Synchronize view while the Integrity status cache refreshes. Before you submit changes, check the status of the Eclipse progress bar and ensure there is no Integrity command activity.
- When a subproject is dropped in another user’s Sandbox, resynchronizing the dropped subproject’s members in the Synchronize view displays an error message. To avoid this, resynchronize the dropped subproject in the Package Explorer.
- When filtered by the active change package, the Synchronize view updates if you change the active change package from `<none>` to a change package, or from one change package to another change package; however, the view does not update when you change the active change package from a change package to `<none>`. To display changes not associated with a change package in the Synchronize view, run the Synchronize wizard, filtering by resources.
- Existing locks on dropped members are not removed. To resolve this, open the Locks view in the Integrity Client GUI and unlock the members.
- Performing the **View Sandbox** command on a dropped subproject incorrectly displays an error message.

- 
- The initial attempt to rename a subproject after adding a member to the subproject displays the error message “the Resource is out of sync with the filesystem.” Refreshing the subproject, then performing the rename command successfully completes the operation.
  - Attempts to revert deferred added members from the Synchronize view do not successfully remove them from the Synchronize or Packages views unless you also delete the file. Reverting the files from the Integrity Client successfully removes them from both views.
  - To avoid potential focus problems when using the Integrity Sharing Wizard, close the Integrity Client if it is open.
  - To hide files, such as the Integrity project registry file (`project.pj`), certain resource filters are enabled by default in Eclipse. By default, these filters apply to the Packages and Navigator views.
  - There are a number of conditions that may cause Eclipse to identify new files in a project, resulting in deferred add operations for non-member files that have not been modified by the user within the Eclipse IDE. For example, in cases where automatic refresh is enabled in Eclipse and files are generated by build scripts, the generated files are identified by Integrity as new files to be added to the project.

If you routinely encounter this situation and do not want Integrity to add such generated files to your projects, you can add the file type to the Eclipse **Ignored Resources** list. Under **Window > Preferences > Team > Ignored Resources**, add the pattern for the file type you want the integration to ignore.



This chapter provides information on using the Integrity integration with IBM Rational Rose Enterprise Edition 2003, version 6.13. The chapter includes information on the following topics:

- “Configuring the Rational Rose Integration” on page 50
- “Using the Rational Rose Integration” on page 50
- “Creating an Integrity Project” on page 51
- “Creating a Sandbox” on page 51
- “Adding Members to an Integrity Project” on page 51
- “Checking Out Members” on page 52
- “Checking In Members” on page 52

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Configuring the Rational Rose Integration

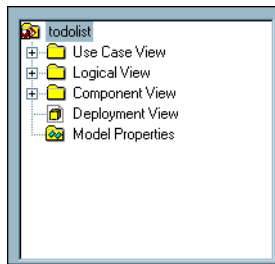
The Integrity Client includes the **File > Integrations** menu action for enabling and disabling the integration with IBM Rational Rose. You can select from a list of available integrations and enable or disable them, as required to work with your preferred application. For more information, see “Configuring Integrations” on page 3.

## Using the Rational Rose Integration

You access version control functionality from within Rational Rose by choosing **Tools > Version Control**, and then selecting one of the following commands:

Command	Function
<b>Add to Version Control</b>	Equivalent to the Integrity <b>New Project</b> and <b>Add Members</b> commands. Brings the current Rational Rose project or selected file(s) under version control. The <b>Create Archive</b> dialog box displays.
<b>Remove from Version Control</b>	Equivalent to the Integrity <b>Drop Members</b> command. Drops the selected file(s) from the Integrity project. The <b>Drop Member</b> dialog box displays.
<b>Start Version Control Explorer</b>	Equivalent to launching the graphical user interface or the Integrity <b>Open Sandbox</b> command. Displays a <b>Sandbox</b> view.
<b>Check In</b>	Equivalent to the Integrity <b>Check In</b> command. Checks in the selected file(s). The <b>Check In</b> dialog box displays.
<b>Check Out</b>	Equivalent to the Integrity <b>Check Out</b> command. Checks out the selected file. The <b>Check Out</b> dialog box displays.
<b>Undo Check Out</b>	Equivalent to the Integrity <b>Revert</b> command. Replaces the selected working file with the revision that was checked out, as it appeared prior to modification, and unlocks the file. The <b>Overwrite</b> dialog box may display.
<b>Get Latest</b>	Equivalent to the Integrity <b>Resynchronize</b> command. Gets the latest version of the selected file and puts it in your working directory. The <b>Overwrite</b> dialog box may display.
<b>File Properties</b>	Equivalent to the Integrity <b>Member Information</b> command. Displays member information for the selected file. The <b>Member Information</b> dialog box displays.
<b>File History</b>	Equivalent to the Integrity <b>View Member History</b> command. Displays the member history for the selected file. The <b>Member History</b> view displays.
<b>Version Control Options</b>	Displays Rational Rose version control options.

The Rational Rose **Browser** panel is enhanced with version control features and indicators. For example, a red check mark displays beside the project to indicate that it is under version control.



## Creating an Integrity Project

### To create an Integrity project in Rational Rose

- 1 In Rational Rose, select the project you want to put under version control.
- 2 Select **Tools > Version Control > Add to Version Control**. The **Add to or Associate with MKS Integration** dialog box displays, showing the project file that will be placed under version control.
- 3 Select one or more files to add to the Integrity project, and click **OK**.
- 4 At the following prompt click **Yes** to browse for a project. The **Add to or Associate with MKS Integration** dialog box displays again.
- 5 Click **OK**.
- 6 At the prompt save your project files. The **Specify the Project to Create** dialog box displays.
- 7 Create a project as described in the *Integrity User Guide*. The **Create Sandbox Wizard** displays.
- 8 Create a Sandbox as described in the *Integrity User Guide*. The **Create Archive** dialog box displays.
- 9 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*. The **Check Out** dialog box displays.
- 10 Check out each member as described in the *Integrity User Guide*.

## Creating a Sandbox

In Integrity, create a Sandbox as described in the *Integrity User Guide*.

## Adding Members to an Integrity Project

### To add members to an Integrity project in Rational Rose

- 1 In Rational Rose, select one or more files to add to the Integrity project.
- 2 Select **Tools > Version Control > Add to Version Control**. The **Add to or Associate with MKS Integration** dialog box displays displaying the project file that will be placed under version control.
- 3 Select one or more files to add to the Integrity project, and click **OK**.
- 4 At the prompt click **Yes** to browse for a project. The **Add to or Associate with MKS Integration** dialog box displays again.

- 
- 5 Click **OK**.
  - 6 At the prompt save your project files. The **Create Archive** dialog box displays.
  - 7 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.
  - 8 Click **OK**.

## Checking Out Members

### To check out members in Rational Rose

- 1 In Rational Rose, select one or more files to check out.
- 2 Select **Tools > Version Control > Check Out**. A list displays, showing all files that will be checked out.
- 3 Click **OK**. The **Check Out** dialog box displays.
- 4 Check out each member as described in the *Integrity User Guide*.

## Checking In Members

### To check in members in Rational Rose

- 1 In Rational Rose, select one or more files to check in.
- 2 Select **Tools > Version Control > Check In**. A list displays, showing all files that will be checked in.
- 3 Click **OK**. The **Check In** dialog box displays.
- 4 Check in each member as described in the *Integrity User Guide*.

---

**NOTE** When using the Rational Rose integration, you are not prompted to confirm the check-in of an unmodified file.

---



**PART III**  
**Microsoft**

The Integrity integration with Microsoft® Visual Studio® 2005/2008/2010/2012 (SDK) allows users to access Integrity commands through Visual Studio, providing a seamless development and configuration management experience.

This chapter discusses the following topics:

- “Before You Start” on page 55
- “Setting Up and Configuring the Integration” on page 56
- “Online and Offline Mode” on page 60
- “Working With Active Change Packages” on page 61
- “Integrity Glyphs in Visual Studio” on page 62
- “Managing Work In Progress” on page 63
- “Managing Assigned Work” on page 66
- “Placing Visual Studio Solutions Under Integrity Source Control” on page 67
- “Working With Visual Studio Files” on page 77
- “Best Practices” on page 81
- “Limitations” on page 83
- “Troubleshooting” on page 84

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Before You Start

Before you set up or use the integration, note the following:

- This guide assumes you know how to use Microsoft Visual Studio and Integrity. For more information about using a product, refer to the appropriate documentation.
- For suggested best practices on using the Visual Studio integration, you should read “Best Practices” on page 81.
- The integration works with the following:
  - Windows 7, Windows Vista, Windows XP, Windows Server 2008/2003
  - Microsoft Visual Studio 2005, 2008, 2010, and 2012
  - Integrity 10
- The Visual Studio integration is designed to use change packages for submitting source code changes to the Integrity repository. Ensure that change packages are enabled on the Integrity Server.

To use change packages, the administrator must also enable Integrity for startup on the Integrity Server (`mksis.startup.im=true`). If change packages are mandatory, Integrity for workflows and documents functionality must be enabled (that is, the policy for `IntegrityManagerEnabled` must be set to true).

For more information on setting properties and policies on the Integrity Server, see the *Integrity Server Installation and Configuration Guide*.

- The following Integrity Server configurations are supported *per* Visual Studio solution:
  - one Integrity Server configured for workflows and documents, and for configuration management
  - one Integrity Server configured for Integrity configuration management
  - one Integrity Server configured for configuration management and one Integrity Server configured for workflows and documents
- Ensure that the Integrity Client path is set correctly for the `PATH` environment variable (that is, the Integrity Client’s `<installdir>/bin` path appears first). Failure to set the path correctly displays a `Package Load Failure` error in Visual Studio. In addition, do not disable packages in Visual Studio. Disabling packages prevents you from selecting Integrity as the source control provider.
- If you are using an older version of the Visual Studio integration, disable the integration in Visual Studio and the Integrity Client before you upgrade to the new Visual Studio integration. To enable the new integration, see “Setting Up and Configuring the Integration” on page 56.

If you have a Visual Studio solution that was placed under Integrity source control using the previous SCC-based Visual Studio integration, you can migrate it for use with this version of the integration. For more information, see “Migrating a Visual Studio Solution from the MKS SCC VS Integration” on page 73.

- If you are upgrading from MKS Integrity 2007 SP 4 or earlier, this release of the Visual Studio integration does not use the MKS Worktray and MKS Change Package view found in previous releases of the Visual Studio integration. To improve management of work in progress and assigned work, this release uses Integrity Work In Progress and Integrity Items views. To use the Integrity Items view, one of your Integrity Servers must be configured for

---

workflows and documents. For more information, see “Managing Work In Progress” on page 63 and “Managing Assigned Work” on page 66.

- When using the Integrity integration with Microsoft Visual Studio, ensure that the necessary Access Control List (ACL) permissions are granted to users. For example, users adding Visual Studio solutions to Integrity source control require the following permissions:
  - `CreateProject`
  - `ModifyProjectAttribute`
  - `CreateSubproject`

For more information on configuring ACLs for Integrity, see the *Integrity Server Installation and Configuration Guide*.

- If you are working in a single Integrity Server environment, you should disable prompts for server information and credentials in the Integrity Client. This provides a more seamless experience with the Visual Studio integration.

If you are working in a multi-server environment, you should disable prompting for credentials, but enabling server prompting so that you can connect to the appropriate Integrity Server. If you do not disable prompting, Visual Studio displays prompts for server information and credentials when you perform Integrity commands. Failure to disable prompting for credentials may display errors when attempting to share a Visual Studio solution. To disable prompting in the Integrity Client, see the *Integrity Getting Started Guide*.

- To avoid potential focus problems when using Visual Studio, close the Integrity Client window, but do not shut down the client.

## Setting Up and Configuring the Integration

This section describes how to set up and configure the Microsoft Visual Studio integration for use with Integrity.

### Installing the Visual Studio Integration

Supported versions of Visual Studio that are currently installed are registered for use with Integrity when you run the Visual Studio integration installer.

---

#### NOTE

If you install another supported version of Visual Studio after you install the integration, you can register the version of Visual Studio by doing the following:

From the **Control Panel > Add or Remove Programs** list, select **Integrity Integration for Microsoft Visual Studio** and click the link for **Click here for support information**.

Click **Repair**.

The Visual Studio integration registers the version of Visual Studio you installed.

---

Before you install the integration, ensure that:

- A supported version of Integrity is installed (“Before You Start” on page 55).
- Visual Studio is shut down. If Visual Studio is running during the installation, it must be restarted after the installation completes to take effect.



- 
- If you are currently using the Visual Studio integration available in Integrity Client 2007 SP5 or 2009, disable the integration in the Integrity Client.

To remove the Visual Studio integration on Windows Vista Enterprise, you must run the Integrity Client as an administrator, and then disable the integration.

### To install the Visual Studio integration

- 1 Run `Integrity_vs_Integration.msi`. The **Integrity Integration for Microsoft Visual Studio Setup** wizard displays.
- 2 Click **Next**.
- 3 Choose a directory to install the integration in. By default, the integration is installed in `C:\Program Files\Integrity\Integrations\Visual Studio`.
- 4 Click **Next**.
- 5 To install the integration click **Install**.
- 6 To exit the Setup Wizard, click **Finish**.

## Enabling the Integrity Plug-In in Microsoft Visual Studio

Visual Studio supports various source control providers. You must specify Integrity as the source control provider.

- 1 In Visual Studio, select **Tools > Options**. The **Options** dialog box displays.
- 2 Select **Source Control > Plug-in Selection**.
- 3 From the **Current source control plug-in** list, select **Integrity**.
- 4 Click **OK**. The Integrity toolbar, Integrity Items, and Integrity Work In Progress views display. If desired, dock the views to a location of your choice in Visual Studio.

For more information on the Integrity toolbar, see “Working With Active Change Packages” on page 61.

For more information on using the Integrity Items view, see “Managing Work In Progress” on page 63.

For more information on using the Integrity Work In Progress view, see “Managing Assigned Work” on page 66.

## Toggling the Integrity Toolbar

The Integrity toolbar displays connection status, a change package list, and toolbar buttons for managing changes to Visual Studio solutions and managing change packages. By default, the Integrity toolbar appears docked when you select **Integrity** as the source control provider.

---

## To toggle the Integrity toolbar

- 1 In Visual Studio, select **Tools > Customize**. The **Customize** dialog box displays.
- 2 From the **Toolbars** tab, toggle **Source Control - Integrity**. The Integrity toolbar disappears or displays in Visual Studio.
- 3 Click **Close**.

Once an open solution is under Integrity source control (see “Sharing a Visual Studio Solution” on page 67), the toolbar buttons in the Integrity toolbar become active.



For more information on connection status, see “Online and Offline Mode” on page 60.

For more information on managing change packages, see “Working With Active Change Packages” on page 61.

For more information on managing changes to Visual Studio solutions, see “Resynchronizing a Visual Studio Solution” on page 76 and “Reverting a Visual Studio Solution” on page 76.

## Setting Preferences

Once you enable the Integrity plug-in, you can configure source control preferences.

### Setting Default Location for Imported/Branched Solutions

You can specify the default location for imported and branched Visual Studio solutions. This path displays when you import or branch a Visual Studio solution.

- 1 Select **Tools > Options**. The **Options** dialog box displays.
- 2 Select **Source Control > Settings**.
- 3 In the **Default Location for “Import Solution” and “Branch Solution”** field, type a path, or click **Browse** to select one. For example, the default location on Windows XP is `c:\Documents and Settings\\My Documents`.
- 4 To save your preferences, click **OK**.

## Working With Keywords

A *keyword* is a placeholder that can be inserted into text-based working files. This placeholder is a special variable (for example, `$Date$`, `$Author$`, `$State$`) used to represent textual information in a working file. Keywords can be expanded (that is, replaced with their literal values) when a revision is checked out.

To enable keywords, set keyword preferences for the appropriate commands, such as check out and check in, in the Integrity Client preferences. For more information, see the *Integrity User Guide*.

To use a keyword, simply include it in a working file, surrounded by dollar signs (for example, `$Date$`) and check the file back into its archive.

---

**NOTE** If you add keywords to solution and project files, Integrity does not expand them on check in.

---

---

## Configuring the Location of VS Solutions and Projects in the Integrity Repository

---

**NOTE** This section is intended for Integrity administrators only. It is an optional feature related to sharing Visual Studio solutions and projects.

---

By default, sharing a Visual Studio solution allows you to choose where in the Integrity repository you want to place a Visual Studio solution and Visual Studio project as a top-level Integrity project and Integrity subproject. In the Integrity repository, the Visual Studio project is placed in the Visual Studio solution's Integrity project subtree. The solution and project are known as being "in-tree" in the repository structure.

As an alternative repository configuration, Integrity administrators can set an Integrity policy that automatically defines for users where top-level Integrity projects and Integrity subprojects are placed in the Integrity repository. If set, users do not choose the location where top-level Integrity projects and Integrity subprojects are placed in the Integrity repository when sharing a Visual Studio solution.

Note the following:

- This policy takes effect immediately; however, it only affects new Visual Studio solutions and projects. Existing shared solutions and projects are not affected.
- Once you add the policy lines *do not* remove them. To enable or disable the policy, change the values of `integration.catalog.vssolution.enabled` and `integration.catalog.vsproject.enabled` to `true` or `false`.
- The `integration.catalog.vsproject` and `integration.catalog.vssolution` settings should not point to the same top-level Integrity project or subproject.
- You must specify configuration paths for top-level Integrity projects and Integrity subprojects. Specifying project paths causes share operations to fail.

You can view a top-level Integrity project's or subproject's path information by viewing the Integrity **Project Information** dialog box.

### To define where top-level Integrity projects and Integrity subprojects are placed in the Integrity repository

- 1 In the Integrity Administration Client, open the **Configuration Management** node and select **Policies**.
- 2 In the right-hand pane, select **Global Policies**, and then select **Policies > Edit**. The **Global Policies** dialog box displays.
- 3 Click the **Other** tab.
- 4 To specify where top-level Integrity projects for Visual Studio solutions are placed, add the following lines:

```
integration.catalog.vssolution=<configuration path>  
integration.catalog.vssolution.enabled=<true/false>
```

To specify where Integrity subprojects for Visual Studio projects are placed, add the following lines:

```
integration.catalog.vsproject=<configuration path>  
integration.catalog.vsproject.enabled=<true/false>
```

---

where

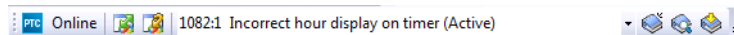
<configuration path> specifies the configuration path of the top-level Integrity project or Integrity subproject, for example, #p=/abcFinancialToolkitApp/abcFinancialToolkitAppsIn.pj.

<true/false> specifies if the policy is enabled or disabled.

- 5 To save your changes, click **OK**.

## Online and Offline Mode

The Visual Studio integration uses connection status with the Integrity Client and an Integrity Server to determine the integration's operating mode. In Visual Studio, the Integrity toolbar displays one of the following status indicators: **Connecting**, **Not Shared**, **Online** or **Offline**.



When you open Visual Studio and no solution is open, the integration cannot determine the connection status, and displays **Not Shared** in the Integrity toolbar. To establish a connection with the client and server, open a Visual Studio solution under Integrity source control or place a solution under Integrity source control. While attempting to establish a connection, Visual Studio displays **Connecting**.

When the integration connects to the Integrity Client and the client establishes a connection with the Integrity Server, the integration switches to **Online** mode and all Integrity commands and views are available.

If the Integrity Client cannot establish a connection with the Integrity Server or the previously established connection is lost, the integration switches to **Offline** mode. In offline mode, you can continue working in Visual Studio, performing only the Integrity commands that do not require a connection to the Integrity Server, such as adding or editing files. Any changes affecting Sandbox members are done without an active change package, requiring you to move the changes to an active change package when you switch to online mode. With the exception of the connection status indicator, all Integrity views and commands that affect the Integrity repository are disabled. Most notably, you cannot rename files or directories, and the **Solution Explorer** only updates glyphs for basic local operations, such as adding, deleting, and editing files.

If you are working in online mode and the server suddenly goes down or there is a poor network connection, the integration automatically switches to **Offline** mode, allowing you to continue working in Visual Studio. When the server connection is re-established, the integration automatically switches to **Online** mode and all Integrity commands and views automatically become active again, allowing you to resynchronize your changes with the Integrity repository.

---

**NOTE** Switching from offline to online mode refreshes the **Solution Explorer**, resynchronizing all resources in the Visual Studio solution and retrieving the latest glyphs. Depending on how many files or Visual Studio projects are visible in the solution, this process may take a long time; however, it occurs in the background and does not prevent you from working in Visual Studio.

---

You can also manually switch the integration to offline mode. Working in offline mode is recommended when you know that you do not have access to the server for long periods of time. For example, if you are working remotely without an Internet connection or if you intentionally disconnect your machine from the network by clicking **Online** and then **Go Offline**, you work in

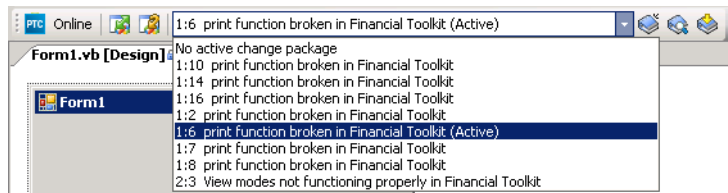
---

offline mode, allowing you to work on local files (edits and adds). When you reconnect to the network by clicking **Offline** and then **Go Online**, move your offline changes to a change package for submission to the Integrity repository.

## Working With Active Change Packages

The Visual Studio integration is designed to use change packages for submitting source code changes to the Integrity repository. The *active change package* is a change package that has been set as the default change package for member operations. This also allows other developers to identify what you are currently working on, because they can see which files you have locked and the associated change package.

The Integrity toolbar displays a change package list that allows you to select the active change package, and toolbar buttons to create, submit, and view change packages. Creating a change package makes it the active change package. For more information of using the toolbar buttons, see “Managing Work In Progress” on page 63.



By default, the active change package label displays the change package ID, change package summary, and **(Active)**, for example, **1:6 print function broken in Financial Toolkit (Active)**. If no active change package exists, **No active change package** displays. Text that exceeds the size of the label is truncated; however, you can hover your mouse over the label (if the change package is the active change package, **Currently Active Change Package** displays in a tooltip). If you are connected to more than one server, the server and port that the change package resides on also appears.



To select a different active change package, click the drop-down arrow button next to the change package list and select a change package or **No active change package**.

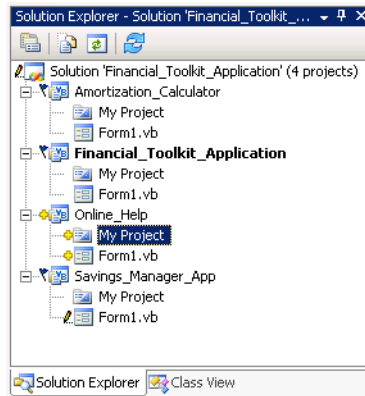
Note the following about using change packages with the Visual Studio integration:

- A change package is a container for *distinct* tasks, not a generic container for all tasks performed during a project. A change package containing too many entries may impact the integration’s performance. Before you perform a task that requires a change package, create a change package (this enables it as the active change package).
- An active change package cannot be used for multiple solutions.
- The active change package must be set in Visual Studio. If you set the active change package in the Integrity Client, the integration does not reflect the change.
- You can move added or modified change package entries from one change package to another. You cannot move renamed or moved change package entries from one change package to another.

---











## Integrity Glyphs in Visual Studio

The Visual Studio **Solution Explorer** and Integrity Work In Progress view are enhanced with Integrity glyphs to indicate Integrity source control status. For example, a flag glyph  displays beside a Visual Studio solution or project if it is under Integrity source control, and a pencil glyph  displays beside a file if it is checked out by you with a lock.






For files edited in Visual Studio, the integration automatically updates glyphs in the **Solution Explorer**.

The following Integrity glyphs display in the **Solution Explorer** and/or Integrity Work In Progress view:

Glyph	Definition
	Visual Studio Solution, Visual Studio project, or Web Application Project under Integrity source control. <b>Note:</b> This glyph does not display for Web sites under Integrity source control.
	Locally modified file.
	Locally added file.
	Locally dropped file.
	Locally renamed/moved file.
	File is a former member.
	Internal error retrieving status.
	Incoming change.
	Locally modified change that includes an incoming change.
	Locally moved/renamed file that includes an incoming change.

---

Glyph	Definition
	Locally modified file that includes a lock by another user.
	File locked by another user.
	Locally moved/renamed file that includes a lock by another user.

## Managing Work In Progress

All Integrity source control operations in the Visual Studio integration use change packages to track work. To manage your work in progress from the context of a change package, the Visual Studio integration includes an Integrity Work In Progress view that displays your current change package as a collection of files, displayed in a Visual Studio solution hierarchy.

From the Integrity Work In Progress view, you can do the following:

- view open change packages
- submit change packages
- move changes between change packages
- view work not associated with change packages
- revert changes
- move unassociated changes into change packages
- resolve conflicts with incoming changes
- discard change packages

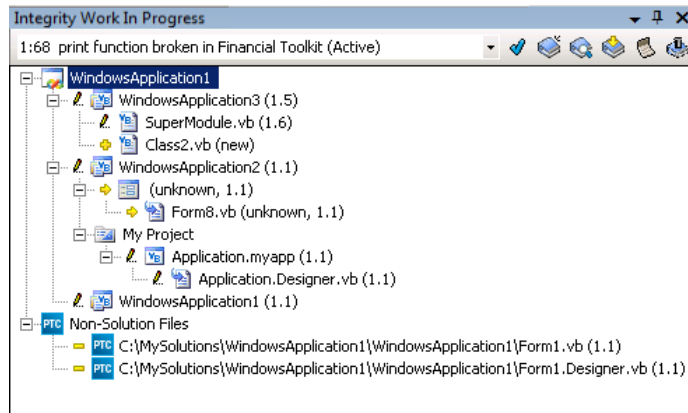
Note the following:

- All changed files are tracked with Integrity locks; however, Visual Studio does not register changes unless there are content modifications to the files. This means files that have not changed do not appear in the Integrity Work In Progress view. If you view the associated change package(s), the files are locked. When you submit the change package(s), the integration re-examines the files for content modifications and if none exist they are reverted (unlocked).
- The Integrity Work In Progress view dynamically updates according to changes in change packages.

### To display the Integrity Work In Progress view

By default, the Integrity Work In Progress view displays when you enable Integrity as the source control provider. If the view is not displayed, select **View > Other Windows > Integrity Work In Progress**. The Integrity Work In Progress view displays.

The Integrity Work In Progress view displays your current change package as a collection of working files, displayed in a Visual Studio solution hierarchy. All nodes in the hierarchy display alphabetically. Multiple CP entries may have to be collapsed so that they are represented as a single working file.



Revision numbers for files display in brackets, for example, `Application.Designer.vb (1.1)`. You can view more detailed information about a file by using the **View Member Properties** command. For more information, see “Advanced Integrity Commands” on page 80.

Note the following:

- After you branch a solution, the revision number that displays beside the solution node is the revision number of the solution file.
- Double-clicking a file in the Integrity Work In Progress view opens it for editing.

### File Status





The Integrity Work In Progress view only displays the status represented by working files. This includes glyphs for added, dropped, moved, renamed, and changed files. For more information, see “Integrity Glyphs in Visual Studio” on page 62.

### Incoming Changes



To notify you when one or more files need to be resynchronized, putting the latest revision in your Sandbox, the Integrity Work In Progress view displays glyphs for conflicts (incoming and outgoing changes), and remote drops by other users. For more information, see “Resynchronizing a Visual Studio Solution” on page 76.

### Managing Change Packages

The change package list allows you to view changes associated with a change package. From the toolbar, you can perform the following change package operations:

Command	Operation
<b>Set Active Change Package</b> (  )	Sets the selected change package as the active change package. ( <b>Active</b> ) appears after the change package name.
<b>Create Change Package</b> (  )	Creates a change package, making it the active change package. The <b>Create Change Package</b> dialog box displays.
<b>View Change Package</b> (  )	Displays the change package for the currently selected change package. If there is no selected change package selected, this command is disabled. The <b>Change Package</b> view displays.
<b>Submit Change Package</b> (  )	Submits the selected change package, committing the associated changes (files) to the Integrity repository. By default, confirmation of a successful submission does not appear. To display a confirmation message, enable the <b>Show Successful Submit</b> option for the <b>Submit Change Package</b> command in the Integrity Client.



Command	Operation
<b>View Integrity Item</b> (  )	Displays the Integrity item associated with the selected change package. The <b>View Item Details</b> view displays.
<b>Discard Change Package</b> (  )	Discards the selected change package and its associated entries.

## Managing Non-Solution Files

Files that are not in the open solution cannot be displayed in the solution hierarchy. Change package entries for non-solution files display as full working file paths in a flat list in the tree under the heading **Non-Solution Files**.

To commit the changes to the Integrity repository, select the non-solution files and move them to a change package by right-clicking and selecting **Move to Change Package**. The **Move Work to Change Package** dialog box displays. After you select a change package and click **OK**, you can submit the change package to commit the changes to the repository.

## Managing Unassociated Changes

Unassociated changes are changes made outside of the context of a change package. They can be made while in offline mode, or they can be made while **No active change package** is selected.

To evaluate these changes and move them to a change package, select **Unassociated Changes** from the top of the change package list in the Integrity Work In Progress view. With the unassociated changes displayed, move them to a change package by right-clicking the files and selecting **Move to Change Package**. The **Move Work to Change Package** dialog box displays. After you select a change package and click **OK**, you can submit the change package to commit the changes to the repository.

---

**NOTE** Visual Studio automatically makes changes to files based on events in Visual Studio, for example, changing the development server port number in Web application project files, and adding debug information to Web site configuration files. If a Visual Studio project is shared and an associated change package is closed before Visual Studio makes changes, those changes appear under **Unassociated Changes** in the Integrity Work In Progress view.

---

## Removing Entries From a Change Package

If you have an edit or deferred add entry in a change package that you do not want to submit with the change package, you can remove the entry, making it an unassociated change.

Note the following:

- Drop, rename, and move entries cannot be moved out of a change package, or from one change package to another.
- In the Integrity Work In Progress view, you cannot move non-solution files from the currently displayed active change package to another change package. To move non-solution files in an active change package, use the Integrity Client GUI.

With a change package displayed in the Integrity Work In Progress view, right-click the entry you want to remove, and select **Move to Change Package**. The **Move Work to Change Package** dialog box displays.

From the change package list, select **No active change package**, and click **OK**. The changes are unassociated with the change package and appear in the Integrity Work In Progress view when you select **Unassociated Changes** from the list.

---

When your changes are complete, you can move the unassociated changes to a change package for submission to the Integrity repository.

### Shortcut Menu Commands

Right-clicking a file in the Integrity Work In Progress view allows you to perform the following Integrity commands:

- **Move to Change Package** (only for edited or deferred add files)
- **Resynchronize** (only for incoming changes)
- **Revert** (only for edited files)
- **View Differences** (displays differences between the working file revision and current changes)
- **View Incoming Differences** (displays differences between the member revision and current working file; only for incoming changes)
- **View Annotated Revision**
- **View Member History**
- **View Member Properties**
- **Change Lock to Exclusive/Non-Exclusive**

## Managing Assigned Work

The Integrity Items view provides Integrity item functionality from within the Visual Studio interface, allowing you to manage your assigned items. This view replaces the Integrity Worktray found in previous releases of the Visual Studio integration.

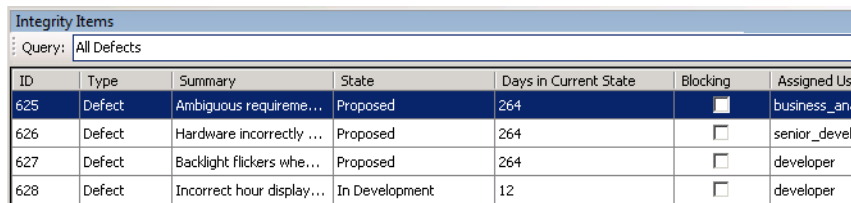
---

**NOTE** If you are currently connected to your default Integrity Server, you are not prompted to establish a connection. If you are not connected to your default Integrity Server, you may be prompted to choose an existing connection (if you are connected to another server), or create a new connection.

---

### To display the Integrity Items view

By default, the Integrity Items view displays when you enable Integrity as the source control provider. If the view is not displayed, select **View > Other Windows > Integrity Items**. The Integrity Items view displays. By default, the Integrity Items view is empty. After you run a query, the Integrity Items view displays items returned by the query.




The screenshot shows the 'Integrity Items' window with a query set to 'All Defects'. The table below represents the data shown in the window.

ID	Type	Summary	State	Days in Current State	Blocking	Assigned Us
625	Defect	Ambiguous requireme...	Proposed	264	<input type="checkbox"/>	business_an
626	Defect	Hardware incorrectly ...	Proposed	264	<input type="checkbox"/>	senior_devel
627	Defect	Backlight flickers whe...	Proposed	264	<input type="checkbox"/>	developer
628	Defect	Incorrect hour display...	In Development	12	<input type="checkbox"/>	developer

The Integrity Items view displays items based on the selected query. By default, no query is selected in the **Query** list. Query criteria must be defined and made visible from Integrity before that query can be used from the Integrity Items view. The **Query** list is built when you display the Integrity Items view and displays the Quick Query by default.






Note the following:

- You can refresh the list of queries by clicking .
- You can filter the **Query** list by typing the name of the query you are looking for. The query list displays the first matching query in the list and auto-completes.
- All queries visible to you display (favorites and non-favorites)

### Toolbar

The following operations are available from the toolbar on the Integrity Items view:

**TIP** Right-clicking an item in the Integrity Items view allows you to perform the following commands: **View Item**, **Edit Item**, and **Create Change Package** (if a Visual Studio solution is open and under Integrity source control).

Command	Operation
<b>View Item</b> (  )	Opens the selected item in the Item Details view.
<b>Edit Item</b> (  )	Opens the selected item for editing. <b>Note:</b> Refreshing an item after an edit is successful only if the edit is invoked from the Integrity Items view. If you view the item, and then edit the item from the Item Details view, the item does not refresh in the Integrity Items view until you run the query again.
<b>Create Item</b> (  )	Creates a new item.
<b>Refresh Query List</b> (  )	Refreshes the <b>Query</b> list, updating the definitions of available queries.
<b>Re-run Query</b> (  )	Runs the selected query in the <b>Query</b> list. When you run a query for the first time, a table appears below the toolbar, where the column names are fields defined by the query's column set. The table is sorted to match the sort direction and fields in the query definition. The default sort direction is the sort defined by the query definition. You can sort the tables by clicking the column headers. <b>Note:</b> Rich content appears as plain text in fields that support rich content.

## Placing Visual Studio Solutions Under Integrity Source Control

This section provides information on placing Visual Studio solutions, projects, Web Application Projects, and Web sites under Integrity source control. In addition, this section describes how to manage solutions under Integrity source control.

### Sharing a Visual Studio Solution

*Sharing* is the process that associates a Visual Studio solution and its Visual Studio projects with an Integrity project (the Visual Studio solution) and Integrity subprojects (the Visual Studio projects), storing the relative layout of the solution and projects in a Sandbox (which may not be the layout in the Integrity repository).

Once you place a Visual Studio solution and its projects under Integrity source control, you can perform Integrity operations, such as checking in files and viewing annotated revisions. To allow other users to work with the solution, they must import the solution (this creates a Sandbox of the solution under Integrity source control). For more information, see “Importing a Visual Studio Solution” on page 70.

---

Note the following:

- The Visual Studio integration supports Web Application Projects and Web sites on file systems. The integration does not support Web sites located via FTP and remote HTTP (HTTP links must be converted to local paths before placing the Web Application Project or Web site under Integrity source control).
- Sharing Visual Studio projects within a Web site is not supported.
- To avoid potential issues with updating the Integrity repository, always submit change packages from Visual Studio, not the Integrity Client.
- Your administrator may set policies that automatically define where on the Integrity Server the corresponding Integrity projects and Integrity subprojects for Visual Studio solutions and projects are placed. If these policies are defined, some of the steps in the following procedure may not appear.

### To share a Visual Studio solution

1 Do one of the following:

- To share an existing Visual Studio solution, select the solution file in the **Solution Explorer** and select **File > Source Control > Share Solution**.
- To share a new Visual Studio solution, select **File > New > Project**. The **New Project** dialog box displays.

Specify the project information and enable **Share Solution**, and then click **OK**.

If you did not disable prompting for server information and credentials in the Integrity Client, a dialog box appears, prompting you to select a connection or click **New** to specify a new server to connect to.

The **Share Solution** dialog box displays.

- 2 Sharing a solution requires associating it with a change package. Select a change package from the **Change Package** list, or create one by clicking **Create**. To proceed, click **Next**.
- 3 In the **Share Name** and **Share Description** fields, provide a unique name and description for the solution. The name and description appear to users when they import the solution in Visual Studio. To proceed, click **Next**.

---

**TIP** You can edit the descriptions of the Visual Studio project's shares, by clicking **Advanced**.

---

4 Specify where on the Integrity Server you want to add the solution. Choose one of the following options, and then click **Next**:

- **In a new Integrity project**  
Specify the root path, project path, and name of the top-level Integrity project, then click **Next**.
- **In a new Integrity subproject within an existing Integrity project**  
From the list, select the top-level Integrity project you want to add the new Integrity subproject to, then click **Next**. To filter the project list, type in the **Show project names containing** field to display top-level projects matching the name you type.  
Specify the name and location of the new Integrity project to be created as a subproject, then click **Next**.

---

- **In an existing Integrity project or Integrity subproject**

**NOTE** If the existing Integrity project contains identical files to the ones in the solution, the files in the Integrity project are updated by the working files in the solution.

---

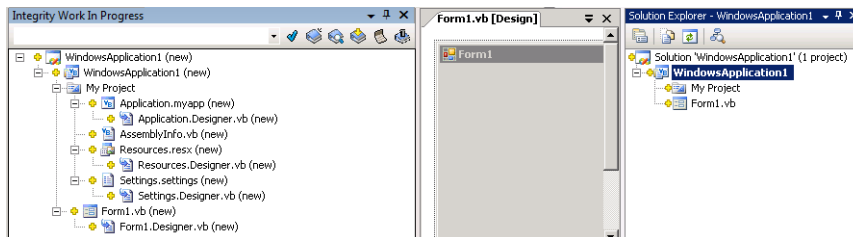
From the list, select the Integrity project you want to add the solution to, then click **Next**. To filter the project list, type in the **Show project names containing** field to display top-level projects matching the name you type.


- 5 Choose an option for adding Visual Studio solution files to the Integrity repository:
  - **Add all files in the Visual Studio solution to source control using the change package <CP>**. This option is enabled by default.
  - **Add only the Visual Studio solution and Visual Studio project files to source control using the change package <CP>**. This option is recommended if you are sharing a Visual Studio solution containing a large number of files (over several hundred). After the Integrity project and Sandbox are created, add the files to the solution in small batches using a separate change package for each batch.

Click **Next**. A summary of your choices displays.

- 6 To share the solution, click **Share**. The integration informs you that the solution will be made public (available for importing) when the change package associated with the share operation is submitted to the Integrity repository.
- 7 Click **OK**.

In the **Solution Explorer** and Integrity Work In Progress view, the solution, project, and its files display the  glyph, indicating they are deferred member adds (files not yet committed to the Integrity repository).





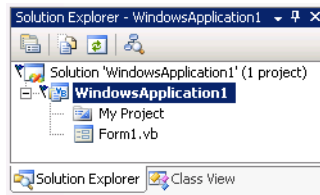
- 8 To submit the solution, project, and its files to the Integrity repository, submit the associated change package by clicking  in the Integrity toolbar or the Integrity Work In Progress view. The **Create Archive** dialog box displays. Follow the procedure for creating archives, as described in the *Integrity User Guide*.

---

**IMPORTANT** To avoid potential issues with updating the Integrity repository, always submit change packages from Visual Studio, not the Integrity Client.

---

After the change package is submitted successfully, the change package and its entries disappear from the Integrity Work In Progress view. In the **Solution Explorer**, the  glyph disappears from the solution, project, and files. The solution and project display the  glyph, indicating they are under source control.



## Importing a Visual Studio Solution

To allow other users to use a solution under Integrity source control, they must import the solution into Visual Studio. Importing a solution creates a Sandbox from the solution.

Note the following:

- Sharing a solution records the information necessary to import a solution; therefore, you must share a solution before you can import it (even if it is already in the repository).
- If the import fails, any Sandboxes created during the import remain on the file system.

### To import a solution

- 1 In Visual Studio, select **File > Source Control > Import Solution**. If you have a Visual Studio solution open that is under Integrity source control, the integration warns you that importing a solution closes the currently open solution. Before you import a solution, Integrity recommends submitting any solution changes to the Integrity repository with an associated change package.


To continue, click **Yes**. The **Import Solution** dialog box displays.

- 2 From the list, select the solution you want to import, then click **Next**. To filter the results, type in the **Show share names containing** text field to display solutions matching the name you type.

---

**NOTE** A branched solution displays the development path name in brackets after the solution name, for example, **FinancialToolkit (ServicePack1)**. For more information on branching a solution, see “Branching a Visual Studio Solution” on page 75.

---

- 3 Select the location on your local drive to create the Sandbox, then click **Next**. You can type the location or click **Browse** to select a directory. A summary of your choices displays.
- 4 To import the solution, click **Import**. The solution displays in the **Solution Explorer**. The solution and projects display the  glyph, indicating they are under Integrity source control.

## Adding a Visual Studio Project to a Shared Solution


You can place a new or existing Visual Studio project under Integrity source control by adding it to the shared Visual Studio solution it belongs to.


### To add a Visual Studio project to a shared Visual Studio solution

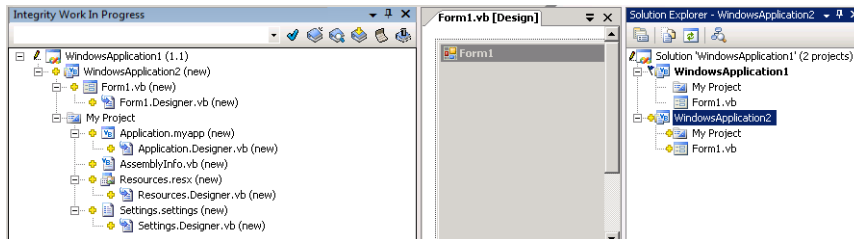
- 1 With a shared solution open, do one of the following:
  - To add an existing Visual Studio project to the solution, right-click the project in the **Solution Explorer** and select **Share Project**.
  - To create a new Visual Studio project and add it to the solution, right-click the solution and select **Add > New Project**.


Specify the project information and click **OK**.



The **Share a Visual Studio Project** dialog box displays.

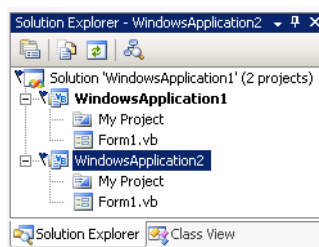
- 2 Sharing a project requires associating it with a change package. Select a change package from the **Change Package** list, or create one by clicking **Create**.
- 3 To add the Visual Studio project to the solution, click **OK**. In the **Solution Explorer** and Integrity Work In Progress view, the project and its files display the  glyph, indicating they are deferred member adds.

Adding a project to a solution requires writing to the solution file. To indicate that the solution file is checked out and locked for editing, a  glyph displays in the **Solution Explorer** and Integrity Work In Progress view.



- 4 To submit the project, and its files to the Integrity repository, submit the associated change package by clicking  in the Integrity toolbar or the Integrity Work In Progress view. The **Create Archive** dialog box displays. Follow the procedure for creating archives, as described in the *Integrity User Guide*.
- 5 The **Check In** dialog box displays for the solution file. Follow the procedure for checking in a member, as described in the *Integrity User Guide*.

After the change package is submitted successfully, the change package and its entries disappear from the Integrity Work In Progress view. In the **Solution Explorer**, the  glyph disappears from the project and its files, and the project displays the  glyph, indicating it is under source control.



## Dropping a Visual Studio Project From a Shared Solution

You can drop a Visual Studio project from a Visual Studio solution under Integrity source control by removing it from the solution. Removing the project removes the project information and project share information from the solution; the corresponding Integrity subproject and any subproject files are not dropped from Integrity source control. By design, this prevents the Integrity subproject from being dropped from Integrity source control if it is shared with another Visual Studio solution.


After you remove the project from the solution in Visual Studio, you must manually drop the subproject in the Integrity Client GUI to completely remove it from Integrity source control.


---


Before you drop the subproject, Integrity recommends verifying that it is not shared with another Integrity project.

### To drop a Visual Studio project

- 1 With an active change package set, right-click the project you want to remove in the **Solution Explorer** and select **Remove**. Visual Studio informs you that the project will be removed.
- 2 Click **OK**. The project disappears from the **Solution Explorer**.

Removing a project from a solution requires writing to the solution file. To indicate that the solution file is checked out and locked for editing, a  glyph displays in the **Solution Explorer** and Integrity Work In Progress view.

- 3 To submit the updated solution to the Integrity repository, submit the associated change package by clicking  in the Integrity toolbar or the Integrity Work In Progress view.
- 4 The **Check In** dialog box displays for the solution file. Follow the procedure for checking in a member, as described in the *Integrity User Guide*.

After the change package is submitted successfully, the change package and its entry disappear from the Integrity Work In Progress view and the **Solution Explorer**. The solution now displays the  glyph, indicating the changes to the solution file are committed to the Integrity repository.

- 5 Select **File > Source Control > View Solution Sandbox**. The **Sandbox** view for the solution displays.
- 6 Select the subproject that corresponds to the Visual Studio project you removed in Visual Studio.
- 7 Select **Member > Drop**. The **Drop Subproject** dialog box displays.
- 8 Follow the procedure for dropping a subproject, as described in the *Integrity User Guide*. The subproject is removed from the Integrity project.


## Importing a Visual Studio Project


Importing a Visual Studio project is useful if you want to add a Visual Studio project in a Visual Studio Solution to another Visual Studio solution. This adds the Visual Studio project to the solution's top-level Integrity project as a shared subproject. Both Visual Studio solutions must be under Integrity source control.

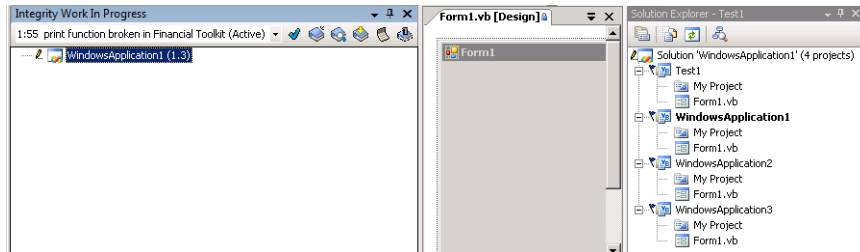
### To import a Visual Studio project


- 1 Open the Visual Studio solution you want to add a project to, and select **File > Source Control > Import Project**. The **Import Project** dialog box displays.
- 2 Importing a project requires associating it with a change package. Select a change package from the **Change Package** list, or create one by clicking **Create**. To proceed, click **Next**.
- 3 From the list, select the solution containing the project you want to import, then click **Next**. To filter the results, type in the **Show share names containing** text field to display solutions matching the name you type.
- 4 Select the location on your local drive to create the Sandbox, then click **Next**. You can type the location or click **Browse** to select a directory. A summary of your choices displays.




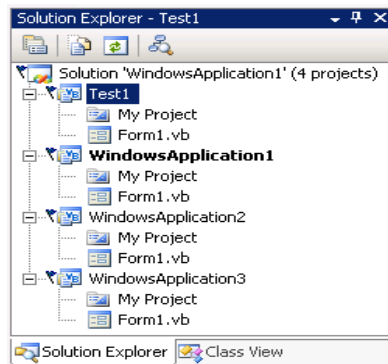
- To import the project, click **Import**. In the **Solution Explorer**, the solution displays the imported project. The project displays the  glyph, indicating it is under Integrity source control.

Importing a project to a solution requires writing to the solution file. To indicate that the solution file is checked out and locked for editing, a  glyph displays in the **Solution Explorer** and Integrity Work In Progress view.



- To submit the updated solution to the Integrity repository, submit the associated change package by clicking  in the Integrity toolbar or the Integrity Work In Progress view.
- The **Check In** dialog box displays for the solution file. Follow the procedure for checking in a member, as described in the *Integrity User Guide*.

After the change package is submitted successfully, the change package and its entry disappear from the Integrity Work In Progress view. The solution now displays the  glyph, indicating the changes to the solution file are committed to the Integrity repository.



## Migrating a Visual Studio Solution from the MKS SCC VS Integration

If you have an existing Visual Studio solution that was placed under version control using the previous MKS SCC-based Visual Studio integration, you can migrate the solution for use with the current integration.

### To migrate a Visual Studio solution from the MKS SCC Visual Studio integration

- In the MKS SCC-based Visual Studio integration, open the Visual Studio solution and remove the SCC binding by selecting **File > Source Control > Change Source Control > Unbind**.
- Ensure that all Visual Studio projects are saved and work properly.
- Enable the new Visual Studio integration, as described in “Setting Up and Configuring the Integration” on page 56.

- 
- 4 Share the solution, as described in “Sharing a Visual Studio Solution” on page 67.

---

**NOTE** After you submit the change package associated with the share operation, some Visual Studio projects may appear in the Integrity Work In Progress view under **Unassociated Changes**. These represent changes for unbinding the projects. When you share a Visual Studio solution, the integration does not submit any changes in existing Sandboxes unless the Visual Studio integration changed the files during the share operation.

---

- 5 If you add a new Visual Studio project to the migrated solution, right-click the project and select **View member properties**. The **Member Properties** dialog box displays.
- 6 Click **Allow The Integration To Add Visual Studio Projects To Source Control Automatically**. The button disappears.

Additional projects added to the solution are automatically added to source control; this button does not appear.

- 7 Click **Close**.

For each solution you migrate, repeat this procedure.

## Ignoring Visual Studio Entities From Integrity Source Control

There may be occasions where you want to ignore certain Visual Studio entities from source control, for example, temporary files (`.tmp`). When you ignore Visual Studio entities from source control, the entities are not annotated with glyphs in the **Solution Explorer** and are not valid selections for Integrity commands. The ignored entities are saved to an `.mkignore` file with the Visual Studio project you applied it to, adding it as a member of the Integrity subproject.

To apply the same ignore list to other projects, you must set them on a per-project basis.

---

**CAUTION** Do not edit the `.mkignore` file.

---

### To ignore Visual Studio entities



- 1 With an active change package set, right-click the Visual Studio project you want to create an ignore list for in the **Solution Explorer**, and select **Ignore from Source Control**. The **Files Ignored From Source Control** dialog box displays.
- 2 To add a file type to the filter list, click **Add** and type the file type. To select multiple files, use wildcard characters (`*` or `?`). For example, to ignore all temporary files, type `*.tmp`.
- 3 To preview the files in the Visual Studio project that will be ignored, select the filter from the ignore list and click **Preview**. A dialog box displays a list of files that will be ignored.

---

**NOTE**

- If a file has a lock on it, you cannot create an ignore filter for that file. The integration warns you about the affected files, but does not add them to the ignore list. Existing locks are not removed.
  - If you choose to ignore file types that are currently in change packages, the integration warns you about the affected files, but does not add them to the ignore list. To ignore the file types, resolve the change packages (submit the change packages or remove specific change package entries), and then add the file types to the ignore list.
- 

- 4 Click **OK**.

- 
- 5 To save your changes, click **OK**. In the Integrity Work In Progress view, the `.mkignore` file displays the  glyph, indicating it is a deferred member add.
  - 6 To submit the file to the Integrity repository, submit the associated change package by clicking  in the Integrity toolbar or the Integrity Work In Progress view. The **Create Archive** dialog box displays. Follow the procedure for creating archives, as described in the *Integrity User Guide*.

After the change package is submitted successfully, the `.mkignore` file disappears from the Integrity Work In Progress view.

## Branching a Visual Studio Solution

Development paths are used to deliberately create a parallel branch of development for the purpose of experimenting with research or performing post-release maintenance. Integrity allows multiple developers to point to the same development path, each using their own variant Sandbox.

When you need to create a new branch (development path) from an existing Visual Studio solution under Integrity source control, you branch the solution. Branching a solution checkpoints the solution, creates a development path on all Integrity projects associated with the solution in the Integrity repository, and creates a Sandbox for the branched solution.

---

**IMPORTANT** Before you branch a solution, you should submit solution changes to the Integrity repository with an associated change package. If you perform the branch operation without committing your changes, the uncommitted changes do not appear in the branched solution.

---

### To branch a solution

- 1 With the Visual Studio solution open that you want to branch, select **File > Source Control > Branch Solution**. The integration warns you that the operation creates a branch of the open solution, based on the data currently committed to the Integrity repository. In addition, the open solution will close and a new Sandbox will be created.

---

**TIP** If no solution is open, you can select **File > Source Control > Branch Solution**; however, you are prompted to select a solution from a list of shared solutions.

---

- 2 To continue, click **Yes**. The **Branch Solution** dialog box displays.
- 3 Branching a solution requires associating it with a change package. Select a change package from the **Change Package** list, or create one by clicking **Create**. To proceed, click **Next**.
- 4 Type a unique name for the **Development Path**, then click **Next**.
- 5 In the **Share Name** and **Share Description** fields, provide a unique name and description for the solution. The name and description appear to users when they import this branch of the solution in Visual Studio. To proceed, click **Next**.

---

**TIP** You can edit the descriptions of projects, by clicking **Advanced**.


---

- 6 Select the location on your local drive to create the Sandbox, then click **Next**. You can type the location or click **Browse** to select a directory. A summary of your choices displays.

---


7 To branch the solution, click **Branch**. The integration informs you that the branched solution will be made public (available for importing) when the change package associated with the branch operation is submitted to the Integrity repository.

8 Click **OK**.


Branching a solution requires writing to the solution file. To indicate that the solution file is checked out and locked for editing, a  glyph displays in the **Solution Explorer** and Integrity Work In Progress view.

9 To submit the branched solution to the Integrity repository, submit the associated change package by clicking  in the Integrity toolbar or the Integrity Work In Progress view.

10 The **Check In** dialog box displays for the solution file. Follow the procedure for checking in a member, as described in the *Integrity User Guide*.


After the change package is submitted successfully, the change package and its entry disappear from the Integrity Work In Progress view. In the **Solution Explorer**, the  glyph disappears. Other users can now create a Sandbox from the branch by importing the solution. In the **Import Solution** dialog box, a branched solution displays the development path name in brackets after the solution name, for example, `FinancialToolkit (ServicePack1)`.

## Resynchronizing a Visual Studio Solution

Resynchronizing a Visual Studio solution under Integrity source control updates your Sandbox with the latest solution, project, and files. For example, if you see the  glyph in the Integrity Work In Progress view and **Solution Explorer**, there is a new member revision available and you should resynchronize the solution.

### To resynchronize a Visual Studio solution

Do one of the following:

- Select **File > Source Control > Resynchronize Solution**.
- Click  in the Integrity toolbar.
- Right-click the solution in the **Solution Explorer** and select **Resynchronize Solution**.

The integration updates the solution, removing any incoming changes from the Integrity Work In Progress view and updating the glyphs in the **Solution Explorer**.

## Reverting a Visual Studio Solution

Reverting a Visual Studio solution reverts all files in the solution containing working file changes; however, it does not remove any recently added files to the solution from the file system. In most Visual Studio projects, a revert operation removes recently added files from the project and associated change package. In a Web site project, a revert operation removes recently added files in the associated change package and displays the files under **Unassociated Changes** in the Integrity Work in Progress view. To remove the files from the web site project, delete them.

---


**IMPORTANT** If change package reviews are enabled, submit any pending revisions before you revert a Visual Studio solution.

---

---

## To revert a Visual Studio solution

Do one of the following:

- Select **File > Source Control > Revert Solution**.
- Click  in the Integrity toolbar.
- Right-click the solution in the **Solution Explorer** and select **Revert Solution**.

A warning dialog box displays warning you that reverting the solution overwrites all of your changes that have not been committed to the Integrity repository, and cannot be cancelled.

To proceed, click **OK**. The integration updates the solution, removing the change package entries from the Integrity Work In Progress view and updating the glyphs in the **Solution Explorer**.

## Checkpointing a Visual Studio Solution

Checkpointing a Visual Studio solution creates a new revision of the associated Integrity project and adds it to the project history. When you checkpoint a solution, you save all the information needed to recreate the solution completely as it existed when you checkpointed it. The saved information includes the solution and project structure and the list of files with their revision numbers.

### To checkpoint a Visual Studio solution

- 1 Open the Visual Studio solution you want to checkpoint.
- 2 Select **File > Source Control > Checkpoint**. The **Checkpoint Solution** dialog box displays.
- 3 Add a **Label** and a **Description**.
- 4 Click **OK**.

## Viewing a Sandbox for a Visual Studio Solution


You can view the top-level Sandbox for an open Visual Studio solution by selecting **File > Source Control > View Solution Sandbox**. The **Sandbox** view displays.

For more information on the Sandbox view, see the *Integrity User Guide*.

## Working With Visual Studio Files

To provide a more seamless Integrity experience within Visual Studio, most basic Integrity file commands, such as adding members and checking out files occur implicitly as deferred operations when you perform the equivalent Visual Studio commands.

Before you perform an Integrity command, ensure that an active change package is selected in the change package list. After you complete the Integrity command, submit the change package to commit the changes to the Integrity repository. For more information on submitting change packages, see “Managing Work In Progress” on page 63.

Most file commands require writing to the solution or project file. When you perform a file command, the solution or project may display a  glyph in the **Solution Explorer** and Integrity Work In Progress view, indicating that the solution or project file is checked out and locked for editing.

---

## Adding Members to an Integrity Project





Adding a file to a Visual Studio project performs a deferred add operation on the file.

---

**NOTE** When using the **Copy** command, the revision history is not copied from the source location to the target location.

---

### To add a member to an Integrity project

- 1 In Visual Studio, add a file to a Visual Studio project under Integrity source control. In the Integrity Work In Progress view and **Solution Explorer**, a  glyph displays next to the new file, and a  glyph displays next to the project file.
- 2 Submit the change package associated with the add. The **Create Archive** dialog box displays. Follow the procedure for creating archives, as described in the *Integrity User Guide*.
- 3 The **Check In** dialog box displays for the project file. Follow the procedure for checking in a member, as described in the *Integrity User Guide*. After the change package is submitted successfully, the change package and its entry disappear from the Integrity Work In Progress view. In the **Solution Explorer**, the  and  glyphs disappear.





## Dropping Members From an Integrity Project

Deleting or removing a file from a Visual Studio project performs a deferred drop operation on the file. After the change package is submitted, the file is deleted from the file system and removed from the Integrity repository.

Note the following:


- For a Web site under Integrity source control, the Visual Studio **Exclude From Project** command simply removes the file from the Web site, leaving the file in the file system and under Integrity source control. The Visual Studio **Delete** command deletes the file from the file system and removes it from the Integrity repository.
- Although you can delete files from a Visual Studio project while in offline mode and without an active change package, the deleted files are not dropped from Integrity source control and do not appear in the Integrity Work In Progress view under **Unassociated Changes**. In addition, the files are deleted from the Visual Studio project and file system, preventing you from moving them to a change package. To allow you to recover or resolve deleted files, you should delete files in online mode and with an active change package.

### To drop a member from an Integrity project


- 1 In Visual Studio, delete or remove a file from a Visual Studio project under Integrity source control. The file is removed from the **Solution Explorer**. In the Integrity Work In Progress view, a  glyph displays next to the file under **Non-Solution Files** because it is not associated with the solution anymore. In the **Solution Explorer** and Integrity Work In Progress view, a  glyph displays next to the project file.
- 2 Submit the change package associated with the drop. The **Drop Member** dialog box displays. Follow the procedure for dropping members, as described in the *Integrity User Guide*.
- 3 The **Check In** dialog box displays for the project file. Follow the procedure for checking in a member, as described in the *Integrity User Guide*. After the change package is submitted successfully, the change package and its entry disappear from the Integrity Work In Progress view. In the **Solution Explorer**, the  and  glyphs disappear.


---

## Checking Out Members

In Visual Studio, open a file for editing and make changes. After you save changes to the file, a  glyph displays next to it in the Integrity Work In Progress view and **Solution Explorer**, indicating the file is checked out by you and has been modified.

## Checking In Members

In Visual Studio, submit the change package associated with a file checked out by you, indicated by a  glyph in the Integrity Work In Progress view and **Solution Explorer**.

The **Check In** dialog box displays for the file. Follow the procedure for checking in a member, as described in the *Integrity User Guide*. After the change package is submitted successfully, the change package and its entry disappear from the Integrity Work In Progress view. In the **Solution Explorer**, the  glyph disappears.





## Renaming Members

Renaming a file in Visual Studio performs a deferred rename operation on the file.

---

**NOTE** If you attempt to rename a file locked by you in another change package, the rename operation is performed in the change package containing the locked file. For example, if `Form.vb` is locked by you in change package `12:1` and your active change package is `12:2`, renaming `Form.vb` to `Form2.vb` performs the rename operation in `12:1` because `Form.vb` already exists in that change package. This also applies to moving a file or locking a file that is a deferred rename operation in a change package.

---

- 1 In Visual Studio, rename a file in a Visual Studio project under Integrity source control. In the **Solution Explorer**, the renamed file displays a  glyph. In the Integrity Work In Progress view, the renamed file displays a  glyph. In Integrity Work In Progress view and the **Solution Explorer**, the Visual Studio project displays a  glyph.
- 2 Submit the change package associated with the rename. The **Check In** dialog box displays for the project file.
- 3 Follow the procedure for checking in a member, as described in the *Integrity User Guide*. The **Check In** dialog box displays for the renamed file.
- 4 Check in the member. After the change package is submitted successfully, the change package and its entries disappear from the Integrity Work In Progress view. In the **Solution Explorer**, the  glyph disappears.

## Moving Members

You can move folders and files between projects in a Visual Studio solution, which performs a deferred move operation.

Note the following:

- Dragging and dropping, or using the **Cut/Copy** command to move a folder or file performs a deferred drop and add operation, effectively erasing the revision history. If you use these commands to move a file in offline mode without an active change package, the file is removed from the Visual Studio project but not from Integrity source control. To move a file, you should use the **Integrity Move From/To** commands in online mode with an active change package.

- Moving a file to a location where a file with the same name exists results in an error.
- If you move an entire tree of files, the source folders are still visible in the **Solution Explorer**. After moving a tree of files, you must manually remove the source folders.

### To move files/folders between projects



1 Do one of the following:


- In the **Solution Explorer**, right-click the file/folder you want to move and select **Integrity Move From**.

Select the project you want to move the file/folder to, then right-click and select **Integrity Move To**.

- Drag the file/folder you want to move and drop it in the desired project.
- Right-click the file/folder you want to move and select **Cut/Copy**.

Select the project you want to copy the file/folder to, then right-click and select **Paste**.

In the **Solution Explorer**, a moved file displays a  glyph. In the Integrity Work In Progress view, a moved file displays a  glyph.

2 Submit the change package associated with the move. After the change package is submitted successfully, the change package and its entry disappear from the Integrity Work In Progress view. In the **Solution Explorer**, the  glyph disappears.

## Advanced Integrity Commands

You can access advanced Integrity member functionality from within Visual Studio by right-clicking a Visual Studio solution, Visual Studio project, or file in the **Solution Explorer** or Integrity Work In Progress view and choosing a command from the short cut menu.

---

**NOTE** In the **Solution Explorer**, short cut menu operations operate on the selected file. The short cut menu in the editor window operates on the currently open file. If Visual Studio does not allow you to select the file and display a short cut menu, you can click the "**Show All Files**" button in the **Solution Explorer**, and use the context menu operations from there.

---

The following Integrity member commands are available:

Command	Function
<b>Resynchronize</b>	Equivalent to the Integrity <b>Resynchronize</b> command. Gets the latest version of the selected file and puts it in your working directory. This command is only available from the <b>Solution Explorer</b> and Integrity Work In Progress view when incoming changes are present. The <b>Overwrite</b> dialog box may display.
<b>Revert</b>	Equivalent to the Integrity <b>Revert</b> command. Replaces the working file with the revision that was checked out, as it appeared prior to modification, and unlocks the file (and removes it from the associated change package). This command is only available from the Integrity Work In Progress view. The <b>Overwrite</b> dialog box may display. Note the following: <ul style="list-style-type: none"> <li>■ You can revert a file in a change package that is not the active change package. Reverting the file removes it from the change package.</li> <li>■ If you add a file to a Visual Studio project and want to revert the operation, remove the file from source control (or if you removed it from the project, add it again). If you rename or move a file and want to revert the operation, move or rename the file again. The <b>Revert</b> command is intended for reverting changes to a working file.</li> </ul>



Command	Function
<b>View differences</b>	Equivalent to the Integrity <b>Differences</b> command. Compares the selected working file with the member revision. Visual Difference automatically launches and displays the two files.
<b>View annotated revision</b>	Equivalent to the Integrity <b>View Annotated</b> command. Displays the annotated revision history of the selected file. The <b>Annotated Revision</b> view displays.
<b>View member history</b>	Equivalent to the Integrity <b>View Member History</b> command. Displays the revision history of the selected file. The <b>Member History</b> view displays.
<b>View member properties</b>	Equivalent to the Integrity <b>View Member Information</b> command. Displays the member information of the selected file. A dialog appears, displaying server name and port number, solution share name, solution project, member name, project share name, project configuration path, member name, Sandbox, member revision, working revision, and development path (if applicable). To view additional Integrity information, click <b>Integrity Member</b> . The <b>Member Information</b> view displays.

## Best Practices

This section describes best practices for using the Visual Studio integration. It also points out efficiencies that can be gained by using the integration in a certain way, as well as identifying any risks, constraints, or other limits within the integration or within a particular implementation of the integration.

- **Displaying working file status for overlapping Sandboxes/subsandboxes**

To ensure that correct status displays for working files in overlapping Sandboxes/subsandboxes, you should:

- Not change CreateSubproject policies (creating subprojects for every folder when adding/moving members) without also manually restructuring existing project trees.
- Avoid using mixed CreateSubproject policies across Visual Studio solutions and projects.

- **Notification of file changes in Visual Studio solutions**

To prevent file change notifications from appearing when you resynchronize or revert a Visual Studio solution, set the following preferences in Visual Studio:

- a) Select **Tools > Options > Environment > Documents**.
- b) Enable **Detect when file is changed outside the environment** and **Auto-load changes, if saved**.
- c) To save your preferences, click **OK**.

- **Creating In-Tree Visual Studio projects and Web sites**

As a best practice, place Visual Studio projects and Web sites in-tree with the associated Visual Studio solution to simplify source control operations for other users. Visual Studio projects and Web sites that are out-of-tree can cause confusion or problems when users import or branch solutions.

- **Setting Up Projects to Reuse Code**

The ability to reuse code is an important part of managing software development. If you set up your project structure so that each component is self-contained, you can share that component with other solutions through the use of Integrity subprojects. By referencing the original subproject, Integrity allows you to share a subproject and the members it contains

---

between two or more projects. Shared subprojects do not have to be located within the same directory structure or project hierarchy.

To share project code, import an existing Visual Studio project to the Visual Studio solution that you want to share the project with. Projects are shared at the solution level; Web sites are shared at the solution root level. For more information on importing Visual Studio projects, see “Importing a Visual Studio Project” on page 72.

- **Creating and sharing a Visual Studio project**

The following option should always be enabled in Visual Studio before you create a Visual Studio project: **Tools > Options > Projects and Solutions > General > Save new Projects when created**. By default, this option is enabled. If this option is disabled, a Visual Studio project cannot be shared.

- **Ignoring files from source control**

There may be occasions where you want to ignore certain Visual Studio entities from Integrity source control, for example, temporary files (`.tmp`). As a best practice, specify the entities you want to ignore before you add files to a Visual Studio project. For best results, ignored files should not be placed under source control.

- **Java memory usage**

If you are working with large solutions or large change packages, increase the Java heap size on the Integrity Client for better performance, for example, 512 MB. For more information, contact PTC-Integrity Support.

- **Working with multiple Visual Studio Solutions**

Opening a Visual Studio solution closes the currently open Visual Studio solution. You should save your changes and submitting any associated change packages before you open another Visual Studio solution.

- **Submitting change packages in Visual Studio**

To avoid potential issues with updating the Integrity repository, always submit change packages from Visual Studio, not the Integrity Client.

- **Resolve conflicts before submitting change packages**

Integrity allows you to resolve conflicts when checking in members; however, PTC recommends resolving conflicts by resyncing the repository before submitting a change package. This avoids potential workflow issues and allows compiling and testing with the incoming changes in the Sandbox.

- **Refactoring in a Team Environment**

When working in a team environment, there are refactoring scenarios that users should be aware of or avoid to prevent conflicts.

---

**IMPORTANT** To allow you to recover or resolve refactored files, PTC strongly recommends that you perform all refactoring operations (add, drop, move, and rename) in online mode with an active change package.

---

#### *Renaming the Same Member:*

David renames `Module1.vb` to `Module2.vb`, but does not submit the change package.

Mary renames `Module1.vb` to `Module3.vb` and submits the change package.

In his Sandbox, David sees that a rename and lock operation are removed from his change package, and that `Module2.vb` is now a former member.

---

To recover and save his work, David must manually make changes.

*Renaming a Locked Member:*

In a change package, Mary has a lock on `Application1.vb`.

David cannot rename `Application1.vb` because of the lock.

David can drop `Application1.vb` and add it as `Application2.vb`, ignoring Mary's lock on `Application1.vb`.

*Adding the Same Member:*

David adds `Form1.vb` to a project, but does not submit the change package.

Mary also adds `Form1.vb` to the same project and submits her change package.

In his Sandbox, David sees that the project file contains a local modification and an incoming change. `Form1.vb` displays as an add operation.

If David submits his change package, an error message displays informing him that the member is already a member of the project. Refreshing the **Solution Explorer** removes the add operation glyph.

The change package can no longer be submitted. Each attempt to submit the change package results in a new member revision with the changes David made before submitting the change package.

- **Backing up Visual Studio share information**

Information about shared Visual Studio solutions and projects is stored to the following file on the Integrity Server:

```
<Integrity Server installdir>/data/vsi/vsibinding.properties
```

PTC recommends backing up this file along with other Integrity Server files; however, do not edit the file. Editing the file may cause shares to not work properly.

- **Viewing visible fields in query results**

When running a query from the Integrity Items view in Visual Studio, query results are displayed with the **ID**, **Type**, **State**, and **Summary** columns, even when additional columns are set as visible. To see all visible fields, you must set the column set to **Custom**.

To see all visible fields when running a query, open the Integrity Client and select the target query. Choose **Query > Edit > Column Set** and select the **Custom** option. When running the query, all visible columns are then displayed.

## Limitations

- While loading a Visual Studio solution, Integrity commands may not appear consistently in shortcut menus.
- Migrating a build Sandbox from the previous SCC-based Visual Studio integration to the current SDK integration is not supported. This is due to the fact that a migration causes the member files to change, which is a limitation of build Sandboxes.
- When two users simultaneously add Visual Studio projects to a Visual Studio solution, the project information is not saved in the merged Visual Studio solution file for the user who

---

submitted the solution file first. This generates an error message stating that Visual Studio cannot load the added project.


To avoid this error, select **Option 2 - resynchronize the member revision via change package, merging as needed**. If you do not select Option 2, you must resynchronize the Visual Studio solution containing the unavailable project.

- If you enabled password prompting on the Integrity Client, the integration cannot use the specified default password. When prompted in the integration, you must re-type your password.
- Moving a file from one Visual Studio project to another may take a long time to complete if your Recycle Bin contains too many items. This is a known issue with Visual Studio. Emptying your Recycle Bin improves the speed of the move operation.
- The Visual Studio integration does not support renaming Visual Studio solutions, Visual Studio projects, or Web sites. You can rename directories; however, you must commit any changes to files in a directory before the rename operation. If you are moving a directory, you must also commit any changes to files in the directory before the move operation.
- Importing a Visual Studio solution containing unshared Visual Studio projects displays an error message for each unshared project in the solution.
- Date-only fields may incorrectly display as date/time fields in the Integrity Items view.
- If you create two top-level Integrity projects for a Web site solution and Web site project, and two separate Sandboxes associated with each Integrity project, when you share the Web site solution in Visual Studio, the Web site project files are not included in the change package used for the share operation. The Web site project files display as **unassociated changes** in the Integrity Work In Progress view and only the solution is shared. To resolve this issue, move the Web site files from the **unassociated changes** to a change package and submit the change package.

## Troubleshooting

To assist you in diagnosing issues that may arise when using the Visual Studio integration, the **Integrityvsi.log** file is created on the machine where Visual Studio and the Integrity Client are installed. The log file records information, warnings, and error messages.

To open the log file, select **Tools > Options > Source Control** and then choose the **Integrity** plug-in. The **Logging** option displays. Click **Logging** and in the right pane click **Open**.

If you require assistance, contact PTC-Integrity Support and provide them with the log file. In addition, include the version number of the Integrity Visual Studio integration and the Integrity Client it is communicating with (in Visual Studio, click  in the Integrity toolbar).



The purpose of this chapter is to assist you in implementing and using the Integrity integration with Microsoft® Visual Studio® .NET 2003. The integration provides a streamlined process for adding projects and solutions under source control, including the **Change Package** field that allows you to pre-select a change package when adding a solution to source control.

This chapter helps you decide how to use the features of Visual Studio .NET 2003 with Integrity for configuration management, and suggests ways to implement the integration so that it has minimal impact on end users and minimal administrative overhead. It also describes how users can access Integrity from Visual Studio .NET 2003 once the integration is implemented.

---

**NOTE** MKS Worktray for Microsoft Visual Studio .NET 2003 is no longer supported.

---

To help you set up and use the integration, this chapter discusses the following topics:

- “Implementing the Integration” on page 87
- “Setting Preferences” on page 90
- “Parallel Development Considerations” on page 91
- “Building Projects” on page 93
- “Command Functionality” on page 94
- “Troubleshooting” on page 101

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Implementing the Integration

This section provides information on implementing the integration and includes the following topics:

- “Assumptions” on page 87
- “Required Permissions” on page 87
- “Implementing the New Project Structure” on page 87
- “Setting Up Projects to Reuse Code” on page 88

Existing Visual Studio .NET, 2002, and 2003 solutions, and projects integrated with earlier versions of Integrity, are not affected when you upgrade to Integrity.

### Assumptions

Before you implement the Visual Studio .NET 2003 integration, you should have a good understanding of both Integrity and Visual Studio .NET 2003, particularly in the areas of managing projects and solutions.

### Required Permissions

When using the Integrity integration with Microsoft Visual Studio, certain additional permissions are required for the user running the following commands:

```
Add Solution to Source Control
Add Selected Projects to Source Control
```

The user adding the project requires the following permissions in the Integrity Access Control List (ACLs):

```
ModifyProjectAttribute
CreateSubproject
```

For more information on configuring ACLs for Integrity, see the *Integrity Server Installation and Configuration Guide*.

### Implementing the New Project Structure

Visual Studio .NET 2003 requires integrated source control systems to use a new project structure, based on the concept of a solution root.

#### Visual Studio .NET 2003 Solution Root

A *solution root* provides a single location in source control where all the contents of a solution are stored, even when the solution contains projects that are stored in different locations in the file system.

For example, if a solution contains both a Web project (such as an ASP.NET application) and a non-Web project (such as a class library), the Web project files will be located in a different directory than the non-Web project files. This is because in Visual Studio .NET 2003, Web project files must be located in the Internet Information Server `/Inetpub` directory. When you add the solution to source control, both the Web and the non-Web project files are added under a parent solution root project.

---

For more information on the solution root concept, see <http://msdn.microsoft.com>.

## How the Solution Root Is Used in Integrity

When you add a multi-project solution to Integrity for configuration management, the following options are available:

- The **Create Project Wizard** provides functionality to create a new project, create a project as a subproject of an existing project, or create a project as a subproject of a new root project.
- Additional subprojects are created for each Web project and for each non-Web project on a different folder or drive, if the SLN had multiple project or subprojects.

For detailed instructions, see “Adding a Solution to Integrity” on page 96.

## Best Practices for Solution Root Project Structure

To make the integration with Visual Studio .NET 2003 work smoothly, consider the following points:

- When creating non-Web projects in Visual Studio .NET 2003, it is important to always create your projects so that they are inside a solution. Always select the **Create directory for Solution** option in the **New Project** dialog box. This creates a directory for your solution and a second directory inside the solution directory for your project.
- When creating any projects that are outside the solution tree, create a blank solution (**File > New > Blank Solution**) and specify a location for it. You can then add your projects to this solution.
- When performing administrative tasks such as checkpointing or creating a development path, it is important to always use the top level (solution root) project. Since this is not possible from within Visual Studio .NET 2003, you should perform these procedures using the Integrity Client.

## Setting Up Projects to Reuse Code

The ability to reuse code is an important part of managing software development. If you set up your project structure so that each component is self-contained, you can share that component with other solutions through the use of Integrity subprojects. By referencing the original subproject, Integrity allows you to share a subproject and the members it contains between two or more projects. Shared subprojects do not have to be located within the same directory structure or project hierarchy.

To share project code, you need to add the project as a shared subproject in Integrity, then add it as an existing project in Visual Studio .NET 2003. Non-Web projects are shared at the solution level; Web projects are shared at the solution root level.

---

**NOTE** Within the integration, sharing a variant subproject is possible but not recommended.

---

### To set up shared subprojects for non-Web applications

- 1 In the Visual Studio .NET 2003 **Solution Explorer** panel, select the solution, then select **File > Source Control > MKS SCC Integration**.
- 2 In the **Sandbox** view, select the solution Sandbox and add the shared subproject.

For more information on how to set up shared subprojects, see the *Integrity User Guide*.



- 
- 3 Resynchronize the new shared subproject.
  - 4 Close the **Sandbox** view.
  - 5 In the Visual Studio .NET 2003 **Solution Explorer** panel, select the solution, then select **Add > Existing Project**.
  - 6 Select the project to add. You are prompted to check out the solution file.
  - 7 Check out the solution file.
  - 8 Check in the solution file.

### To set up shared subprojects for Web applications

- 1 In Integrity, select and open the solution root project and add the shared subproject.

---

**IMPORTANT** You must share the subproject using the same name as the original subproject, for example, `WebApplication1/project.pj`.

---

For more information on how to set up shared subprojects, see the *Integrity User Guide*.

- 2 Create a Sandbox from the newly shared subproject.
- 3 In the Visual Studio .NET 2003 **Solution Explorer** panel, select the solution, then select **Add > Existing Project**.
- 4 Select the project to add. You are prompted to check out the solution file.
- 5 Check out the solution file.
- 6 Check in the solution file.

### To share project members

- 1 Select a project in Visual Studio .NET 2003 and choose **Source Control > Share**. The **Add Members From Archive** wizard displays.
- 2 Select a member to be added from archive and click **Finish**. The **Add Member From Archive** dialog box displays.
- 3 To add the member from archive, click **OK**. The selected member is added to the Visual Studio .NET 2003 project and shares the same archive as the member in the source project.

---

**NOTE** If the member revision is updated in the source project, it is not updated in the project it is shared to. The member revision in the shared project must be updated manually.

---

### Setting Up Master Projects

If you want to group multiple solutions, you need to create a master project to contain the solutions.

To group solutions under a master project, you need to create your project structure in Integrity, then add your Visual Studio .NET 2003 solutions to it.

---

### To set up Master Projects

- 1 In Visual Studio .NET 2003, create your solution.
- 2 In Integrity, create your master project.  
For more information, see the *Integrity User Guide*.
- 3 Create the subproject that you want to use as your solution root.  
For more information, see the *Integrity User Guide*.
- 4 Create a subproject inside the subproject you created in step 3 and name it `<solution name>/project.pj`.
- 5 Create a Sandbox from the subproject you created in step 4 and place it in your solution directory.
- 6 In Visual Studio .NET 2003, select your solution and click **File > Source Control > Add Solution to Source Control**. You are prompted to add the solution files to your Sandbox location.
- 7 Repeat steps 1-6 for additional solutions.

## Setting Preferences

You can now set Integrity Client preferences from within Visual Studio .NET 2003 using the **Preferences Configuration** option.

### To set Integrity preferences

- 1 Select **Tools > Options**. The **Options** dialog box displays.
- 2 Click to expand the **Source Control** folder in the left pane, and select **SCC Provider**.
- 3 In the right pane, click **Advanced**. The Integrity Client **Preferences Configuration** panel displays, allowing you to modify settings for the Integrity Client and Integrity.
- 4 Make the necessary changes to the preferences. For more information, see “Required Settings” on page 90.
- 5 Click **OK** to save the changes and close the **Preferences Configuration** panel.
- 6 To save the changes and close the **Options** window, click **OK**.

### Required Settings

You must also set the following Integrity Client preferences to make the integration work smoothly:

- Under the **Integrity Client** node, click the **Connection** folder and in the right pane, clear the **Prompt for User Name** and **Prompt for Password** options for connection preferences. This eliminates the need for users to enter this information when they perform an action that interacts with Integrity.
- Under the **Configuration Management** node, click the **Commands** folder and under the **Add Members** command, select the **Create Subprojects** option. This enables subprojects to be created for sharing project components.

---

**NOTE** If you select the **Create Subprojects** option after adding non-Web projects, and then add new members to those projects, see “Troubleshooting” on page 101 to resolve the issue. If you do not resolve the issue, you will encounter Sandbox detection problems when performing operations in the integration.

---

- Under the **Configuration Management** node, click the **Commands** folder and under the **Check In** command, select the **Check In if Unchanged** option. In Visual Studio .NET 2003, there are several cases where multiple files are used to represent one thing. When a change is made, it is typically made to one of the set of files, but Visual Studio .NET 2003 checks out all associated files and attempts to check in all associated files. Selecting this option prevents users from getting potentially confusing messages.

For more information on setting preferences, see the *Integrity Server Installation and Configuration Guide*.

## Parallel Development Considerations

Parallel development represents a critical aspect of configuring a software project. The primary challenge is to allow the work of separate development teams on the same files and reliably merge those changes.

The Microsoft Visual Studio .NET integration provides two possible methods for supporting parallel development (optimistic locking and development paths). The best method to use depends on your development environment. For more information see:

- “Optimistic Locking” on page 91
- “Development Paths” on page 92

---

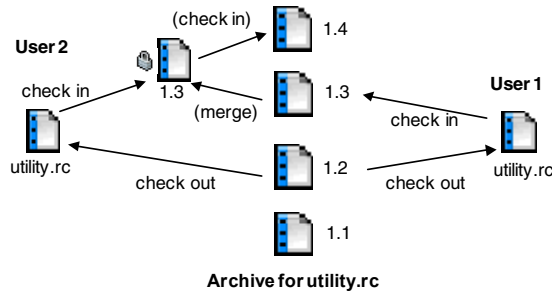
**NOTE** The integration between Visual Studio .NET 2003 and Integrity does not support non-exclusive locking.

---

## Optimistic Locking

Optimistic locking allows multiple users to check out and edit any file, even if another user has checked it out. When you check in a file that has been checked out with optimistic locking, the following steps take place:

- The latest revision of the file is silently checked out with an exclusive lock.
- If the latest revision has changed since the file was checked out, the file being checked in is merged with the latest revision. This can happen automatically or manually, depending on how you set up your policies (for more information, see “Setting Up Optimistic Locking” on page 92). Any merge conflicts are reported so the user can resolve them.
- The file is checked in as a new revision.



## Setting Up Optimistic Locking

The following Integrity policies need to be reviewed when setting up optimistic locking. Integrity policies are set up through the Integrity Administration Client. For more information, see the *Integrity Server Installation and Configuration Guide*. These policies display under the **Other** tab of the policy editor.

- `SCC.optimisticLocking` (default is `false`)

If this policy is set to `false` (the default), optimistic locking is disabled. Setting this policy to `true` enables optimistic locking.

- `SCC.optimisticLocking.autoMerge` (default is `true`)

If this policy is set to `true` (the default), when files are checked in they are automatically merged with the current member revision if there are no conflicts. If this policy is set to `false`, users are given the option of automatically or manually merging on checkin. Users should be given the option to merge manually if they are working with structured files (for example, HTML, XML).

- `SCC.optimisticLocking.autoCheckinAfterMerge` (default is `confirm`)

If this policy is set to `confirm` (the default), when a user checks in a file that requires merging, the user must confirm that their files should be automatically checked in after merging. If this policy is set to `yes`, files are automatically checked in after merging. If this policy is set to `no`, users have the option of testing their merged files before checking them in.

- `SCC.optimisticLocking.autoCheckInIfBinary` (default is `confirm`)

Binary files cannot be merged. If the latest revision has changed since the file was checked out, the file being checked in overwrites the latest revision. If this policy is set to `confirm`, when a user checks in a binary file that is different from the latest revision, they are asked to confirm the check in. If this policy is set to `yes`, binary files are automatically checked in and overwrite the latest revision. If this policy is set to `no`, users have the option of not overwriting the latest revision.

You must also ensure that source control dialogs are disabled in Visual Studio .NET 2003 (they are disabled by default). To check the setting, in Visual Studio .NET 2003 click **Tools > Options > Source Control**, and on the General tab make sure the **Display silent check out command in menus**, and the **Display silent check in command in menus** are both selected.

## Development Paths

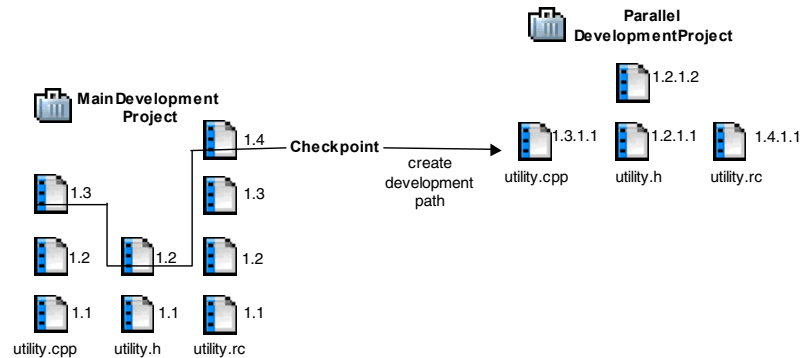
Development paths are used to deliberately create a parallel branch of development for the purpose of experimenting with research or performing post-release maintenance. Integrity allows multiple developers to point to the same development path, each using their own variant Sandbox.

---

If you use branching to support parallel development, experienced users can also use branching instead of development paths for experimental work. This eliminates the administrative work associated with development paths.

If you use optimistic locking to support parallel development, all experimental work must be done on development paths.

To make sure you get the correct version of all project files in your development path, you need to establish a record – or baseline – of the project at the time you create the development path. Checkpointing a project in Integrity saves a copy of the project as a baseline revision in the project's history.



### To create development paths

- 1 In Integrity, checkpoint the project at the solution root level.
- 2 In Integrity, create a development path from the checkpointed revision.

---

**IMPORTANT** Make sure all projects have the same development path.

---

- 3 In Visual Studio .NET 2003, create a variant Sandbox from the project (**File > Open From Source Control**). For more information, see the *Integrity User Guide*.

## Building Projects

At key development stages, you need to build a static version of an entire project based on a particular project revision. To make sure your build includes the correct version of all the files in your solution, you should do the following:

- 1 In Integrity, checkpoint the project at the solution root level.
- 2 In Integrity, create a development path from the checkpointed revision.
- 3 In Visual Studio .NET 2003, create a build Sandbox from the project (**File > Open From Source Control**).

---

**NOTE** Build projects are not explicitly supported in the integration with Integrity. For each Sandbox you create, you must manually select the project revision required for the build project.

---

---

## Command Functionality

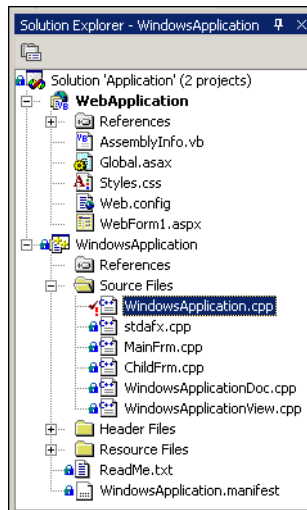
This section describes how to access Integrity functionality through Microsoft Visual Studio .NET 2003.

You can access basic version control functionality, such as checking in a file, from within Visual Studio .NET, by selecting one or more files in the **Solution Explorer** panel, then selecting **File > Source Control** (or right clicking), then selecting one of the following commands:

Command	Function
Add Solution to Source Control	Equivalent to the Integrity <b>New Project</b> command. Creates an Integrity project from the selected Visual Studio .NET solution (this includes all related Visual Studio .NET projects in the solution). The <b>Create Project Wizard</b> displays. For more information, see “Adding a Solution to Integrity” on page 96.
Add Selected Projects to Source Control	Creates a top-level Integrity project outside of the solution root structure for the selected Visual Studio .NET Web project. <b>Important:</b> This command is not recommended if you have an existing solution root. Instead of this command, you can also use the <b>Check In</b> command to add new projects to source control. If the SLN file is not added to source control, this command also launches the <b>Create Project Wizard</b> . If the SLN file is already under source control, you are prompted to add the selected projects to the same location. If change packages are enabled, the <b>Create Subproject</b> dialog box displays.
Open From Source Control	Equivalent to the Integrity <b>New Sandbox</b> command. Selection must be subproject of the solution root that contains the SLN file. Creates a Sandbox from a solution on the server. If you are creating a build Sandbox, see “Joining Development of a Solution” on page 98.
Change Source Control	Allows you to connect/disconnect from the Integrity Server or bind a solution that was added to Integrity outside of the integration. You are prompted to check out the SLN file. To update the bindings, click <b>OK</b> . For more information, see “Joining Development of a Solution” on page 98. <b>Note:</b> The <b>Browse</b> function is not applicable to Integrity.
Get Latest Version/Get	Equivalent to the Integrity <b>Resynchronize</b> command. Gets the latest version of the selected file and puts it in your working directory. The <b>Overwrite</b> dialog box may display. <b>Note:</b> If you resync a member that was checked out with optimistic locking, the working file becomes read-only.
Check Out	Equivalent to the Integrity <b>Check Out</b> command. Checks out the selected file(s). The <b>Check Out</b> dialog box displays. For more information, see “Checking Out Members” on page 100.
Check In	Performs the following possible operations: <ul style="list-style-type: none"><li>■ Check in a member (displays <b>Check In</b> dialog box)</li><li>■ Add a project to source control (creates subproject and subSandbox)</li><li>■ Add members to source control (displays <b>Create Archive</b> dialog box)</li></ul> For more information, see “Checking In Members” on page 100.
Undo Checkout	Equivalent to the Integrity <b>Revert</b> command. Replaces the working file with the revision that was checked out, as it appeared prior to modification, and unlocks the file. The <b>Overwrite</b> dialog box may display.





Command	Function
History	Equivalent to the Integrity <b>View Member History</b> command. Displays the revision history of the selected file. The <b>Member History</b> view displays.
Share	Equivalent to the Integrity <b>Add Member From Archive</b> command. The <b>Add Member From Archive</b> wizard displays. For more information, see “To share project members” on page 89. <b>Note:</b> If the member revision is updated in the source project, it is not updated in the project it is shared to. The member revision in the shared project must be updated manually.
Exclude ... from Source Control	Disables version control operations for the selected file(s).
Compare Versions	Equivalent to the Integrity <b>Differences</b> command. Compares the selected working file with the member revision. Visual Difference automatically launches and displays the two files.
Integrity Properties	Equivalent to the Integrity <b>Member Information</b> command. Displays member information for the selected file. The <b>Member Information</b> dialog box displays.
Add Project From Source Control	Equivalent to the Integrity <b>New Sandbox</b> command. Creates a Sandbox from the selected Visual Studio .NET project on the server, and adds that project to the current solution. Use shared subprojects as described in “Setting Up Projects to Reuse Code” on page 88. <b>Important:</b> This command is not recommended if you have an existing solution root.
MKS SCC Integration	Equivalent to launching the Integrity graphical user interface or the Integrity <b>Open Sandbox</b> command. Displays a <b>Sandbox</b> view. <b>Note:</b> If there is no selection, the default view displays.
Refresh Status	Equivalent to the Integrity <b>Scan for Changes</b> command. Updates the status and archive information of the selected file(s).

The Visual Studio .NET **Solution Explorer** panel is also enhanced with version control features and indicators. For example, a lock icon is displayed beside a file if it is under version control, and a red check mark with an exclamation mark displays beside a file if it is checked out by you with a lock.



---

## Solution Explorer Icons

Icon	Definition
	Displays when the file is under version control but is not currently being accessed.
	Displays when checking out the file using optimistic locking. The file is writable.
	Displays when you check out the file with a lock, or the file is locked by you.
	Displays when someone else checks out the file with a lock, or file is locked by that person.

## Adding a Solution to Integrity

By adding a solution to Integrity, you automatically add all of the projects and files contained in that solution.

When adding a solution, subprojects are created for Web projects. However, by default, subprojects are not created for non-Web projects. To create subprojects for non-Web projects, before performing this procedure, select the **Create Subprojects** option in **Preferences Configurations** dialog box for the **Add Members** command.

---

**NOTE** If you select the **Create Subprojects** option in **Preferences Configurations** dialog box for the **Add Members** command after adding non-Web projects, and then add new members to those projects, see “Troubleshooting” on page 101 to resolve the issue. If you do not resolve the issue, you will encounter Sandbox detection problems when performing operations in the integration.

---

When adding a solution, all files in the solution are added as members. For information on the recommended project structure, see “Best Practices for Solution Root Project Structure” on page 88.

A Sandbox is automatically created for the solution, containing corresponding subSandboxes for the subprojects.

---

**IMPORTANT** If you added a solution to Integrity from outside of Visual Studio .NET instead of as described in this procedure, you must bind the files by selecting **Source Control > Change Source Control**. Then from the **Change Source Control** dialog box, select the files in the list, and then click **Bind**.

---

### To add a solution to Integrity using Visual Studio .NET

- 1 In Visual Studio .NET 2003, select the solution you want to put under version control.
- 2 For a Visual Studio .NET 2003 solution, select **File > Source Control > Add Solution to Source Control**. The **Create Project Wizard** displays.

---

**NOTE** If the administrator needs to modify the **ChangePackageEnabled** policy on the Integrity Server, all active change packages should be in a closed state before making the modification. The administrator should advise users accordingly.

If any active change packages remain open during a policy modification, the **si closecp** command can be used to close the affected change packages.

---



---

3 For the project you originally selected, choose one of the available options:

- **Create a new project** and click **Finish**. The **Specify Project** dialog box displays. In the **File name** field, specify the name of the project you want to create on the Integrity Server.
- **Create as a subproject of an existing project** and click **Next**. In the **Project Name** field, specify the name of the existing Integrity configuration management project or click **Browse** to select it from the list of existing registered projects on the Integrity Server. In the **Subproject Name** field, specify the name of the subproject you are creating. (By default, the name is `<solution name>.root.`) To have the changes take effect, click **Finish**.
- **Create as a subproject of a new root project** and click **Next**. For the **Project Name** field, click **Create New** to create the new root project on the Integrity Server. The **Specify Project** dialog box displays. In the **File name** field, enter the name of the new root project and click **OK** to accept the changes. To have the changes take effect, click **Finish**.

For example, for three related projects (`Lunar1`, `Solar2`, and `Stellar3`) that are not currently under Integrity control, you begin by selecting the third option to **Create as a subproject of a new root project**. Select the `Lunar1` project and create it as a subproject at a new project root defined by you, for example, `system`. The next project (`Solar2`) can then be created by selecting the second option, **Create as a subproject of an existing project**. Because `Solar2` is a related project, it is created as a subproject of the existing top level Integrity configuration management project, `system`. For the remaining project, `Stellar3`, you repeat the process used for `Solar2`, and create it as a subproject of the existing `system` project.

---

**NOTE** By default, if you create a chain of directories, all subfolders in the directory are also created as subprojects. If a subfolder does not include a `.pj` file, `project.pj` is also added to the subproject string.

---


4 If your project contains files, the **Create Archive** dialog box displays. Modify the **Create Archive** options as described in the *Integrity User Guide* or online help.

---

**NOTE** If Integrity finds an existing archive, the **Existing archive detected** dialog displays. For more information on sharing archives, see the *Integrity User Guide* or online help.

---

For each Web project, subprojects are created using the Web project names.

Each file under version control corresponding to a Integrity member is marked in the Solution Explorer with the  icon.

Integrity also automatically creates a Sandbox with the project name in your .NET workspace. If an existing Sandbox is detected in the solution directory, the **Get SCC Project Path** dialog box displays, asking you if you want to use the existing Sandbox for source control operations.

---

**NOTE** If you want to use a new Sandbox, click **No** to return to the **Create Project Wizard**.

---

To use the existing Sandbox for source control operations, click **Yes**. The **Solution Root Location Wizard** displays.

- 
- 5 The solution root represents the location where future Web projects will be created. Select from the available options:
    - Use the identified project or click **Browse** to select another project.
    - Create a new project or subproject on the Integrity Server.
    - Disable solution root functionality for this solution.
  - 6 To use the selected solution root option, click **Finish**.

## Adding a New Project to Source Control

If a solution is already under version control, but you have now created additional projects, those projects can be added separately to Integrity as either top level projects or subprojects of the solution. A Sandbox is automatically created for the projects or subprojects.

### To create an Integrity configuration management project in Visual Studio .NET

- 1 In Visual Studio .NET, select the project you want to put under version control. Then select **File > Source Control > Check In**. The **Specify the Project to Create** dialog box displays.

---

**NOTE** If the SLN file is bound through **Change Source Control** and Web projects are added, the **Solution Root Location Wizard** displays. The **Solution Root Location Wizard** also displays if the SLN file is bound through **Change Source Control** and it contains pre-existing Web projects.


---

If you are adding a non-Web project, a Sandbox is created automatically. For subdirectories to be added as subprojects in the Sandbox, you must select the **Create Subprojects** option for the **Add Members** command.

If change packages are enabled, the **Create Subproject** dialog box displays. Select the appropriate change package from the list and click **OK**.

The **Create Archive** dialog box displays.

- 2 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.

Each file under version control corresponding to an Integrity member is marked in the Solution Explorer with the  icon.

## Joining Development of a Solution

You can join the development of a solution already under source control by creating a Sandbox corresponding to that solution on your local machine.

If you are creating a build Sandbox, see “Building Projects” on page 93.

---

**NOTE** If you are creating a Sandbox for a master project that was created and placed under source control before using Integrity (but not migrated to a solution root), each time the **Create Sandbox Wizard** displays for a Web project, you must select the Integrity project that corresponds to that Web project.

---

---

## To create a Sandbox in Visual Studio .NET

Select the subproject of the solution root containing the Visual Studio .NET solution (SLN file) under version control that you want to create a Sandbox from. Select **File > Source Control > Open From Source Control**. The **Create Sandbox Wizard** displays. Create a Sandbox as described in the *Integrity User Guide*.

---

**NOTE** If an existing Sandbox is detected in the solution directory, the **Get SCC Project Path** dialog box displays. To add the solution to source control, click **Yes**.

---

## Binding a New Variant Solution or Project

When first creating a new variant solution, Visual Studio .NET displays a message to indicate that the binding is different from what is recorded. This message must be addressed prior to the addition of any new project; however, this message can be generally ignored for continued operations in the existing solution and projects.

## To correct the bindings for a new variant solution or Web project

- 1 In Visual Studio .NET, choose **File > Source Control > Change Source Control**, and in the **Source Control** window, click **Unbind**. Once the solution is unbound, click **Bind** to rebind the solution. This corrects the solution binding to point it to the variant project.
- 2 If the member is not already checked out, you are then prompted to check out the solution control file. After checking out the solution control file, check the file back in to source control.

Once the binding is corrected, new Web projects are then correctly added as variants and users creating another instance of the variant will no longer have to correct the bindings for that instance.

## Adding Members to an Integrity Project

You can add new members to source control by checking them in.


---

**NOTE** If you previously added non-Web members to source control without creating subprojects for them, and are adding new members to those projects, see “Troubleshooting” on page 101 to resolve Sandbox detection errors.

---

## To add members to an Integrity project in Visual Studio .NET

- 1 In Visual Studio .NET, select one or more files to add to the Integrity configuration management project.
- 2 Select **File > Source Control > Check In**. The Visual Studio .NET **Check In** dialog box displays.
- 3 Click **Check In**. The **Create Archive** dialog box displays.
- 4 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.

Each file under version control corresponding to an Integrity member is marked in the Solution Explorer with the  icon.

---

## Checking Out Members


### To check out members in Visual Studio .NET

- 1 In Visual Studio .NET, select one or more files to check out.
- 2 Select **File > Source Control > Check Out**. The Visual Studio .NET **Check Out** dialog box displays.
- 3 Click **Check Out**. The **Check Out** dialog box displays.
- 4 Check out each member as described in the *Integrity User Guide*.

The following **Check Out** dialog box option is specific to the Visual Studio .NET integration:


**Use Optimistic Locking** specifies if to use optimistic locking when checking out the member. Clear this option to get an exclusive lock on the member.

---

**IMPORTANT** If optimistic locking is enabled, you cannot have an exclusive lock on the member. The member displays in the Solution Explorer with a  icon. For more information, see “Optimistic Locking” on page 91.

In the Integrity GUI, there is no icon marker for members that have optimistic locks.

---

The file displays in the Solution Explorer with a  icon denoting that it is checked out with a lock.

## Checking In Members

If optimistic locking is enabled, your checked in changes are merged with the latest revision, rather than contained in their own revision. Binary files cannot be merged, but instead they overwrite the member revision. For more information, see “Optimistic Locking” on page 91.

---

**NOTE** If another user has an exclusive lock on a member that you have an optimistic lock on, you must wait until that user checks in the member (thereby releasing the exclusive lock) before you can check in your changes. For more information, see “Optimistic Locking” on page 91.

---

### To check in members in Visual Studio .NET

In Visual Studio .NET, select one or more files to check in. Select **File > Source Control > Check In**. The Visual Studio .NET **Check In** dialog box displays. Click **Check In**. The **Check In** dialog box displays. Check in each member as described in the *Integrity User Guide*.

---

**IMPORTANT** If optimistic locking is enabled with auto merge, the changes are merged with the latest revision automatically. If auto merge is not enabled, then you are presented with the option to merge manually.

If you are checking in a binary file, you are prompted to overwrite the member revision with the working file contents.

For more information on optimistic locking, see “Building Projects” on page 93.

---

---

# Troubleshooting

Problem	Solution
The file is not a current or destined or pending member or subSandbox of project (after <b>Create Subprojects</b> option is selected in <b>Add Member</b> command preferences).	Occurs when trying to check out a member after adding Visual Studio .NET non-Web projects to source control as Integrity subprojects, when non-Web projects were previously added as subfolders only. This occurs because the <b>Create Subproject</b> option was selected in the Integrity Client preferences after already adding .NET projects, and then new members were added to the non-Web project. If a non-Web project is added to source control with the <b>Create Subprojects</b> option not selected in the client preferences, it is added as a sub folder in the Integrity configuration management project. If you select the option at a later date, any new members added to the non-Web project will create a subproject in the folder for that member. <b>Workaround:</b> Before continuing to use source control with any members that were added before enabling Create Subprojects, migrate them from the sub folder to the subproject. If you do not migrate the sub folder members to the non-Web subproject, you will encounter Sandbox detection problems when performing operations in the integration.
Working file revision contents failed to revert after cancelling a check in operation.	Occurs when cancelling a check in operation on an optimistically locked member after previously accepting a merge operation. By accepting the merge, you have accepted the introduction of another user's work into your Sandbox working file. <b>Workaround:</b> None
Sandbox not populated with members after creating a new Sandbox through the VS .NET integration.	Occurs when selecting the solution project file rather than the project that contains the SLN file during an <b>Open From Source Control</b> operation. <b>Workaround:</b> Select SLN file when performing <b>Open From Source Control</b> operation.
Sandbox for Web project created as top level Sandbox when should be subSandbox. Sandbox on main trunk when should be on same variant as solution.	Occurs when you added the Web project to source control using the <b>Add Selected Projects to Source Control</b> command. <b>Workaround:</b> Drop the Web projects from Integrity, and then add the Web projects from the .NET integration using the <b>Check In</b> command.
Error Message: "You already have a local copy of the solution or project you selected. Use the Open Solution command on the File menu to open the local copy."	Occurs when you are attempting to join the development of a solution already under source control by creating a Sandbox corresponding to that solution on your local machine, but that the following has occurred: <ul style="list-style-type: none"><li>■ You attempted to create the Sandbox through the integration using the <b>Open From Source Control</b> command.</li><li>■ The user who created the solution added an existing project, which is located outside of the solution tree, to that solution before placing it under source control.</li></ul> <b>Workaround:</b> Create a Sandbox for the solution using the Integrity Client instead of the integration. When creating the Sandbox, be sure to select the subproject that contains the SLN file for the solution, not the root project. Then open the Sandbox through the integration using the <b>Open From Source Control</b> command.
Error Message: "Refresh source code control status: The operation could not be completed"	Occurs when you are attempting to add a project to source control using the <b>Add Selected Projects to Source Control</b> command. <b>Workaround:</b> In the Integrity Client connection preferences, in the <b>Prompt for</b> area, clear the <b>User Name</b> and <b>Password</b> options.

The Integrity integration with Microsoft® Visual Basic® allows you to access the configuration management functionality of Integrity when working within a Visual Basic integrated development environment.

This chapter provides instructions on using the integration and includes the following topics:

- “Using the Microsoft Visual Basic Integration” on page 103
- “Creating an Integrity Project” on page 104
- “Creating a Sandbox” on page 104
- “Adding Members to an Integrity Project” on page 104
- “Checking Out Members” on page 105
- “Checking In Members” on page 105

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

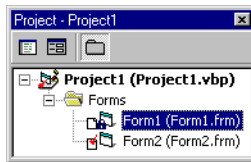
## Using the Microsoft Visual Basic Integration

This section describes how to access Integrity through Microsoft Visual Basic 6.0.

You can access version control functionality from within Visual Basic by choosing **Tools > MKS Integrity SCC Extension**, then by selecting one of the following commands:

Command	Function
<b>Get Latest Version</b>	Equivalent to the Integrity <b>Resynchronize</b> command. Gets the latest version of the selected file and puts it in your working directory. The <b>Overwrite</b> dialog box may display.
<b>Check Out</b>	Equivalent to the Integrity <b>Check Out</b> command. Checks out the selected file(s). The <b>Check Out</b> dialog box displays.
<b>Check In</b>	Equivalent to the Integrity <b>Check In</b> command. Checks in the selected file(s). The <b>Check In</b> dialog box displays.
<b>Undo Check Out</b>	Equivalent to the Integrity <b>Revert</b> command. Replaces the selected working file with the revision that was checked out, as it appeared prior to modification, and unlocks the file. The <b>Overwrite</b> dialog box may display.
<b>Show History</b>	Equivalent to the Integrity <b>View Member History</b> command. Displays the revision history of the selected file. The <b>Member History</b> view displays.
<b>Show Differences</b>	Equivalent to the Integrity <b>Differences</b> command. Compares the selected working file with the member revision. Visual Difference automatically launches and displays the two files.
<b>MKS Integrity SCC Extension Properties</b>	Equivalent to the Integrity <b>Member Information</b> command. Displays member information for the selected file. The <b>Member Information</b> dialog box displays.
<b>Add Project to MKS Integrity SCC Extension</b>	Equivalent to the Integrity <b>New Project</b> command. Brings the selected Visual Basic project under version control.
<b>Add Files to MKS Integrity SCCExtension</b>	Equivalent to the Integrity <b>Add Members</b> command. Adds the selected file(s) to the Integrity project.
<b>Create Project from MKS Integrity SCC Extension</b>	Equivalent to the Integrity <b>New Sandbox</b> command. Creates a Sandbox from the project under version control.
<b>Run MKS Integrity SCC Extension</b>	Equivalent to launching the graphical user interface or the Integrity <b>Open Sandbox</b> command. Displays a <b>Sandbox</b> view.
<b>Options</b>	Equivalent to the Integrity <b>Preferences</b> command. Configures how Integrity functions, both within Visual Basic and by itself. If you select the <b>Advanced</b> option, the <b>Preferences</b> dialog box displays. The <b>Preferences</b> dialog box displays.
<b>Refresh File Status</b>	Equivalent to the Integrity <b>Scan for Changes</b> command. Updates the status and archive information of the selected file(s).

The Visual Basic Project window is enhanced with version control features and indicators. For example, a check mark beside a file icon indicates the file is currently checked out. When a file is read-only, a small lock displays next to the file icon in the project window.



## Creating an Integrity Project

### To create an Integrity configuration management project in Visual Basic

- 1 Open the Visual Basic project you want to put under version control.
- 2 Select **Tools > MKS Integrity SCC Extension > Add Project to MKS Integrity SCC Extension**. The **Specify the Project to Create** dialog box displays.
- 3 Create a project as described in the *Integrity User Guide*. The **Create Sandbox Wizard** displays.
- 4 Create a Sandbox as described in the *Integrity User Guide*. The **Add Files to MKS Integrity SCC Extension** dialog box displays.
- 5 Select the file(s) you want to add to the Sandbox.
- 6 Click **OK**. The **Create Archive** dialog box displays.
- 7 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.

## Creating a Sandbox

### To create a Sandbox in Visual Basic

Open the Visual Basic project you have under version control. Select **Tools > MKS Integrity SCC Extension > Create a Project from MKS Integrity SCC Extension**. The **Create Sandbox Wizard** displays. Create a Sandbox as described in the *Integrity User Guide*.

## Adding Members to an Integrity Project

### To add members to an Integrity configuration management project in Visual Basic

- 1 In Visual Basic, select one or more files to add to the Integrity project.
- 2 Select **Tools > MKS Integrity SCC Extension > Add Files to MKS Integrity SCC Extension**. The **Add Files to MKS Integrity SCC Extension** dialog box displays.
- 3 Select one or more files to add to the Sandbox.
- 4 Click **OK**. The **Create Archive** dialog box displays.
- 5 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.



---

## Checking Out Members

### To check out members in Visual Basic

- 1 In Visual Basic, select one or more files to check out.
- 2 Select **Tools > MKS Integrity SCC Extension > Check Out**. A list displays, showing all files that can be checked out.
- 3 Select one or more files to check out.
- 4 Click **OK**. The **Check Out** dialog box displays.
- 5 Check out each member as described in the *Integrity User Guide*.

## Checking In Members

### To check in members in Visual Basic

- 1 In Visual Basic, select one or more files to check in.
- 2 Select **Tools > MKS Integrity SCC Extension > Check In**. A list displays, showing all files that can be checked in.
- 3 Select one or more files to check in.
- 4 Click **OK**. The **Check In** dialog box displays.
- 5 Check in each member as described in the *Integrity User Guide*.

The Integrity integration with Microsoft® Visual C++® 6.0 allows you to access the configuration management functionality of Integrity when working within a Visual C++ integrated development environment.

This chapter provides instructions on using the integration and includes the following topics:

- “Configuring the Visual C++ Integration” on page 107
- “Using the Microsoft Visual C++ Integration” on page 107
- “Creating an Integrity Project” on page 108
- “Creating a Sandbox” on page 108
- “Adding Members to an Integrity Project” on page 108
- “Checking Out Members” on page 108
- “Checking In Members” on page 109

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Configuring the Visual C++ Integration

The Integrity Client includes the **File > Integrations** menu action for enabling and disabling the integration with Microsoft Visual C++. You can select from a list of available integrations and enable or disable them, as required to work with your preferred application. For more information, see “Configuring Integrations” on page 3.

## Using the Microsoft Visual C++ Integration

This section describes how to access Integrity through Microsoft Visual C++.

Access version control functionality from within Visual C++ by choosing **Project > Source Control**, then selecting one of the following commands:

Command	Function
<b>Get Latest Version</b>	Equivalent to the Integrity <b>Resynchronize</b> command. Gets the latest version of the selected file and puts it in your working directory. The <b>Overwrite</b> dialog box may display.
<b>Check Out</b>	Equivalent to the Integrity <b>Check Out</b> command. Checks out the selected file(s). The <b>Check Out</b> dialog box displays.
<b>Check In</b>	Equivalent to the Integrity <b>Check In</b> command. Checks in the selected file(s). The <b>Check In</b> dialog box displays.
<b>Undo Check Out</b>	Equivalent to the Integrity <b>Revert</b> command. Replaces the working file with the revision that was checked out, as it appeared prior to modification, and unlocks the file. The <b>Overwrite</b> dialog box may display.
<b>Add to Source Control</b>	Equivalent to the Integrity <b>New Project</b> command or the Integrity <b>Add Members</b> command. If you choose a <b>DSP</b> or <b>DSW</b> file in Visual C++, then select <b>Add to Source Control</b> , the Visual C++ project is placed under version control. If you choose an individual Visual C++ project file, then select <b>Add to Source Control</b> , the file is added to version control as a member. The <b>Create Archive</b> dialog box displays.
<b>Remove from Source Control</b>	Equivalent to the Integrity <b>Drop Members</b> command. Drops the selected file(s) from the Integrity project. The <b>Drop Member</b> dialog box displays.
<b>Show History</b>	Equivalent to the Integrity <b>View Member History</b> command. Displays the revision history of the selected file. The <b>Member History</b> view displays.
<b>Show Differences</b>	Equivalent to the Integrity <b>Differences</b> command. Compares the selected working file with the member revision. Visual Difference automatically launches and displays the two files.
<b>MKS Integrity SCC Extension Properties</b>	Equivalent to the Integrity <b>Member Information</b> command. Displays member information for the selected file. The <b>Member Information</b> dialog box displays.
<b>Refresh Status</b>	Equivalent to the Integrity <b>Scan for Changes</b> command. Updates the status and archive information of the selected file(s).
<b>MKS Integrity SCC Extension</b>	Equivalent to launching the graphical user interface or the Integrity <b>Open Sandbox</b> command. Displays a <b>Sandbox</b> view.

---

## Creating an Integrity Project

### To create an Integrity project in Visual C++

- 1 In Visual C++, select the project you want to put under version control.
- 2 Select **Project > Source Control > Add to Source Control**. The **Specify the Project to Create** dialog box displays.
- 3 Create a project as described in the *Integrity User Guide*. The **Create Sandbox Wizard** displays.
- 4 Create a Sandbox as described in the *Integrity User Guide*. The **Add to Source Control** dialog box displays.
- 5 Select one or more files to add to the Integrity project.
- 6 Click **OK**. The **Create Archive** dialog box displays.
- 7 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.

## Creating a Sandbox

### To create a Sandbox in Visual C++

- 1 In Visual C++, select **File > Open Workspace**. The **Open Workspace** dialog box displays.
- 2 Select the Visual C++ project that you want to create a Sandbox from.
- 3 Click **Source Control**. The **Create Sandbox Wizard** displays.
- 4 Create a Sandbox as described in the *Integrity User Guide*.
- 5 To add the project members to the Sandbox, select the **Populate Sandbox** option.

## Adding Members to an Integrity Project

### To add members to an Integrity configuration management project in Visual C++

In Visual C++, select one or more files to add to the Integrity project. Select **Tools > Source Control > Add to Source Control**. The **Create Archive** dialog box displays. Add members to the project as described in the *Integrity User Guide*.

## Checking Out Members

### To check out members in Visual C++

- 1 In Visual C++, select one or more files to check out.
- 2 Select **Project > Source Control > Check Out**. A list displays, showing all files that can be checked out.
- 3 Select one or more files to check out.
- 4 Click **OK**. The **Check Out** dialog box displays.
- 5 Check out each member as described in the *Integrity User Guide*.

---

## Checking In Members

### To check in members in Visual C++

- 1 In Visual C++, select one or more files to check in.
- 2 Select **Project > Source Control > Check In**. A list displays, showing all files that can be checked in.
- 3 Select one or more files to check in.
- 4 Click **OK**. The **Check In** dialog box displays.
- 5 Check in each member as described in the *Integrity User Guide*.

The Integrity to Microsoft® Project integration combines the powerful project management features of Microsoft Project 2003, 2007, and 2010, with the flexible workflow of Integrity, allowing project managers to track and monitor project status, cost, and readiness, as well as potential associated risks.

The `Integrity2009_MSPProject2003_Integration.zip` and `Integrity2009_MSPProject_Client_Template.zip` files are available for download from the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>).

To help you set up and use the integration, the following topics are included:

- “Overview” on page 111
- “Before You Start” on page 111
- “Pre-installation Administration” on page 114
- “Installing the Microsoft Project Integration” on page 117
- “Repairing or Removing the Integration” on page 119
- “Microsoft Project Configuration” on page 119
- “Using the Microsoft Project Integration” on page 123

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Overview

The Microsoft Project integration works through an add-in component that provides access to the available functionality through the Microsoft Project **Tools** menu. The integration supports:

- **Creation of new Integrity items from Microsoft Project tasks**

Creating a new item from a task adds more detail to a task, such as a history of state changes, attachments, related items, and change package information. Using the **Synchronize All Tasks** command, you can automatically create items based on existing tasks in Microsoft Project. A hyperlink to the Integrity item also displays next to the Microsoft Project task. Once an item is created, you can track and monitor the item's state changes by synchronizing the task in Microsoft Project.

- **Creation of new Microsoft Project tasks from existing Integrity items returned in a query**

Synchronizing by query adds the individual Integrity items as task items in the Microsoft Project file. The **Synchronize Tasks By Query** command essentially populates the Microsoft Project file with items defined only in the target Integrity query. For each item found in the Integrity query, the integration creates a task in Microsoft Project.

- **Synchronization of selected or linked tasks**

Maintaining synchronization between Integrity and Microsoft Project is carried out using either the **Synchronize Linked Tasks** or the **Synchronize Selected Tasks** commands.

Using the **Synchronize Linked Tasks** command, Integrity publishes and retrieves updates for all tasks in the Microsoft Project file that have been linked between Integrity and Microsoft Project. Linked tasks are those tasks that have been previously synchronized between Integrity and Microsoft Project.

Using the **Synchronize Selected Tasks** command, selected linked tasks are updated with the information from Microsoft Project, and selected unlinked tasks are added and linked to Integrity.

- **Conflict detection for resources or field contents**

Conflict detection operates in cases where a change is entered for the field contents or resources (assigned users) associated with a task in Microsoft Project, and different information is entered for the linked item in Integrity. The integration provides the **Conflict Detected** dialog box to help you resolve the identified differences.

## Before You Start

This section provides details on basic system requirements, integration components, assumptions, and known items for the Microsoft Project integration as follows:

- "Integrity Server Requirements" on page 112
- "Client Requirements" on page 112
- "Integration Components" on page 112
- "Assumptions" on page 113
- "Key Considerations" on page 113

---

## Integrity Server Requirements

- General requirements for the Integrity Server are as described in the *Integrity Server Installation and Configuration Guide*. The Integrity Server must be configured to accept remote API connections (see “Configuring the Integrity Server” on page 114).
- Integrity database requirements are as described in the *Integrity Server Installation and Configuration Guide*.
- The Integrity Server has Integrity for Application Lifecycle Management (ALM) installed. For more information, see “Installing the Integrity ALM template” on page 116.
- The required XML mapping template is installed.

## Client Requirements

- Microsoft Project 2003/Microsoft Project Server 2003.  
Microsoft Project 2007.  
Microsoft Project 2010.
- Microsoft .NET Framework 1.1 runtime with required service packs installed. If Microsoft .NET Framework 2.0 or greater is installed but version 1.1 is not, you cannot install the integration. If .NET Framework 2.0 or greater is installed *in addition* to version 1.1, there is no effect on the integration.
- Administrators require the Integrity Administration Client for configuring ACL permissions and setting up Integrity workflows.
- The integration user must have a valid user ID in the realm and must be allowed the `ViewAdmin` permissions under the `mks:im` access control list (ACL). For more information on configuring ACL permissions, see the *Integrity Server Installation and Configuration Guide*.

---

**NOTE** To use the Microsoft Project integration, users do not require the Integrity Client.

---

## Integration Components

The Microsoft Project integration is available for download from the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>). The Microsoft Project integration includes the following components (available as ZIP files):

- `Integrity2009_MSProject2003_Integration.zip`—the Integrity to Microsoft Project Integration install (with Setup Wizard), including the `MKSProjectTemplate.mpt` Project template.
- `Integrity2009_MSProject_Client_Template.zip`—the XML mapping templates that configure the integration.

Sample mapping templates work with the Application Lifecycle Management solution and are available for download from the Integrity Support Center.



---

## Assumptions

- You know how to use Microsoft Project. For more information about using the product, refer to the appropriate documentation from the product vendor.
- You are using one of the sample XML mapping templates and have installed ALM 2009 on your system.
- If you are modifying the XML mapping template, you understand and can use XML.
- You have made any required modifications to the internal field values in the XML mapping template to reflect the ALM installed field name. For example, if you installed ALM with a prefix, that same prefix is present in the internal field values contained in the XML mapping template. For general information on configuring the XML mapping template, see “Configuring a Mapping Template” on page 188. For more detailed information, contact PTC-Integrity Support.

## Key Considerations

- For existing linked tasks, changes to the task/sub-task hierarchy cannot be made from within Integrity. Any changes to the task/sub-task hierarchy must be made in the project file from within Microsoft Project.
- The Microsoft Project integration does not support commas (,) in user names. Commas contained in user names are changed to spaces. Semi-colons (;), and open and closed brackets are supported in the integration.
- The Microsoft Project integration does not set the following fields:

**Constraint Type**  
**Request/Demand**  
**Resource Type**  
**Status**  
**Work Contour**

These fields are read-only and therefore Integrity cannot change the values directly. Other fields (such as **Start Date**, **Finish Date**, or **Status**) can indirectly drive the values for these fields.

---

**NOTE** To avoid errors, read-only fields should be set to the direction value of **in**. If a read-only field is set to a bidirectional value (**both**) or the value of **out**, the resulting error is recorded in the log file.

---

- If you are working with two Microsoft Project files that contain shared items in Integrity, running synchronizations with both projects open at the same time does not synchronize information across the project files. All projects are handled independently by the integration.
- Conflict detection may not function correctly when the Integrity Server and the database reside on separate machines, and clocks for the two machines are not in sync. In this situation, conflict detection errors can occur if an item modification timestamp pre-dates an item creation timestamp.

To ensure that conflict detection works correctly, use a clock synchronization utility to keep the two machine clocks in sync.

- When entering data, Microsoft Project includes a known limitation of 255 characters for text fields, outline code fields, and for values and characters in Enterprise Resource Multi-Value (ERMV) fields. For fields containing more than 255 characters, Microsoft Project truncates the

---

data and adds an ellipsis (...). A Microsoft Project Notes field can hold more than 255 characters; however, a Notes field only displays a maximum of 255 characters before truncation occurs.

To avoid having truncated data exported to Integrity, consider the following:

- If you routinely have data in excess of 255 characters, only synchronize from Integrity to a Notes field in Microsoft Project. Avoid performing bidirectional synchronizations.
- If you need to synchronize data bidirectionally, map the Project Notes field to a 255-character text field in Integrity. If necessary, use an Integrity event trigger for editing operations to copy only the first 255 characters from the Integrity field to the Project Notes field.

For more information on the 255 character limit in Microsoft Project, refer to Microsoft's product documentation.

- For purposes of running the Microsoft Project integration, only Integrity 2007 or greater can connect to Integrity Server 2009. Older versions of the Microsoft Project XML mapping templates (for example, `MSProjInt.xml` for Integrity 2006) are not supported with Integrity 2009.

## Pre-installation Administration

Before users can install and use the Microsoft Project integration, an administrator with access to the Integrity Server must carry out certain configuration tasks for both the Integrity Server and Integrity. Pre-installation administration tasks include configuring the Integrity Server to allow for remote API connections and installing the Integrity ALM template. Depending on the Microsoft Project fields used in your organization, the administrator may also need to modify the XML mapping template that configures the integration.

This section provides the following information on the required pre-installation administration tasks:

- "Configuring the Integrity Server" on page 114
- "Configuring Integrity" on page 115
- "Setting Permissions" on page 116
- "Modifying the Integration XML Mapping Template" on page 116

## Configuring the Integrity Server

Before users can work with the Microsoft Project integration, the administrator must configure the Integrity Server to allow remote API connections. This configuration sets the connection policy for a server integration point.

The two policies that control API access to the Integrity Server are the connection and authentication policies. Settings for these policies are specified in the following file on the Integrity Server:

```
<installdir>/config/client/IntegrityClientSite.rc
```

where `<installdir>` is the path to the directory where you installed the Integrity Server.

---

The required settings for these policies depend on your integration point; however, defaults are set for the server integration points. For the Microsoft Project integration, changes are required to the connection policy only. No changes are required for the authentication policy settings.

### **Connection Policy**

Within the `IntegrityClientSite.rc` file, the default connection policy is specified by the following property:

```
daemon.connectionPolicy=mks.ic.common.policy.  
ICAllowSpecificConnectionPolicy
```

The default setting allows only a specific set of IP addresses to connect. The required setting allows remote API connections.

### **To configure the API connection policy for the Integrity Server**

- 1 Open the following file in a text editor:

```
<installdir>/config/client/IntegrityClientSite.rc
```

where `<installdir>` is the path to the directory where you installed the Integrity Server.

- 2 Comment out the default policy (that is, insert the `#` symbol), and then uncomment the following property:

```
daemon.connectionPolicy=mks.ic.common.policy.  
ICAllowAllConnectionPolicy
```

---

**TIP** When you uncomment a property (by removing the `#` symbol), you enable the required property.

---

This allows all clients to request an API connection; however, all connecting clients still require the proper authentication.

If there is a requirement to specify individual users who are allowed to connect, use a comma-delimited list of the user IDs to define the property for `daemon.validUsersList`. Using a comma-delimited list does not change the connection policy setting.

- 3 Save and close the file.
- 4 To have the changes take effect, restart the Integrity Server.

---

**NOTE** For more information on configuring the Integrity Server to allow remote API connections, see the *Integrity Integrations Builder API Guide*.

---

## **Configuring Integrity**

This version of the Microsoft Project integration provides improved flexibility for configuring the integration. You can set up the basic configuration using the Integrity ALM template. The sample XML mapping templates work with the types that are defined for ALM.

As an alternative, the basic configuration can be extended and modified to suit the workflows and types used for projects within your organization. To create a custom configuration for the Microsoft Project integration, contact PTC-Integrity Support.

The next step in setting up the integration is to configure Integrity by installing the Integrity ALM template.

---

## Installing the Integrity ALM template

You install the Integrity ALM template using the Integrity Administration Client. Before beginning the installation, refer to the Integrity Administration Client online help for detailed instructions.

---

### IMPORTANT

Ensure that the internal field values in the XML mapping template reflect the Integrity field names as installed for your workflow or ALM installation. For example, if you installed ALM with a prefix, ensure that the same prefix is present in the internal field values contained in the XML mapping template. For information on configuring the XML mapping template, see “Configuring a Mapping Template” on page 188. For more detailed information, contact PTC-Integrity Support.

For more information on working with Integrity ALM, see the documentation included with the ALM solution download, available from the Integrity Support Center:

<http://www.ptc.com/support/integrity.htm>

---

To install the ALM template for use with the Microsoft Project integration, note the following important information:

- Before installing the ALM template, it is important that you back up your Integrity Server database. Follow the database vendor’s recommended procedures for performing the backup.
- To begin the installation, right click the **Workflows and Documents** node in the tree pane, and select **Install Solution** from the shortcut menu. The **Solution Installer** wizard takes you through the rest of the installation process.
- During the installation, make sure to map the listed **Group** and **User** roles to existing groups and users in your organization.

## Setting Permissions

The integration user must have a valid user ID in the realm and must be allowed the `viewAdmin` permissions under the `mks:im` access control list (ACL). For more information on configuring ACL permissions, see the *Integrity Server Installation and Configuration Guide*.

## Modifying the Integration XML Mapping Template

The Microsoft Project integration is based on a configurable XML mapping template. The following sample XML mapping templates are pre-configured to allow you to use all the supported features of the Microsoft Project integration:

```
MS_Project_ALM_Test_Objective.xml.sample
MS_Project_ALM_Work_Items.xml.sample
```

The mapping templates work with Integrity for Application Lifecycle Management and are available for download from the Integrity Support Center.

By default, the XML mapping template is structured to work with the types installed with Integrity ALM. You can also create additional XML mapping template files to support multiple workflows for different Microsoft Project files. Each mapping template you create must have a unique ID.

---

If you need to modify the existing mapping template or create additional templates, contact PTC-Integrity Support. For general information on modifying the XML template, see “Configuring a Mapping Template” on page 188.

---

**IMPORTANT** If you need to work with additional features, such as date fields, contact PTC-Integrity Support for assistance.

---

### To modify the Project integration XML mapping template

- 1 From the **Gateway-Templates** ZIP on the Integrity Support Center, extract the sample template files to the following directory on the Integrity Server:

`<installdir>\data\gateway\mappings\`

where `<installdir>` is the path to the directory where you installed the Integrity Server.

- 2 Rename the files to remove the `.sample` extension, and then modify the XML mapping fields as required to correspond to your existing workflow or ALM installation. Ensure that the internal field values used in the XML mapping template correspond to the Integrity field names as installed for your existing workflow or ALM installation.

## Installing the Microsoft Project Integration

You install the Microsoft Project integration using the Setup Wizard provided with the **Project-Integration** ZIP file. The integration ZIP file is available for download from the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>).

During the installation, you may be prompted to install updates for Microsoft .NET Framework 1.1, depending on the Microsoft updates installed on your system. If you are prompted to perform an update, follow the links to download the required redistributable program from the Microsoft Windows Update site at [windowsupdate.microsoft.com](http://windowsupdate.microsoft.com).

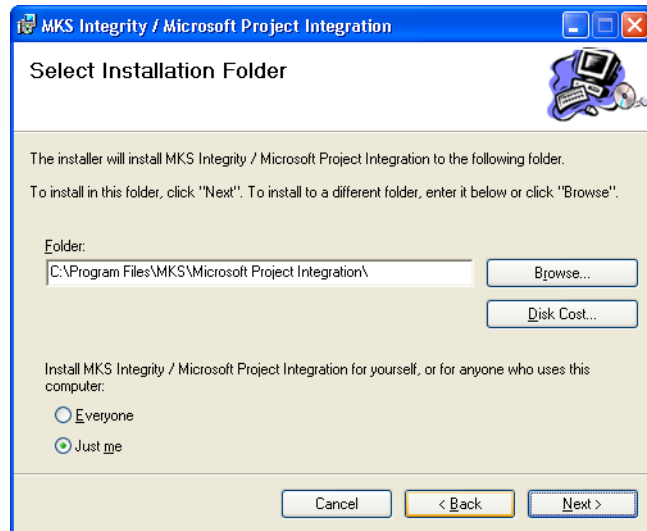
The following updates are required and must be installed in the sequence shown:

- .NET Framework 1.1 Redistributable (Developer version)
- .NET Framework 1.1 Service Pack 1

If Microsoft .NET Framework 2.0 or greater is installed but version 1.1 is not, you cannot install the integration. If .NET Framework 2.0 or greater is installed *in addition* to version 1.1, there is no effect on the integration.

### To install the Microsoft Project integration

- 1 Ensure that you close Microsoft Project before installing the integration.
- 2 From the **Project-Integration** ZIP file, extract all files to a common directory, and run the `setup.exe` file. The Setup Wizard displays.
- 3 To continue, click **Next**. The **Select Installation Folder** panel displays.



- 4 By default, the following folder is specified:

`C:/Program Files/MKS/Microsoft Project Integration`

To specify another directory for the installation, click **Browse** and choose the target directory.

To determine the amount of disk space required for the installation, click **Disk Cost**. The Disk Space dialog box summarizes information on the **Volume**, **Disk Size**, **Available**, and **Required** disk space. To return to the Setup Wizard, click **OK**.

- 5 On the same panel, you can also configure the option that allows access to the integration while using Microsoft Project. To allow access to yourself only, select **Just Me**. To allow access to anyone using your computer, select **Everyone**. By default, the option is set to **Just Me**.
- 6 To continue, click **Next**. The **Confirm Installation** panel displays.
- 7 To start the installation, click **Next**. The **Installing Panel** displays, indicating the progress for the installation. When the installation is complete, the **Installation Complete** panel displays.
- 8 To exit the Setup Wizard, click **Close**.

---

**IMPORTANT** The Microsoft Project integration uses certain files installed with the .NET Framework. Use the Microsoft Windows Update site at [windowsupdate.microsoft.com](http://windowsupdate.microsoft.com) to check for any critical updates to the .NET Framework.

---

- 9 From the common directory you unzipped to, copy the `MKSProjectTemplate.mpt` file to the Microsoft templates directory on the machine using the integration. On Windows, this directory is:

`C:/Documents and Settings/<username>/Application Data/  
Microsoft/Templates/`

You can now open Microsoft Project and configure the required Microsoft Project properties. For more information, see "Microsoft Project Configuration" on page 119.

---

## Repairing or Removing the Integration

You can use the Setup Wizard to repair or remove the Microsoft Project integration.

---

**TIP** You can also remove the Microsoft Project integration using Windows **Start > Control Panel > Add or Remove Programs**.

---

### To repair or remove the Microsoft Project integration using the Setup Wizard

- 1 Ensure that you close Microsoft Project before attempting to repair or remove the integration.
- 2 From the **Project-Integration** ZIP file, extract and run the **setup.exe** file. The Setup Wizard displays.
- 3 Select one of the available options as required:
  - To repair the integration with Integrity, select the option for **Repair Integrity to Microsoft Project Integration**.
  - To remove the integration with Integrity, select the option for **Remove Integrity to Microsoft Project Integration**.
- 4 To continue the selected operation, click **Finish**. The integration with Integrity is repaired or removed, according to the option you selected.

## Microsoft Project Configuration

To use the integration with Microsoft Project, certain custom properties and fields are created for each project file you are working on. You can also enable logging to assist with any diagnostic work. This section provides information on the following topics:

- “Integrity Custom Project Properties” on page 119
- “Integrity Custom Project Fields” on page 120
- “Enabling SSL Communication” on page 121
- “Using Microsoft Project Server and Enterprise Resources” on page 121
- “Logging Configuration” on page 122

## Integrity Custom Project Properties

To use the integration with Microsoft Project, certain custom properties are configured for each project file you are working on. The required custom properties are created automatically when you run the first synchronization operation. The required integration properties are configured under select **File > Properties**.

---

**IMPORTANT** The **MKS Integration ID** property is automatically generated after the first synchronization. It includes a unique identification key and has the form:

*project name.mpp:identification key*

Do not modify this automatically generated property.

---

The following properties are automatically configured for each Microsoft Project .mpp file when you run a synchronization with Integrity:

Name	Type	Value
<b>MKS Integrity Server Host Name</b>	Text	Name of the Integrity Server.
<b>MKS Integrity Server Port</b>	Number	Port number used by Integrity to connect to the Integrity Server. By default, the port number is 7001. Check with the administrator to confirm the correct port number.
<b>MKS Template Name</b>	Text	By default, references the integration XML mapping template installed on the Integrity Server. The mapping template is configured to use the supported features of the Microsoft Project integration. You can also modify the template to suit the workflows and types used for your projects. For more information on modifying the template, see "Modifying the Integration XML Mapping Template" on page 116. <b>Note:</b> This property can also be set to reference another template file.
<b>MKS Integration ID</b>	Number	Unique generated identifier key. <b>Important:</b> Do not modify the generated value, even if you rename the project file.

## Integrity Custom Project Fields

The integration also requires certain custom fields to work with the default template installed with Integrity. The required custom fields are pre-defined in the `MKSProjectTemplate.mpt` template file provided with the integration. To use the Integrity template, you must copy the `MKSProjectTemplate.mpt` file to the Microsoft templates directory on the machine using the integration. Once this step is completed, the required fields are created dynamically if they do not already exist.

The custom project fields include:

Integrity Custom Project Field	Description
<b>MKS Issue</b>	Identifier number of the item in Integrity. Column displayed automatically when using the <code>MKSProjectTemplate.mpt</code> file as the template.
<b>MKS User ID</b>	User name of the user in Integrity. This is the login ID rather than the full name. Column displayed automatically on the <b>Resources Sheet</b> view when using the <code>MKSProjectTemplate.mpt</code> file as the template.
<b>MKS Issue Type</b>	Type of item in Integrity. In the default configuration, this is one of <b>Project, Requirement, Test Objective, Test Plan, Work Item, Feature, Or Task</b> . Column displayed automatically when using the <code>MKSProjectTemplate.mpt</code> file as the template.
<b>MKS State</b>	Initial state for items of the defined type. Field is visible in Integrity, but no column displays in Microsoft Project. <b>Note:</b> To view Integrity state information in Microsoft Project, you must create a new column for <code>MKS State</code> . <b>Important:</b> Once you create the <code>MKS State</code> column, all tasks must have a valid initial state for the associated Integrity items of that type. If no valid initial state exists for the type, an error occurs during all subsequent synchronizations.



Integrity Custom Project Field	Description
<b>MKS Effort</b>	Pick list field with options of <b>Small</b> , <b>Medium</b> , <b>Large</b> , and <b>Very Large</b> . Information is published from Integrity to Microsoft Project only. Column not displayed by default. To display the column in Microsoft Project, select a task view and choose <b>Insert &gt; Column</b> . In the <b>Column Definition</b> dialog box, select <b>MKS Effort</b> from the <b>Field Name</b> list.
<b>MKS Risk</b>	Pick list field with options of <b>Low</b> , <b>Medium</b> , and <b>High</b> . Information is published from Integrity to Microsoft Project only. Column not displayed by default. To display the column in Microsoft Project, select a task view and choose <b>Insert &gt; Column</b> . In the <b>Column Definition</b> dialog box, select <b>MKS Risk</b> from the <b>Field Name</b> list.
<b>MKS Requirement ID</b>	Is the content of the <b>External ID</b> field in Integrity. Information is published from Integrity to Microsoft Project only. Column not displayed by default. To display the column in Microsoft Project, select a task view and choose <b>Insert &gt; Column</b> . In the <b>Column Definition</b> dialog box, select <b>MKS Requirement ID</b> from the <b>Field Name</b> list.

## Enabling SSL Communication

When using the Integrity integration with Microsoft Project, you can choose a Secure Sockets Layer (SSL) connection. To configure SSL for the Microsoft Project integration:

- 1 Open the Microsoft Project (.mpp) file and select **File > Info > Project Information > Advanced Properties**.
- 2 Click the **Custom** tab and in the **Name** field, add a property for **MKS SSL Connection**.
- 3 In the **Type** field, select **Yes or no**. In the **Value** field, choose the **Yes** option. To accept the changes click **Add** and then **OK**.
- 4 Save and re-open the .mpp file. The SSL connection to Microsoft Project is now enabled. For Microsoft Project, SSL communications are enabled on a per-session basis, therefore, if you close Microsoft Project, you must reset the **MKS SSL Connection** property to use secure communications when you start a new session. To deactivate SSL during the same Project session, you can set **MKS SSL Connection** to **No**.

---

**NOTE** To use an SSL connection with the Microsoft Project integration, you must also enable SSL communications for the server-side API. The API authenticates its session via a separate HTTP connection to the Integrity Server. To allow this authentication to use the SSL connection, the Certificate Authority associated with the SSL certificate must be registered with the JRE used by the Integrity Server.

---

## Using Microsoft Project Server and Enterprise Resources

Microsoft Project Server stores a pool of Enterprise Resources. To make use of Enterprise Resources within a project, they must first be imported into the project file. In Microsoft Project Server, this is done either through the menu action for **Insert > New Resource From > Project Server** or through the menu action for **Tools > Build Team from Enterprise**. Once imported into the Microsoft Project file, the Enterprise Resources are treated as normal resources for purposes of editing the project.

The mechanism for linking Integrity users with resources in Microsoft Project is based on a required custom resource field (**MKS User ID**). The **MKS User ID** field provides the link between the two systems and allows the Microsoft Project resource to own the resource field for purposes

---

of representation, reporting, and control. The Integrity user identifier (as based on the user's Integrity logon) is expected to exactly match the value in the **MKS User ID** custom resource field. This custom field exists in the **MKSProjectTemplate.mpt**, but can be added to any project file as a custom text field for resources.

To make use of the Microsoft Project Server Enterprise Resources, you must edit and record the Integrity user ID information with those resources. When this information is imported into the Microsoft Project file, it is represented by the **MKS User ID** field. For more information on working with Enterprise Resources in the Integrity integration, contact PTC-Integrity Support.

---

**NOTE** The Integrity **Synchronize Resources** operation pre-loads the Microsoft Project file with resources from the Integrity user base and also populates the custom **MKS User ID** field. To make use of Enterprise Resources in a synchronization operation, resources in the Microsoft Project file must have user information populated in the **MKS User ID** field. Whether the information is populated automatically or manually, is a matter of preference.

---

### To use Microsoft Project Enterprise Resources in Integrity

- 1 From the Microsoft Project menu, select **Tools > Enterprise Options > Open Enterprise Resource Pool** and check out the resources you want associated with Integrity users.
- 2 From the menu, select **Insert > Column** and in the resource sheet, insert a custom text field. Ensure that this is the same custom text field that is referenced as the **MKS User ID** field. In the **MKSProjectTemplate.mpt** file, this custom text field is **Text20**.
- 3 In the created custom text field, enter the Integrity user ID for each Enterprise user. Save these resources back to the Microsoft Project Server resource pool.
- 4 Insert the Enterprise Resources into the Microsoft Project file.
- 5 If required, rename the created custom text field to **MKS User ID**. This step is not required if you are using the **MKSProjectTemplate.mpt** and you used **Text20** as the custom field in step 2.
- 6 You can now synchronize tasks between Microsoft Project and Integrity.

## Logging Configuration

To enable logging for the Microsoft Project integration, you must add the **MKS Enable Logging** custom property for each project file you are working on. By default, the **MKS Enable Logging** property is set to **No** and logging is not activated.

The **MKS Enable Logging** custom property is configured in the **.mpp** file by selecting **File > Properties**. Once enabled, log and error messages are sent to `<temp dir>/projint.log` on the machine running the integration.

### To configure logging for the Microsoft Project integration

- 1 Open the Microsoft Project (**.mpp**) file and select **File > Properties**.
- 2 Click the **Custom** tab and in the **Name** field, add the following property:  
**MKS Enable Logging**
- 3 In the **Type** field, select **Yes** or **no**.
- 4 In the **Value** field, choose the **Yes** option.
- 5 To accept the changes click, **OK**.

- 
- 6 Save and re-open the `.mpp` file. Log and error messages are sent to `<temp_dir>/projint.log` file and can be checked to assist with any diagnostic work.

---

**TIP** To deactivate logging, set **MKS Enable Logging** to **No**.

---

## Using the Microsoft Project Integration

This section provides instructions on using the integration to synchronize information between the `.mpp` project file and items in the Integrity database. Once a synchronization operation is completed successfully, tasks are then linked between Integrity and Microsoft Project.

Before performing any other synchronizations, it is important to first synchronize resources (users) for the project. If you add resources before synchronizing, those resources are included as entries in addition to the resources provided through the synchronization. You can delete the unwanted resources by selecting the associated resource number, right clicking, and choosing **Delete Resource** from the shortcut menu.

Linked tasks are those tasks that have been previously synchronized between Integrity and Microsoft Project. Unlinked tasks are tasks or items that exist either in Microsoft Project or Integrity and have not yet been synchronized through the integration. When a task is linked, the **MKS Issue ID** displays in the **MKS Issue** field of the task. When a synchronization is complete, the Microsoft Project status bar displays **Ready**.

---

**IMPORTANT** Ensure that the date format used in your Microsoft project file matches the default Microsoft Project date format—`month/day/year`. If you need to use another date format, contact PTC-Integrity Support for assistance.

---

Prior to a synchronization, you can link tasks manually by entering the **MKS Issue ID** number in the **MKS Issue** field. For any new tasks that are not linked in this way, Integrity creates a new Integrity item when the next synchronization runs.

Integration functionality is accessible through the Microsoft Project **Tools** menu, allowing you to synchronize linked tasks, selected tasks, tasks in a single Integrity query, all tasks, and resources. The integration also provides conflict detection to help you resolve conflicts in the case of field content and assigned resources (users).

---

**IMPORTANT** For task related synchronization, you must be in a task oriented view. Synchronizing resources is independent of the view.

---

### **Handling of Empty Date Fields**

If a synchronization includes an Integrity item with an empty date field, that date field information is removed from the item when it is populated to Microsoft Project. If a date field contains a value and you perform a synchronization, that field information cannot be removed or unset in future synchronizations.

The following scenarios show how the integration handles empty and populated date fields:

- Where there are no tasks in the Microsoft Project `.mpp` file, and you perform a synchronization with one or more Integrity items containing an empty date field, the date information in the created Microsoft Project task will depend on the date information of the original Integrity item. Wherever the Integrity item included a valid date, the created Microsoft Project task will

---

also have a date. Wherever the Integrity item contained an empty date field, the created Microsoft Project task will have no date.

- Where Microsoft Project task contains a valid date and the corresponding Integrity item has an empty date field, a synchronization will populate the Integrity item with the date information from the Microsoft Project task.
- Where a Microsoft Project task was previously populated with a date from an Integrity item and that date is removed in the project file but remains in the Integrity item, a synchronization will cause the Microsoft Project task to be repopulated with the date information from the Integrity item.

### **Key Tasks**

The primary steps for using the integration are:

- “Specifying Task Types” on page 124. This task includes specifying tasks types using the **MKS Issue Type** custom project field.
- “Setting Task Relationships” on page 125. This task includes setting task relationships using the outline level in Microsoft Project.
- “Synchronizing Resources” on page 126. This task includes synchronizing project resources (or users) using the **Synchronize Resources** command.
- “Synchronizing All Tasks” on page 126. This task includes publishing task items from an existing Microsoft Project to Integrity using the **Synchronize All Tasks** command.
- “Synchronizing Tasks By Query” on page 126. This task includes publishing items from an existing Integrity query to Microsoft Project using the **Synchronize Tasks By Query** command. Each item from the query is linked to a task in Microsoft Project to create a new project.
- “Synchronizing Linked Tasks” on page 127 and “Synchronizing Selected Tasks” on page 127. These tasks include maintaining the status of items in Microsoft Project and Integrity using the **Synchronize Linked Tasks** and **Synchronize Selected Tasks** commands as required.
- “Detecting Conflicts” on page 128. This task includes resolving any conflicts detected during synchronizations.

## **Specifying Task Types**

When creating a task in Microsoft project, you specify the type of Integrity item (for example **Project, Requirement, Test Objective, Test Plan, Work Item, Feature, or Task,**) using the **MKS Issue Type** custom project field. Once the task is assigned to a type, the next synchronization creates that task as an item of the specified type in Integrity. The mapped fields are then dependent on this type.

The **MKS Issue Type** custom project field is displayed automatically in Microsoft Project when using the **MKSProjectTemplate.mpt** file as the template. For more information on custom project fields, see “Integrity Custom Project Fields” on page 120.

---

**NOTE** Using the default template, the **MKS Issue Type** is a required field. If you are not using the pre-configured **MKSProjectTemplate.mpt** file, you must create this field manually.

---

---

## To specify the type of task using Microsoft Project

In the Microsoft Project file, create the required line items for the tasks. In the **MKS Issue Type** column, enter the type that will be used for the linked item in Integrity (for example, **Project**, **Requirement**, **Test Objective**, **Test Plan**, **Work Item**, **Feature**, **Task**). Run the required synchronization—**Synchronize All Tasks** or **Synchronize Selected Tasks**—to publish the information to Integrity. For more information, see “Synchronizing All Tasks” on page 126 or “Synchronizing Selected Tasks” on page 127.

The tasks are created as items of the specified types in Integrity.

## Setting Task Relationships

When manually adding tasks in Microsoft Project, you can create relationships between tasks using the outline level. To relate tasks, you set the main task at the top outline level and indent the related subtasks as required. Each task is indented one additional level to the task it is related to. In Microsoft Project, the outline level is controlled through **Project > Outline > Indent** or **Outdent**. For more information, refer to the documentation provided with Microsoft Project.

Once the parent/child relationships are set using the outline level in Microsoft Project, the next synchronization creates the specified relationships between the items in Integrity. Integrity maps task relationships downward through the hierarchy. The relationships are then shown when you view the linked item in Integrity.

---

**IMPORTANT** If the integration is configured for the default ALM solution as outlined in this guide, the relationship hierarchy cannot be changed from within Integrity once the relationship hierarchy is set in Microsoft Project and a synchronization is run to link with items in Integrity. Custom configurations can be created to allow greater flexibility.

---

For example, if you set the following hierarchy of relationships, running a synchronization that links to items in Integrity:

```
Requirement 1
  Feature 1
  Feature 2
Requirement 2
```

and then from Integrity attempt to change the hierarchy to:

```
Requirement 1
  Feature 1
Requirement 2
  Feature 2
```

The relationship hierarchy in Microsoft Project is not updated. To change the relationship hierarchy, you must make the required changes through Microsoft Project.

---

**IMPORTANT** If the integration encounters an unknown type during a synchronization, the unknown type may be removed from existing relationships in Integrity. For example, the **Test** type used in the ALM template is not recognized by the integration.

---

## To set the task relationship using Microsoft Project

In the Microsoft Project file, create the required line items for the tasks. Using **Project > Outline > Indent** or **Outdent**, set the outline level required to relate the tasks, where each task is indented one additional level to the task it is related to. Run the required synchronization—**Synchronize All Tasks** or **Synchronize Selected Tasks**—to publish the information to Integrity. For more information, see “Synchronizing All Tasks” on page 126 or “Synchronizing Selected Tasks” on page 127.

---

The corresponding relationships are created for the linked items in Integrity.

---

**NOTE** For information on valid relationships, see the documentation included with the ALM solution download, available from the Integrity Support Center at:  
<http://www.ptc.com/support/integrity.htm>

---

## Synchronizing Resources

The **Synchronize Resources** command publishes resources to both Microsoft Project and Integrity and ensures that resources (users) are created/updated in the project file, and updated in Integrity. If the Integrity user does not have a full name, the **MKS User ID** is used as the resource name. If a full name is later added in Integrity, the resource name is updated to the full name in the next synchronization.

---

**IMPORTANT** Before performing any other synchronizations, you must first synchronize resources (users) for the project. If you add resources before synchronizing, those resources are included as entries in addition to the resources provided through the synchronization. You can delete the unwanted resources by selecting the associated resource number, right clicking, and choosing **Delete Resource** from the shortcut menu.

---

### To synchronize resources

From the Microsoft Project menu, select **Tools > Synchronize Resources**. Enter the Integrity Server **Details** and **Connection Details** if prompted, and from the list, choose the XML mapping template you want to use. A connection to Integrity is opened and all resources are synchronized. The Microsoft Project status bar displays **Ready** to indicate the synchronization is complete.

---

**TIP** You can also link resources manually by entering the user's login ID in the **MKS User ID** field.

---

## Synchronizing All Tasks

The **Synchronize All Tasks** command essentially allows you to populate Integrity with task items defined only in the Microsoft Project file. For each task item in Microsoft Project, the integration creates an item in Integrity, or updates the item if the task is already linked.

### To synchronize all tasks

From the Microsoft Project menu, select **Tools > Synchronize All Tasks**. Enter the **Integrity Server Details** and **Connection Details** if prompted and, from the list, choose the XML mapping template you want to use. A connection to Integrity is opened and all tasks are synchronized. The Microsoft Project status bar displays **Ready** to indicate the synchronization is complete.

## Synchronizing Tasks By Query

The **Synchronize Task By Query** command allows you to synchronize tasks based on a defined Integrity query. Synchronizing by query adds the individual Integrity items as task items in the Microsoft Project file. The **Synchronize Tasks By Query** command essentially populates the Microsoft Project file with items defined only in the target Integrity query. For each item found in the Integrity query, the integration creates a task in Microsoft Project.

---

You can use this command to start a new .mpp file based on an existing query in Integrity. For example, for a query based on a specific type—**All Features**—you could create new project file that included all items of the type **Feature** found by the Integrity query.

---

**NOTE** When synchronizing Integrity items with Microsoft Project, new resources are created and assigned to the linked tasks, if the items have users (resources) that do not already exist in Microsoft Project.

---

### To synchronize a tasks by Integrity query

From the Microsoft Project menu, select **Tools > Synchronize Tasks By Query**. Enter the **Integrity Server Details** and **Connection Details** if prompted, and from the list, choose the XML mapping template you want to use. The **Select an Integrity Query** dialog box displays, retrieving the list of available Integrity queries. From the list of Integrity queries, select a query to synchronize by. To accept your selection, click **OK**.

The synchronization with Integrity is performed and the Microsoft project file is populated with tasks based on items in the Integrity query. The Microsoft Project status bar displays **Ready** to indicate the synchronization is complete.

## Synchronizing Linked Tasks

When you perform a synchronization of linked tasks, Integrity publishes and retrieves updates for all tasks in the Microsoft Project file that have been linked between Integrity and Microsoft Project. Linked tasks are those tasks that have been previously synchronized between Integrity and Microsoft Project. Unlinked tasks are tasks or items that exist either in Microsoft Project or Integrity and have not yet been synchronized through the integration. When a task is linked, the **MKS Issue ID** displays in the **MKS Issue** field of the task.

When synchronizing linked tasks, you do not need to select every task in a set of relationships. If a relationship is mapped (as to predecessor, successor, outline parent, or outline children), Integrity automatically selects the related tasks, and new, related tasks are created in Integrity as required.

---

**NOTE** Prior to a synchronization, you can link tasks manually by entering the **MKS Issue ID** number in the **MKS Issue** field. For any new tasks that are not linked in this way, Integrity creates a new Integrity item when the next synchronization runs.

---

### To synchronize a linked task

From the Microsoft Project menu, select **Tools > Synchronize Linked Tasks**. Enter the **Integrity Server Details** and **Connection Details** if prompted, and from the list, choose the XML mapping template you want to use. A connection to Integrity is opened and all linked tasks are synchronized. The Microsoft Project status bar displays **Ready** to indicate the synchronization is complete.

## Synchronizing Selected Tasks

The Microsoft Project integration includes functionality that allows you to synchronize selected project tasks, whether those tasks are linked or unlinked. When synchronizing selected tasks, you highlight the project tasks you want to synchronize, run the **Synchronize Selected Tasks** operation and the selected tasks are published to Integrity.

Using the **Synchronize Selected Tasks** command, selected linked tasks are updated with the information from Microsoft Project, and selected unlinked tasks are added and linked to Integrity.

---

When synchronizing selected tasks, you do not need to select every task in a set of relationships. If a relationship is mapped (predecessor, successor, outline parent, or outline children) Integrity automatically selects the related tasks, and new, related tasks are created in Integrity as required.

### **To synchronize selected tasks**

From the Microsoft Project menu, select **Tools > Synchronize Selected Tasks**. Enter the **Integrity Server Details** and **Connection Details** if prompted, and from the list, choose the XML mapping template you want to use. A connection to Integrity is opened and all selected tasks are synchronized. The Microsoft Project status bar displays **Ready** to indicate the synchronization is complete.

## **Detecting Conflicts**

Conflict detection operates in cases where a change is entered for the field contents or resources (assigned users) associated with a task in Microsoft Project, and different information is entered for the linked item in Integrity. The integration provides the **Conflict Detected** dialog box to help you resolve the identified differences.

### **To resolve an identified conflict**

Review the content of the **Conflict Detected** dialog box where the information is displayed in the **Microsoft Project** and **Integrity** text fields. In the **Resolution** field, enter the correct information that will be used to populate both Microsoft Project and Integrity. To accept the change, click **OK**. The resolution information is published to both Integrity and Microsoft Project. For example, the following dialog box shows the user `mchang` being set as the resource for the item or task.





This chapter provides information on the Integrity integration with Microsoft® Word as installed with the `MKSWordImporterSetup.msi` installer available for download from the Integrity Support Center as part of the `MKSIntegrity2009-2007_SP6_MSWord2003_Integration.zip` file.

This chapter is intended only for users who are using the Word integration that is installed with the `MSI` installer, and who want to export requirements from Word documents into Integrity. The instructions provided here do not apply to the new Integrity Client functionality for editing items and documents in Microsoft Word.

---

**NOTE** For information on the new Integrity Client functionality for editing items and documents in Word (including embedded objects and quick export of items/documents to Word format), see the *MKS Gateway User Guide*.

---

This Integrity integration with Microsoft Word provides support for rich content in Word documents. The integration is intended to work with Integrity for Requirements Management 2007 or Integrity for Application Lifecycle Management 2009.

This chapter includes the following topics:

- “Assumptions” on page 131
- “Overview of the Integration” on page 131
- “System Requirements” on page 133
- “Configuring the Integrity Server” on page 133
- “Installing and Uninstalling the Integration” on page 134
- “Customizing the Word Integration” on page 135
- “Exporting Word Documents to Integrity” on page 141
- “Troubleshooting” on page 143

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Assumptions

Before you use the Word integration, PTC assumes that you fully understand the following:

- the hardware platform and Windows operating system you are installing the Integrity Client on
- Integrity for managing workflows and documents
- Integrity for Requirements Management or Integrity for Application Lifecycle Management
- Microsoft Word

If you are customizing the Word integration, you should have a working knowledge of XML mapping files and the MCFG file used for the Word integration, as well as knowledge of XSLT.

## Overview of the Integration

The Word integration provides a one-way transfer of data from Word into Integrity; any changes made to requirements in the original Word file after the export are not made to their corresponding items in Integrity.

To convert the content of the Word document into Integrity items, a Document Structure Definition (DSD) parses the Word document. More specifically, the DSD maps the content and structure of the document to types and fields in your Integrity workflow, determining the Integrity items that are created. Your Integrity professional services representative configures the DSD.

---

**NOTE** To ensure an accurate conversion, your Integrity workflow must be consistent with the DSD. For example, if a type has a mandatory field in the workflow, it must also be mandatory in the DSD.

---

To export Word documents that include rich content (tables, embedded images, and formatted text), the data is converted to HTML, then mapped to rich content fields in Integrity items. In Integrity, rich content is expressed using a limited set of HTML elements and attributes. For more information, see “Rich Content Support” on page 131.

---

**NOTE** Rich content fields are supported only with Integrity for Requirements Management. Exporting a Word document containing rich content to the Application Lifecycle Management (Process) Solution does not preserve the rich content formatting.

---

The Word integration is subject to rules for user and group visibility, editability, and relevance for fields in the target Integrity item type you are exporting to. For more information, contact your Integrity administrator.

### Rich Content Support

The integration supports rich content in Word documents and maintains that content during the export process. More specifically, the integration supports:

- headings and heading levels
- ordered and unordered lists
- embedded tables

- 
- embedded images
  - embedded OLE objects

For the complete list of HTML elements and attributes supported in Integrity, see the *Integrity User Guide*. Unsupported HTML elements and attributes in the document are silently discarded when new items are created.

### **Headings**

Headings are mapped from the available Word heading styles (Heading1 to Heading 9) to the available HTML heading tags (h1 to h6). Headings that extend beyond Heading 6 are converted to h6.

Correctly defined numbered headings (i.e. numbered sections) remain as headings. If incorrectly defined in Word, numbered headings are converted to lists.

If you include a table of contents in your document, page numbers are appended to the end of headings instead of retaining the spacing between the heading and page number, for example, 1) Title Page.....1 displays as 1) Title Page1.

### **Fonts, Sizes, and Styles**

Specific font families are not preserved during the export; however, font styles (bold, underline, italic, and strikethrough) are preserved. In addition, font sizes are ignored in normal text.

### **Lists**

Numbered and bulleted lists are supported.

### **Tables**

Irregularly shaped cells (L-shaped) are unsupported.

### **Embedded Objects**

The integration exports an image snapshot of the embedded object and the object itself. If the embedded object is exported into a rich content field, you can edit the embedded object. The Integrity Document view allows you to edit the embedded item via an external application.

### **Embedded Images**

The Word integration supports EMF (Windows Metafile), GIF, JPG, and PNG image formats in Word documents. EMF images are converted to PNG format in the new items. Autoshapes and WordArt in Word documents are not supported.

### **Hyperlinks**

Hyperlinks in the Word document are preserved.

---

## System Requirements

For the integration to work effectively, you must have the following installed and configured correctly:

- Integrity Server, configured to allow API connections. For more information, see “Configuring the Integrity Server” on page 133.
- Microsoft Windows XP Professional/Vista Business or Enterprise.
- Integrity Client 2007, 2009, or 10.
- Microsoft Word 2003 Service Pack 2, Word 2007, or Word 2010.

---

**NOTE** Although the integration supports Word 2003 and 2007, the integration requires documents saved in Word 2007 (**docx**) format. If you are exporting a Word document that is not in Microsoft Word 2007 (**docx**) format, the integration prompts you to save a copy of the document in **docx** format. To save Word 2003 documents in **docx** format, the Microsoft Office Compatibility Pack must be installed, which you can download from:

<http://go.microsoft.com/fwlink?LinkID=77512>.

---

If not currently installed on your system, the following components must be installed before you run the integration installer:

- Microsoft .NET Framework 2.0 and 3.0

---

**NOTE** The Word integration uses the Microsoft .NET framework. PTC recommends running Windows Update to check for any critical updates to the .NET framework.

---

- Microsoft Visual Studio 2005 Tools for Office Second Edition runtime
- Microsoft Visual Studio 2005 Tools for Office Language Pack
- Windows Installer 3.1
- Microsoft Office 2003/2007 Primary Interop Assemblies

You can download these components from <http://www.microsoft.com> or run the Microsoft Office installer and select them during the install.

### **Connecting to the Integrity Server**

For each Word session, you are prompted to authenticate with the Integrity Server, even if the Integrity Client is already connected.

## Configuring the Integrity Server

Before users can work with the Microsoft Word integration, the administrator must configure the Integrity Server to allow remote API connections to the server. This enables integration users to connect to and run commands on the server. The default connection policy allows only a specific set of IP addresses to connect. The required setting allows all clients to connect.

The connection policy is specified in the following file on the Integrity Server:

```
<installdir>\config\client\IntegrityClientSite.rc
```

where `<installdir>` is the path to the directory where you installed the Integrity Server.

---

### To set the Integrity Server connection policy

- 1 In a text editor, open the `IntegrityClientSite.rc` file.
- 2 Comment out the following default policy:

```
daemon.connectionPolicy=mks.ic.common.policy.  
ICAllowSpecificConnectionPolicy
```

- 3 Uncomment the following policy:

```
daemon.connectionPolicy=mks.ic.common.policy.  
ICAllowAllConnectionPolicy
```

---

**TIP** To comment out a policy, insert a # symbol as the first letter of the property. To uncomment a policy, remove the # symbol.

---

This allows all clients to request an API connection; however, all connecting clients still require the proper authentication.

If there is a requirement to specify individual users who are allowed to connect, use a comma-delimited list of the user IDs to define the property for `daemon.validUsersList`. Using a comma-delimited list does not change the connection policy setting.

- 4 To have the changes take effect, restart the Integrity Server.

---

**NOTE** For more information on configuring the Integrity Server to allow remote API connections, see the *Integrity Integrations Builder API Guide*.

---

## Installing and Uninstalling the Integration

Before you install/uninstall the Word integration:

- Administrative privileges are required to install the integration.
- Shut down the Integrity Client and all instances of Word.
- If Word is your default editor for Microsoft Outlook, shut down Outlook.

---

**NOTE** The Word integration provided with Integrity 2006 is no longer supported. If you have this earlier version of the integration, uninstall it before installing the later integration. For information on uninstalling the earlier Word integration, refer to the original product documentation.

---

### To install the Word integration

- 1 Extract the contents of `MKSIntegrity2009-2007_SP6_MSWord2003_Integration.zip` to a temporary directory.
- 2 In the temporary directory, run `setup.exe`. The **MKS Word Document Importer Setup Wizard** displays.
- 3 To continue, click **Next**. The **Select Installation Folder** panel displays. Accept the default path location or specify another location by browsing or typing the path of the directory where you want to install the integration.
- 4 Click **Next**. The **Confirm Installation** panel displays.

- 
- 5 To start the installation, click **Next**.
  - 6 Once the integration is successfully installed, click **Close** to exit.

### To uninstall the Word integration

- 1 From the **Add or Remove Programs** control panel (Windows XP) or **Uninstall a Program** control panel (Windows Vista), select **MKS Word Document Importer**.
- 2 Click **Remove** (Windows XP), or right click and select **Uninstall** (Windows Vista).
- 3 To confirm, click **Yes**.

## Customizing the Word Integration

This section outlines the administrative tasks required for customizing the Word integration for your site. The Word integration relies on XML mapping templates and DSD files to perform document exports. The XML templates are found under:

```
<Integrity Server installdir>\data\gateway\mappings
```

The MCFG rule set files (or DSD files) are in a compressed file with the `.mcfg` extension. The files can be extracted using a ZIP utility application.

Each MCFG file includes:

- The `mapping_name.txt` file, containing the mapping template name. This file identifies the XML mapping template by matching the value of the name attribute of the mapping tag. For example, if `mapping_name.txt` contains `ABC` as the content, then only the XML template containing the following tag is picked up for processing:

```
<mapping name="ABC" template-version="2.1">
```
- The `name.txt` file, containing the title name that displays in the **Document Structure Definition** list in the **Export to MKS Integrity Wizard**.
- The `static.txt` file, containing all definitions and properties for fields displayed in the **Export to MKS Integrity Wizard**.
- The `process-items.xlst` file, that translates the document and prepares it for export to Integrity.

This section discusses the following topics:

- “Customizing the DSD” on page 136
- “Importing the DSD” on page 138
- “Importing a Local DSD” on page 139
- “Understanding the XML Mapping File” on page 140
- “Understanding the Static TXT File” on page 139
- “Understanding the Process Items XLST File” on page 141

---

## Customizing the DSD

By default, the Word integration allows for the export of requirement documents into Integrity. You can also create additional DSDs to use with different document types (or domains), for example, specification, test, or input documents. This section describes how you can create a custom DSD for other document types.

Once you have created a custom DSD for a new document type, the specified identifier will be available for selection from the list in the Word Importer dialog box.

Sample XML mapping templates for use with the Word integration are included in the `Integrity2009_Server_Templates.zip` file that is available for download from the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>).

Sample MCFG files are provided in the `Integrity2009_MSWord_Client_Templates.zip`, also available for download from the Integrity Support Center.

You extract the sample XML mapping template files, including:

```
MS_Word_Requirements_Document.xml.sample
MS_Word_Test_Suite.xml.sample
```

to the following directory on the Integrity Server:

```
<Integrity Server installdir>\data\gateway\mappings\
```

The XML mapping template must be renamed to remove the `.sample` extension, and then modified to reflect your workflow or ALM installation. The modified XML mapping template then becomes your production mapping template for the Word integration.

---

### TIP

The XML template file name should differentiate between the default template and the custom one being created.

Ensure that the internal field values in the XML mapping template reflect the Integrity field names as installed for your workflow or ALM installation. For example, if you installed ALM with a prefix, ensure that the same prefix is present in the internal field values contained in the XML mapping template.

For general information on modifying an XML mapping template, see “Configuring a Mapping Template” on page 188. For more detailed information, contact PTC-Integrity Support.

---

## To customize the DSD

---

**IMPORTANT** Before making any modifications to your MCFG/DSD files, you should create backup copies in a safe location.

---

- 1 Make note of the name of your production XML mapping template.
- 2 For each specific import that will be supported, you must create a new MCFG (DSD). By default, Integrity includes a condensed version and non-condensed version. You can use the default versions to create two versions (condensed and non-condensed) for your document type.

If you have not already imported the MCFG file, it is available in the

`<integration installdir>` on the Integrity Client or by default in the following directory:

```
C:\Program Files\MKS\MKS Word Document Import\Rulesets
```



---

If you have imported the MCFG file, it resides in:

```
C:\Documents and Settings\\.mks\integrations
\Word30\RuleSets
```

---

**NOTE** The MCFG file is a ZIP file that can be opened and repackaged using a ZIP utility. Copy the MCFG file to a temporary location and change the file extension from `.mcfg` to `.zip`. You can then extract the files using any ZIP utility.

---

- 3 Using a ZIP utility, extract the contents of the MCFG file, which includes `mapping_name.txt`, `name.txt`, `static.txt`, and
- 4 Update the `mapping_name.txt` file with the name of the custom mapping template. For example, if you are working with the `MS_Word_Requirements_Document.xml.sample` template, you would add `Microsoft Word Requirements Document` as the mapping name.
- 5 Update `name.txt` with a new identifier. The specified identifier will be available for selection from the list in the Word Importer dialog box.

To have this modified DSD to appear in the list with the default DSDs, change its name by modifying the contained `name.txt` file. The `name.txt` file contains a single line of text representing the name that will appear in the **Document Structure Definition** list in the export wizard. If this DSD is going to replace an existing DSD file, you do not need to change its name.

---

**IMPORTANT** By default, the `name.txt` file is read-only. Depending on your text editor, you may need to set the file to read-write access to save your changes.

---

- 6 In the directory where you extracted the MCFG contents, open the `static.txt` file and in the `category` line, change the last (ninth) parameter to a comma separated list of the allowable categories in the your document.

When editing the `static.txt` file, ensure that field relationship rules are respected.

---

**IMPORTANT** By default, the `static.txt` file is read-only. Depending on your text editor, you may need to set the file to read-write access to save your changes.

---

For additional information on the `static.txt` file, see “Understanding the Static TXT File” on page 139.

- 7 Save the files and re-package the MCFG file.

To re-package the MCFG file, select the required TXT files and add them to a ZIP archive in the directory where you extracted the MCFG contents. Rename the ZIP file to change the extension from `.zip` to `.mcfg`. If you want to replace the existing MCFG, give the re-packaged file the same name as the original MCFG file.

- 8 After importing the DSD, you can export the defined document type to Integrity. For more information, see “Importing the DSD” on page 138.

---

**IMPORTANT**

- When rendering rich content, broken image links are displayed if Word cannot find the following file in the DSD:

`internal_attachments_name.txt`

If `internal_attachments_name.txt` is not found, Word defaults to using `RQ_Text Attachments` as the attachment field name for the rich content attachment links. This default behavior causes the broken image links in the rich content.

To resolve this issue, you must add a file that contains the actual attachment field name in the DSD (`internal_attachments_name.txt`). Ensure that the internal field values in the DSD reflect the Integrity field names as installed for your workflow or ALM installation. For example, if you installed ALM with a prefix, ensure that the same prefix is present in the internal field values contained in the DSD.

- If you were previously using the Word integration with an RM 2007-based solution and the rich content field did not have `RQ_Text Attachments` mapped as the attachment field, you must add the `internal_attachments_name.txt` file to the RM MCFG files and ensure that the file contains the correct attachment field name. To add the `internal_attachments_name.txt` file, follow the instructions for “Customizing the Word Integration” on page 135.

The ALM 2009 MCFG files contain the `internal_attachments_name.txt` file. This file uses the `ALM_` prefix to illustrate where changes are required. You must modify the contents of the `internal_attachments_name.txt` file if your ALM 2009 solution is installed with no prefix or a prefix other than `ALM_`.

---

## Importing the DSD

Once you have customized the DSD and re-packaged the MCFG, you can install and import the modifications to the integration. Once imported, the defined DSDs are available to all Word integration users.

### DSD Storage Locations

DSD files are found in the following locations:

- `C:\Program Files\MKS\MKS Word Document Importer\RuleSets`  
(For DSDs that were shipped with the Integrity)
- `C:\Documents and Settings\\.mks\integrations\Word30\RuleSets`  
(For DSDs that were imported using **Import Definitions**)

### To import the MCFG file

---

**IMPORTANT** Before making any modifications to your MCFG/DSD files, you should create backup copies in a safe location.

---

You import the modified DSD by doing one of the following:

- If you have renamed the DSD (by modifying the `name.txt` file), you can import the MCFG file that you have created using **MKS > Import Definitions** in Word.
- If you are replacing an existing DSD file, you must do the following:
  - a) Shut down Word and the Integrity Client. In addition, if Microsoft Outlook is configured to use Word as its default editor, you must also shut down Outlook.
  - b) Browse to the DSD storage locations previously specified and remove any copies of the MCFG file that you are replacing.
  - c) From within Word, use **MKS > Import Definitions** to import the new MCFG file.

---

When importing a document using the modified DSD, the field values should now correspond to the new values that you set in the `static.txt` file.

## Importing a Local DSD

The administrator or Integrity professional services representative typically configures global DSDs for all users. The global DSDs reside in `<document importer installdir>\RuleSets`; however, as a user, you can import your own local DSDs that override the global DSDs.

### To import a local DSD

- 1 In Word, select **MKS > Import Definitions**. A standard browse dialog box displays.
- 2 Browse to the location of the DSD file (MCFG) and select it. The local DSD is added to the following directory:

```
C:\Documents and Settings\\.mks\integrations\Word30\RuleSets
```

This DSD is available from the **Document Structure Definition** list when you export your document(s).

## Understanding the Static TXT File

The Word integration populates fields that are defined in the MCFG (or DSD files), and then prompts for inputs from the user through the **Export to MKS Integrity Wizard**. The fields displayed in the export wizard are called external fields, which in turn map to internal fields that are defined within Integrity.

Each DSD file includes a `static.txt` file which contains all the definitions and properties for fields used in the **Export to MKS Integrity Wizard**. For example, the `Assigned User` field is defined as:

```
Assigned User|Assigned User|String|true| User Name ||false|false|
User Name1,User Name2
```

The `Project` field is defined as:

```
Project|Project|String|true|/Release2|Project value to be used for all
items.|false|false|
```

The preceding line contains a number of properties for the `Project` field. For example, the `Default` value property (that is, the 5th property – in the preceding example, this property has a value of `"/Release2"`).

The line is separated into segments using the `'|'` symbol. Each segment represents a property to define the field as follows:

Field Name	Description
Name	The external value for the field name.
Display Name	The field name displayed in the <b>Export to MKS Integrity Wizard</b> interface.
Type	Value (for example, <code>string</code> ) defining the type of field.
Mandatory	Boolean field. A mandatory field in Integrity. Displays in red font in the <b>Export to MKS Integrity Wizard</b> interface.
Value -- default value	The default field value.
Description	Field description (free form text).

Field Name	Description
Hidden	Boolean field. Field hidden from the user in Preview and Results panels of the <b>Export to MKS Integrity Wizard</b> .
Content	Boolean field. Displayed value in the <b>Content</b> section of <b>Preview</b> and <b>Results</b> panels of the <b>Export to MKS Integrity Wizard</b> .
Allowed Values	A comma separated list for pick values. Displayed in a drop-down list in the <b>Export to MKS Integrity Wizard</b> interface.

## Understanding the XML Mapping File

The mapping of external and internal fields is specified in the XML mapping file (for example, the `MS_Word_Requirements_Document.xml` file). The XML mapping file allows the integration to retrieve the required values from the correct locations.

Using the example of a **System Requirement**, the internal **Assigned User** field in Integrity would be defined with the following mapping:

```
<map name="System Requirement">
  <set-attribute name="rmtype" value="content" />
  <field internal="Category"
    direction="in"
    on-create-only="true">
    <default>System Requirement</default>
  </field>

  <field external="Summary"
    internal="ALMText"
    field-type="richcontent"
    direction="in"
    on-create-only="true"
    required="true" />

  <field external="Assigned User"
    internal="Assigned User"
    direction="in"
    on-create-only="true" />
</map>
```

With this configuration, when the export wizard performs the export, the integration will map the value of the external field **Assigned User** to the internal field **Assigned User**.

---

**IMPORTANT** For fields that are mandatory in Integrity, you must define a default value in the XML mapping template. If a mandatory field is left blank, the export operation fails.

---

---

## Understanding the Process Items XLST File

The integration retrieves the value of a newly-created field through the `process-items.xlst` file, as contained in the content folder of the DSD. The `process-items.xlst` file translates the document and prepares it for export to Integrity.

For example, to configure the `process-items.xlst` file for **Assigned User**, add the following lines under the `<xsl:template name="OutputFields">` tag:

```
<mks:field>
  <xsl:attribute name="name">
    <xsl:value-of select="$assigned_user_key"/>
  </xsl:attribute>
  <xsl:value-of select="local-variables:stringValueOf($assigned_user_key)"/>
</mks:field>
```

These lines are used to retrieve the value of the **Assigned User** field. The variable `$assigned_user_key` should be defined in the same file under the field variable declaration segment and can be defined as:

```
<xsl:variable name="assigned_user_key" select="'Assigned User'"/>
```

## Exporting Word Documents to Integrity

---

**NOTE** Processing time and performance is affected by data size, network communication between the client and server, and server load.

---

### To export Word documents to Integrity

1 Do one of the following:

- To export the entire contents of a document currently open in Word, select **MKS > Export Active Document**.
- To export portions of a document currently open in Word, select the relevant portions, then select **MKS > Export Selection**.
- To export multiple documents not currently open in Word, select **MKS > Export Multiple Documents**. The **Select Documents for Export** dialog box displays.

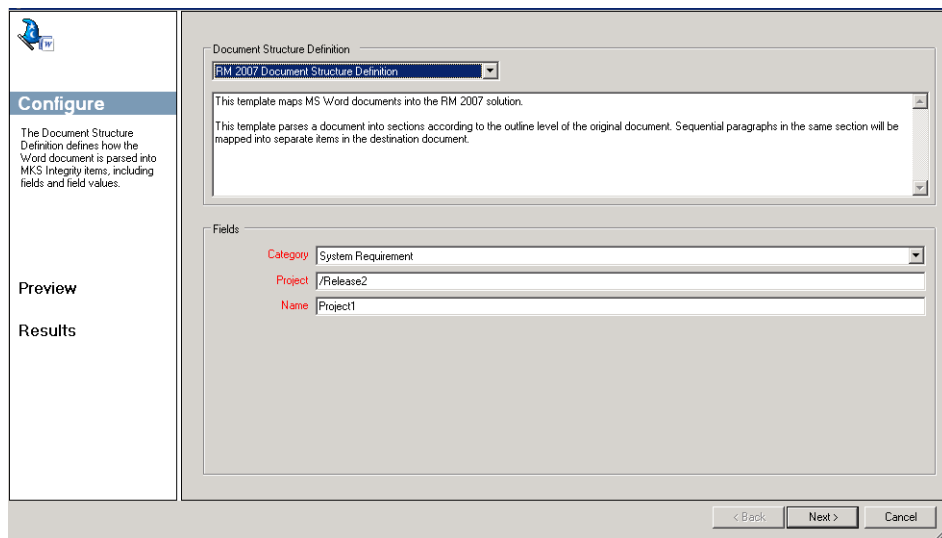
Select the documents you want to convert by clicking **Add**, then click **OK**.

---

**NOTE** You cannot preview batch conversions.

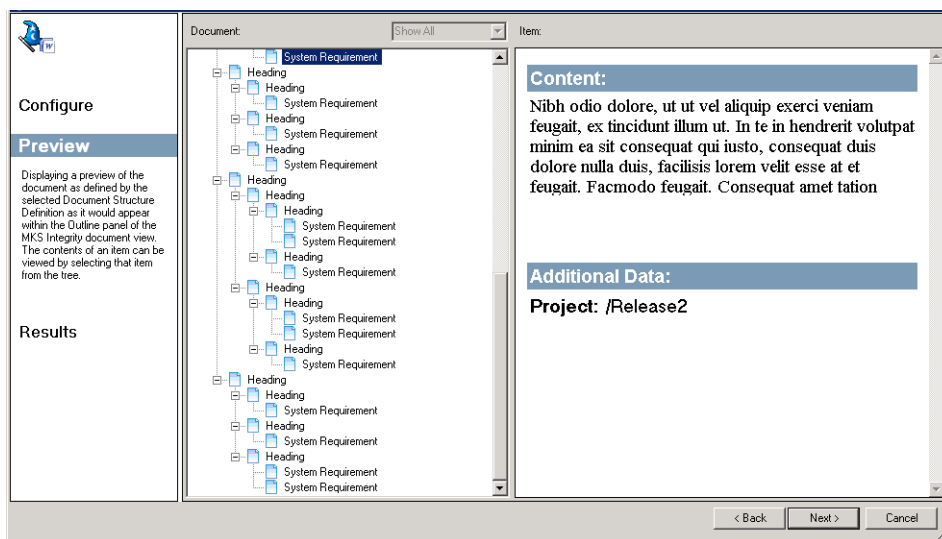
---

The **Export to MKS Integrity Wizard** displays.



- 2 From the **Document Structure Definition** list, select the DSD to parse the document.
- 3 Under **Fields**, complete or change the default field values. Mandatory fields appear red.
- 4 Do one of the following:
  - To apply the DSD to the document and see a preview, click **Next** or **Preview** in the left-hand pane. A progress bar in the wizard indicates status. The **Preview** panel displays a preview of the document as it would appear in the Outline panel of the Integrity Document view.
  - To skip the preview and view the final results, click **Results** in the left-hand pane and proceed to step 7. The integration applies the DSD to the document and creates new items.

**NOTE** If you are exporting a Word document that is not in Microsoft Word 2007 (**docx**) format, the integration prompts you to save a copy of the document. Click **OK**, then save the document as a **docx** file.



- 5 In the **Document** pane, you can view the content by expanding and collapsing segments and selecting items. Item content displays in the **Item** pane.

- 
- To create the items and view the final results, click **Next** or **Results** in the left-hand pane. The **Results** panel displays.
  - The **Results** panel displays a summary of the document with icons to indicate the success (🟢) or failure (🔴) of exported items. Successfully exported items display item IDs.

The filter at the top of the wizard allows you to **Show All** or **Show Errors**.

---

**NOTE** You can copy the results to the clipboard by right-clicking on the tree or pressing CTRL+C.

---

- To close the wizard, click **Close**. If errors occurred, the integration displays which document(s) did not successfully export. Click **OK**.

---

**NOTE** If the initial document submission fails, the next submission may attempt to reuse any items that were successfully created the first time and merge them with the new items. If you are exporting selections from a document, the integration does not attempt to reuse and merge items. For more information, see “Troubleshooting” on page 143.

---

- You can now switch to the Integrity Client to work with the new items.

## Troubleshooting

Integrity provides two log files to troubleshoot errors. By default, the `C:\Documents and Settings\\Local Settings\Temp` directory contains the following log files:

- `MSWord-log.txt` displays errors with the Word integration
- `MSWord-APIsession.log` displays errors with the API

If errors occur, consider the following:

- Potential errors include: communication with the API, data processing with the item mapper, or committing processed items and attachments to Integrity.
- Possible reasons for errors occurring during the conversion process include: connection errors, misconfiguration of the Document Structure Definition, or data volume.
- If an item has been partially submitted, you can attempt to fix the input and resubmit the failed parts, or cancel the submission and re-submit.
- Specifying an incorrect field in the **Export to MKS Integrity Wizard** displays an exception error message.

The Integrity to Microsoft® Excel® integration combines the powerful spreadsheet features of Microsoft Excel 2003, 2007, and 2010 (Excel), with the flexible workflow of Integrity. Users can work with Integrity item data in either product and keep the information in sync. This integration can be used for any situation where Excel's functionality might be useful, for example, for gathering requirements or managing features for projects.

---

**NOTE** The Integrity integration with Excel uses XML functionality and therefore requires Microsoft Excel 2003/2007 Office Professional Edition.

---

The integration also allows the import and export of requirement documents between Integrity and Excel. Unstructured documents can be created in Excel and structure added using the Integrity Document view. The integration works with both embedded and attached documents in Integrity.

The integration ZIP file (`MKSIntegrity2009&2007_SP3_MSExcels2003_Integration.zip`) is available for download from the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>). The integration ZIP file can be found under the **Downloads** tab.

To help you set up and use the integration, this chapter includes the following topics:

- “Overview” on page 145
- “Before You Start” on page 145
- “Setting the Integrity Server Connection Policy” on page 147
- “Installing the Microsoft Excel Integration” on page 147
- “Repairing or Removing the Integration” on page 148
- “Customizing the Microsoft Excel Integration” on page 149
- “Using the Microsoft Excel Integration” on page 152

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---



---

## Overview

The Microsoft Excel integration uses the Excel data list (or table) to store Integrity data. The integration supports users who need to use the features of Excel to manage and process data stored within Integrity.

The integration allows you to use Integrity data within Excel to produce requirement documents, reports, and charts, and to perform data analysis for project planning. Users can also import Integrity data into an Excel sheet and publish data in an existing Excel sheet to Integrity.

An Excel data list comprises a series of rows containing related data. The integration works through an add-in component that provides access to functionality through the Microsoft Excel **Data** menu. The integration supports:

- **Creation of new Integrity items from data entered in Microsoft Excel**

You can use the integration to create an empty data list in Excel, and manually enter new data as required. You can then publish this data to Integrity, which automatically creates Integrity items based on the existing rows in the data list.

- **Creation of new data in Microsoft Excel from existing Integrity items**

You can import Integrity items based on an Integrity query or, in the case of the requirement document type, by document (item) ID. This creates a new data list in Excel and populates it with Integrity data. For each item found in the Integrity query, the integration creates a row in the Excel data list.

- **Data synchronization**

Once an item is created, you can track and monitor state changes for that item by resynchronizing the data list in Excel. This both publishes and retrieves updates for all the rows in the Excel data list.

## Before You Start

This section provides details on basic system requirements, integration components, assumptions, and key considerations for the Microsoft Excel integration as follows:

- “Integrity Server Requirements” on page 145
- “Client Requirements” on page 146
- “Integration Components” on page 146
- “Assumptions” on page 146
- “Key Considerations” on page 146

## Integrity Server Requirements

- General requirements for the Integrity Server are as described in the *Integrity Server Installation and Configuration Guide*.
- Integrity database requirements are as described in the *Integrity Server Installation and Configuration Guide*.
- Integrity Server 2009 or Integrity Server 10.

---

## Client Requirements

- Microsoft Excel 2003 (Microsoft Office Professional Edition, Service Pack 2 or greater).  
Microsoft Excel 2007 (Microsoft Office Professional Plus 2007).  
Microsoft Excel 2010.
- Microsoft .NET Framework 1.1 runtime with required service packs installed. If Microsoft .NET Framework 2.0 or greater is installed, but version 1.1 is not, you cannot install the integration. If .NET Framework 2.0 or greater is installed *after* version 1.1, there is no effect on the integration.
- Administrators require the Integrity Administration Client for configuring ACL permissions and setting up Integrity workflows.
- Users do not require the Integrity Client to use the Microsoft Excel integration.

## Integration Components

The Microsoft Excel integration includes the following components:

- `MKSIntegrity2009&2007_SP3_MSExcel2003_Integration.zip`—the Integrity to Microsoft Excel Integration install (with Setup Wizard) is available for download from the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>).
- The XML mapping templates that configure the integration, including:

```
MS_Excel_Defects.xml.sample  
MS_Excel_Requirements_Document.xml.sample  
MS_Excel_Test_Suite.xml.sample  
MS_Excel_RM2007_Requirement_Document.xml.sample
```

The sample XML mapping templates are contained in `Integrity2009_Server_Templates.zip`, available for download from the Integrity Support Center.

---

**CAUTION** If you have previously modified the XML mapping template files, ensure that you create backup copies of your modified templates *before* extracting any new template files provided with this integration.

---

## Assumptions

- You know how to use Microsoft Excel. For more information about using the product, refer to the appropriate documentation from the product vendor.
- If you are modifying a mapping template, you understand and can use XML. If you need to modify a template and do not understand XML, contact PTC-Integrity Support.

## Key Considerations

- If you are using an English version of Excel on a machine with regional settings configured for a non-English language, an error may occur when running the integration (`Old Format or Invalid Type Library`). For more information on this error and how to resolve it, browse to the following Microsoft Support Web page:

<http://support.microsoft.com/kb/320369/en-us>

---

## Setting the Integrity Server Connection Policy

Before users can work with the Microsoft Excel integration, the administrator must configure the Integrity Server to allow remote API connections to the server. This enables integration users to connect to and run commands on the server. The default connection policy allows only a specific set of IP addresses to connect. The required setting allows all clients to connect.

The connection policy is specified in the following file on the Integrity Server:

```
<installdir>\config\client\IntegrityClientSite.rc
```

where `<installdir>` is the path to the directory where you installed the Integrity Server.

### To set the Integrity Server connection policy

- 1 In a text editor, open the `IntegrityClientSite.rc` file.
- 2 Comment out the following default policy:

```
daemon.connectionPolicy=mks.ic.common.policy.  
ICAllowSpecificConnectionPolicy
```

- 3 Uncomment the following policy:

```
daemon.connectionPolicy=mks.ic.common.policy.  
ICAllowAllConnectionPolicy
```

---

**TIP** To comment out a policy, insert a # symbol as the first letter of the property. To uncomment a policy, remove the # symbol.

---

This allows all clients to request an API connection; however, all connecting clients still require the proper authentication.

If there is a requirement to specify individual users who are allowed to connect, use a comma-delimited list of the user IDs to define the property for `daemon.validUsersList`. Using a comma-delimited list does not change the connection policy setting.

- 4 To have the changes take effect, restart the Integrity Server.

---

**NOTE** For more information on configuring the Integrity Server to allow remote API connections, see the *Integrity Integrations Builder API Guide*.

---

## Installing the Microsoft Excel Integration

You install the Microsoft Excel integration using the Setup Wizard provided with the `MKSIntegrity2009&2007_SP3_MSExcel2003_Integration.zip` file. The integration ZIP file is available for download from the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>).

During the installation, you may be prompted to install updates for Microsoft .NET Framework 1.1, depending on the Microsoft updates installed on your system. If you are prompted to perform an update, follow the links to download the required redistributable program from the Microsoft Windows Update site at:

<http://windowsupdate.microsoft.com>

---

The following updates are required and must be installed in the sequence shown:

- .NET Framework 1.1 Redistributable (Developer version)
- .NET Framework 1.1 Service Pack 1

---

**NOTE** If Microsoft .NET Framework 2.0 is installed but version 1.1 is not, you cannot install the integration. If .NET Framework 2.0 is installed *after* version 1.1, there is no effect on the integration.

---

### To install the Microsoft Excel integration on a client machine

- 1 Ensure that you close Microsoft Excel before installing the integration.
- 2 From the `MKSIntegrity2009&2007_SP3_MSExcels2003_Integration.zip` file supplied with the integration, extract all files to a common directory, and run the `setup.exe` file. The Setup Wizard displays.
- 3 To continue, click **Next**. The **Select Installation Folder** panel displays.
- 4 By default, the following directory is specified:

`C:\Program Files\MKS\Microsoft Excel Integration`

To specify another directory for the installation, click **Browse** and choose the target directory.

To determine the amount of disk space required for the installation, click **Disk Cost**. The Disk Space dialog box summarizes information on the **Volume**, **Disk Size**, **Available**, and **Required** disk space. To return to the Setup Wizard, click **OK**.

- 5 On the same panel, you can also configure the option that allows access to the integration while using Microsoft Excel. To allow access to yourself only, select **Just Me**. To allow access to anyone using your computer, select **Everyone**. By default, the option is set to **Just Me**.
- 6 To continue, click **Next**. The **Confirm Installation** panel displays.
- 7 To start the installation, click **Next**. The **Installing Panel** displays, indicating the progress for the installation. When the installation is complete, the **Installation Complete** panel displays.
- 8 To exit the Setup Wizard, click **Close**.

---

**IMPORTANT** The Microsoft Excel integration uses certain files installed with the .NET Framework. To check for any critical updates to the .NET Framework, use the Microsoft Windows site at:

<http://windowsupdate.microsoft.com>

---

You can now open Microsoft Excel on a client machine and configure the required Microsoft Excel properties. For more information, see “Customizing the Microsoft Excel Integration” on page 149.

## Repairing or Removing the Integration

You can use the Setup Wizard to repair or remove the Microsoft Excel integration.

---

**TIP** You can also remove the Microsoft Excel integration using Windows **Start > Control Panel > Add or Remove Programs**.

---

---

## To repair or remove the Microsoft Excel integration using the Setup Wizard

- 1 Ensure that you close Microsoft Excel before attempting to repair or remove the integration.
- 2 From the `MKSIntegrity2009&2007_SP3_MSExcel2003_Integration.zip` file, extract and run the `setup.exe` file. The Setup Wizard displays.
- 3 Select one of the available options as required:
  - To repair the integration with Integrity, select the option for **Repair Integrity to Microsoft Excel Integration**.
  - To remove the integration with Integrity, select the option for **Remove Integrity to Microsoft Excel Integration**.
- 4 To continue the selected operation, click **Finish**. The integration with Integrity is repaired or removed, according to the option you selected.

## Customizing the Microsoft Excel Integration

This section includes the following topics to assist with customizing the integration with Microsoft Excel integration:

- “XML Mapping Template” on page 149
- “Integrity Custom Properties” on page 151
- “Enabling SSL Communication” on page 151
- “Logging” on page 152

### XML Mapping Template

The following XML mapping templates provide sample mappings for running the Microsoft Excel integration with Integrity for Application Lifecycle Management 2009:

- `MS_Excel_Defects.xml.sample`, providing a sample mapping configuration for ALM 2009 Defects. By default, the template supports the `Defect` type.
- `MS_Excel_Requirements_Document.xml.sample`, providing a sample mapping configuration for ALM 2009 Requirement Documents. By default, the template supports the `Requirement` type.
- `MS_Excel_Test_Suite.xml.sample`, providing a sample mapping configuration for ALM 2009 test suites. By default, the template supports the `Test case` type. For more information on test suites, see the *Integrity User Guide*.
- `MS_Excel_RM2007_Requirement_Document.xml.sample`, providing a sample mapping for migration from 2007 to 2009.

---

**NOTE**

- Ensure that the internal field values in the XML mapping template reflect the Integrity field names as installed for your workflow or ALM installation. For example, if you installed ALM with a prefix, ensure that the same prefix is present in the internal field values contained in the XML mapping template. For general information on configuring the XML mapping template, see “Configuring a Mapping Template” on page 188. For more detailed information, contact PTC-Integrity Support.
  - If changes are made to the fields visible with the supported type, the selected template must be modified.
  - The external name for **Summary** should not be changed.
- 

**To modify the Excel integration XML mapping template**

- 1 From the `Integrity2009_Server_Templates.zip` file, extract the sample mapping templates to the following directory on the Integrity Server:

```
<installdir>\data\gateway\mappings\
```

where `<installdir>` is the path to the directory where you installed the Integrity Server.

---

**CAUTION** If you have previously modified the XML mapping template files, ensure that you create backup copies of your modified templates *before* extracting any new template files provided with this integration.

---

- 2 Rename the files to remove the `.sample` extension, and then modify the XML mapping fields as required to correspond to your existing workflow or ALM installation. Ensure that the internal field values used in the XML mapping template correspond to the Integrity field names as installed for your existing workflow or ALM installation.

To use the Excel integration with a type other than those defined in a template, you must create a copy of the template and change the name attribute to indicate the type it will be used with. All field mappings must then be modified to correspond with the selected type. For general information on modifying the XML mapping templates for Excel, see “Configuring a Mapping Template” on page 188.

**To import a mapping template**

To integrate a subset of the fields in any template, you import the mapping template into Excel. You can then drag and drop the required fields from the template into an Excel data list. For more information, see “Importing the XML Mapping Template” on page 153.

To import a mapping template in Excel 2003, select **Data > MKS Integrity > Import Map**.

To import a mapping template in Excel 2007, click the **Add-Ins** tab and select **MKS Integrity > Import Map**.

---

**NOTE** If a field is mandatory in the Integrity workflow, it should be designated as mandatory in XML mapping template so that users do not attempt to publish without first entering data for that field. Similarly, if a field is required in the XML mapping template, then that field should also be set as mandatory in the Integrity workflow so that users cannot create an item without first completing that field.

---

---

## Integrity Custom Properties

To use the integration with Microsoft Excel, certain custom properties are configured for each worksheet file you are working on. The required custom properties are created automatically when you run the first synchronization operation.

For Excel 2003, the required integration properties are configured under **File > Properties**.

For Excel 2007, the required integration properties are configured under **Office Button > Prepare > Properties**. Click the **Document Properties** panel and **Advanced Properties**. Properties are configured under the **Custom** tab in the **Properties** box.

The following properties are automatically configured for each Microsoft Excel **.xls** file when you run a synchronization with Integrity:

Name	Type	Value
<b>MKS Integrity Server Hostname</b>	Text	Name of the Integrity Server.
<b>MKS Integrity Server Port</b>	Number	Port number used by Integrity to connect to the Integrity Server. By default, the port number is 7001. Check with your Integrity administrator to confirm the correct port number.
<b>MKSIntegrityXmlMap_1</b> <b>MKSIntegrityXmlMap_2</b> <b>MKSIntegrityXmlMap_3</b> <b>MKSIntegrityXmlMap_4</b> ... incremented by one for each data list used with the integration	Text	The name of the XML mapping template used with the integration. The templates are configured to use the supported features of the Microsoft Excel integration. You can also modify the templates to suit the workflows and types used for your projects. For more information on modifying a template, see "Customizing the Microsoft Excel Integration" on page 149.

## Enabling SSL Communication

When using the Integrity integration with Microsoft Excel, you can choose a Secure Sockets Layer (SSL) connection. To configure SSL for the Microsoft Excel integration:

- 1 Open the Microsoft Excel (**.xls**) file and click the Microsoft Office Button. From the menu, select **Prepare > Properties**. Click the **Document Properties** panel and select **Advanced Properties**.
- 2 Click the **Custom** tab and in the **Name** field, add a property for **MKS SSL Connection**.
- 3 In the **Type** field, select **Yes or no**. In the **Value** field, choose the **Yes** option. To accept the changes click **Add** and then **OK**.
- 4 Save and re-open the **.xls** file. The SSL connection to Microsoft Excel is now enabled. For Microsoft Excel, SSL communications are enabled on a per-session basis, therefore, if you close Excel, you must reset the **MKS SSL Connection** property to use secure communications when you start a new session. To deactivate SSL during the same Excel session, you can set **MKS SSL Connection** to **No**.

---

**NOTE** To use an SSL connection with the Microsoft Excel integration, you must also enable SSL communications for the server-side API. The API authenticates its session via a separate HTTP connection to the Integrity Server. To allow this authentication to use the SSL connection, the Certificate Authority associated with the SSL certificate must be registered with the JRE used by the Integrity Server.

---

---

## Logging

Logging is enabled by default for the Microsoft Excel integration. Log and error messages are sent to a temporary subdirectory under the home directory of the user. On Windows, the file for client logging is:

```
C:\Document and Settings\\Local Settings\Temp\excelintc.log
```

The file for API logging is:

```
C:\Document and Settings\\Local Settings\Temp\excelintc.log
```

Logging and error messages can be used to assist with any diagnostic work. The log files are retained until the integration is re-started.

---

**NOTE** The location for log files depends on the value set for the system variable **TMP**. If the system variable is not set, then the value of the user variable **TEMP** is used.

---

## Using the Microsoft Excel Integration

Integration actions are available only within Excel workbooks. Each Integrity item is represented by a row, and each item field is represented by a column. Related fields are presented as a list of related Integrity item IDs.

The integration also allows you to work with requirement documents, providing the ability to create new documents and edit existing ones. Existing requirement documents can also be retrieved from Integrity into Excel.

You can have multiple data lists/tables within an Excel sheet. You can also manage and analyze the data in the list independently of data outside the list. Synchronizing data between Integrity and Microsoft Excel does not affect non-Integrity data.

---

**IMPORTANT** Ensure that the date format used in your data list matches the default Microsoft Excel date format. For more information on date formats and how they are handled by the integration, see “Date Fields and Formats” on page 159.

---

You can perform any of the following tasks within the Excel data list:

- rename or delete columns
- change the order of columns
- add columns of non-Integrity data

---

**NOTE** A new row added to the Excel data list becomes a new Integrity item upon synchronization.

---

This section provides information on the following topics:

- “Importing the XML Mapping Template” on page 153
- “Creating a New List” on page 154
- “Retrieving Items From Integrity” on page 154
- “Synchronizing Data” on page 155
- “Working With Requirement Documents” on page 157



- 
- “Detecting Conflicts” on page 159
  - “Working With Special Fields” on page 159

## Importing the XML Mapping Template

The **Import Map** function loads the fields from the XML mapping template into the Excel sheet. After importing the mapping template, you can then display the XML source and use drag and drop to customize the sheet with only the fields you need.

Once the fields are mapped, you can also rename the column headers with a new display name or change the order of the columns without affecting your ability to publish and retrieve updates. Updates to the sheet retrieve and publish data only for the selected subset of fields. For example, if you did not map the `Project` field in your sheet, you cannot publish data to the `Project` field for the Integrity item.

---

**TIP** If you require all the fields contained in the mapping template, you can use the **Create New List** function to import the complete mapping template and automatically create the required columns.

---

### Importing a mapping template using Microsoft Excel

- 1 To import the mapping template using Microsoft Excel 2003, select **Data > MKS Integrity > Import Map**. After connecting to the Integrity Server, the **Excel Integration Configuration** dialog box displays.

To import the mapping template using Microsoft Excel 2007, click the **Add-Ins** tab and select **MKS Integrity > Import Map**. After connecting to the Integrity Server, the **Excel Integration Configuration** dialog box displays.

- 2 To complete the import, choose the required template from the **Select a Template** list and click **OK**.

---

**NOTE** If you select the **Microsoft Excel Requirements Document Template**, you can also enter information under **Document Properties**. For more information, see “Editing an Existing Requirement Document” on page 158.

---

- 3 After importing the mapping template, you can then drag and drop individual fields from Integrity into the Excel data list.
- 4 To display the template fields in Excel 2003, select **Data > XML > XML Source**. The **XML Source** pane displays all the fields from the mapping template that you imported. To customize your Excel sheet, drag and drop the required fields from the **XML Source** pane onto the sheet in the column where you want the field heading to appear.

---

**TIP** You can also create a column by right-clicking the target field name under **XML Source** and then choosing **Map element** from the shortcut menu. In the dialog box, specify or choose the column cell where you want the field heading to appear and click **OK**.

---

To display the template fields in Excel 2007, click the Microsoft Office Button and then click **Excel Options**. Under **Popular**, select the option for **Show Developer tab in the Ribbon** and click **OK**. In the **XML** section of the **Developer** tab, click **Source** and the **XML Source** pane displays all the fields from the mapping template that you imported.

- 
- 5 To remove a mapped field, right click the bolded field name under **XML Source** and choose **Remove element** from the shortcut menu. To remove the column from the sheet, you must manually delete it.

## Creating a New List

The **Create New List** function imports the complete mapping template and automatically creates the required columns in the Excel sheet. **Create New List** creates an empty data list, but does not allow you to use drag and drop to select XML fields.

To populate the sheet with data, you can use either the **Get Items** or **Synchronize with Query** function. For more information, see “Retrieving Items From Integrity” on page 154 and “Synchronizing Using Queries” on page 156.

When working with a requirement document, you can also use the **Synchronize** command to populate the data sheet. For more information, see “Creating a New Requirement Document” on page 157.

---

### IMPORTANT

- You cannot create item ID numbers. Item ID numbers are created automatically by Integrity.
  - You can create multiple lists within a single Excel sheet. You can also have multiple lists containing Integrity and non-Integrity data.
  - If you are creating a large document, you can improve performance when adding requirements by using the **Synchronize with Query** command.
- 

### Creating a new list using Microsoft Excel

- 1 To create a new data list using Microsoft Excel 2003, select **Data > MKS Integrity > Create New List**. The **Excel Integration Configuration** dialog box displays.

To create a new data list using Microsoft Excel 2007, click the **Add-Ins** tab and select **MKS Integrity > Create New List**. The **Excel Integration Configuration** dialog box displays.

- 2 From the **Select a Template** list, choose the required template. This imports the mapping template and uses the template fields to create the column headings. For a list of the fields included in the available templates, see “Customizing the Microsoft Excel Integration” on page 149.

You can also enter data in the list and publish it to Integrity by clicking **Data > MKS Integrity > Synchronize**. For each row in the Microsoft Excel data list, the integration creates an item in Integrity.

## Retrieving Items From Integrity

The **Get Items** function allows you to import the mapping template and retrieve all items from the Integrity query you select. For example, if you selected a query called **All Projects**, **Get Items** creates a new data list that includes all items found by that query.

---

### IMPORTANT

- Do not attempt to import Integrity data into an Excel data list that contains existing data.
  - The **Select a Query** list displays only those queries that you have added as favorites within Integrity. For more information on working with queries, see the *Integrity User Guide*.
-

---

If Integrity items cannot be retrieved, an error message displays, providing information on the location to check for additional details (the **MKS Status (Reserved)** column or the log file). If the **MKS Status (Reserved)** column is not mapped, then an error status dialog box will display information. If the **MKS Status (Reserved)** column is mapped, then any errors are reported in the **MKS Status (Reserved)** column for each row of data.

---

**NOTE** For more information on the log file, see “Logging” on page 152.

---

### Retrieving items from Integrity using Microsoft Excel

- 1 To retrieve items from Integrity using Microsoft Excel 2003, select **Data > MKS Integrity > Get Items**, and if required, connect to the Integrity Server. The **Excel Integration Configuration** dialog box displays.

To retrieve items from Integrity using Microsoft Excel 2007, click the **Add-Ins** tab and select **MKS Integrity > Get Items**, and if required, connect to the Integrity Server. The **Excel Integration Configuration** dialog box displays.

- 2 From the **Select a Template** list, choose the required template and then select a query from the **Select a Query** list. To complete the operation, click **OK**.

All Integrity items found in the query are retrieved into the Excel sheet, with a row for each item found in the query.

## Synchronizing Data

The **Synchronize** function allows you to publish updates from Excel to Integrity, and from Integrity into Excel. Changes can include updates to existing Integrity items or the creation of new items in Integrity.

When publishing data to Integrity, the list you are synchronizing should not contain any empty rows. Depending on the setup of the XML mapping template (for example, if there is no mandatory field, or if a default value is set for a required field), an empty row can cause a new item to be created in Integrity.

The synchronization publishes to Integrity any updates made to item data in the Microsoft Excel data list. Updates to items in Integrity are also retrieved into Excel.

Any non-Integrity data in Excel is not affected by the synchronization. The order of the rows in the Excel list is also not affected. Formatting applied to any fields in Excel is not affected.

If the same data has been updated in both Excel and Integrity, this causes a conflict when you attempt to synchronize the data. Conflicts are logged and must be resolved manually. For more information, see “Detecting Conflicts” on page 159.

---

**NOTE**

- Before running a synchronization, save all changes in your Excel worksheet.
  - You can only synchronize one data list per operation.
  - If you synchronize a data list that contains a highlighted table cell, an error occurs. To correct the error, move the cursor to de-select the table cell, and repeat the synchronization operation.
-

---

## Publishing data from Microsoft Excel

- 1 To synchronize the data between Microsoft Excel 2003 and Integrity data, click **Data > MKS Integrity > Synchronize**.

To synchronize the data between Microsoft Excel 2007 and Integrity data, click the **Add-Ins** tab and select **MKS Integrity > Synchronize**.

- 2 You are prompted to confirm the synchronization operation. When the synchronization completes, the Microsoft Excel status bar displays **Ready**.

## Synchronizing Using Queries

The **Synchronize with Query** function works in two directions: to retrieve items from Integrity into the Excel sheet, and to publish updates from the Excel sheet to Integrity. When a synchronization is complete, the Microsoft Excel status bar displays **Ready**.

When publishing data to Integrity, the list you are synchronizing should not contain any empty rows. Depending on the setup of the XML mapping template (for example, if there is no mandatory field, or if a default value is set for a required field), an empty row can cause a new item to be created in Integrity.

If new items are added through a synchronization based on a query, the new items are appended to the bottom of the list. Error information is displayed in the **MKS Status (Reserved)** column.

---

### NOTE

- You can only resynchronize one data list per operation. Synchronizing data between Integrity and Microsoft Excel does not affect non-Integrity data.
  - Excel removes empty cells from its output when publishing to Integrity; therefore, deleted content will be updated with values from Integrity upon synchronization.
  - If you add a new requirement to an Excel list using an existing document ID, that requirement is added to the specified document when you run **Synchronize with Query**.
  - If you are creating a large document, you can improve performance when adding requirements by using the **Synchronize with Query** command.
- 

## Retrieving and publishing data using Microsoft Excel

- 1 To synchronize query-based data between Microsoft Excel 2003 and Integrity data, click **Data > MKS Integrity > Synchronize with Query**. The **Excel Integration Configuration** dialog box displays.

To synchronize query-based data between Microsoft Excel 2007 and Integrity data, click the **Add-Ins** tab and select **MKS Integrity > Synchronize with Query**. The **Excel Integration Configuration** dialog box displays.

- 2 Under **Query**, click the **Select a Query** list and select a query to base your synchronization on. The default query will be the one you used to create your data initially. Click **OK** and the integration runs the synchronization.

---

## Working With Requirement Documents

The Integrity integration with Excel allows you create a new data list in Excel and publish it as a single requirement document to Integrity. The new document is created as plain text with a flat structure. Any included subdocuments are also handled by the integration when you run the commands for **Get Items**, **Synchronize**, or **Synchronize with Query**. For more information, see “Creating a New Requirement Document” on page 157.

---

**NOTE** The procedures for working with requirements documents also apply to test suites.

---

The integration also allows you to edit existing requirement documents using both Excel and Integrity. Using the **Get Items** command, you can retrieve an existing document from Integrity and then edit it in Excel. You can also use the **Synchronize** command to publish and retrieve changes made in either Excel or Integrity. Existing documents can also be edited in rich text using the Document view in Integrity, with updates exported to Excel using the Synchronize command. Updates can be made using **Document Properties**. For more information, see “Editing an Existing Requirement Document” on page 158.

To change the structure of a document, you should use the Integrity Document view. After making the required changes, you can update the Excel sheet by using **Synchronize** to retrieve the updated document from Integrity. For more information, see “Adding Structure to a Document” on page 159.

When working with requirement documents, you should select one of the following templates from the **Select a Template** list:

**Microsoft Excel Requirements Document**  
**Microsoft Excel Test Suite**  
**Microsoft Excel migrated RM 2007 Requirement Document Template**

### Creating a New Requirement Document

You can use Excel to create a document from an initial set of requirements. When edited in Excel, each listed requirement is created at the top level (root) within the parent document.

The integration searches for different requirement types depending on column values. It then runs the associated commands in Integrity to create new document segments, or to add content to new or existing segments. The document ID and type is maintained for each synchronization.

When using the sample templates, the **Category** column indicates whether the Excel row is a comment, heading, or requirement. The sample mapping template can be customized to reflect the information mapping used in your requirement documents.

New rows are also added at the top level. If restructuring is required, this should be done using the Document view provided by Integrity Requirements. For more information, see “Adding Structure to a Document” on page 159.

---

**NOTE** For any errors where the requirement document is not created successfully, re-run the **Create New List** command, and copy the existing content to the new list.

---

### To create a new requirement document

- 1 To create a new requirement document using Microsoft Excel 2003, select **Data > MKS Integrity > Create New List**. The **Excel Integration Configuration** dialog box displays.

To create a new requirement document using Microsoft Excel 2007, click the **Add-Ins** tab and select **Data > MKS Integrity > Create New List**. The **Excel Integration Configuration** dialog box displays.

- 
- 2 From the **Select a Template** list, choose the **Microsoft Excel Requirements Document** and in the **Project** field under **Document Properties**, specify the project name. Text entered in the **Summary** field is used as the item summary. If no text is entered, the requirement document displays a null summary.

---

**NOTE**

- To create a new requirement document, a valid project name must be specified in the **Project** field. If the **Project** field is blank, the integration uses the default value specified for **Project** in the mapping template.
  - If the **Summary** field is blank, the integration uses the default value specified for **Summary** in the mapping template.
  - When creating a list and synchronizing, an error message displays if you have entered an incorrect project name in the **Project** field. To resolve the error, correct the project value in the **Project** column of the Excel data list and then synchronize with Integrity. The project and its contents are then created correctly.
- 

- 3 To import the template, click **OK**.
- 4 To generate the requirement document (item) ID, select **MKS Integrity > Synchronize** and click **OK**.
- 5 Enter new rows and text in the Excel data list as required to create the content in your requirement document. You can use the **Category** column to specify whether the Excel row is a comment, heading, or specific type of requirement. Each listed requirement is created at the top level within the parent document.
- 6 To publish the updates and create the necessary related items in Integrity, select **MKS Integrity > Synchronize** and click **OK**. Integrity automatically creates the required items and relationships.

## Editing an Existing Requirement Document

The integration supports the editing of existing requirement documents using both Excel (plain text) and Integrity (rich text). The **Category** column allows you to specify the type of content being added, whether it is a heading, comment, or requirement information. New rows inserted in Excel are added at the end of the existing content in the document.

To edit an existing document in Excel, you retrieve the document using **Get Items** and **Document Properties**. Edits made in Excel can then be published to Integrity using the **Synchronize** command.

### To retrieve an existing document from Integrity for editing in Excel

- 1 To retrieve an existing document for editing in Microsoft Excel 2003 select **Data > MKS Integrity > Get Items**. The **Excel Integration Configuration** dialog box displays.  
To retrieve an existing document for editing in Microsoft Excel 2007, click the **Add-Ins** tab and select **Data > MKS Integrity > Get Items**. The **Excel Integration Configuration** dialog box displays.
- 2 Under **Document Properties**, enter the **Document ID** (the associated Integrity item number).
- 3 Click **OK**, and the requirement document is retrieved into the Excel table list.
- 4 Edit the rows as required to update the document.
- 5 To publish the information back to Integrity, run **MKS Integrity > Synchronize**.

Edits made in Integrity can be exported to Excel using **Document Properties** and specifying a **Document ID** to identify the requirement document you want to update.

---

**NOTE**

- You can only work with one requirement document per sheet. For Excel workbooks with multiple sheets, you can import requirements using one Excel sheet for each transaction.
  - Excel does not support rich text; therefore if you edit a rich text entry using Excel, it is exported as plain text into Integrity. Inserted image attachments are not dropped; however, they must be manually restored to their previous location in the document.
- 

## Adding Structure to a Document

To change the structure of a document, you should use the Integrity Document view. For more detailed information on working with requirement documents in the Integrity Document view, see the *Integrity User Guide*.

## Detecting Conflicts

If the same data has been updated in both Excel and Integrity, you will see a conflict when you attempt to synchronize the data. Conflicts are logged and noted in the **MKS Status** column. Details of the field conflicts are included in the log file and must be resolved manually.

To review information on an identified conflict, refer to the `excelint.log` file found in the temporary directory. On Windows, this is under the home directory of the user:

```
C:\Document and Settings\\Local Settings\Temp
```

For the API, conflict information is logged to the `excelintc.log` file in the same temporary directory.

## Working With Special Fields

### Formulas

Excel formulas can be used within Integrity data. When the data is published to Integrity, it uses the resulting field value. Depending on the field mapping used in the mapping template, the formula is replaced with the field value when a synchronization brings updated data from Integrity.

### Date Fields and Formats

The integration uses two date formats—an external (Excel) format and an internal (Integrity) format. The external Excel format must be specified for date mapping to occur. If the internal Integrity date format is not specified, it defaults to the simple date (month/day/year).

---

**TIP** It is good practice to always specify the internal Integrity format.

---

Once a date field is set with a specified format in the XML mapping template, the value for that date field should not be left empty in either Integrity or Excel. Leaving a date field empty means the date format cannot be satisfied and causes the retrieval and/or publishing of that item to fail.

---

Within Integrity there are two date formats:

**MM/dd/yyyy** - (month/day/year) simple date; default format if not otherwise specified

**MM/dd/yyyy hh:mm:ss** - date+time (24hr format)

---

**NOTE** The internal Integrity date format defaults to the simple date format; however, if time is required, the secondary date+time format must be specified.

---

Within Excel, there are three date types - **date** (date only), **time** (time only), and **datetime** (date+time). The date types do not affect the display within Excel, but do affect the way data is handled during import and export, or for purposes of calculations.

Currently, the integration supports only the **date** type in Excel. Time information can be included, but Excel calculations for such fields will not use time values.

For the integration, the following external Excel formats are supported:

**"yyyy-MM-dd"** - (year-month-day) simple date

**"yyyy-MM-dd hh:mm:ss a"** - date field with time (*display* only) value

---

**IMPORTANT** Using a date format other than one of the supported formats can cause Excel to convert date values to display strings, making further date calculations impossible for that value.

---

### **Example**

The following provides an example of the XML mapping for date formats:

```
<!-- For a field that is a date-only field -->
<field
  external="New_Date"
  internal="New Date"
  data-type="date"
  external-date-format="yyyy-MM-dd"
  internal-date-format="MM/dd/yyyy"/>
<field
  external="Modified_Date"
  internal="Modified Date"
  direction="out"
  date-type="string" />
```

The date-only format is supported in the integration with Excel; however, any date/time fields in Integrity do not appear in Excel. Ensure that the date format used in your data list matches the date format used by Microsoft Excel.

---

**NOTE** If you want to use any other date formats, contact PTC-Integrity Support for assistance.

---







# **PART IV**

## **Other**

Sybase, HP, CA

The integration with Sybase® PowerBuilder® 10.5 allows users to access Integrity configuration management commands from within the PowerBuilder interface. This chapter provides information on configuring the integration and using the available commands. The following topics are discussed:

- “Configuring the PowerBuilder Integration” on page 164
- “Using the PowerBuilder Integration” on page 164
- “Creating an Integrity Project” on page 166
- “Creating a Sandbox” on page 166
- “Adding Members to an Integrity Project” on page 166
- “Checking Out Members” on page 167
- “Checking In Members” on page 167

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Configuring the PowerBuilder Integration

The Integrity Client includes the **File > Integrations** menu action for enabling and disabling the integration with Sybase PowerBuilder. You can select from a list of available integrations and enable or disable them, as required to work with your preferred application. For more information, see “Configuring Integrations” on page 3.

Before you can use Integrity for configuration management in your PowerBuilder projects, you must select Integrity as the source control system. This does not create Integrity configuration management projects or Sandboxes, but allows you to access the necessary commands when working in PowerBuilder.

---

**IMPORTANT** You must enable Integrity as the source control system for any PowerBuilder workspace in which you want to perform any other Integrity operation.

---

### To enable Integrity as the source control system in PowerBuilder

- 1 In the PowerBuilder Workspace panel, select the workspace file containing the PowerBuilder project you want to put under version control.
- 2 Right click and select **Properties**. The **Properties of Workspace** dialog box displays.
- 3 Click the **Source Control** tab and under **Source Control System**, select **MKS SCC Integration**.
- 4 To enable Integrity as the source control system, click **OK**.

## Using the PowerBuilder Integration

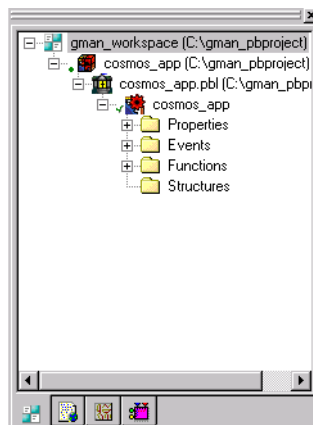
This section describes how to access Integrity through Sybase PowerBuilder. You can access version control functionality through the **Entry > Source Control** menu (only visible when the Library Painter is open), or by right clicking an object, and then selecting one of the following commands:

Command	Function
<b>Get Latest Version</b>	Equivalent to the Integrity <b>Resynchronize</b> command. Gets the latest version of the selected object and puts it in your working PowerBuilder library (.pb1) directory. The <b>Overwrite</b> dialog box may appear.
<b>Check Out</b>	Equivalent to the Integrity <b>Check Out</b> command. Checks out the selected object(s). The <b>Check Out</b> dialog box displays.
<b>Check In</b>	Equivalent to the Integrity <b>Check In</b> command. Checks in the selected object(s). The <b>Check In</b> dialog box displays.
<b>Undo Check Out</b>	Equivalent to the Integrity <b>Revert</b> command. Replaces the selected working file with the revision that was checked out, as it appeared prior to modification, and unlocks the object. The <b>Overwrite</b> dialog box may appear.
<b>Add to Source Control</b>	Equivalent to the Integrity <b>Add Members</b> command. Adds the selected object(s) to the Integrity project. The <b>Create Archive</b> dialog box displays.

Command	Function
<b>Remove from Source Control</b>	Equivalent to the Integrity <b>Drop Members</b> command. Drops the selected object(s) from the Integrity project. The <b>Drop Members</b> dialog box displays.
<b>Show History</b>	Equivalent to the Integrity <b>View Member History</b> command. Displays the revision history of the selected object. The <b>Member History</b> view displays.
<b>Show Differences</b>	Equivalent to the Integrity <b>Differences</b> command. Compares the selected object with the member revision. Visual Difference automatically launches and displays the two files.
<b>Refresh Status</b>	Equivalent to the Integrity <b>Scan for Changes</b> command. Updates the status and archive information of the selected object.
<b>MKS SCC Integration Properties</b>	Equivalent to the Integrity <b>Member Information</b> command. Displays member information for the selected object. The <b>Member Information</b> dialog box displays. <b>Note:</b> From the shortcut menu, select <b>SCC Properties</b> .
<b>Advanced Options</b>	Equivalent to the Integrity <b>Preferences</b> command. The <b>Preferences Configuration</b> window displays.
<b>Run MKS SCC Integration</b>	Equivalent to launching the graphical user interface or the Integrity <b>Open Sandbox</b> command. Displays the <b>Sandbox</b> view. <b>Note:</b> From the shortcut menu, select <b>Run Source Control Management Tool</b> .

Note the following about using the PowerBuilder integration:

- Files checked out by you are flagged with a green check mark. Files checked out by other users are flagged with a red check mark.
- When performing version control commands on a PowerBuilder project, make sure the **PBL** you are working with corresponds to the workspace you have open in the Workspace panel.
- The PowerBuilder workspace window is enhanced with version control features and indicators. For example, a green check mark beside an object indicates that the object is currently checked out.



- When refreshing the status of a member through the **Refresh Status** command, PowerBuilder displays a message asking permission to overwrite the selected member. This message informs you that PowerBuilder is overwriting read-only files when it copies them from the **PBL**.

---

To avoid these messages, right click and select **Properties** from the shortcut menu. The **Properties of Workspace** dialog box displays. From the dialog box, select the **Suppress prompts to overwrite read-only files** option.

## Creating an Integrity Project

**To create an Integrity configuration management project in PowerBuilder**

---

**NOTE** You must create a folder for the Integrity project and Sandbox before you create a project or Sandbox.

---

- 1 In PowerBuilder's Workspace panel, select the workspace file containing the PowerBuilder project you want to put under version control.
- 2 Right click and select **Properties**. The **Properties of Workspace** dialog box displays.
- 3 Next to the **Project** field, click the browse button. The **Specify the Project to Create** dialog box displays.
- 4 Create a project as described in the *Integrity User Guide*. The **Create Sandbox Wizard** displays.
- 5 Create a Sandbox as described in the *Integrity User Guide*.
- 6 Fill in additional options of the **Properties of Workspace** dialog box as needed and click **OK**.

## Creating a Sandbox

PowerBuilder does not directly support the creation of Sandboxes. For more information on working with Integrity Sandboxes from within PowerBuilder, contact PTC- Integrity Support.

## Adding Members to an Integrity Project

**To add members to an Integrity project in PowerBuilder**

- 1 In the PowerBuilder Workspace panel, select the workspace file.
- 2 Right click and from the shortcut menu, select **Add to Source Control**. The **Add to Source Control** dialog box displays.
- 3 Select one or more objects to add to the Integrity project.
- 4 Click **OK**. The **Create Archive** dialog box displays.
- 5 Modify the **Create Archive** options as necessary as described in the *Integrity User Guide*.

---

## Checking Out Members

### To check out members in PowerBuilder

- 1 From PowerBuilder's Workspace panel, select one or more objects to check out.
- 2 Right click and from the shortcut menu, select **Check Out**. The PowerBuilder **Check Out** dialog box displays.
- 3 Select one or more objects to check out.
- 4 Click **OK**. The **Check Out** dialog box displays.
- 5 Check out each member as described in the *Integrity User Guide*.

---

**NOTE** To check out a revision without placing a lock, add the following to the **PB.INI** file:

`SccCheckoutNoLock=1`

If a library section already exists, add the `SccCheckoutNoLock` option at the end of that section. If there is no library section, add the option at the end of the **PB.INI** file.

---

## Checking In Members

### To check in members in PowerBuilder

---

**NOTE** PowerBuilder does not prompt you when checking in an unmodified member. You should scan for changes before checking in a member.

---

- 1 From PowerBuilder's Workspace panel, select one or more objects to check in.
- 2 Right click, and from the shortcut menu, select **Check In**. The PowerBuilder **Check In** dialog box displays.
- 3 Select one or more objects to check in.
- 4 Click **OK**. The **Check In** dialog box displays.
- 5 Check in each member as described in the *Integrity User Guide*.





# HP Quality Center Requirements and Defect Modules

---

The Integrity integration with the HP Quality Center 9.2 Requirements and Defect Modules allows you to map requirement and defect items between Integrity and Quality Center 9.2.

The integration references the user fields in Quality Center to map to fields in Integrity. To provide a direct link between a Quality Center requirement or defect and an Integrity item, a linked field is created in Quality Center. In the field you have designated as the linked field in Quality Center, you can enter the Integrity item ID that you want the requirement to correspond to.

This integration is intended to work with Integrity for Application Lifecycle Management 2009. You can also configure the integration to work with previous workflows (for example, with workflows installed for Integrity for Requirements Management 2007 and ALM 2.1).

This chapter provides information on the following topics:

- “Integration Overview” on page 170
- “Assumptions” on page 171
- “System Requirements” on page 171
- “Installation and Configuration Overview” on page 171
- “Step 1: Configuring the Integrity Server” on page 172
- “Step 2: Configuring Quality Center” on page 173
- “Step 3: Configuring Integrity” on page 176
- “Step 4: Installing the Integration” on page 177
- “Step 5: Configuring the Integration” on page 178
- “Step 6: Running the Integration” on page 182
- “Troubleshooting” on page 183

---

**IMPORTANT** With the release of Integrity 10.0, the default installation directory of the Integrity Client has changed. This change affects integrations that were installed with the Integrity Client 2009 or earlier. For more information, see “Integrity Client Default Installation Directory” on page 3.

---

---

## Integration Overview

The integration references the user fields in Quality Center to map to fields in Integrity. To provide a direct link between a Quality Center requirement or defect and an Integrity item, a linked field is created in Quality Center. In the field you have designated as the linked field in Quality Center, you can enter the Integrity item ID that you want the requirement to correspond to.

You can run the integration with many different configurations to suit particular projects. Each configuration and server must have their own requirement and defect process configuration file and log file.

RM and defect process configuration files set the parameters for connecting to the Quality Center server and for handling logging and e-mail notifications. The sample file included with the integration can be used as a basis for your own configuration. To learn about the valid field values for this file, see “Modifying Process Configuration and Mapping Files” on page 179.

---

**NOTE** Each `processConfig.xml` file contained in the directory defined in the `hp-im-integration.bat` file is run each time the integration is run.

---

The XML mapping configuration files included with the integration are the sample templates you can use as a basis for the field mappings between Quality Center and Integrity. To learn about the valid field values for these files, see “Modifying Process Configuration and Mapping Files” on page 179.

Once the items are mapped, any updates to these fields are reflected in the item/requirement and/or item/defect when the integration is run. You can map integer, pick, float, logical, date, short text, long text, state, user, and group data types.

A log file reports any errors encountered while the integration is running. The path to the log file is specified in the process configuration files. Any errors that cause the integration to fail are logged, and can be reported via e-mail notification. Non-fatal errors are logged and the integration continues to run. You can specify one e-mail address or a set of e-mail addresses in the process configuration file.

To learn how to resolve specific errors or situations, see “Troubleshooting” on page 183.

---

**NOTE**

- The integration supports basic rich content synchronized between Integrity and Quality Center, for example: **bold**, underline, *italic*, and ~~strikethrough~~ are preserved.
  - Links between defects and requirements in Quality Center are not captured by the integration. As a result, the corresponding issues in Integrity will not be linked unless you link them manually. Similarly, relationships between Integrity requirements and tasks are not maintained during the integration and need to be added manually.
  - Folder names containing spaces that are listed in the `hp-im-integration.bat` file may cause issues with the integration. Ensure that you convert all folder names that have spaces to their **DOS** file system short name equivalent. To determine the short name of a folder, run the `dir/x` command from a command prompt on the required folder.
-

---

## Assumptions

Before you use the integration, you should fully understand:

- The hardware platform and Windows operating system on which the integration is installed.
- Integrity.
- Integrity for Requirements Management or Integrity for Application Lifecycle Management (ALM).
- HP Quality Center, including the **Requirements** and **Defect** modules.

For more information on Quality Center, browse to [www.hp.com](http://www.hp.com) and search for Quality Center 9.2.

- XML.

---

### NOTE

This integration uses XML files to create the mappings between Integrity and Quality Center. This guide assumes that if you are modifying the XML files, you understand and can use XML.

For general information on configuring the XML mapping template, see “Configuring a Mapping Template” on page 188. For more detailed information, contact PTC-Integrity Support.

---

## System Requirements

For the integration to work effectively, you must have the following installed and configured correctly:

- Integrity Server 2009 or later
- HP Quality Center 9.2
- JVM 1.5.0\_11 or higher

## Installation and Configuration Overview

The Integrity integration with Quality Center is provided in the `MKSIntegrity2007_QC92_IM_Integration.zip` file, available for download from the Integrity Support Center. This ZIP file includes the installable `QC92_IM_Integration_Install.zip` file.

Sample XML mapping templates and process configuration files are provided in the `Integrity2009_Server_Templates.zip` file and are available for download from the Integrity Support Center.

---

The following list is a high-level overview of the steps required to install and configure the integration:

- “Step 1: Configuring the Integrity Server” on page 172  
Configuring the Integrity Server to allow remote API connections.
- “Step 2: Configuring Quality Center” on page 173
  - a) “Creating and Editing Projects” on page 174
  - b) “Creating MKS ID Field for Requirements and Defects” on page 174
  - c) “Creating Additional Requirements User Fields” on page 175
  - d) “Adding the User Fields to All Requirement and Defect Types” on page 176
- “Step 3: Configuring Integrity” on page 176  
Configuring Integrity by creating the applicable required fields.
- “Step 4: Installing the Integration” on page 177  
Installing the integration from the `QC92_IM_Integration_Install.zip` file.
- “Step 5: Configuring the Integration” on page 178.
  - a) Create a new Integrity query.
  - b) Create the applicable process configuration files (`processConfig.xml`), each with a unique name.
  - c) Create a new mapping configuration file each for requirements and/or defects or edit the existing one.
  - d) Specify the location of the configuration files in the `hp-im-integration.bat` file.
  - e) In the Integrity Administration Client, set the **Attachments** fields to **Visible** on the required types.
- “Step 6: Running the Integration” on page 182  
Schedule the execution of the `hp-im-integration.bat` file.  
If you have any issues running the integration, see “Troubleshooting” on page 183.

## Step 1: Configuring the Integrity Server

Before users can work with the Quality Center integration, the administrator must configure the Integrity Server to allow remote API connections. This configuration sets the connection policy for a server integration point.

The two policies that control API access to the Integrity Server are the connection and authentication policies. Settings for these policies are specified in the following file on the Integrity Server:

```
<installdir>/config/client/IntegrityClientSite.rc
```

where `<installdir>` is the path to the directory where you installed the Integrity Server.

The required settings for these policies depend on your integration point; however, Integrity has set defaults for the server integration points. For the Quality Center integration, changes are required to the connection policy only. No changes are required for the authentication policy settings.

---

## Connection Policy

Within the `IntegrityClientSite.rc` file, the default connection policy is specified by the following property:

```
daemon.connectionPolicy=mks.ic.common.policy.  
ICAllowSpecificConnectionPolicy
```

The default setting allows only a specific set of IP addresses to connect. The required setting allows remote API connections.

### To configure the API connection policy for the Integrity Server

- 1 Open the following file in a text editor:

```
<installdir>/config/client/IntegrityClientSite.rc
```

where `<installdir>` is the path to the directory where you installed the Integrity Server.

- 2 Comment out the default policy (that is, insert the `#` symbol), and then uncomment the following property:

```
daemon.connectionPolicy=mks.ic.common.policy.  
ICAllowAllConnectionPolicy
```

---

**TIP** When you uncomment a property (by removing the `#` symbol), you enable the required property.

---

This allows all clients to request an API connection; however, all connecting clients still require the proper authentication.

If there is a requirement to specify individual users who are allowed to connect, use a comma-delimited list of the user IDs to define the property for `daemon.validUsersList`. Using a comma-delimited list does not change the connection policy setting.

- 3 Save and close the file.
- 4 To have the changes take effect, restart the Integrity Server.

---

**NOTE** For more information on configuring the Integrity Server to allow remote API connections, see the *Integrity Integrations Builder API Guide*.

---

## Step 2: Configuring Quality Center

You should configure Quality Center before you configure Integrity. In order to configure Quality Center to work properly with Integrity, you need to:

- “Creating and Editing Projects” on page 174
- “Creating MKS ID Field for Requirements and Defects” on page 174
- “Creating Additional Requirements User Fields” on page 175

---

## Creating and Editing Projects

Project settings must be configured for each individual Quality Center project referenced by the integration. For more information on creating a new project or editing an existing project, refer to the Quality Center product documentation.

Each project you create must have an appropriate integration user with permissions for creating and editing custom fields.

### To create a new project

- 1 Open Quality Center and click the **Site Administrator** link to log in.
- 2 Click **Create Project**.
- 3 Log out of **Site Administrator** before you “Creating MKS ID Field for Requirements and Defects” on page 174.

---

**TIP** Create one project per domain.

---

## Creating MKS ID Field for Requirements and Defects

The integration requires the **MKS ID** custom field by default. After you add this field to both requirements and defects you then need to add it to the applicable column set. The **MKS ID** custom field allows the Integrity item number to be displayed in the project.

You must have administrator rights to create custom fields in Quality Center.

---

**NOTE** For the Requirements module, all items are imported into Integrity as requirements (instead of requirement documents) unless you set the **MKS\_Type** field to **Requirement Document**. At minimum, the top-level requirement folder of Quality Center should be set to **Requirement Document** before the initial run of the integration.

---

### To create MKS ID custom fields

- 1 Connect to Quality Center and click the **Quality Center** link.
- 2 Type your login credentials and select the project you created in “Creating and Editing Projects” on page 174 or select an existing project.
- 3 Click **Tools > Customize > Customize Project Entities** and in **Quality Center - Project Customization**, click **Project Entities**.

---

**TIP** Use the Quality Center **Tools** menu, not the Web browser's **Tools** menu.

---

- 4 Under **Requirement**, select **User Fields**, and create the **MKS ID** custom field as follows:

Field Property	Value
Field Name	RQ_USER_01
Field Label	MKS ID
Field Type	String
Field Length	40

The options for **Masked** should be set specifically for your project.

- 
- Under **Defect**, select **User Fields**, and create the **MKS ID** custom field as follows:

Field Property	Value
Field Name	BG_USER_01
Field Label	MKS ID
Field Type	String
Field Length	40

The options for **Searchable** (defects only) and **Masked** should be set specifically for your project.

- Click **Return** to get back to the **Project Customization** page. The next step is to display the **MKS ID** field for requirements and defects in the Quality Center column heading.

The next section provides procedures for creating and displaying the **MKS ID** field in Quality Center. For more detailed information on creating custom fields in Quality Center, see the Quality Center product documentation.

### To display MKS ID custom fields in Quality Center

- In Quality Center, select **View > Select Columns**. The **Select Columns** dialog box displays.

---

**TIP** Use the Quality Center **View** menu, not your Web browser's **View** menu.

---

- Under **Available Columns**, select **MKS ID** and click the right arrow to move your selection to **Visible Columns** for both the **Requirement** and the **Defect** column set.

To accept the changes, click **OK**. The **MKS ID** displays in the requirement or defect item.

## Creating Additional Requirements User Fields

You must create additional Requirements user fields for the integration to function properly. Required fields can be used to pass information to or from Integrity to Quality Center. Any additional custom fields must be strings of an appropriate length.

---

**NOTE** The defect types require only the **BG\_USER\_01** field. To learn how to create this field, see "Creating MKS ID Field for Requirements and Defects" on page 174.

---

To set up new fields for use between Integrity and Quality Center, define the new field as follows:

- Set up the custom field in Quality Center.
- Map the new field in your production copies of the mapping files, located in:

```
<integration installdir>\MappingConfigFiles
```

For more information on mapping fields, see the comments provided within that file.

### To create requirement fields in Quality Center

- Connect to the Quality Center server.
- Under **Requirement**, select **User Fields** to create the custom fields. Refer to the mapping template for the required field values.
- Save your changes.

---

## Adding the User Fields to All Requirement and Defect Types

By default, the user fields `RQ_USER_01`, `RQ_USER_02`, and `RQ_USER_03` are only added to the `Unspecified` requirement type. You must add all three fields to all five requirement or defect types for them to function correctly.

Select **Customize > Requirements Types** and move all three user fields from `Not in Type` to `In Type` individually for each of the five other requirement or defect types.

## Setting the MKS Type in Quality Center

The **MKS Type** must be set when you create requirements in QC 9.2. For example, with the ALM 2009 workflow, you would choose the `Requirement Document` type when creating documents and `Requirement` type for requirements.

---

**IMPORTANT** The creation of your requirement fails if you do not define the MKS type for each requirement document. If the type is left blank, a requirement rather than a document is created.

---

## Step 3: Configuring Integrity

The next step is to create and modify the required custom fields in Integrity. Ensure that the fields are added for the applicable Requirement type in Integrity. For more information about creating fields, see the *Integrity Server Administration Guide*.

- 1 Open the Integrity Administration Client.
- 2 Under the **Workflows and Documents** node select **Fields**.
- 3 Right click and select **Create**. The **Create Field** dialog box displays.
- 4 Create the custom fields as required. Refer to the mapping template for the required field values.

For example, in the ALM 2009 workflow, the following custom fields would be required:

**ALM 2009 Required Custom Fields**

Name/Display Name	Data Type	Added To
QC Requirement Name	Short Text	Requirement
QC Id	Short Text	Requirement, Requirement Document, Change Request
QC Direct Cover Status	Short Text	Requirement, Requirement Document

- 5 Type the same field name as the field label you defined in the Quality Center project entities.
- 6 To save the new field, click **OK**.
- 7 Under the **Types** node, select the **Requirement** type.
- 8 Right click and select **Edit Type**. The **Edit Type** dialog box displays.



- 
- 9 In the left pane, select **Visible Fields** and enable the new field to ensure that it is visible to all of the groups using the integration. For more information on creating fields and editing types, see the *Integrity Server Administration Guide*.
  - 10 Repeat steps 5 - 7 for **Defect**.
  - 11 Save your changes.

## Configuring the Document View in Integrity

To provide ease of navigation between Integrity and Quality Center, you can also configure the Integrity Document view to display the same field layout structure that is displayed in Quality Center. You configure the Document view using the Integrity Client.

### To configure the Document view in Integrity

- 1 Select **View > Options**. The **Options** dialog box displays.
- 2 Under the **Title** tab, the following field formats can be modified:  

```
Outline column format
Title format
Tab title format
```
- 3 To save your changes, click **OK**.

---

**NOTE** For detailed instructions on configuring the Integrity Document view, see the *Integrity User Guide*.

---

## Step 4: Installing the Integration

The integration install (`QC92_IM_Integration_Install.zip`) is contained in `MKSIntegrity2007_QC92_IM_Integration.zip`, available for download from the Integrity Support Center:

<http://www.ptc.com/support/integrity.htm>

---

**NOTE** Ensure you install Integrity, Quality Center, and the integration on machines that register the same time in the same time zones. If the times and time zones are not in sync, you may experience problems completing the integration.

---

### To install the Quality Center integration

- 1 Extract the contents of `MKSIntegrity2007_QC92_IM_Integration.zip` to a temporary location on your `c:\` drive. This ZIP file includes the `QC92_IM_Integration_Install.zip` file.
- 2 Extract `QC92_IM_Integration_Install.zip` to `<integration installdir>`.
- 3 Log in to Quality Center. The `OTAClient.dll` file is created automatically in your chosen Quality Center directory, for example:  

```
C:\Program Files\Common Files\Mercury Interactive\TDAPIClient
```
- 4 Update the `hp-im-integration.bat` file (located in the Quality Center installation directory) with the path to `OTAClient.dll`.
- 5 Ensure that you have JVM installed and point to the JVM install location in the `hp-im-integration.bat` file instead of the JVM version included with Quality Center.
- 6 Proceed to “Step 5: Configuring the Integration” on page 178.

---

## Step 5: Configuring the Integration

After you install the integration, the next step is to configure the applicable components. The majority of the configuration is done using the XML process configuration and mapping configuration files.

Additional details on process configuration and mapping configuration files are discussed under:

- “Process Configuration XML Files” on page 180
- “XML Mapping Configuration Templates” on page 180

Sample XML mapping configuration templates and process configuration files are provided in the `Integrity2009_Server_Templates.zip` file, which is also available for download from the Integrity Support Center.

To download the sample configuration files, go to the Integrity Support Center:

<http://www.ptc.com/support/integrity.htm>

To configure the Quality Center integration

- 1 Create an Integrity query to search for all of the requirements you want to synchronize. Specify the query name or the query definition in the `processConfig.xml` file.

To learn how to create a query, see the *Integrity User Guide*.

- If the query times out, increase the query timeout by running

```
im diag --diag=setintpolicy --param=QueryTimeOut
--param=<specify seconds>
```

- If the number of items returned exceeds the server query maximum, increase the limit by configuring the `mksis.im.queryGovernor` property (formerly `mksis.im.maxQueryIssueCount` in the `<Integrity Server installldir>\config\properties\im.properties` file). For more information on configuring `mksis.im.queryGovernor`, see the *Integrity Server Installation and Configuration Guide*.

---

**TIP** You can determine the existing query limit by running the following command:

```
im diag --diag=policies
```

---

- 2 Create a new process configuration file for each Quality Center requirement and defect module project. Each file requires a unique name. You will require one `processConfig.xml` per server and one per configuration. You may require one or more for defects, one for requirements, one or more for ALM defects, one for ALM requirements, etc.

Place the newly created process configuration file in:

```
<integration installldir>\ProcessConfigFiles
```

---

**IMPORTANT** The integration uses different `processConfig.xml` files for each defect or requirement module, Quality Center project, and associated server.

Integrity runs all `*.xml` files that are contained in the directory specified in the `.bat` file. Any process configuration files you do not want to run must be renamed to something other than `*.xml`.

Mapping files are only used if they are referred to in the process configuration files. You may have as many mapping configuration files as required with any name you choose.

---

---

Refer to the requirement and defect `processConfig.xml.skeleton` files for additional information. To learn how to configure the files, see “Modifying Process Configuration and Mapping Files” on page 179.

- 3 Create or edit the existing mapping configuration files located in:

```
<integration installldir>\MappingConfigFiles
```

If required, you can modify the process and configuration mapping files to suit your application environment. See “Modifying Process Configuration and Mapping Files” on page 179 for field-level details.

- 4 Specify the location of the configuration files in the `hp-im-integration.bat` file. All `.xml` files contained in the above directory are referenced.
- 5 In the Integrity Administration Client, set the **Attachments** fields to **Visible** on the **Requirement** and **Requirement Document** types. If you want attachments to be visible on defects, do the same in **Task** and/or the **Requirement** type.

## Modifying Process Configuration and Mapping Files

Sample XML mapping templates are extracted to the `<integration installldir>\MappingConfigFiles` directory during installation.

Sample process configuration files are extracted to the `<integration installldir>\ProcessConfigFiles` directory during installation.

Together, these files are used to configure the integration. The mapping configuration files can be configured for your application environment.

---

**NOTE** Ensure that the internal field values in the XML mapping template reflect the Integrity field names as installed for your workflow or ALM installation. For example, if you installed ALM with a prefix, ensure that the same prefix is present in the internal field values contained in the XML mapping template.

For general information on configuring the XML mapping template, see “Configuring a Mapping Template” on page 188. For more detailed information, contact PTC-Integrity Support.

---

### Key Considerations

- For each `processConfig.xml` file, you can only specify one Quality Center server, one Integrity Server, one Quality Center project, and one defect or requirement module. For example, if you have one Quality Center server with one project and one Integrity Server, and you want to use defects and requirements, you must use two process configuration files.
- Mandatory fields in Integrity must have a value when an item is created. If mandatory fields do not have mappings to Quality Center fields, the integration cannot create an Integrity item. The default XML templates provided with the integration provide the necessary mappings and default values for mandatory fields in the Requirement type in Integrity.

When creating any additional mandatory fields, ensure that all mandatory Integrity fields for the requirement type are mapped to the appropriate Quality Center fields. Ensure that the fields mapped in Quality Center are also mandatory.

For more information on configuring mandatory fields, see the *Integrity Server Administration Guide*.

- The Quality Center integration user (connecting to Integrity) must have the required permissions for projects in Quality Center and for type and field visibility in Integrity. The

---

integration cannot map data to a type or field that is invisible to the integration user. For more information on visibility rules, see the *Integrity Server Administration Guide* or online help.

- Updates to Integrity requirement items will fail if the Integrity user configured to run the Quality Center integration does not have permission to create or modify a requirement item. If permission is then granted to the user before the next run of the integration, the missed updates are not be picked up because the date of the update precedes the last run time.
- Moving a requirement to a different parent is not supported in Quality Center. If you re-parent a requirement in Quality Center and then attempt to run the integration, you may not see the corresponding change in Integrity.
- Related requirements in Integrity cannot have the same name in the target Quality Center project.

### To modify the process configuration and XML mapping template files

---

#### NOTE

- XML code is case-sensitive.
  - If you need to further customize the template to suit the workflows used in your organization, contact PTC-Integrity Support.
- 

- 1 Copy the files and save them in the same directory for as many integrations as you require (separate directories for each).
- 2 Modify and save the copied versions to create the required mappings.

### Process Configuration XML Files

The `processConfig.xml` file controls the Quality Center adapter settings, Integrity connections, and error logging and notification.

Sample process configuration files are available with the integration. The skeleton files include comments to assist with customization.

### XML Mapping Configuration Templates

Sample XML mapping configuration templates are available for the Quality Center integration. The XML mapping configuration files allow you to map ALM and defect items to requirements in Quality Center. The sample files include comments to assist with customization.

The general option for `direction=""` defines the allowed direction of information transfer. Valid values are `both`, `in`, or `out`.

- `both` moves information in both directions between Integrity and Quality Center.
- `in` moves information into Integrity from Quality Center.
- `out` moves information out of Integrity to Quality Center.

An internal field is a field referenced in Integrity. An external field is a field in Quality Center.

Additional fields are mapped as `external="ALM_USER_nn" internal="MKS field name"` and so on. For more information, see "Creating Additional Requirements User Fields" on page 175.

---

For general information on configuring XML mapping templates, see “Configuring a Mapping Template” on page 188. For more detailed instructions, contact PTC-Integrity Support.

---

**IMPORTANT** Ensure that the internal field values in the XML mapping templates reflect the Integrity field names as installed for your workflow or ALM installation. For example, if you installed ALM with a prefix, ensure that the same prefix is present in the internal field values contained in the XML mapping template.

---

## Setting Up E-mail Notification

This section provides information about configuring the contents of the `processConfig.xml` file for e-mail notification for logging integration errors.

Contents of the e-mail message are configurable using number-encoded tags that expand to populate the message with the related information. The following tag codes are available:

Tag Code	Information Populated
{0}	Data Mapper Instance
{1}	Adapter Name
{2}	Date/Time
{3}	Log File Name
{4}	Log Message
{5}	Exception Class
{6}	Exception Message
{7}	Exception Stack Trace

For example,

```
An error occurred at {2} with the following log message:"{4}". For more information, refer to the {3} file.
```

expands to the following message:

```
An error occurred at 01/09/06 2:50 PM with the following log message: "An error occurred when mapping item 26 Could not save modified item 26: The following fields may not be edited [ Summary ]". For more information, refer to the QCDataMapper.log file.
```

## Setting Up Attachment Handling

The mapping files define the mapping of attachments between Integrity and Quality Center.

### Key Considerations

- Ensure that you monitor and periodically clean up the attachments folder specified in the process configuration files.
- Non-file attachments are not supported, for example, links. However, you can use the field `<field name="QC-ADDLINKS">http://HOSTNAME:PORT/im/viewissue?selection={ID}</field>`; to create attachment links. This field is further discussed in the process configuration file description in “Step 5: Configuring the Integration” on page 178.
- To use attachments with defects, you have to make Attachments visible on `Task`.

- 
- To change the way attachments are handled, modify the following field mappings:

```
<field external="_ATTACHMENT" internal="Attachments" direction="both" field-type="attachment" clobber="false"/>
```

where

- `external="_ATTACHMENT"` is the unique field name the integration uses for Quality Center attachments.

---

**CAUTION** Do not modify the `external="_ATTACHMENT"` field.

---

- `internal="Attachments"` is the field name Integrity uses for attachments.

---

**CAUTION** Do not modify the `internal="ATTACHMENTS"` field.

---

- `direction="both"` defines the allowed direction of information transfer. Valid values are `both`, `in`, or `out`. `both` moves information in both directions between Integrity and Quality Center. `in` moves information into Integrity from Quality Center. `out` moves information out of Integrity and into Quality Center.
- `on-create-only="false"` defines when attachments can be created. Valid values are `true` or `false`.
- The value `clobber=false` defines how attachments are handled by the integration. The value only applies to the handling of attachments in Quality Center when they are exported to Integrity.

The value `clobber=false` means that upon refresh, the new list of attachments is appended to any existing list of attachments. Revised attachments are updated in the list and new attachments are added to the existing list.

The value `clobber=true` means that upon refresh, the new list of attachments is used to replace any existing list of attachments. If the revised requirement has no attachments, then the update deletes any previously existing attachments.

---

**NOTE** To synchronize attachments in both directions, set `clobber` to `true` or the synchronization of the second attachment from Quality Center to Integrity fails.

---

## Step 6: Running the Integration

To run the integration, schedule the execution of the `hp-im-integration.bat` file. You should schedule a batch run of the integration during off hours to avoid potential errors that may result from users having Quality Center windows open in edit mode.

The `hp-im-integration.bat` file must be run twice during the initial load of requirements to ensure all data is completely synchronized between Quality Center and Integrity.

An error occurs if a requirement or a defect is created in Integrity, synchronized to Quality Center, and subsequently deleted. To remove the linkage, you must delete either the `MKS ID` or the original requirement in Quality Center.

---

### Key Considerations

- Links between defects and requirements in Quality Center are not captured by the integration. As a result, the corresponding issues in Integrity are not linked unless you link them manually. Similarly, relationships between Integrity requirements and tasks are not maintained during the integration and must be added manually.
- For the requirements module, all items are imported to Integrity as requirements unless you set the `MKS_Type` field to `Requirement Document`. The top-level requirement directory of QC 9.2 should be, at minimum, set to `Requirement Document` before the initial run of the integration.

## Troubleshooting

Log files store information about historical synchronizations. Integrity allows you to specify the name and location of the Quality Center integration log file. You must specify one log per scheduled integration. The log file name and location is specified using the `<log-file></log-file>` mapping in the `processConfig.xml` files.

To learn how to configure the process configuration file, see “Modifying Process Configuration and Mapping Files” on page 179.

In addition, the `IntegrityClientSite.rc` file may provide additional Integrity Server information.

Potential errors include: failed initial load synchronizations or updates, extra logging information in the `IntegrityClientSite.rc` log file, communication with the API, data processing with the mapping files.

Possible reasons for errors occurring during the initial load or update process include: connection errors, misconfiguration of the mapping files or process configuration file, running simultaneous integrations.

When problems occur in the Quality Center integration, an attempt is made to rectify the problem on the next run. This is accomplished using a defined number of retries in the `processConfig.xml` files.

If the problems are not resolved, the retry list (and subsequently, the integration runtime), may become long. To solve this:

- check the log files for any issues and fix them.
- delete the files in the `<installdir>/data` directory. Doing so will remove some of the retry attempts and keep the integration going.

If additional errors occur, consider the following:

- If you cannot run a synchronization because you have received an error indicating that integration is already running (and you are sure there is not one already running), delete the `.lck` file.
- Adjust the number of retries in the `processConfig.xml` file. This setting controls the number of times the integration attempts to update a requirement in the Quality Center Requirements module. If a requirement exceeds the number of failures it is no longer synchronized.

- 
- Re-run the integration if a synchronization failed during an initial load:
    - Ensure that all mappings in the mapping files contain values.
    - Reset the passwords in the Quality Center connection settings and/or the Integrity connection settings in the `processConfig.xml` file. Change `QC-PASSWORD-ENCRYPTED` to `false`.
    - Delete the `.lrt` and `.dat` files specific to each Quality Center project (four in total).
    - Run the integration again.
  - If a fatal error occurs during an initial load and the synchronization stops, try the following and re-run the integration:
    - Verify that the integration log does not end with  
`***** DATA MAPPER END *****`
    - Verify the existence of the `hs_err_pidXXXXX.log` log file corresponding to the halted integration.
    - If both of the above are true:
      - Delete corresponding `.lrt` and `.dat` files.
      - Delete the `.lck` (lock) file.
      - Run a query against the `REQ` table in the Quality Center database to determine if a requirement with the name `New_Requirement` exists.

Sample query:

```
SELECT * FROM REQ WHERE ALM_REQ_NAME LIKE 'New Requirement'
```

    - If a requirement named `New_Requirement` is found, delete it.
  - If a requirement update to Quality Center fails, check that the requirement is not locked. If it is locked, wait until the requirement is unlocked or remove the entry from the locks table inside the Quality Center database. The failed requirement is updated if the number of failed attempts does not exceed the maximum number of retries defined in the process configuration file. Run the integration again.



This chapter contains general information on the Integrity integration with CA Endeavor® Change Manager (CA Endeavor).

The following topics are discussed:

- “Understanding the CA Endeavor Integration” on page 186
- “Integrity Commands” on page 186

---

# Understanding the CA Endeavor Integration

The Integrity integration with CA Endeavor allows you to combine the highly customizable workflow management functionality of Integrity with the change management functionality of Endeavor within a mainframe software development environment.

The integration is designed so that specific events in Endeavor will trigger specific commands in Integrity. This allows you to connect development changes in Endeavor to the workflow management in Integrity.

## Integrity Commands

Integrity and Endeavor communicate through the COBOL Adapter, which in turn uses the Integrity API to run the commands in Integrity. The following Integrity commands can be configured to run once a specified Endeavor event occurs:

- `im connect`
- `im viewissue`
- `im editissue`
- `im createissue`<sup>1</sup>
- `im createcp`
- `im createcprent`<sup>1</sup>
- `im viewcp`
- `im editcprent`<sup>1</sup>
- `im cps`

Depending on the configuration of the integration, one or more of the available commands is initiated in Integrity. For more information on these commands, see the *Integrity CLI Reference Guide for Workflow and Documents*. For more information on Endeavor change packages, see the *Integrity User Guide* or online help.

---

**NOTE** Depending on the configuration of the integration, you may or may not receive confirmation from Integrity when commands are run, and you may or may not be required to provide input.

---

For more information on configuring and implementing this integration, visit the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>). A guide is available with the integration download and includes useful examples. Additionally, you may consult the *Integrity Integrations Builder API Guide*.

---

<sup>1</sup> For more information on these commands, go to the Integrity Support Center (<http://www.ptc.com/support/integrity.htm>).



**PART V**  
**Appendixes**

## Configuring a Mapping Template

---

This appendix provides information on the XML mapping template file that is used to configure the integrations with Microsoft Excel, Project and Word. The mapping template file defines the necessary two-way relationships between internal Integrity items and external items. The two-way mappings allow each set of items to remain synchronized such that a change made in one set of items is reflected in the other.

To assist you in editing a mapping template, this appendix includes the following topics:

- “Overview” on page 189
- “Mapping Template XML Elements” on page 190
- “Mapping Template Example” on page 199

---

**IMPORTANT** The mapping templates included with Integrity are functioning examples and should always be edited with care. If you are not familiar with XML, contact PTC - Integrity Support for further assistance.

---

---

## Overview

While the configuration is an XML file like an IIF document or the Wizard Configuration file, it is more than a way of representing data. It is a small programming language that lets you define the process of importing and exporting items from an external data source and Integrity internal items.

The Gateway Configuration file processes items one at a time and determines the complete set of mapping rules to apply to the item before actually applying any of them. This allows later steps to change or negate previous mapping rules. It also means that regardless of what mappings are scheduled to be applied, it is always the original values of the internal and external fields that are used.

There are two other important concepts for configuration files: conditional mappings and levels.

Conditional mappings are mapping instructions that only apply when they are activated by a `<map-conditional>` element (or the `<map-translation>` sub-element). These conditional mappings are contained within `<map>` elements. Until mappings are activated, they are ignored. Once a `<map>` is activated, the mapping instructions within that element become active. This can include additional `<map-conditional>` elements and the corresponding `<map>` elements; although such `<map>` elements are also ignored until they are activated.

Levels are a related concept. The process begins with the main instructions at level 1. Each time a `<map>` element is activated, the process moves up a level and remains at that new level until the `</map>` tag is reached. At that point, the process moves back to the previous level.

The importance of levels is in the way they interact with the `<map-conditional>` and `<map>` elements. When a `<map-conditional>` element is attempting to find a matching `<map>`, only `<map>` elements at the current level of processing are examined. This means that if you have a `<map-conditional>` element within a `<map>` element, the corresponding `<map>` elements that can be activated by that `<map-conditional>` must also appear within the same `<map>` element. Additionally, once a `<map-conditional>` on a given level has activated a `<map>` element and its contents, all other `<map-conditional>` elements at the current level are ignored.

---

## Mapping Template XML Elements

This section describes the various XML elements of the mapping template, and provides information on how to configure those elements.

### **<mapping>**

The **<mapping>** element is the highest level XML element. As such, it contains all other elements. There is only one **<mapping>** element in the file.

#### **Attributes:**

##### **name**

The **name** attribute specifies the name of the particular Gateway conversion process defined in the file. This name is the identifier through which an Gateway configuration is selected for use by the Gateway Wizard. It is expected to be unique across all known Gateway configurations.

##### **template-version**

The **template-version** attribute specifies the version of the XML template used for creating the Gateway Configuration file. This documentation describes version 2.1. This value is required for the proper operations of Gateway.

**Parent Element:** none (highest level XML element)

**Sub-Elements:** **<description>**, **<set-property>**, **<link-field>**, **<field>**, **<map-conditional>**, **<map>**

### **<description>**

The **<description>** element specifies a description of the particular conversion process defined in the file. The Gateway Wizard displays this description when prompting for the selection of the Gateway configuration to use.

There is one **<description>** element in a configuration element.

**Parent Element:** **<mapping>**

**Sub-Elements:** none

### **<set-property>**

The **<set-property>** element sets an item control property to a given value. For example:

```
<set-property name="owner" value="external"/>
```

When importing items into Integrity, there are a few common properties worth mentioning:

#### **prototype**

This property helps Integrity identify the type of item it is importing. Valid values include **DOCUMENT** and **CONTENT**. Commonly, this **prototype** property is provided directly within the definition of the item data being imported, but the Gateway configuration may be used to override and set the item's **prototype** property.

#### **owner**

This property helps Integrity determine whether the item is externally or internally owned. Valid values are **"external"** and **"internal"**. In the case of a document, for example, externally owned data would mean that Integrity should delete any content that is no longer present within the data being re-imported.

---

### **prior-data-field**

This property, in the case where a document is being re-imported, identifies to the attachment field within which an IIF document representing the previously imported data would be found. This property is only relevant to document items.

### **prior-data-name**

This property, in the case where a document is being re-imported, identifies to the name of the attachment file (an IIF document) that represents the previously imported data. Previously imported data is used to help identify the set the changes to be performed on the data.

#### **Attributes:**

##### **name**

The **name** attribute specifies the name of the property to set.

##### **value**

The **value** attribute specifies the value to which the property indicated by the name attribute is to be set.

**Parent Element:** <mapping>, <map>

**Sub-Elements:** none

### **<link-field>**

The <link-field> element automatically creates a link between a unique ID field in the external data source and the internal data. By creating such a link, it is much easier to update existing fields when re-importing previously imported data.

The <link-field> element has four basic forms. The first form is:

```
<link-field field-type="id">
```

When using this form, each item in the external data source has a unique identifier (similar to the primary key of a database) which is then used as the Integrity identifier.

The second form is:

```
<link-field external="REQID" field-type="id">
```

In this form, a field in the external data named **REQID** contains the Integrity identifier for the corresponding field in the internal data.

The third form is:

```
<link-field internal="DATA_KEY">
```

As with the first form, the external data source has a unique identifier but instead of using it as the Integrity identifier, it is stored in the internal field named **DATA\_KEY**.

Finally, the fourth form is:

```
<link-field external="REQ_ID" internal="DATA_KEY">
```

With this form, the external data field named **REQ\_ID** is linked with the internal data field named **DATA\_KEY**.

---

**Attributes:****external**

The **external** attribute specifies the name of a field in the external data source.

**internal**

The **internal** attribute specifies the name of a field in the internal data.

**field-type**

The **field-type="id"** attribute specifies that a field is an **id** field.

**hash-code**

Some external data sources (for example, Oracle databases) do not allow database look-ups using text strings, and only allow look-ups using integers. To accommodate such sources, you can specify the **hash-code** attribute which specifies a field name in which a hash code based on the identifier in question is stored.

**Parent Element:** <mapping>, <map>

**Sub-Elements:** none

**<field>**

The <field> element defines the mapping between fields in the external data source and fields in the internal data. The simplest version of this tag names only the external and internal fields. For example:

```
<field external="Type" internal="RQ_Share Category"/>
```

**Attributes:****external**

The **external** attribute specifies the name of a field in the external data source.

**internal**

The **internal** attribute specifies the name of a field in the internal data.

**direction**

The **direction** attribute specifies the direction of the relationship between the external data field and the internal data field.

direction	Description
in	The internal data field is updated from the external data field.
out	The external data field is updated from the internal data field.
both	The external and internal data field can both be updated from the other, depending on context.
none	This is a special value that indicates that all mappings which have been declared so far are to be ignored and not applied.

When the direction attribute is not specified, the behavior is as though **direction=both** was specified.



---

## field-type

The **field-type** attribute identifies what type of content the field holds. Possible values include:

field-type	Description
<b>type</b>	Integrity item type
<b>relationship</b>	Integrity relationship field
<b>attachment</b>	An attachment
<b>richcontent</b>	Rich content (that is, formatted text) possibly with links and graphics
<b>date</b>	Date with no time
<b>date-time</b>	Date and time

## data-type

The **data-type** attribute determines how the data in the field is to be interpreted. Possible values include:

data-type	Description
<b>xhtml</b>	Specifies that rich content in a field is in XHTML as opposed to an Integrity Rich Content format.

## attachment

The **attachment** attribute is used when mapping fields containing rich content to identify the Integrity attachment field that should be used to store images embedded in the rich content.

## external-date-format

The **external-date-format** attribute specifies the date format used in the external data source. For example, **EEE MM/dd/yy**.

This attribute is only meaningful if **field-type="date"** or **field-type="date-time"** is also specified; otherwise, it is ignored.

## clobber

The **clobber** attribute defines how existing attachments associated with the field are handled. The clobber attribute is only meaningful when **field-type="attachment"** is also specified; otherwise, it is ignored.

The **clobber** attribute can be set to **true** or **false**.

clobber	Description
<b>true</b>	All attachments for the field are replaced with the associated version of the attachment regardless of whether or not the attachment has change.
<b>false</b>	Existing attachments for the field are not removed and only updated attachments are modified.

When no **clobber** attribute is specified, the behavior is as though **clobber=false** was specified.

---

## on-create-only

The `on-create-only` attribute can be set to `true` or `false`.

on-create-only	Description
<code>true</code>	The field is only populated on its initial creation. If the field already exists, nothing is done.
<code>false</code>	The field is populated whether or not it already exists.

## value-translation-type

In some cases, data in the corresponding external and internal fields are stored in two different ways. For example, the external field may contain a range of integer values, while the internal field simply contains string values of `Low`, `Medium`, and `High`. The `value-translation-type` attribute specifies the relationship in such cases.

value-translation-type	Description
<code>string-string</code>	A string value in the external field corresponds to a different string value in the internal field.
<code>intrange-string</code>	A range of integers in the external field corresponds to a string value in the field.
<code>integer-integer</code>	An integer value in the external field corresponds to a different integer value in the internal field.

When the `value-translation-type` attribute is specified, you must also specify a `<value-translation>` sub-element which defines the exact method of translating from one to another.

## comparable

The `comparable` attribute has no meaning to the mapping process itself; however, it is used by the Gateway Wizard to help determine the set of fields displayed in the differencing and changes preview prior to starting the import process. Valid values include:

comparable	Description
<code>true</code>	The field is to be included in the differencing and changes preview.
<code>false</code>	The field is not to be included in the differencing and changes preview.

**Parent Element:** `<mapping>`, `<map>`

**Sub-Elements:** `<default>`, `<value-translation>`

### `<default>`

The `<default>` element specifies a value that is assigned to a field when either only an external or internal field name is named or the internal or external field from which it should be updated does not exist.

The following example sets the `RQ_Category` field in the internal data to a value of `Heading` when the internal data items are being first created.

```
<field internal="RQ_Category" direction="in" on-create-only="true">
  <default>Heading</default>
</field>
```

---

**Attributes:** none

**Parent Element:** <field>

**Sub-Elements:** none

### <value-translation>

When a <field> element has a **value-translation-type** attribute attached, the <value-translation> element defines which values from the external data source correspond to values in the internal data.

The following example shows how this can be used to define the correspondence between ranges of integers in the external field and a single string value in the internal field.

```
<field external="Priority" internal="RQ_Priority" direction="both"
value-translation-type="inrange-string">
  <value-translation external="0..250" internal="Low" />
  <value-translation external="251..500" internal="Medium" />
  <value-translation external="501..750" internal="High" />
  <value-translation external="751..1000" internal="Critical" />
</field>
```

**Attributes:**

#### **external**

The **external** attribute defines which value (or range) in the external field corresponds to the value or (range) in the internal field as defined by the **internal** attribute. The type of value (or range) is specified by the **value-translation-type** attribute of the parent <field> element.

#### **internal**

The **internal** attribute defines which value (or range) in the internal field corresponds to the value or (range) in the external field as defined by the **external** attribute. The type of value (or range) is specified by the **value-translation-type** attribute of the parent <field> element.

**Parent Element:** <field>

**Sub-Elements:** none

### <map-conditional>

The <map-conditional> element specifies that the mapping of a field is conditional on the value of a field or its properties. Based on the value of the field or property, a set of specialized mapping rules (defined by a <map> element) is activated.

In its simplest form, the value of the specified field or property is used as the name of the <map> to be activated. For example:

```
<map-conditional property="prototype" />
```

examines the value of the **prototype** property and looks for a <map> element with a name that matches the value of **prototype**. If found, that <map> is activated and the specialized mapping rules defined within that <map> are applied. Using this example, if the value of the **prototype** property is **CONTENT** and a <map name="CONTENT"> element exists, that <map> is activated.

---

Once a `<map-conditional>` element has activated a `<map>`, all other `<map-conditional>` elements at that level are ignored. For example:

```
<map-conditional external="Description" />
<map-conditional property="prototype" />
```

first looks for a `<map>` that has a name that matches the value of the external field name `Description`. If such a `<map>` is found, it is activated and the second `<map-conditional>` is ignored. However, if no such `<map>` is found, the second `<map-conditional>` is examined to see if there is a `<map>` that matches the name of the `prototype` property.

If the field or property whose value is being compared to `<map>` names does not exist in the current item, the `<map-conditional>` is ignored.

#### Attributes:

##### external

The `external` attribute specifies the name of a field in the external data source whose value is tested to find a matching `<map>` element.

##### internal

The `internal` attribute specifies the name of a field in the internal data whose value is tested to find a matching `<map>` element.

##### property

The `property` attribute specifies an item control property whose value is tested to find a matching `<map>` element.

##### direction

The `direction` attribute specifies that a matching `<map>` is only to be activated if the direction of the relationship between the external data source field and the internal data field matches the specified value. Possible values include:

direction	Description
in	The internal data field is updated from the external data field.
out	The external data field is updated from the internal data field.
both	The external and internal data field can both be updated from the other, depending on context.

For example:

```
<map-conditional property="prototype" direction="in">
```

only activates a matching `<map>` for the `prototype` value if the internal field is being updated from the external field. That is, it is only activated if the mapping is an import operation.

##### default-map

The `default-map` attribute specifies the name (as defined by a `<map>` element) of a set of specialized mapping rules that are to be applied when the value being matched to `<map>` names matches no `<map>` elements and has no other recognized value (that is, it is a null value or its value is not represented in any `<map-translation>` sub-element).

**Parent Element:** `<mapping>`, `<map>`

**Sub-Elements:** `<map-translation>`

---

## <map-translation>

The <map-translation> element is a sub-element of the <map-conditional> element. It lets you specify a list of values (that do not match a <map> name) for the field or property being examined in the <map-conditional> and associate them with a <map> element. The specified <map> is activated if the examined field or property has the given value.

For example:

```
<map-conditional external="Category">
  <map-translation external="Software" map-name="Application" />
  <map-translation external="Utility" map-name="Application" />
</map-conditional>
<map name="Application">
  ...
</map>
```

In this case, if the value of the external field named `Category` is `Application`, then <map name="Application"> is activated. If the value is not `Application`, the <map-translation> sub-elements come into play and if the value of the `Category` field is either `Software` or `Utility`, then the <map name="Application"> is also activated.

When no `external` attribute is specified in a <map-translation> sub-element, then any non-null value for the field or property specified in the parent <map-conditional> element which has not already been matched activates the specified <map>.

For example:

```
<map-conditional external="Category">
  <map-translation external="Software" map-name="Application" />
  <map-translation external="Utility" map-name="Application" />
  <map-translation map-name="Non-Application" />
</map-conditional>
<map name="Application">
  ...
</map>
<map name="Non-Application">
  ...
</map>
```

In this example, if the value of the `Category` field is either `Application` or `Non-Application`, the <map> element with that name is activated. If the field has another value, the <map-translation> sub-elements specify that a value of `Software` or `Utility` for that field activates <map name="Application"> and any other non-null value activates <map name="Non-Application">.

You can only specify one <map-translation> sub-element without an external attribute per <map-conditional> element.

### Attributes:

#### external

In a <map-translation> element, the external attribute does not refer to the name of a field in the external data source, but rather it specifies a possible value for the field or property in the

---

parent `<map-conditional>` element that is being matched to `<map>` names. In the example, the `external` attributes in the `<map-translation>` elements specify possible values for the external field named `Category`.

### **map-name**

The `<map-name>` attribute specifies the name (as defined by a `<map>` element) of a set of specialized mapping rules that are to be activated when the field or property being matched in the parent `<map-conditional>` has a value that matches the value specified by the `external` attribute.

When no `external` attribute is specified and the field or property is being matched has a non-null value, the specified `<map-name>` is activated regardless of the value of that field or property. There can only be one such `<map-translation>` sub-element per `<map-conditional>`.

**Parent Element:** `<map-conditional>`

**Sub-Elements:** none

### **`<map>`**

The `<map>` element defines a specialized set of mappings (using `<field>` elements) that are only performed when activated by a `<map-conditional>` element. When a `<map>` element is not activated.

You can nest `<map>` elements, allowing you to specialize the mappings to be performed even further.

For example:

```
<map-conditional property="prototype"/>
...
<map name="CONTENT">
  <map-conditional external="Category"/>
  ...
  <map name="Software Requirements">
  ...
</map>
  <map name="Hardware Requirements">
  ...
</map>
</map>
```

In this example, the `prototype` property of the item is examined and the main level of the Gateway Configuration file (the level where `<map-conditional property="prototype"/>` resides) is searched looking for a `<map>` with a name that matches the value of the `prototype` property. Thus, if `prototype` has a value of `CONTENT`, the `<map name="CONTENT">` element is activated. Within that `<map>`, the `<map-conditional external="Category">` element examines the value of the external data source field named `Category`, which in this example, identifies a category of requirement and looks to activate a `<map>` with a name matching the value of `Category`. Such a matching `<map>` must exist at the same level as that `<map-conditional>` within the `<map>` element. In this case, the `<map name="Software Requirements">` or `<map name="Hardware Requirements">` is activated if the `Category` field has the value `"Software Requirements"` or `"Hardware Requirements"`, respectively.

---

By using nested <map> elements, you can create complex AND/OR structures for your conditionals. For example, the above example represents the following logic:

- If the `prototype` is "CONTENT" AND the `Category` field is "Software Requirements", use the `Software Requirements` mappings.
- If the `prototype` is "CONTENT" AND the `Category` field is "Hardware Requirements", use the `Hardware Requirements` mappings.

**Attributes:**

**name**

The `name` attribute specifies a name for this particular set of mapping rules. This is the name specified with the `default-map` attribute of the <map-conditional> element or the `map-name` attribute of the <map-translation> element.

**Parent Element:** <mapping>, <map>

**Sub-Elements:** <set-property>, <link-field>, <field>, <map-conditional>, <map>

## Mapping Template Example

The following mapping template shows how the various XML elements work together to configure the Microsoft Excel integration to run with Integrity ALM 2009.

The sample template includes mapping components for Document and Content:

```
<?xml version="1.0" ?>
<mapping name="Microsoft Excel Requirements Document" template-version="2.1">
  <!--
    Sample Mapping configuration for ALM 2009 Requirement Documents.
    The ALM_ prefix on the internal field names will need to be changed to
    match the prefix used on the ALM 2009 solution installation.
  -->
  <map-conditional property="prototype" />
  <!-- Linkage field where Excel is storing the Integrity Item ID
  -->
  <link-field external="Item_ID" field-type="id"
    required="false" />
  <!-- Note the default value should be changed to reflect server and solution
  settings -->
  <field internal="Type"
    on-create-only="true"
    field-type="type">
    <default>ALM_Requirement</default>
  </field>
  <!-- Note that the Project field's external name should not be changed and the
  internal name should only be changed to reflect server and solution settings -
  -->
  <field external="Project"
    internal="Project"
    on-create-only="true">
```

---

```

    <default>/Projects/Release2</default>
</field>

<!-- Note that the Description field's external name should not be changed and
the internal name should only be changed to reflect server and solution
settings -->

<field external="Description"
    internal="ALM_Text"
    on-create-only="false" />

<!-- Source Document / Segment Root -->
<map name="DOCUMENT">

    <!-- Attribute is required so Excel will recognize this mapping as a
document mapping -->

    <set-attribute name="rmtype" value="segment"/>

    <!-- override the global Type field -->
    <field internal="Type"
        on-create-only="true"
        field-type="type">
        <default>ALM_Requirement Document</default>
    </field>

    <!-- Set the Shared Category value for a Document -->
    <field internal="Shared Category"
        on-create-only="false">
        <default>Document</default>
    </field>

    <!-- Set a default Document Title -->
    <field internal="ALM_Document Short Title"
        on-create-only="true">
        <default>Excel Created Document</default>
    </field>

    <!-- Note that the Description field's external name should not be changed
and the internal name should only be changed to reflect server and solution
settings -->

    <!-- override the global Description field -->
    <field external="Description"
        internal="ALM_Shared Text"
        on-create-only="false" />

</map>

<!-- Document Content -->
<map name="CONTENT">

    <!-- Set a default Category value for document content. -->
    <field external="Category" internal="Category" on-create-only="false">
        <default>Comment</default>
    </field>

</map>
</mapping>

```





# Index

## A

- active change packages
  - IBM Eclipse Platform 22, 33
  - Microsoft Visual Studio SDK 61
- active item
  - IBM Eclipse Platform 33
- adding members
  - CodeGear Delphi 12
  - IBM Eclipse Platform 36
  - Microsoft Visual Basic 104
  - Microsoft Visual C++ 108
  - Microsoft Visual Studio .NET 99
  - Microsoft Visual Studio SDK 78
  - Rational Rose 51
  - Sybase PowerBuilder 166
- adding solution
  - Microsoft Visual Studio .NET 88
  - Microsoft Visual Studio SDK 70
- administrator, described 2
- annotated subprojects
  - IBM Eclipse Platform 21
- attachments
  - HP Quality Center 181
  - images in Excel 159
  - Word 143

## B

- before you start 3
- best practices
  - IBM Eclipse Platform 46
  - Microsoft Visual Studio .NET 88
  - Microsoft Visual Studio SDK 81
- Borland Delphi (see CodeGear Delphi Architect)
- Borland JBuilder (see CodeGear JBuilder)

## C

- CA Endevor Change Manager 185
- Change Package view
  - Implementer 35
- change package, active
  - IBM Eclipse Platform 33
  - option 22
  - Microsoft Visual Studio SDK 61
- check in, members
  - CodeGear Delphi 12
  - IBM Eclipse Platform 37
  - Microsoft Visual Basic 105
  - Microsoft Visual C++ 109
  - Microsoft Visual Studio .NET 100
  - Microsoft Visual Studio SDK 79
  - Rational Rose 52
  - Sybase PowerBuilder 167
- check out, members
  - CodeGear Delphi 12
  - IBM Eclipse Platform 37
  - Microsoft Visual Basic 105
  - Microsoft Visual C++ 108
  - Microsoft Visual Studio .NET 100
  - Microsoft Visual Studio SDK 79
  - Rational Rose 52

- Sybase PowerBuilder 167
- client requirements
  - Microsoft Excel 146
  - Microsoft Project 112
- CodeGear Delphi Architect integration 8
  - adding members to project 12
  - checking in members 12
  - checking out members 12
  - commands 9
  - commit browser 10
  - configuring 9
  - considerations 9
  - creating project 11
  - creating Sandbox 11
- CodeGear JBuilder
  - general information 14
  - see also IBM Eclipse Platform
- column sets
  - configuring in Eclipse/Worktray 34
- commit browser
  - CodeGear Delphi Architect 10
- comparing revisions
  - IBM Eclipse Platform 41
- conditional mappings 189
- configuring
  - CodeGear Delphi Architect 9
  - HP Quality Center 173, 178
  - IBM Eclipse Platform 3.4 19
  - IBM Eclipse Platform 3.5+ 20
  - IBM Eclipse Worktray 34
  - IBM Rational Rose 50
  - integrations 3
  - Integrity for Quality Center 176
  - mapping template 188
  - Microsoft Excel 149
  - Microsoft Project 119
    - Integrity custom fields 120
    - Integrity custom properties 119
    - logging 122
  - Microsoft Visual Studio SDK 56
- conflict detection
  - Microsoft Excel 159
  - Microsoft Project 128
- considerations
  - CodeGear Delphi Architect 9
  - HP Quality Center
    - attachments 181
    - config files 179
    - running 183
  - Microsoft Excel 146
  - Microsoft Project 113
  - Microsoft Visual Studio SDK 55
- creating
  - project
    - CodeGear Delphi 11
    - HP Quality Center 174
    - IBM Eclipse Platform 25
    - Microsoft Visual Basic 104
    - Microsoft Visual C++ 108
    - Microsoft Visual Studio .NET 98
    - Microsoft Visual Studio SDK 70

- Sybase PowerBuilder 166
- project sets, IBM Eclipse Platform 28
- Sandbox
  - CodeGear Delphi 11
  - IBM Eclipse Platform 25
  - Microsoft Visual Basic 104
  - Microsoft Visual C++ 108
  - Microsoft Visual Studio .NET 98
  - Microsoft Visual Studio SDK 70
  - Sybase PowerBuilder 166
- custom DSD, creating 135
- custom fields
  - HP Quality Center 174

**D**

- date fields
  - Microsoft Excel 159
  - Microsoft Project 117, 123
- date format
  - Microsoft Excel 152
  - Microsoft Project 123
- deactivate integrations, with Integrity 4
  - IBM Eclipse Platform 23
- Delphi Architect (see CodeGear Delphi Architect integration)
- detecting conflicts
  - Microsoft Excel 159
  - Microsoft Project 128
- development paths
  - Microsoft Visual Studio .NET 92
- Document Structure Definition
  - custom 135
  - importing 138
  - importing, local 139
- Document view
  - HP Quality Center 177
  - Microsoft Word 132, 142
- drop member
  - Microsoft Visual Studio SDK 78
- dropped integrations 3
- dropping members
  - IBM Eclipse Platform 36
- DSD
  - custom 135
  - importing 138
  - importing local 139
  - storage locations (Word) 138

**E**

- Eclipse (see IBM Eclipse Platform integration)
- e-mail notification
  - general 2
  - HP Quality Center 181
- enabling integrations
  - for IDE 3
  - IBM Eclipse Platform 3.4 19
  - IBM Eclipse Platform 3.5+ 20
  - Microsoft Visual Studio SDK 56
- Endevor (see CA Endevor Change Manager)
- Excel (see Microsoft Excel integration)
- exporting
  - Word documents 141

**F**

- fields, Excel
  - date 159
  - special 159

- formulas, Excel 159

## G

- glyphs
  - Microsoft Visual Studio SDK 62

## H

- History view
  - IBM Eclipse Platform 41
- HP Quality Center integration 169
  - adding user fields 176
  - assumptions 171
  - attachment handling 181
  - configure 178
  - configuring
    - Document view 177
    - Integrity 176
    - Quality Center 173
  - considerations
    - attachments 181
    - config files 179
    - running 183
  - creating project 174
  - custom fields 174
  - e-mail notification 181
  - hs\_err\_pid.log 184
  - ID field 174–175
  - installation overview 171
  - installing 177
  - log files 183
  - mapping config XML files 180
  - mapping/config files 179, 180
  - overview 170
  - process config XML files 180
  - process files 179, 180
  - QCDataMapper.log 181
  - requirements user fields 175
  - setting Type 176
  - system requirements 171
  - troubleshooting 183

- hs\_err\_pid.log 184

## I

- IBM Eclipse Platform integration 17
  - active change packages 33
  - active item 33
  - adding members to project 36
  - adding projects to source control 25, 27
  - advanced Integrity commands 37
  - annotated subprojects 21, 31
  - best practices 46
  - checking in members 37
  - checking out members 37
  - comparing revisions 41
  - configuring 19
  - configuring column sets 34
  - creating
    - project sets 28
    - projects 25
    - Sandboxes 25
  - displaying Integrity information 32
  - dropping members from project 36
  - enabling for version 3.4 19
  - enabling for version 3.5+ 20
  - enabling on Linux 19
  - History view 41
  - icon decorations 31

- Implementer 35
- importing project sets 29
- importing projects 26
- installing 19
- Integrity commands 36
- Integrity Worktray 34
- label decorations 30
- limitations 47
- Linux 19
- online/offline mode 24
- preferences 21
- pre-steps 18
- project sets
  - sharing projects 28
- refactoring 40
- removing
  - projects 27
- setting up
  - integrated workspace 25
  - preferences 21
- Sharing Wizard 25, 27
- submitting changes 37, 38
- supported versions 18
- Synchronize Wizard 44
- Synchronizer 42
- synchronizing project 42
- Team Ignored directories 29
- team synchronizing 42
- unassociating
  - projects 27
- uninstalling 23
- upgrading/downgrading locks 33
- using 23
- Worktray view 34
- IBM Rational Rose integration 49
  - adding members to project 51
  - browser 50
  - checking in members 52
  - checking out members 52
  - commands 50
  - configuring 50
  - creating project 51
  - creating Sandbox 51
- icon decorations
  - IBM Eclipse Platform integration 31
- ID field, HP Quality Center
  - creating 174
  - displaying 175
- Implementer
  - Change Package view 35
- Implementer (see Implementer)
- importing
  - DSD (Word) 138
  - Enterprise Resources, Project 121
  - local DSD (Word) 139
  - project
    - Visual Studio SDK 72
  - project sets, IBM Eclipse Platform 29
  - projects, IBM Eclipse Platform 26
  - solution, Visual Studio SDK 70
  - template map
    - Excel 150, 153
- installing
  - HP Quality Center 177
  - IBM Eclipse Platform 3.4 19
  - IBM Eclipse Platform 3.5+ 20
  - IDE integrations 3
  - Microsoft Excel 147
  - Microsoft Project 117
  - Microsoft Word 134
- integrated workspace
  - IBM Eclipse Platform 25, 30
- integrations, general
  - configuring 3
  - deactivating 4
  - dropped 3
  - new 3
  - pre-steps 5
  - supported 3
  - tips 6
  - using 3
- Integrity SCM integrations
  - CodeGear Delphi Architect 8
  - CodeGear JBuilder 14
  - deactivating 4
  - enabling 3
  - IBM Eclipse Platform 17
  - IBM Rational Rose 49
  - Microsoft Visual Basic 103
  - Microsoft Visual C++ 107
  - Microsoft Visual Studio SDK 54
  - Sybase PowerBuilder 164
- Integrity Server
  - configuration
    - HP Quality Center 172
    - Microsoft Excel 147
    - Microsoft Word 133
  - requirements
    - HP Quality Center 171
    - Microsoft Excel integration 145
    - Microsoft Project 112
- Integrity Server (see Integrity Server)
- Integrity Synchronizer
  - IBM Eclipse Platform 42
- Integrity workflow integrations
  - CA Endeavor Change Manager 185
  - HP Quality Center 169
  - integrated workspace 30
  - Microsoft Excel 144
  - Microsoft Project 110
  - Microsoft Word 130
- Integrity Worktray
  - IBM Eclipse Platform 34
  - column sets 34
  - configuring 34
- Integrity Worktray view
  - IBM Eclipse Platform 34
- Integrity, roles 2
- item, active
  - IBM Eclipse Platform 33
- Items view (see MKS Items view)
- items, Visual Studio SDK
  - creating 67
  - editing 67
  - refreshing queries 67
  - re-running queries 67
  - viewing 67

## J

- Java API libraries
  - Eclipse on Linux 19
- JBuilder (see CodeGear JBuilder)

## K

- keywords
  - Microsoft Visual Studio SDK 58

## L

- label decorations
  - IBM Eclipse Platform 30
- limitations
  - IBM Eclipse Platform 47
  - Microsoft Excel 146
  - Microsoft Project 113
- Linux, IBM Eclipse integration 19
- locks, IBM Eclipse
  - upgrading/downgrading 33
- log files
  - excelint.log 152, 159
  - excelintc.log 152
  - hs\_err\_pid.log 184
  - mksvsi.log 84
  - MSWord-APISession.log 143
  - projint.log 122
  - QCDataMapper.log 181
- logging
  - HP Quality Center 183
  - Microsoft Excel 152
  - Microsoft Project 122
  - Microsoft Visual Studio SDK 84
  - Microsoft Word 143

## M

- mapping levels 189
- mapping template
  - conditional mappings 189
  - levels 189
  - overview 189
  - XML elements 190–199
- mapping\_name.txt file (Word) 135, 137
- master projects
  - Microsoft Visual Studio .NET 89
- MCFG file, Word 136, 139
  - extracting 137
  - importing 138
  - re-packaging 137
- member, adding
  - CodeGear Delphi 12
  - IBM Eclipse Platform 36
  - IBM Rational Rose 51
  - Microsoft Visual Basic 104
  - Microsoft Visual C++ 108
  - Microsoft Visual Studio .NET 99
  - Microsoft Visual Studio SDK 78
  - Sybase PowerBuilder 166
- member, checking in
  - CodeGear Delphi 12
  - IBM Eclipse Platform 37
  - IBM Rational Rose 52
  - Microsoft Visual Basic 105
  - Microsoft Visual C++ 109
  - Microsoft Visual Studio .NET 100
  - Microsoft Visual Studio SDK 79
  - Sybase PowerBuilder 167
- member, checking out
  - CodeGear Delphi 12
  - IBM Eclipse Platform 37
  - IBM Rational Rose 52
  - Microsoft Visual Basic 105
  - Microsoft Visual C++ 108
  - Microsoft Visual Studio .NET 100
  - Microsoft Visual Studio SDK 79
  - Sybase PowerBuilder 167
- member, dropping

- IBM Eclipse Platform 36
- Microsoft Visual Studio SDK 78
- member, moving
  - Microsoft Visual Studio SDK 79
- member, renaming
  - Microsoft Visual Studio SDK 79
- Mercury Quality Center (see HP Quality Center)
- Microsoft Excel integration 144
  - client requirements 146
  - configuring server 133, 147
  - considerations 146
  - creating new list 154
  - custom properties 151
  - customizing 149
  - date fields/formats 159
  - date format 152
  - detecting conflicts 159
  - excelint.log 152, 159
  - excelintc.log 152
  - formulas 159
  - get items 154
  - image attachments 159
  - importing
    - template map 150, 153
  - installing 147
  - Integrity Server requirements 145
  - logging 152
  - mapping templates 149
  - publishing data 155
  - removing 148
  - repairing 148
  - retrieving items 154
  - retrieving/publishing data 156
  - setting server connection 147
  - setting up
    - Integrity Server 147
  - special fields 159
  - SSL communications 151
  - synchronize 155
  - synchronize with query 156
  - system requirements 145
  - uninstalling 148
  - using 152
- Microsoft Project integration 110
  - client requirements 112
  - components 112, 146
  - configuring 119–123
  - considerations 113
  - custom project fields 120
  - custom properties 119
  - customizing Integrity 115
  - date fields 117, 123
  - date format 123
  - detecting conflicts 128
  - Enterprise Resources 121
  - importing Enterprise Resources 121
  - installing 117
  - Integrity Server requirements 112
  - logging 122
  - MKS Integration ID 119
  - Project Server 121
  - projint.log 122
  - removing 119
  - repairing 119
  - setting up 114
    - integration template 116
    - Integrity 115
    - permissions 116

- SSL communications 121
- synchronizing
  - all tasks 126
  - by query 126
  - linked tasks 127
  - resources 126
  - selected tasks 127
- system requirements 111
- task relationships 125
- task types 124
- uninstalling 119
- using 123
- Microsoft Visual Basic integration 103
  - adding members to project 104
  - checking in members 105
  - checking out members 105
  - creating project 104
  - creating Sandbox 104
- Microsoft Visual C++ integration 107
  - adding members to project 108
  - checking in members 109
  - checking out members 108
  - creating Integrity SCM project 108
  - creating Sandbox 108
- Microsoft Visual Studio .NET integration 86
  - adding members to project 99
  - adding solution 88
  - best practices 88
  - checking in members 100
  - checking out members 100
  - creating
    - project 98
    - Sandbox 98
  - development paths 92
  - implementing 87
  - joining solution 98
  - master projects 89
  - optimistic locking 91
  - parallel development 91
  - preferences 90
  - reusing code 88
  - solution root 87
  - troubleshooting 101
- Microsoft Visual Studio SDK integration 54, 66
  - active change packages 61
  - adding files 78
  - adding members 78
  - adding project to shared solution 70
  - advanced commands 80
  - best practices 81
  - branching solution 75
  - checking in members 79
  - checking out members 79
  - checkpointing solution 77
  - configuring 56
  - creating items 67
  - creating Sandbox 70
  - default solution location 58
  - defining top-level projects 59
  - dropping members 78
  - dropping project from shared solution 71
  - editing items 67
  - enabling plug-in 57
  - file status 64
  - glyphs 62
  - ignoring entities 74
  - importing project 72
  - importing solution 70
  - incoming changes 64
  - keywords 58
  - limitations 83
  - managing change packages 64
  - migrating solution from SCC 73
  - MKS Items view 66
  - mksvsi.log 84
  - moving members 79
  - non-solution files 65
  - online/offline mode 60
  - preferences 58
  - pre-steps 55
  - project/solution location 59
  - refreshing queries 67
  - removing change package entries 65
  - renaming members 79
  - re-running queries 67
  - resynchronizing solution 76
  - reverting solution 76
  - setting up 56
  - sharing solution 67
  - solution root 82
  - toolbar 67
  - troubleshooting 84
  - unassociated changes 65
  - viewing items 67
  - viewing Sandbox 77
  - Work In Progress 63
  - Work In Progress view
    - displaying 63
- Microsoft Word integration 130
  - active document export 141
  - assumptions 131
  - custom DSD 135
  - Document view 132, 142
  - DSD storage locations 138
  - exporting documents 141
  - importing
    - DSD 138
    - local DSD 139
  - importing DSD 138
  - importing local DSD 139
  - installing 134
  - integration errors 143
  - logging 143
  - mapping\_name.txt file 135, 137
  - MCFG file 136, 139
    - extracting 137
    - importing 138
    - re-packaging 137
  - MSWord-APISession.log 143
  - name.txt file 135, 137
  - overview 131
  - process-items.xlst file 135, 141
  - rich content support 131
  - setting server connection 133
  - setting up
    - Integrity Server 133
  - static.txt file 135, 137, 139
  - system requirements 133
  - troubleshooting 143
  - understanding XML mapping 140
  - uninstalling 135
- MKS Integration ID
  - Microsoft Project 119
- MKS Items view
  - displaying 66
- Microsoft Visual Studio SDK

toolbar 67  
MKS Work In Progress view (see Work In Progress view)  
MKS Worktray  
Microsoft Visual Studio SDK (see MKS Items view)  
moving member  
Microsoft Visual Studio SDK 79

## N

name.txt file (Word) 135, 137  
new integrations 3

## O

online/offline mode  
IBM Eclipse Platform 24  
Microsoft Visual Studio SDK 60  
optimistic locking  
Microsoft Visual Studio .NET 91

## P

parallel development  
Microsoft Visual Studio .NET 91  
plug-ins  
Microsoft Visual Studio SDK 57  
PowerBuilder (see Sybase PowerBuilder)  
preferences, setting  
IBM Eclipse Platform 21  
active change package 22  
annotated subprojects 21  
drop sandbox 22  
Microsoft Visual Studio .NET 90  
Microsoft Visual Studio SDK 58  
process-items.xlst file (Word) 135  
understanding 141  
project  
creating  
CodeGear Delphi 11  
HP Quality Center 174  
IBM Eclipse Platform 25  
IBM Rational Rose 51  
Microsoft Visual Basic 104  
Microsoft Visual C++ 108  
Microsoft Visual Studio .NET 98  
Microsoft Visual Studio SDK 70  
Sybase PowerBuilder 166  
removing  
IBM Eclipse Platform 27  
synchronizing  
IBM Eclipse Platform 42  
unassociating  
IBM Eclipse Platform 27  
Project (see Microsoft Project integration)  
project sets, IBM Eclipse Platform  
creating 28  
importing 29  
sharing 28  
projects, IBM Eclipse Platform  
importing 26

## Q

QCDataMapper.log 181  
Quality Center (see HP Quality Center integration)

## R

Rational Rose (see IBM Rational Rose)  
refactoring  
IBM Eclipse Platform 40  
removing

Microsoft Excel 148  
Microsoft Project 119  
Microsoft Word 135  
removing  
project  
IBM Eclipse Platform 27  
renaming member  
Microsoft Visual Studio SDK 79  
repairing  
Microsoft Excel 148  
Microsoft Project 119  
requirements, Quality Center  
adding user fields 176  
user fields 175  
rich content, Word 131  
embedded images 132  
embedded objects 132  
fonts 132  
headings 132  
hyperlinks 132  
lists 132  
tables 132  
roles, when using Integrity 2

## S

Sandbox  
creating  
CodeGear Delphi 11  
IBM Eclipse Platform 25  
IBM Rational Rose 51  
Microsoft Visual Basic 104  
Microsoft Visual C++ 108  
Microsoft Visual Studio .NET 98  
Microsoft Visual Studio SDK 70  
Sybase PowerBuilder 166  
setup  
Integrity  
HP Quality Center 176  
Microsoft Project 115  
Integrity Server  
HP Quality Center 172  
Microsoft Excel 133, 147  
Microsoft Word 133  
Microsoft Project 114  
integration template 116  
permissions  
Microsoft Project 116  
shared subprojects  
Microsoft Visual Studio .NET 88  
Sharing Wizard 25, 27  
shortcut menu 66  
solution  
Microsoft Visual Studio .NET 87, 88  
Microsoft Visual Studio SDK  
adding project 70  
branching 75  
checkpointing 77  
dropping project from shared 71  
importing 70  
migrating from SCC 73  
resynchronizing 76  
reverting 76  
sharing 67  
viewing Sandbox 77  
solution root  
Microsoft Visual Studio .NET 87  
Microsoft Visual Studio SDK 82  
special fields, Excel 159

---

SSL communications  
  Microsoft Excel 151  
  Microsoft Project 121

static.txt file (Word) 135, 137  
  understanding 139

submit changes  
  IBM Eclipse Platform 37, 38

supported integrations 3

Sybase PowerBuilder integration 163  
  adding members to project 166  
  checking in members 167  
  checking out members 167  
  creating  
    project 166  
    Sandbox 166  
  enabling Integrity 166

Synchronize Wizard 44

synchronizing  
  IBM Eclipse Platform  
    project 42  
  Microsoft Excel  
    data 155  
    with query 156  
  Microsoft Project  
    all tasks 126  
    by query 126  
    linked tasks 127  
    resources 126  
    selected tasks 127

system requirements  
  HP Quality Center 171  
  Microsoft Excel 145  
  Microsoft Project 111  
  Microsoft Word 133

**T**

task relationships  
  Microsoft Project 125

task types  
  Microsoft Project 124

team synchronization  
  running 44  
  symbols 43  
  Synchronize Wizard 44

team synchronizing  
  IBM Eclipse Platform 42

Test Director (see HP Quality Center)

toolbar  
  Implementer 35  
  Integrity Worktray, Eclipse 34  
  MKS Items view 67  
  toggling 57

troubleshooting  
  HP Quality Center 183  
  Microsoft Visual Studio .NET 101  
  Microsoft Visual Studio SDK 84

Microsoft Word 143

## U

unassociating  
  project  
    IBM Eclipse Platform 27

uninstalling  
  IBM Eclipse Platform 23  
  Microsoft Excel 148  
  Microsoft Project 119  
  Microsoft Word 135

user, described 2

## V

Visual Basic (see Microsoft Visual Basic)  
Visual C++ (see Microsoft Visual C++)  
Visual Studio .NET (see Microsoft Visual Studio .NET)  
Visual Studio SDK (See Microsoft Visual Studio SDK)

## W

WebSphere (see IBM Eclipse Platform integration)  
Word (see Microsoft Word integration)

Word documents  
  exporting 141  
  importing DSD 138  
  importing local DSD 139

Work In Progress view  
  displaying 63  
  file status 64  
  incoming changes 64  
  managing change packages 64  
  Microsoft Visual Studio SDK 63  
  non-solution files 65  
  removing change package entries 65  
  shortcut menu 66  
  unassociated changes 65

Worktray  
  Implementer Change Package view 35

Worktray (see Integrity Worktray)

Worktray view  
  Eclipse (see Integrity Worktray view)  
  Visual Studio (see MKS Items view)

## X

XML files  
  HP Quality Center  
    mapping config 180  
    process config 180

XML mapping file, Word 140

XML template  
  conditional mappings 189  
  elements 190–199  
  levels 189  
  overview 189