PTC Integrity™ Upgrading Guide

PTC Integrity 10.9

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

Contents

1

Upgrading to Integrity 10.9

This document contains information that you want to know before upgrading to Integrity 10.9. The information is relevant for upgrades from Integrity 2009 SP7 and Integrity 10.0 through 10.9.

Note

- Integrity 2009 SP7 is the minimum supported version when upgrading a source-only database to Integrity 10.9 that contains no Integrity relationship data.
- Integrity 10.7 is the minimum supported version that you can upgrade automatically (including database migration) to Integrity 10.9. Integrity servers before release 10.7 cannot be upgraded automatically. If your version of Integrity is earlier than 10.7, contact PTC - Integrity Support for assistance.
- Integrity 10.6 extended localization support on the client and server from English and Japanese to three additional languages: German, Chinese Simplified, and Chinese Traditional. However, PTC does not currently support or recommend changing the server language. For example, you cannot change the server language from English to Chinese Simplified.
- Integrity 10.8 introduces a capability for the administrator to localize their configuration and add translation strings for the display name and description attributes of the administrative objects **Fields**, **Test Result Fields**, **Types**, and **States**. During an upgrade of an existing server to Integrity 10.8 and later, the display name and description of the administrative objects (standard set and custom created) are stored in the server locale. The server locale cannot be changed during an upgrade. For example, if an administrator installs an Integrity 10.7 server and earlier, using English as the installer locale, then during an upgrade to Integrity 10.8 and later, the display name and description attributes of the administrative objects **Fields**, **Test Result Fields**, **Types**, and **States** are stored in the English locale.

  For detailed information about Configuring Localization, see the *PTC Integrity Server Administration Guide*.

For the most up-to-date upgrading information made available to you since the publication of this document, see article CS229118 in the Integrity Support Center.

For the most current product platform support information, go to http://www.ptc.com/partners/hardware/current/support.htm.

For the most current Knowledge Base articles, go to http://www.ptc.com/support/integrity.htm.

For detailed information about configuring the supported databases for the Integrity server, see the *PTC Integrity Server Administration Guide*.

For detailed information about new features, general notes, fixed issues, and known issues in Integrity 10.9, see the *PTC Integrity Release Notes*.

Your Upgrade Path

Before upgrading, PTC recommends reviewing this entire guide. Depending on your current version of Integrity, your upgrade path may require a full install using an executable or you may be able to upgrade using a service pack install.

If you are upgrading to Integrity 10.9, you must upgrade using the full install executable. Prior to upgrading to Integrity 10.9, the relationship data from the old relationship table must already have been migrated to the new relationship table using a 10.7 or 10.8 Integrity server. For more information, see Database Considerations on page 41.

A full install using the executable is required when upgrading to Integrity 10.9 from the following product versions:

- Integrity 2009 SP7
- Integrity 10.0
- Integrity 10.1
- Integrity 10.2
- Integrity 10.3
- Integrity 10.4
- Integrity 10.5
- Integrity 10.6
- Integrity 10.8

This guide describes the process for installing using the full install executable.

Before You Start

Before you begin your upgrade to Integrity 10.9, review the following important information:

General Considerations on page 10

Integrity Product Version Considerations on page 12

Document Versioning Considerations on page 26

Database Considerations on page 41

General Considerations

Have a backup plan

Develop a plan to use in case an upgrade fails. By retaining the previous version of the server, you reduce downtime if the upgrade fails. Retaining the previous version of the server allows you to restore the database and then contact PTC - Integrity Support for assistance.

Backup and verify your database

Perform a full database backup and keep the backup on site until it is determined that the upgrade is successful. Verify the database backup to ensure it can be restored if required.

Consider integrated components

Consider the dependencies between integrated components when upgrading. Both Integrity and Implementer can refer to items and change packages that are stored in Integrity. You must keep the Integrity data in sync forIntegrity and Implementer data to accommodate these dependencies.

For example, if upgrading problems cause you to restore Integrity from a weekly backup,Integrity and Implementer could end up referring to items that do not exist or contain out-of-date information.

Do not reuse backed up configuration files

If you upgrade using the full install executable, note that old configuration files cannot be reused with the new release. Copying and pasting files from the old installation into the new server directory causes problems if configuration properties have been renamed or new variables added. A utility is provided for migrating configuration files.

In addition, many properties are now contained in the database. Once properties are successfully migrated to the database with the `isutil` utility, they can be further modified in the Integrity Administration Client GUI or from the CLI using the `integrity setproperty`, `si setproperty`, `im setproperty`, and `aa setproperty` commands; or `im diag` and `si diag` commands.

> **Note**
>
> The documentation files pointed to in `documentationlist.properties` and `installationlist.properties` must be manually updated from the installation DVD and their locations manually updated in those files. For more information, see the *PTC Integrity Server Administration Guide*.

Plan to migrate Custom Certificates

If you upgrade using the full install executable, note that custom CA Root certificates stored in the `cacerts` keystore are not migrated during the upgrade. You must plan to import such certificates manually after the upgrade.

If you do not migrate the certificates, the Integrity server can fail to start due to an inability to establish the proper certificate trust.

Always stop the Integrity server service before installing the upgrade

Stop the Integrity server service, but do not uninstall the Integrity server until after the new server is installed. Retaining the old installation ensures that a rollback is possible if you run into difficulties with the upgrade.

Integrity Product Version Considerations

New installation directories

For Integrity server 10 product versions, the default installation directory is:

```
installdir/Integrity/IntegrityServer10
```

In addition, the installer prevents you from installing the server in an existing installation directory.

For the Integrity client, the default installation directory is:

```
installdir/Integrity/IntegrityClient10
```

PTC recommends reviewing and updating any scripts you use that reference the installation directories.

RCS repository is no longer supported in Integrity 10.0 and later versions

As of Integrity 10.0, the legacy RCS repository is no longer supported for configuration management. To support a more robust repository type for configuration management,Integrity exclusively supports the database repository. For more information, see the *PTC Integrity Server Administration Guide*.

Note

- If you are currently using an RCS repository for configuration management, you must migrate to a database repository in Integrity 2009 SP7 before upgrading to Integrity 10.0 and later versions.
- If you are currently using a database platform that was supported by the RCS repository in previous releases of Integrity but is not currently supported by the database repository, you must migrate to a supported database platform.
- If a database migration is not an option or you require assistance, contact PTC - Integrity Support.

⚠ **Caution**

If you are upgrading an Integrity server that was originally configured for the RCS repository, ensure that your new database is configured as case sensitive. Contact PTC - Integrity Support for assistance.

With the mandatory use of the database repository, the property for `mksis.backend=db` is now required on the Integrity server. The `mksis.backend` property is configured in the following properties file on the server:

`*installdir*/config/properties/is.properties`

If `mksis.backend` is missing or incorrectly set, the Integrity server does not start and an error occurs (*MKS141153 The RCS repository is no longer supported for Source*). To resolve the error, the administrator must add `mksis.backend=db` to the `is.properties` file.

For more information on configuring properties in the `is.properties` file, see the *PTC Integrity Server Administration Guide*.

Integrity 10.8 and later does not support PTC System Monitor 3.0

Integrity 10.8 and later is only compatible with PTC System Monitor (PSM) 4.0. It is not supported with PSM 3.0 and earlier PSM releases. This is due to an incompatibility with the version of Java that is used with both products. This incompatibility does not allow the Integrity server to start. If you are currently running PSM 3.0 and earlier, you must upgrade to 4.0 before upgrading to Integrity 10.8 and later.

Project files are now in client-side database

As of Integrity 10.8 and later, project information is stored in a client-side database, and consequently there are no `.pj` files in Sandboxes. Project files still display as virtual project files (with the `.pj` file extension) in Integrity interfaces. Instead of project files, project information is stored in a client-side database in the `.mks` directory of the system on which the Integrity client is installed. The location of the `.mks` directory is specified by the `MKS_IC_INSTANCE_DIR` environment variable. By default, on Windows the `.mks` directory can be located in the home directory of the user.

Note

An Integrity user's `.mks` directory must have sufficient space available to fit three copies of the client-side database. The amount of space needed depends on how many Sandboxes the user has, and a minimum of 50 MB available space is recommended.

Large amounts of Test Results data in Oracle databases can increase Integrity 10.6 upgrade time (275480, 950629)

If you are using an Oracle database that contains a large amount of Test Results data, the database migration step of the Integrity server upgrade can take a long time. During the upgrade, a message warning of this appears and provides instructions to look for status updates in the migration log (`dbinstall.log`) found in the server installation log directory.

Java Garbage Collector default change (142919, 943799)

As of Integrity 10.5 and later, the Java garbage collector default for the Integrity server has changed to concurrent mark sweep collector (CMS). If your implementation of Integrity uses custom changes to the garbage collector settings, it is recommended that you test the new Integrity server installation in a test environment before upgrading the Integrity server used in the production environment.

Upgrading database to Integrity 10.3 can take additional time (740029, 807473)

In Integrity 10.3, three new columns have been added to the Issues table. During the database migration step of the Integrity upgrade, for each existing item (issue), a value is assigned to one of the columns; possibly increasing upgrade time. The columns have been added for future use.

Trigger bean updates to support deactivating and activating development paths (1010427, 1071406)

In Integrity 10.8 and later, the following updates have been made:

- The following new trigger bean classes are available: `ScriptActivateVariantArgumentsBean` and `ScriptDeactivateVariantArgumentsBean`. These classes make the variant name being activated available to script authors.
- For the `ScriptProjectBean` class, the following new methods are available: `deactivateVariant` and `activateVariant`. These methods deactivate and activate variant projects.
- For the `ScriptProjectBean` class, the `getVariants` method has been updated to return both active and inactive variants. If you only want

active variants to be returned, you must update your scripts to use `getActiveVariants` instead.

- `getActiveVariants`, `getInactiveVariants`

For more information, see the Javadocs.

Methods in `ScriptCopyTreeResultBean` deprecated (631048, 203946)

As of Integrity 10.1 and later, the following methods in the `ScriptCopyTreeResultBean` are deprecated and should no longer be used:

- `getResultMap()`
- `getResultingIssue()`

Instead, use the following methods:

- `getResultMapV2()`
- `getResultingIssues()`

For more information on using `ScriptCopyTreeResultBean` methods, see the "Event Trigger Java Documentation" documentation link on the Integrity server Homepage.

Change to Document ID Field Definition (494359, 517828)

With Integrity 10.1 and later, the definition of the Document ID field is not editable by Integrity administrators. If your implementation of Integrity has a change from the default field definition, the field is changed to the default during the database migration step of the upgrade, with a warning in the migration log file `*installdir*/log/dbinstall.log`.

An example of a warning follows:

The computation definition of the “Document ID” field is incorrectly
set to “isMeaningful()”, it has been changed to “DocumentID()” The
'how to run computation' attribute of the “Document ID” field is
incorrectly set to “static”, it has been changed to “dynamic” The
'store to history' attribute of the “Document ID” field is incorrectly
set to “weekly”, it has been changed to “never”

For further assistance, contact PTC Integrity Support.

Document model enforced when using event triggers (412912, 496076)

Integrity 10.2 introduces the following changes that enforce the document model for change triggers:

- A branch of the backing shared item is caused by a change trigger edit to either the Author or Reuse node of any shared item that is one of the following:
  - Referenced either by an Author node and at least one Reuse node

- ○ Referenced by two or more Reuse nodes (no Author is required in this second case

- Change triggers cannot perform a significant edit on a node in Share mode. If an attempt is made, the following error is returned:

  Changes are not permitted when the \"Reference Mode\" field is set to \"Share\".\n \n Trigger script \"{0}\" invoked by trigger \"{1}\" attempted to modify field \"{2}\" of item #{3}. Please notify the Integrity Administrator about this error.
  Ensure that your implementation of change triggers does not rely on the previous behavior.

Change to command output for `im viewfield` (404859, 735922)

With Integrity 10.2, the output for the `im viewfield` command has changed to include system managed fields. Scripts that use this command require updating.

New Integrity server property enabled by default (683801)

With Integrity 10.2, the following Integrity server property is introduced and enabled by default: `mksis.includeIntegrityGUILinks`. For more information, see entry 683801 in the *Integrity Release Notes*, and the *PTC Integrity Server Administration Guide*.

Method in `ScriptSourceTraceBean` deprecated (727249)

As of Integrity 10.3, the `getFileName()` method in the `ScriptSourceTraceBean` is deprecated. Although the method currently functions, it will no longer be actively supported in future product releases. For full support, it is recommended that you use the following methods instead:

- `getMemberName()`
- `getSubprojectName()`
- `getProject()`

For information on using `ScriptSourceTraceBean` methods, including the limitations of the `getFileName()` method, see the "Event Trigger Java Documentation" documentation link on the Integrity server Homepage.

Removed methods in com.mks.api.IntegrationPointFactory class (919859)

In Integrity 10.4, the following methods are removed from the `com.mks.api.IntegrationPointFactory` class in Integrity Java API 4.12:

- `public static String getAPIVersion()`
- `public IntegrationPoint createLocalIntegrationPoint() throws APIException`

- `public IntegrationPoint createIntegrationPoint(String host, int port) throws APIException`
- `public IntegrationPoint createIntegrationPoint(String host, int port, boolean secure) throws APIException`

In releases before Integrity 10.4, these methods were deprecated and instructions were provided to migrate to other methods where you would be required to explicitly request an API version. If you currently use these methods in a custom integration, PTC recommends reviewing the Java docs for Integrity Java API 4.12 and updating your integration code accordingly.

If you currently use these methods in any trigger scripts, the scripts do not work with Integrity 10.4 and later. PTC recommends reviewing your trigger scripts for any use of the Java API's `IntegrationPointFactory`. Consider revising your scripts to use the trigger's `ScriptEnvironmentBean.createAPISessionBean` instead of directly using the Java API within your scripts.

Note

For further information and instructions, see article CS135671 in the Integrity Support Center at http://www.ptc.com/support/integrity.htm

Support for Integrity client 10.3 (and later) Help when the Integrity server is upgraded (826302)

If your implementation of Integrity includes Integrity 10.3 (and later) Integrity clients connecting to newer version Integrity servers, PTC recommends considering one of the configuration options in the "Integrity Help Backward Compatibility" topic in the *PTC Integrity Server Administration Guide*.

Installing FlexNet and a Java Runtime Environment

To control the use of Integrity components, FlexNet Publisher License Server 11.10.1 is distributed with Integrity 10.2 and later. This version of the FlexNet Publisher License Server installer does not include a Java Runtime Environment (JRE).

You must install JRE 1.5 or higher before installing FlexNet. Download the JRE from http://www.java.com. For more information, refer to the FlexNet Publisher documentation included with the FlexNet installer.

Clients connecting to a proxy server can view Item Presentation Template images residing behind a firewall (775983, 912068)

In Integrity 10.5 and later, an Integrity client connecting to a proxy Integrity server can now view Item Presentation Template (IPT) images residing on the main Integrity server or a network location that is inaccessible by clients, due to network restrictions, such as a firewall. To transfer IPT images across the firewall, images are cached on the proxy server and Java Remote Method Invocation (RMI) calls are performed between the client, proxy server, and main server.

If you use Integrity clients 10.4 and earlier to connect to proxy and main Integrity servers 10.5 and later, the IPT images display as broken image icons. To resolve this issue, contact PTC - Integrity Support.

Move Subproject command supports changing the path name case on a case-insensitive database (109145, 669648)

When moving a subproject on a case-insensitive database, Integrity clients 10.5 and later can now change the path name case. For example, `Test/project.pj` can be changed to `test/project.pj`.

This functionality is not available for Integrity clients 10.4 and earlier. However, earlier clients can pick up path name case changes made by 10.5 and later clients. To pick up a path name case change with earlier clients, do the following:

1. Resynchronize the affected Sandbox.
2. Manually change the case of the subproject path name on the file system.

If users do not make these changes, the affected project name in wizard panels and title bars can be rendered incorrectly.

For Integrity clients 10.5 and later, PTC recommends resynchronizing the affected changes in your Sandbox before performing additional operations. This ensures that you are working with the latest changes and reduces potential configuration errors.

Note

Users cannot change the case of the subproject path name with transactional change packages or Change Package Reviews enabled. If transactional change packages or Change Package Reviews are enabled, users must drop the subproject in a change package, manually change the case of the subproject path name, and then use a new change package to add the subproject as a shared subproject.

Updated `ScriptAPISessionBean` uses Integrity API 4.11 for local and remote API calls (939795)

Before Integrity 10.5, the `ScriptAPISessionBean` used the most current version of the Integrity API. In Integrity 10.5 and later, if the `ScriptAPISessionBean` is used for an API call within the local server or for remote API sessions, the `ScriptAPISessionBean` now uses Integrity API 4.11. This allows a 10.5 and earlier server to make local and remote API calls through the `ScriptAPISessionBean` in event triggers into 10.6 servers and later. Using the same Integrity API version for local and remote API sessions also ensures that trigger script authors can create trigger scripts that do not need to account for multiple versions of the Integrity API responses. When working with event triggers and the API, PTC recommends making note of the API version difference.

For more information, refer to the Java docs and the *PTC Integrity Integrations Builder Guide*.

Test result metadata fields of the Date data type represented in the API as a datetime value (939795)

Before Integrity 10.5, the API representation of a test result metadata field incorrectly returned a `string` value instead of a `datetime` value for the Date data type. In Integrity 10.5 and later, all versions of the Integrity API correctly return a `datetime` value.

Compatibility of Item Presentation Templates in Integrity 10.6 and later (966764)

If you open an Integrity 10.5 and earlier item presentation template (IPT) in an Integrity 10.6 and later release, Integrity adds a header, populated with the `Item Created Information` and `Item Modified Information` read-only fields. If there is a logo in the IPT, it is placed in the header, based on the logo alignment property. Integrity also applies the default text style to fields in the header.

If you open an Integrity 10.6 and later IPT in Integrity 10.5 and earlier, Integrity ignores the read-only fields in the header and moves any custom fields to the first tab in the IPT. As of Integrity 10.6, Integrity no longer include a logo property for IPTs because header layout can be customized just like the layout of any tab. This allows you to add more than one image anywhere in the header.

Enhanced project visibility for project administrators (961444)

To provide enhanced project security, project administrators can only view and edit the projects to which they are assigned. For example, visible projects display in the Integrity Administration Client GUI > **Projects** node and when using the `im projects` command.

To reflect this enhancement from the CLI, the `im dynamicgroups` and `im editdynamicgroups`commands contain the following added options and updated option behavior.

- `im dynamicgroups --fields=`*membership* displays only the projects to which the project administrator is assigned.

  To indicate that a project does not have any groups and users as members of the dynamic group, specify the new `nomembers` keyword. For example, specify `--membership=/ Project=nomembers`. Previously, a blank space indicated that a project did not have any groups and users as members of the dynamic group. For example, the following was previously acceptable:`--membership=/ Project=` .

⚠ Caution

If you are a project administrator, the `im dynamicgroups--fields= membership` command displays a subset of the projects in your Integrity configuration. If a super administrator uses that list of projects when specifying `im editdynamicgroup--membership`, the membership for the projects that the project administrator does not have permission to view are removed.

- `im editdynamicgroups --membership` processes only the projects to which the project administrator is assigned.

  To indicate that a project does not have any groups and users as members of the dynamic group, specify the new `nomembers` keyword. For example, specify `--membership=/ Project=nomembers`. Previously, a blank space indicated that a project did not have any groups and users as members of the dynamic group. For example, the following was previously acceptable: `--membership=/ Project=`.

  To inherit the membership from the parent project to the dynamic group, specify the `inherit` keyword. For example, specify `--membership=/Project=inherit`. Previously, a membership list was specified.

  To allow a project administrator to set membership for a specific project to which he or she is assigned, use the `--projectMembership=`*project*`= inherit|nomembers|`*per-project-membership* option, where:

- *per-project-membership* is in the form *user-list*|*group-list*|*user-list*:*group-list*

where:

- *user-list* is in the form `u=`*username*`[`*,username*`]`
- *group-list* is in the form `g=`*groupname*`[`*,groupname*`]`
  - `inherit` specifies that the membership for the parent project is to be inherited by the dynamic group.
  - `nomembers` specifies that the project does not have any groups and users as members of the dynamic group

Note

Specifying users but no groups removes any existing groups. Similarly, specifying groups but no users removes any existing users.

To correctly work with dynamic groups, PTC recommends using the options provided with these commands. For more information, see the 10.6 and later version of the CLI man pages.

If a project administrator is using a 10.5 and earlier Integrity Administration Client, the GUI and CLI commands return results based on 10.5 and earlier behavior.

Configuration Options For Non-Build Subprojects When Creating a Development Path (972187, 972193)

When creating a development path from a project that contains non-build subprojects (normal or variant subprojects), the options are configurable from the Integrity client. However, an administrator can override and enforce a specific option on the Integrity server.

If one of these options is enforced on a 10.6 and later Integrity server and a 10.5 and earlier Integrity client creates a development path, the default behavior is enforced. All subprojects that are configured differently from the parent project retain their existing configuration.

Improperly using `im dynamicgroups` or `im editdynamicgroup` can result in lost membership data (96020, 148955, 976751)

The `--fields=membership` and `--membership` options were designed so that a script could update project membership for a dynamic group by retrieving the membership for all projects with `im dynamicgroups`

`--fields=membership`, making changes to the list, and then returning the list to `im editdynamicgroup --membership`.

In Integrity 10.6 and later, the `im dynamicgroups --fields=membership` and `im editdynamicgroup --membership` commands have been changed so that they only return or edit the memberships for projects that the project administrator has rights to administer. For example, if a user is a project administrator, the list of members returned from `im dynamicgroups --fields=membership` only includes the projects that the project administrator is allowed to administer. Similarly, if a project administrator attempts to edit membership using `im editdynamicgroup --membership`, Integrity only updates those projects that the project administrator can administer.

With these changes in Integrity 10.6 and later, scripts should not call `im dynamicgroups --fields=membership` as a project administrator and then use the returned list to call `im editdynamicgroup --membership` as a different project administrator. If a project is missing from the list passed to `im editdynamicgroup --membership`, Integrity sets that project's membership to inherit its parent project's membership. This means that using these two commands with different project administrators causes the memberships for some projects to inherit from their parent project, losing their membership data if the two project administrators administer a different set of projects.

Revision description audit tags in text files (976757)

In Integrity 10.6 and later, revision annotation `--- Added comment ---` tags are replaced with `– Added comment –` tags. As a result, users can see additional differences when differencing files from a Sandbox that contains the `$Log: IntUpgradeGuideStartProdVersion.dita $` keyword. These differences that contains the `Revision 1.26 2015/07/24 20:59:10IST Flett, David (dflett)` keyword. These differences that contains the keyword. These differences that contains the `$Revision: 1.40 $` keyword. These differences that contains the keyword. These differences are resolved by updating the working file in the Sandbox. New tags use the format `– Added comment –`. Users do not see additional differences.

Creating, viewing, editing, copying, and deleting configuration management policies from the CLI (949198)

From the CLI, administrators can create, view, edit, copy, and delete configuration management policies using the following commands:

- `si copypolicysection`
- `si deletepolicysection`
- `si setpolicysection`

- `si viewpolicysection`
- `si viewpolicysections`

Note

The `si setpolicysection` command replaces the `si createpolicysection` command (a previously unsupported command that was removed in Integrity 10.6). To create and edit policies, use the `si setpolicysection` command and modify any existing scripts that refer to the `si createpolicysection` command.

These commands are also published commands supported by PTC for use with the Integrity Java API. For more information, see the *PTC Integrity Integrations Builder Guide*.

Using scripts or the Integrity Java API, administrators can automate the setup of configuration management policies. For more information on CLI commands, see the CLI man pages.

Note

To manage your configuration management policies, PTC recommends using the Integrity Administration Client.

Manually disable and lock global Change Package Description Template configuration management policy following an upgrade to Integrity 10.7 and later (1003803)

Following an upgrade to Integrity 10.7 and later, administrators who do not intend to configure Change Package Description templates should consider globally disabling and locking the **Change Package Description Template** policy to ensure optimal performance. The policy can be disabled and locked through the Integrity Administration Client policy editor. For more information on modifying server configuration management policies, see the *PTC Integrity Server Administration Guide*.

Deprecated options for the `im editissue` command (976630)

When using the `im editissue` command in Integrity 10.6 and later, the following options are deprecated and should no longer be used:

- `--addRelationships=`*value*
- `--removeRelationships=`*value*

Instead, use the newer options:

- `--addFieldValues=`*value*
- `--removeFieldValues=`*value*

In addition, the `--addRelationships=`*value* option is deprecated in the `im createissue` and `im copyissue` commands. Instead, use the newer `--addFieldValues=`*value* option.

Although the deprecated options continue to work, PTC does not recommend using them for new scripts because they will be removed in a future release. Existing scripts using the deprecated options should use the new options.

New method added to `ScriptDynamicGroupBean` (917243)

To check the user membership of a dynamic group based on the project, the `isUserMemberOf` method was added to `ScriptDynamicGroupBean` in Integrity 10.6. This updated method can be used in an event trigger to restrict the following:

- creating an item
- editing a project on an item (cannot change to specific projects)
- recording time entries
- editing test results
- label operations on items

For more information, see the "Event Trigger Java Documentation" documentation link on the Integrity server Homepage.

Referencing checkpoints in unregistered configuration management projects (982628)

In Integrity 10.6 and later, project paths are referenced using the repository location. This allows you to reference checkpoints in unregistered (dropped) configuration management projects. If a checkpoint is currently referenced and the corresponding project is later dropped, the checkpoint remains accessible as read-only. For example, you can continue to open the referenced checkpoint from an SI project field or create a build Sandbox from the checkpoint for auditing purposes.

Before upgrading, note the following:

- Although no existing data is migrated when you upgrade to Integrity 10.6 and later, project paths are referenced using the repository location, which can affect existing scripts and triggers. PTC recommends reviewing any existing scripts and triggers.
- Integrity 10.5 and earlier clients cannot view metrics links for SI Project fields that reference build projects created with Integrity 10.6 and later.
- If you are using different versions of the Integrity client and Integrity server and dedicated servers, the behavior is dependent on the version of the Integrity server dedicated to configuration management. For an Integrity server 10.5 and earlier that is dedicated to configuration management, a flat path is stored for build projects.

Custom integrations can possibly require extra steps for Integrity upgrade (999628)

If you are upgrading to Integrity 10.9 from 10.7 or 10.8, the following additional upgrade steps are not required. However, if you are upgrading to Integrity 10.9 from an earlier Integrity version other than 10.8 or 10.7, and you have a custom integration, additional steps for updating your integration can be necessary. You must perform these extra steps if all of the following criteria apply:

- Your environment uses a private copy of the Integrity C API
- Your environment uses the Integrity server as the integration point
- SSL is used for secure communication with the server integration point

If your environment meets all of these criteria, you must update your copy of the Integrity C API to the version provided with Integrity 10.9. If you are using a Microsoft Excel or Project integration, you must also update to the currently supported version of the integration software. For supported versions, see the latest Integrations document on the PTC website. If you do not upgrade the Integrity C API, the integration is unable to connect to the secure port of the Integrity server.

Shared Sandboxes not supported as of Integrity 10.8

Shared Sandboxes are not supported for Integrity 10.8 and later. After the Integrity client upgrade, only the owner of the Sandbox continues to have access to the Sandbox through the Integrity client. Other users who were sharing the Sandbox can no longer access that Sandbox.

Sandboxes used for the Staging and Deploy functionality no longer supported as of Integrity 10.8

In Integrity 10.8 and later, the Sandboxes used for the Staging and Deploy functionality are no longer supported. The migration of such Sandboxes from earlier versions of Integrity to Integrity 10.8 and later is also no longer supported.

Applying Integrity 10.8 service pack prevents restart of Integrity Agent 10.7

The Staging and Deploy functionality is no longer supported in Integrity 10.8 and later. If the Staging and Deploy functionality is enabled (`mksagent.startup.sd=true`) in the `agent.properties` file, the Integrity Agent fails to start. A `FATAL` category log message in the `agent.log` and `FATAL.log` files is also logged. The log message indicates that the Staging and Deploy functionality is no longer supported. Applying the Integrity 10.8 service pack to Integrity Agent 10.7 disables the Staging and Deploy functionality by updating an existing `mksagent.startup.sd=true` property to `mksagent.startup.sd=false` in the `agent.properties` file. This automatic disabling of Staging and Deploy functionality on the Integrity Agent is required to support remote automated patching.

Backup files generated by Integrity 10.8 during multiple-row editing are not compatible with Integrity 10.9

During multiple-row editing of a document, unsaved changes are stored in a backup file so that these changes can be recovered if an unexpected shutdown of the Integrity client occurs. The backup files generated by Integrity 10.8, which introduced a beta version of multiple-row editing, cannot be opened by Integrity 10.9. Before upgrading an Integrity client from 10.8 to 10.9, you want to ensure that all documents are saved or closed successfully. Otherwise, after the client is upgraded, you are unable to recover changes from a 10.8 backup file. Backwards compatibility is planned for releases subsequent to 10.9. For example, an Integrity 11.0 client will be able to open backup files from Integrity 10.9. Future releases will also support recovering pending imports, which are new in Integrity 10.9.

Document Versioning Considerations

Integrity 10.5 introduced enhanced document versioning capabilities.

If you are upgrading from Integrity 10.4 and earlier to Integrity 10.9, consider the following:

Integrity client support for document versioning (930902)

If you are working with document versions using an earlier Integrity client, it is possible for item operations not to work or to produce unexpected behavior.

To make full use of the document versioning capabilities introduced in Integrity 10.8, PTC recommends upgrading your Integrity client to 10.8 and later.

Hyperlinks in the Item Details view (930902)

Integrity 10.4 an earlier do not display hyperlinks to document or content version information in the header and **History** tab of the Item Details view. Earlier clients display **Created by** and **Modified by** information only.

Searching for live and versioned documents using the Document ID field (935528)

In the Integrity client GUI, the **Document ID** field supports searching for live and versioned documents. For example, typing `184` returns the contents of the live document. Typing `184–1.2` returns the contents of the versioned document.

Integrity clients 10.4 and earlier do not support searching for versioned documents using the **Document ID** field.

New item query filters (916659)

From the **Items** query filter in the GUI and Web interface, two new sub-filters are available: **Is versioned** and **Is live**. From the CLI, the sub-filters are: `item.versioned` and `item.live`.

After upgrading, existing queries display all items (versioned and live). To display live items only (pre-upgrade query behavior), modify the queries to include the **Item is live** filter.

Note

As a best practice, PTC recommends including the **Item is live** filter in all queries for live items. This improves the accuracy of query results.

New item ID query filters (933679)

For item ID query filters, you can specify the following conditions to display live and versioned items by item ID:

- Display a specific live item only. For example, **ID** = `123` and **Is live** returns live item `123`.
- Display a live item and all versions of the item. For example, **ID** = `123` and **include versions** returns `123, 123-1.0, 123-1.1`, and `123-1.2`.

- Display a range of live items. For example, **ID** > `123` and < `128` and **Is live** returns `124, 125, 126,` and `127`.
- Display a specific versioned item. For example, **ID** = `123-1.0` returns `123–1.0`.

Note

You cannot display a range of versioned item. For example, you cannot display **ID** > `123-1.0` and < `123-2.0`.

Queries containing versioned item IDs, the **Is live** query filter, or the **include versions** query filter are not visible in Integrity clients 10.4 and earlier. In addition, charts, reports, and dashboards containing those queries are not visible in earlier clients.

New document ID query filters (935528)

For document ID query filters in the GUI and Web interface, you can specify the following conditions to display content that is included in live and versioned documents:

- Display the contents of a live document. For example, **Document ID** = `123` and **Is live** returns content that is included in live document `123`.
- Display the content from a range of live documents. For example, **Document ID** > `123` and < `128` and **Is live** returns content that is included in live documents `124, 125, 126,` and `127`.
- Display the contents of a versioned document. For example, **Document ID** = `123-1.0` returns content that is included in versioned document `123–1.0`.

Note

You cannot display the content from a range of versioned documents. For example, you cannot display content from **Document ID** > `123-1.0` and < `123-2.0`.

Queries containing versioned document IDs or the **Is live** query filter are not visible in Integrity clients 10.4 and earlier. In addition, charts, reports, and dashboards containing those queries are not visible in earlier clients.

Updated document query filters and computed expressions (934500)

From the **Items** query filter in the GUI or Web interface, the following sub-filters are updated:

- **Is a document node** returns all items (live and versioned) that are a document model node type.
- **Is a content node** returns all items (live and versioned) containing references to shared items.

From the CLI, the sub-filters are: `item.node` and `item.content`.

After upgrading, existing queries display all items (versioned and live). To display live items only (pre-upgrade query behavior), modify the queries to include the **Item is live** filter.

For computed expressions, the following item functions are updated:

- `IsNode()` returns `true` if the items (live and versioned) are a document model node type.
- `IsContent()` returns `true` if the items (live and versioned) contain references to shared items.

Existing computed expressions include all items (versioned and live). To include live items only (pre-upgrade behavior), modify the computed expressions to include the `IsLive()` function.

Note

As a best practice, PTC recommends including the `IsLive()` function in all computed expressions for live items. This improves the accuracy of computations.

Updating fields in versioned items (922959, 929457)

Although a document version represents a record of a document at a specific point in the document's history, some fields in individual versioned items can continue to update based on the field definition. These are known as *live fields*, indicated by the live field icon .

The following fields always update based on the field definition:

- item backed picklist fields
- query backed relationship fields

- range fields (An icon displays if the referenced field is configured to update based on the field definition.)
- field value attribute fields (An icon displays if the referenced field is configured to update based on the field definition, which can include any of the above fields, a field on a live item, or where the backing relationship is editable.)

Note

The asterisk (*) icon displays based on the current field configuration, not the field configuration at the time of versioning. For example, changing a field configuration to be editable or allow updates after versioning displays an asterisk (*) icon in all item versions.

When creating or editing computed fields, an available option specifies how values are computed in versioned items: **Allow Computation Updates on Versioned Items** (GUI) and `[no]allowComputationUpdatesOnVersion` (CLI). By default, computation values on versioned items continue to update based on the computed field definition. To record computation values at the time of versioning and prevent further updates, clear the **Allow Computation Updates on Versioned Items** checkbox.

For existing computed fields, this checkbox is selected. Select or clear it based on your requirements.

Note

Before versioning a document containing a computed field, PTC recommends carefully considering whether you want the computed field to **Allow Computation Updates on Versioned Items**. If you change the option after versioning the document, it is possible for historical views of versioned items to not display the expected field value.

New item functions for computed expressions (938099)

For computed expressions, the following item functions are available:

- `IsLive()` returns `true` if item type node or segment is live.
- `IsVersioned()` returns `true` if item type node or segment is versioned.

After upgrading, existing computed expressions include all items (versioned and live). To include live items only (pre-upgrade behavior), modify the computed expressions to include the `IsLive()` function.

For example, to include live items only in the following existing computed expression:

```
aggregate("ALM_Documented By", sum(IsSegment() ? 1 : 0))
```

add the `IsLive()` function:

```
aggregate("ALM_Documented By", sum(IsSegment() and IsLive() ? 1 : 0))
```

Note

As a best practice, PTC recommends including the `IsLive()` function in all computed expressions for live items. This improves the accuracy of computations.

Displaying live or versioned items in reports (931016)

In report recipes, you can specify the following report tags to filter by live or versioned items:

- `<%filter%>(item is live)<%endfilter%>` filters by live items.
- `<%filter%>(item is versioned)<%endfilter%>` filters by versioned items.

After upgrading, existing reports display all items (versioned and live). To display live items only (pre-upgrade report recipe behavior), modify the report recipes to include the live items report tag filter.

For example, to include live items only in a report recipe similar to **Detail - HTML, Column, Relationships**, modify the `beginrelationshipsdetail` line to:

```
<%beginrelationshipsdetail &relationshipsdetailfields%><%filter%>
(item is live)<%endfilter%>
```

Note

If a report recipe does not include the appropriate filter, users can modify the backing query to include a query filter that displays live or versioned items.

Report recipes containing report tag filters for live and versioned item IDs display an error message in Integrity clients 10.4 and earlier.

Note

As a best practice, PTC recommends including the `(item is live)` filter in all report recipes for live items. This improves the accuracy of reports.

Displaying live fields and ambiguous computed fields in reports (939682)

The Integrity client GUI and Web interface include icons to indicate live fields (⚡) and ambiguous computed fields (⚠) in versioned items. However, reports cannot display these icons. To indicate whether field values in a report are live or ambiguous, the following report tags are available:

Report Tag	Description
`&fieldislive`	Displays `true` or `false` to indicate whether the field is live.
`&fieldisambiguous`	Displays `true` or `false` to indicate whether the field is an ambiguous computation.
`&relationshipfieldislive`	Displays `true` or `false` to indicate whether the relationship field is live.
`&relationshipfieldisambiguous`	Displays `true` or `false` to indicate whether the relationship field is an ambiguous computation.
`<%islive%>fieldname<%endislive%>`	Instead of displaying the field name, displays `true` or `false` to indicate whether the field is live.
`<%isambiguous%>fieldname<%endisambiguous%>`	Instead of displaying the field name, displays `true` or `false` to indicate whether the field is an ambiguous computation.
`<%relationshipisliveL#%>fieldname<%endrelationshipisliveL#%>`	Displays `true` or `false` to indicate whether the field is live for the item at the specified relationship level.
`<%relationshipisambiguousL#%>fieldname<%endrelationshipisambiguousL#%>`	Displays `true` or `false` to indicate whether the field is an ambiguous computation for the item at the specified relationship level.
`&fieldgroupcomputeislive`	If a field contains a defined group computed field, displays `true` or `false` to indicate whether the

Report Tag	Description
	computation is live. **Note** If there is no group computed field for the field, an empty string displays.
`&fieldgroupcomputeisambiguous`	If a field contains a defined group computed field, displays `true` or `false` to indicate whether the computation is ambiguous. **Note** If there is no group computed field for the field, an empty string displays.
`<%groupcomputeisambiguous%>`	Used within the `<%iterategroupcompute%>` report tag, displays `true` or `false` to indicate if the associated `<%groupcompute%>` is ambiguous.
`<%groupcomputeislive%>`	Used within the `<%iterategroupcompute%>` report tag, displays `true` or `false` to indicate if the associated `<%groupcompute%>` is live.

After upgrading, existing reports containing ambiguous computed fields do not display any data for the ambiguous computed fields. Live fields continue to display data for the live fields. Modify existing reports as needed to display appropriate data.

Integrity clients 10.4 and earlier ignore the new report tags in report recipes.

Report recipes that include the item ID field in JavaScript code (944738)

If document versioning is enabled and you have an existing report recipe that includes the item ID field in JavaScript code, you must put quotes around the item ID field.

For example:

```
<script type="text/javascript">
if(REPORT_ISSUES == "") REPORT_ISSUES = <%<%builtin ID%>%>;
else REPORT_ISSUES = REPORT_ISSUES+","+<%<%builtin ID%>%>;
section = new Array();
level = 0;
```

```
ReportID = <%<%builtin ID%>%>;
<script>
```

Modify to:

```
<script type="text/javascript">
if(REPORT_ISSUES == "") REPORT_ISSUES = "<%<%builtin ID%>%>";
else REPORT_ISSUES = REPORT_ISSUES+","+"<%<%builtin ID%>%>";
section = new Array();
level = 0;
ReportID = "<%<%builtin ID%>%>";
<script>
```

Defining rules for live and versioned items (945966, 938098)

When defining rules in Integrity, you can specify conditions for live and versioned document model items. For example, you can create an event trigger rule to run on versioned items only or an e-mail notification rule that sends an e-mail when a user edits a specific live item.

With items, you can:

- define a rule to match live items only. For example, **Item is live** matches live items only.
- define a rule to match versioned items only. For example, **Item is versioned** matches versioned items only.

After upgrading, existing rules display all items (versioned and live). To display live items only (pre-upgrade behavior), modify the rules to include the **Item is live** condition.

Note

As a best practice, PTC recommends including the **Item is live** condition in all rules for live items. This improves the accuracy of rules.

With item IDs, you can:

- define a rule using a live item ID to match a single live item. For example, **ID**= `123` and **Item is live** match `123`.
- define a rule using a versioned item ID to match a single versioned item. For example, **ID**= `123–1.0` matches `123–1.0`.

Note

- You cannot define a rule using a live item ID to match the live item and all versions of the item.
- You cannot define a rule using live or versioned item IDs to match a range of live or versioned items. For example, you cannot use **ID**> `123–1.0` and < `123-2.0` to match of range of live or versioned items.

With document IDs, you can:

- define a rule using a live document ID to match a single live document. For example, **Document ID**= `123` and **Item is live** match content that is included in live document `123`.
- define a rule using a versioned document ID to match a single versioned document. For example, **Document ID**= `123–1.0` matches content that is included in versioned document `123–1.0`.

Note

You cannot define a rule using live or versioned document IDs to match a range of live or versioned documents. For example, you cannot use **Document ID**> `123-1.0` and < `123-2.0` to match a range of live or versioned documents.

When defining a new rule with live or versioned item conditions from the Integrity Administration Client, a message displays that Integrity clients earlier than 10.5 cannot evaluate this rule correctly, potentially resulting in unexpected behavior.

Integrity API updates to computed field values in workflows and documents items (939795)

Integrity API 4.13 (introduced in Integrity 10.5) includes support for distinguishing literal null values from null values returned by ambiguous computed field values in workflows and documents items.

When viewing a workflows and documents item in previous Integrity releases, the API representation for a computed field value matched the representation of a non-computed field value for the same workflows and documents field data type. In Integrity API 4.13, computed field values are now represented by an API item of `im.Computation` model type containing the following API fields:

- `value` indicates the computation value
- `isAmbiguous` indicates the boolean value of whether the computation is ambiguous
- `isDynamic` indicates the boolean value of whether the computation is dynamic or static

Note

With this update, earlier versions of the Integrity API can return null values with no context for simple data types.

For more information on ambiguous computations, see the *PTC Integrity Server Administration Guide*.

Changes to `LocalTriggerManager.ScriptIssueDeltaBean.revision` method (941478)

With the introduction of document versioning functionality, the trigger method `ScriptIssueDeltaBean.revision` has been modified to perform a check-in operation. The following changes have been made to the associated parameters:

- The `conditionalSignificantEdit` parameter is ignored and internally set to true. Document model items are only checked in when they contain significant changes.
- The `conditionalModified` parameter is ignored and internally set to false. Document model items are only checked in when they contain significant changes.
- The `recurseInclude` parameter is ignored and internally set to true. All included documents are automatically checked in.
- The `recurseReference` parameter is ignored and internally set to true. All referenced documents are automatically checked in.
- The `recurse` parameter applies only to content items. If the selection is a segment, or a node that points to a segment, the option is ignored and internally set to true.

If you have any trigger scripts that currently use the `ScriptIssueDeltaBean.revision` method, PTC recommends that you review those scripts and update them as required.

Changes to `LocalTriggerManager.ScriptServerBean` class (945403)

With the introduction of document versioning functionality, the `LocalTriggerManager.ScriptServerBean` class has been modified to include the following new methods to look up the new versioning-related fields:

- `getLiveItemIDFieldBean` for the **Live Item ID** field
- `getMajorVersionIDFieldBean` for the **Major Version ID** field
- `getMinorVersionIDFieldBean` for the **Minor Version ID** field

If you have any trigger scripts that currently use the `LocalTriggerManager.ScriptServerBean` class, PTC recommends that you review those scripts and update them as required.

Computed fields and ambiguous computations (939364, 944630)

With document versioning, a computed field containing an ambiguous computation returns an empty value. If a computed field can contain an ambiguous computation, as a best practice, PTC recommends wrapping the expression in the `IsEmpty()` function. This ensures that any empty computed field values stored to history are as a result of an ambiguous computed expression.

From the CLI, the `!` decorator indicates that a computed field contains an ambiguous computation. To toggle the display of the `!` decorator, specify the `--[no]showDecorators` option for the following commands:

- `im exportissues`
- `im issues`
- `im viewissue`

By default, `--showDecorators` is specified for `im viewissue` and `--noshowDecorators` is specified for `im issues/im exportissues`. If you have any scripts that use these commands, PTC recommends that you review those scripts and update as required.

Creating, editing, and viewing hyperlinks to versioned items (940895)

Integrity clients 10.4 and earlier cannot create, edit, or view (click) hyperlinks to versioned items in the Item Details view.

Custom integrations and versioned item data (941917)

If you use the Integrity API to develop custom integrations, note the following about how API versions handle versioned item data:

- Integrity API 4.13 and later can include versioned items, as specified by an Integrity 10.5 and later query definition.
- Integrity API 4.12 and earlier excludes versioned items by default. The client/command removes versioned items from the query results. There is no change to operation of the query on the server, and there is no modification to the query definition sent to the server.

When you write new integrations or upgrade existing integrations, PTC recommends that you upgrade the API version to 4.13 and specify if you want to include versioned items by using an appropriate query definition. This ensures that your integrations use the appropriate item data.

Charts and ambiguous computed expressions (940753)

Charts do not display ambiguous computed expression values in versioned content item data. If a chart contains an ambiguous computed expression, an error message displays.

After upgrading, backing queries in existing charts display all items (versioned and live). Modify backing queries in existing charts as needed to display appropriate data. If you choose to include versioned content item data in charts, modify the chart definition to exclude computed expressions that refer to document context (**Document ID**, **Contains**, **Contained By** fields, or the `Aggregate ByDocument()` function). If you do not exclude these computed expressions, they can return ambiguous results, preventing the chart from running.

Changes to the `LocalTriggerManager.ScriptIssueBean` and `LocalTriggerManager.ScriptIssueDeltaBean` classes (946348)

With the introduction of document versioning functionality, the `LocalTriggerManager.ScriptIssueBean` and `LocalTriggerManager.ScriptIssueDeltaBean` classes were modified to include the `isFieldValueAmbiguous(String fieldName)` method to determine if a computed field value (in the context of a versioned item) is ambiguous.

`isFieldValueAmbiguous(String fieldName)` returns true if (in the context of a versioned item) the stored value for the provided computed field is null because of an ambiguous computation.

If you have any trigger scripts that currently use the `LocalTriggerManager.ScriptIssueBean` or `LocalTriggerManager.ScriptIssueDeltaBean` classes to inspect field values, PTC recommends reviewing and updating the scripts where appropriate to account for ambiguous computations.

Changes to the `LocalTriggerManager.ScriptIssueBean` class, `ScriptIssueBean.toString()` method, and `ScriptIssueDeltaBean.toString()` method (946174)

With the introduction of document versioning functionality, the `LocalTriggerManager.ScriptIssueBeanclass`, `ScriptIssueBean.toString()`, and `ScriptIssueDeltaBean.toString()` methods were updated to support e-mail notifications containing versioned items.

The `LocalTriggerManager.ScriptIssueBean` class was modified to add the `getIssueIDString()` method to retrieve the versioned item ID from an item.

The `ScriptIssueBean.toString()` and `ScriptIssueDeltaBean.toString()` methods for issue beans were modified to include the versioned item ID.

To use these new methods, the following trigger scripts included with Integrity 10.5 and later were updated:

- `email.js`
- `signatureRequired.js`
- `postLinkedIssue.js`
- `PromoteIssue.js`
- `javamail.js`
- `hello.js`
- `escalate.js`
- `emailAdvanced.js`
- `dependentStatus.js`

If you have any trigger scripts that currently use these methods or you are currently using the trigger scripts that were updated in Integrity 10.5 and later, PTC recommends reviewing and updating your scripts where appropriate.

Document versioning and the ALM solution (945333)

If you have an existing solution installed, such as the ALM solution, enabling document versioning can affect fields and admin objects in the solution. For example, it is possible for certain reports or triggers in the ALM solution not to work or to produce unexpected behavior.

Note

With the release of Integrity 10.6 and later, the ALM solution download is no longer maintained or available for download from the Integrity Support Center. Contact your PTC Account Representative to learn more about available ALM solution offerings from PTC.

For more information on known issues, browse the Knowledge Base at:

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

Some Knowledge Base articles indicate potential workarounds or fixes that can be made to correct known issues.

Specifying and Displaying Versioned Item IDs (926736)

Versioned item IDs cannot be specified or displayed for the following:

Integrity Component	Affected Area
Relationships View	When loading a branch in the Relationships view (GUI), the versioned item ID does not display. After the branch is loaded, the versioned item ID displays.
Viewing and Editing Items	Computed fields and range fields (if using an FVA to ID) do not display versioned item IDs.
Export Items to Microsoft Excel	`im exportissues` (CLI) and **Export Items to Excel** (GUI/Web) do not export versioned item IDs.
CLI Commands	You cannot specify versioned item IDs with the following CLI commands: • `im printissue` • `im propagatetraces` • `im branchsegment -insertLocation`, `-parentID` • `im issues -focusIssueID` • `im viewsegment -focusIssueID`, `-sessionID` • `im viewduplicates`
Visual Studio and Eclipse Integrations	Versioned items are not returned in the integrations.
Integrity Web Services	You cannot specify versioned item IDs for input and output does not display versioned item IDs.
Document Read and Review Application	You cannot specify versioned item IDs.

Logging	• Audit log does not display versioned item IDs. • Client and server logs display some versioned item IDs.
E-mail Notification	Relationship field values do not display versioned item IDs.

Database Considerations

Migrate relationship data to new database table prior to upgrade

Integrity 10.7 introduced a new database storage model and table for relationship data. Prior to upgrading to Integrity 10.9, the relationship data from the old relationship table must already have been migrated to the new relationship table using a 10.7 or 10.8 Integrity server.

Note

If the migration has not occurred prior to attempting an upgrade to Integrity 10.9, the upgrade does not succeed but the database is still usable to run the original version of Integrity. If you are performing a silent install, the following error is logged in `dbinstall.log`:

```
ERROR(0): Database migration aborted. The migration
to the IIDeltaMap table must be performed prior to
upgrading to Integrity 10.9 (and later). For migration
information, see the Integrity 10.8 version of the PTC
Integrity Upgrading Guide.
```

The new relationship table is more compact and grows at a significantly slower rate than the relationship table that it replaces. For more information on migrating relationship data, consult the 10.8 version of the *PTC Integrity Upgrading Guide*.

Grant `SELECT_CATALOG_ROLE` to the Oracle database user

For Oracle databases, PTC recommends granting `SELECT_CATALOG_ROLE` to the database user to ensure that the correct SQL logging and query plan retrieval permissions are available.

Case-sensitive collation for Microsoft SQL server databases

If you are using Microsoft SQL Server as your database, the Integrity server requires a case-sensitive collation. For example, you can use `SQL_Latin1_General_CP1_CS_AS`.

Oracle error in Document view

When working in the Integrity **Document** view, an error can occur after applying a field filter for a full-text field. The error (*java.sql.SQLSyntaxErrorException: ORA-00918: column ambiguously defined*) occurs only when the Integrity server is running on an Oracle database version earlier than 11.2.0.1. The Integrity client erroneously shows that the failed field filter is active when it is not.

The underlying cause of the error is related to a known Oracle issue (Doc ID 5702433.8, Bug 5702433 "ORA-918 from ANSI join with a CONTAINS clause"). The defect is addressed in the base version of Oracle 11g Release 2 (11.2.0.1).

Duration of an Integrity 2009 database migration

Migrating an Integrity 2009 database to Integrity 10 can require several hours depending on certain factors. During this time, the installer can appear to have made little progress. You can check the progress of the upgrade by viewing the `dbinstall.log` file located in the following directory on the server:

`installdir``/log/dbinstall.log`

For more information on factors that increase migration time, contact PTC - Integrity Support.

Duration of database migration

Database migrations can take longer than normal if the backing database is Microsoft SQL Server and the `IssueDeltas` and `IssueDeltaAtoms` tables contain a large amount of data (where either one of the tables is large or both tables are large collectively). The duration of the migration is dependent on the speed of your database server and the size of your data; so a test migration is recommended as part of planning the upgrade window.

SQL Server transaction log affects disk size allocation

If your implementation of Integrity is backed by a Microsoft SQL Server and the `IssueDeltaAtoms` and `IssueDeltas` contain a large amount of data, then a large amount of disk space is needed for the SQL Server transaction logs. Prior to the upgrade, test the upgrade on test servers to determine the size of the disk space that is needed for the SQL Server transaction logs. Then use that information to set the maximum size of the database transaction log. Also ensure that there is sufficient hard disk space size allocated where the transaction log resides.

Error when using Oracle 12c with Integrity 10.8

If your Integrity server is running on an Oracle 12c database, the `ORA-01792: maximum number of columns in a table or view is 1000` error may occur. This error is related to a known Oracle 12c issue (Doc ID

1951689.1, Bug 17376322 “Select Statement Throws ORA-01792 Error”). This defect is addressed by the Oracle patch 19509982.

Compatibility Support

The next several topics provide information about compatibility:

- Integrity Client and Server Compatibility on page 43
- Configuration Management Repository Compatibility on page 44
- Dedicated Integrity Server Compatibility on page 44
- Integrity Server and Integrity Agent Compatibility on page 45
- Integrity API Compatibility on page 45
- Implementer and Integrity Server Compatibility on page 45

Integrity Client and Server Compatibility

Integrity server 10.9 supports connections from Integrity client 2009 SP7 and 10.0 through 10.9.

If you are currently using an FSA-enabled proxy server and upgrading from an earlier release to Integrity 10.9, you do not have to perform the upgrade all at once. Integrity server 10.9 supports connections from a proxy Integrity server 2009 SP7 and an Integrity server for 10.0 through 10.9.

Note the following:

- Upgrade the components of the system in the following order:

  1. Workflows and Documents and Test Management-enabled servers
  2. Configuration Management-enabled servers
  3. FSA proxy servers
  4. Clients

- By default, the `servicepack.policy` file specifies the minimum Integrity client version and service pack that can connect to Integrity server 10.9. However, you can configure these values as new service packs are released. For more information on configuring the `servicepack.policy` file, refer to the *PTC Integrity Server Administration Guide*.
- Integrity supports the installation of multiple Integrity clients on a single machine. For example, a 10.9 client and a 10.8 client. This is useful for accessing functionality available in specific releases and connecting to different versions of the Integrity server. For more information, see the *PTC Integrity Server Administration Guide*.
- The Web interface version does not depend on the client version.

- ViewSets edited with a new Integrity client may be unusable on older clients, and will have an adverse impact on your users if those ViewSets are configured to be mandatory. For example, a ViewSet edited with a 10.9 Integrity Administration Client and published to a 10.9 server may only be usable for 10.9 clients. Older clients will continue to be able to use existing ViewSets that have not been edited by an 10.9 Integrity client. Particular care should be taken when working with mandatory ViewSets in a mixed version environment. A mandatory ViewSet edited with a new Integrity client may cause the older client to become unusable, due to new functionality introduced in newer releases.
- If some users who are working within a project will be using an older Integrity client, ensure that you do not deactivate development paths in those projects. Older clients do not interact correctly with deactivated development paths.
- Ensure the Integrity Administration Client and Integrity server versions match. Administrative operations are not supported when using an Integrity Administration Client that is a different version than the Integrity server version.
- Integrity 10.9 (and later) includes the **Ignore Keywords** policy that prevents keywords from being expanded and unexpanded in text files worked on by users of configuration management functionality. For Integrity 10.8 (and earlier) clients, this policy works for **Member ▸ Check In** and **Member ▸ Rename** operations. For all other operations that have keyword settings, those keyword settings apply, even when the policy is set.

Configuration Management Repository Compatibility

Updates of earlier versions of the configuration management database repository to later versions are handled automatically by the Integrity server installer, in the same manner that configuration management repositories are automatically updated.

The RCS-Style repository is no longer supported as of Integrity 10.0 and later.

Dedicated Integrity Server Compatibility

You can have multiple Integrity servers in your environment, each dedicated to specific functionality. A common practice is to have one server dedicated to Workflows and Documents functionality, and one or more servers dedicated to Configuration Management functionality.

The Workflows and Documents server is the central server and runs the current Integrity version. Any number of Configuration Management servers can be connected to the central Workflows and Documents server. Connected Configuration Management servers can run any mix of Integrity versions from 2009 SP7 through the current version.

Integrity Server and Integrity Agent Compatibility

Integrity server 10.9 supports connections from Integrity Agent 2009 SP7 and Integrity 10.0 through 10.9. If you are upgrading from an earlier release to Integrity Agent 10.9, you do not have to perform the upgrading all at once.

Integrity API Compatibility

Integrity provides an Application Program Interface (API) for integrating third-party products with Integrity. The API provides a framework to invoke Integrity commands and receive responses. Command compatibility via the API is constrained by the commands that are supported for the API and by the command changes, if any, that affect compatibility.

The following table identifies the supported compatibility configurations for API language bindings with Integrity integration points:

API Language	API Version	Released in Integrity Version
Integrity Java/C API	4.16	10.8
	4.15	10.7
	4.14	10.6
	4.13	10.5
	4.12	10.4
	4.11	10.0
	4.10	2009
Integrity Web Services API	10.2	10.2
	10	10.0
	9.7	2009 SP7
	9.0	2009 SP6

Implementer and Integrity Server Compatibility

Implementer integrates with Integrity to provide functionality for workflows and documents, configuration management, or both.

Supported Databases

Supported Databases

You can find information about supported databases in Integrity Product Platforms.

Dropped Database

The following database versions are no longer supported in Integrity 10.1 and later:

- Microsoft SQL Server 2005 [1]
- Oracle 10g R2
- Oracle 11g R1[2]
- IBM DB2 for Linux, Unix, and Windows (all versions)
- IBM DB2 iSeries (all versions)

⚠ **Caution**

Ensure database upgrades are performed before upgrading the Integrity server.

Database Migration

Database schema migrations are automatically handled during the installation process when you upgrade using the full install executable. During the installation, you select your database type and database connection information. You are then prompted to migrate the database schema you used with the existing Integrity for use in this release. For more information, see the *PTC Integrity Server Administration Guide*.

[1] Integrity 10.1 supports a minimum of Microsoft SQL Server 2008 SP3 or Microsoft SQL Server 2008 R2 SP1.

[2] Oracle 11g R1 is no longer supported in Integrity 10.8 and later.

Note

- The installer does not permit you to create new tables if it detects any existing Integrity tables. If there are existing data, your choices are to migrate the data or exit the installation.
- For Oracle databases, ORA-01555 errors can occur if you do not have sufficiently large rollback segments. Consult your Oracle product documentation for more information.
- The database migration eliminates duplicate ACLs in the following way:
  - If the ACL is a complete duplicate (same ACL name, same principal, same; permission name, and same grant/deny value), then the duplicate ACL is deleted.
  - If the ACL is a partial duplicate, such as conflicting values of grant/deny, then the ACL with the deny value wins.

If the migration fails, you must resolve the error, restore the database, and then attempt the migration again.

Note

If the migration fails due to a `Database Transaction Log Size Exceeded` error, you can correct the problem and then rerun the migration without first restoring the database.

Debugging and Diagnosing Database Issues

During the installation, the following file is created for debugging and diagnostic purposes when a database is upgraded:

`installdir/log/dbinstall.log`

where *installdir* is the path to the directory where you installed the Integrity server.

Integrations Support

Supported Integrations

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

http://www.ptc.com/partners/hardware/current/support.htm

Implementer Integration

When upgrading your Integrity server, specific issues with Implementer integrations can occur. If you are upgrading an Implementer integration, contact PTC - Integrity Support for assistance, and consult the documentation for Implementer.

Getting Ready to Upgrade

When to Upgrade

PTC provides an alert system that publishes Integrity Alerts for all new HotFixes, as well as other pertinent product information (such as deprecated support for operating systems, databases, and releases). You can select the criteria of interest to you and sign up for e-mail notifications to alert you to the relevant updates. You can also review and search all existing alerts. For more information, go to the Integrity Support Center at:

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

To avoid disruption in service to your users, perform the upgrade in off hours. If that is not possible, remember to inform your users that you are performing an upgrade and that the Integrity is to be temporarily unavailable.

Information Required for Upgrade

Be ready with your customer information since the installation program prompts you to provide this during upgrading. Information is saved in the `*installdir*/config/properties/ is.properties` file and is automatically displayed on the Integrity server Homepage.

Backing Up

Backing Up Your Database

You cannot roll back without an existing database backup. It is essential that you back up your existing database before starting the upgrade.

Performing regular backups of your database should already be part of your normal operations.

Note

- Always stop the Integrity server before performing any database maintenance.
- If you changed the default location of the server database files, make sure the files that you are copying are from this new location.

Backing Up Integrity Server Directory

You want to back up the *installdir* directory before running the installation. A backup of this directory allows you to restore the Integrity server directory if the need arises.

Note

To ensure the database is not in use during a backup, stop the Integrity server before creating any backups.

Performing a Trial Run

Because Integrity is a mission-critical application for your enterprise, it is recommended that you perform a test of the upgrade in the form of a trial run before installing live on the production server.

Upgrading for UNIX Users

The upgrading instructions in the next sections apply to both Windows and UNIX users, except as follows for UNIX systems:

- Make sure the environment variable *$DISPLAY* is set, if necessary.
- References to “services” do not apply to UNIX.
- Install the Integrity server in a new location on UNIX without uninstalling the existing (previous) server.
- Ensure that the new server replacing the existing server has been installed, configured, tested, and is performing as needed. Then uninstall the existing (previous) server by running the following file:

  ```
  installdir/uninstall/IntegrityServerUninstall
  ```

  For more information on stopping the server, see the *PTC Integrity Server Administration Guide*.

PTC recommends that you stop the Integrity server before installing. This ensures that the previous Integrity server is not running while upgrading your database.

Upgrading to Integrity 10.9

If you are upgrading to Integrity 10.9 from 10.8, you must upgrade using the full install executable.

A full install using the executable is required when upgrading to Integrity 10.9 from supported versions other than 10.7. For more information, see Your Upgrade Path on page 10.

This guide describes the process for installing using the full install executable.

Single Server

The following is a high-level description for upgrading a single server:

1. Stop the existing Integrity.
2. Install 10.9 to a different directory than the one in which the existing Integrity server resides.
3. Migrate the server configuration files, such as properties, policies, reports, and triggers. Make manual adjustments to configuration files as required.
4. Upgrade Integrity clients.
5. Uninstall the existing (previous) Integrity server.

Multiple Servers

The following is a high-level description for upgrading multiple servers:

1. Plan your upgrade sequence, determining how to minimize downtime for each server. The recommended upgrade sequence follows:
   a. Workflows and Documents and Test Management-enabled servers
   b. Configuration Management-enabled servers
   c. FSA proxy servers
   d. Clients
2. Implement your upgrade sequence. For each server:
   a. Stop the existing Integrity server.
   b. Install Integrity server 10.9 to a different directory than the one in which the existing Integrity resides.

c. Migrate server configuration files, such as properties, policies, reports, and triggers. Make manual adjustments to configuration files as required.

3. Upgrade Integrity clients.
4. Uninstall all existing (previous) Integrity servers.

Using Admin Staging to Upgrade Multiple Servers

As a best practice to avoid losing any in progress changes while upgrading, PTC recommends a three-tiered admin staging configuration for upgrading staging and production servers.

Consider the following admin staging configuration:

Development (staging server) > *Test* (staging server) > *Production* (production server)

This configuration reduces the risk of losing any in progress changes on the Development and Test servers when the upgrade occurs.

1. Migrate all of the changes you want to retain from Development to Test and from Test to Production.
2. Ensure you have reliable backups of all of the servers in your admin staging configuration.
3. Ensure you have a rollback plan.
4. On the Development server:

   a. From the command line, run `mksis stop` to stop the server.

   b. To uninstall the Windows service for your server, run `mksis remove`. This removes the service only; it does not uninstall the program files.

   c. Install the new server using the **Upgrade of an existing server** option. Specify the Development server's database as the database to upgrade, and point to the existing Integrity server install directory as the server to upgrade. As part of this process, the Windows service is installed.

   d. From the new *`New_ServerInstall`*`/bin`, run `isutil -c migrateServerConfig` *Existing_ServerInstall*.

      where *New_Server_install* is the new Integrity server 10.9 directory and *Existing_ServerInstall* is the existing (previous) Integrity server directory.

      This creates a very large amount of output, including files that were not migrated because of a conflict. This output does not appear in the `serverMigration.log` directory.

   e. Manually migrate the customized contents of any of the following *Existing_ServerInstall* directories: `/data/public_html`, `/data/triggers`, `/data/gateway`.

5. On the Test server:

    a. From the command line, run `mksis stop` to stop your server.

    b. To uninstall the Windows service for your server, run `mksis remove`. This removes the service only; it does not uninstall the program files.

    c. Install the new server using the **Upgrade of an existing server** option. Specify the Test server's database as the database to upgrade, and point to the existing Integrity server install directory as the server to upgrade. As part of this process, the Windows service is installed.

    d. From *`New_ServerInstall`*`/bin`, run `isutil -c migrateServerConfig` *Existing_ServerInstall*.

        where *New_ServerInstall* is the new Integrity server 10.9 directory and *Existing_ServerInstall* is the existing (previous) Integrity server directory.

        This creates a very large amount of output, including files that were not migrated because of a conflict. This output does not appear in the `serverMigration.log` directory.

    e. Manually migrate the customized contents of any of the following *Existing_ServerInstall* directories: `/data/public_html`, `/data/triggers`, `/data/gateway`.

6. On the Production server:

    a. From the command line, run `mksis stop` to stop your server.

    b. To uninstall the Windows service for your server, run `mksis remove`. This removes the service only; it does not uninstall the program files.

    c. Install the new server using the **Upgrade of an existing server** option. Specify this server's database as the database to upgrade, and point to the existing Integrity server install directory as the server to upgrade. As part of this process, the Windows service is installed.

    d. From the *`New_ServerInstall`*`/bin`, run `isutil -c migrateServerConfig` *Existing_ServerInstall*.

        where *New_ServerInstall* is the new Integrity server 10.9 directory and *Existing_ServerInstall* is the existing (previous) Integrity server directory.

        This creates a very large amount of output, including files that were not migrated because of a conflict. This output does not appear in the `serverMigration.log` directory.

    e. Manually migrate the customized contents of any of the following *Existing_ServerInstall* directories: `/data/public_html`, `/data/triggers`, `/data/gateway`.

7. Restart the Production server.

8. Restart the Test server.
9. Restart the Development server.
10. To confirm that the Test server is functioning properly, start the Admin Migration Wizard on the Test server.
11. To confirm that the Development server is functioning properly, start the Admin Migration Wizard on the Development server.

Installing Integrity 10.9 Server

This section contains instructions for installing the Integrity server from the full install executable.

Determining the Correct license.dat File

As you prepare to install Integrity 10.9, you must determine the correct `license.dat` file to use.

If you are upgrading an existing server, you can continue to use the same `license.dat` file. On the existing server, check the *`Existing_ServerInstall`*`/config/properties/is.properties` file, and search for the `mksis.licensePath` property, which lists the path to the `license.dat` file. *Existing_ServerInstall* is the existing Integrity server directory.

If you are moving the FlexNet server to new hardware, PTC recommends using the PTC Licensing Tool (https://www.ptc.com/apps/licenseManager/auth/ssl/index.jsp) to get a license transfer. This is not necessary if the Integrity server is moving to new hardware, but the FlexNet server is not moving.

You can also use the PTC Licensing Tool to get a fresh copy of the license file.

Note

If you are upgrading to Integrity server 10.9 from an Integrity server earlier than 10.1, determine the correct `license.dat` file before the upgrade. This file includes information customer information, such as `PTC_Customer_Number` and `PTC_Contract_Numbers`.

Step 1: Stop Existing Integrity Server

Before attempting to install Integrity server 10.9, stop the existing Integrity server. Always stop the server using the `mksis stop` command. Stopping the server using this method ensures that the service is also stopped; this is a requirement before installing a new Integrity server.

⚠ **Caution**

Never perform a hard stop of the Integrity server. Archives can become corrupted if a checkin operation is being performed when a hard stop is used to stop the Integrity server.

For more information on stopping the existing Integrity server, see the *PTC Integrity Server Administration Guide* for that release.

Note

This release installs the service named `Integrity 10`. You need to uninstall the service from the previous Integrity server release before installing Integrity server 10.9. For information on installing and uninstalling the service, see the *PTC Integrity Server Administration Guide*.

Step 2: Install Integrity 10.9 Server

As part of the installation and upgrade, you must also perform the following tasks:

1. Specify the installation type (server host or proxy only).
2. Specify a suitable installation directory.
3. Specify the appropriate server host name and port number for Integrity server 10.9.
4. Upgrade your existing database.
5. Migrate the server configuration files and admin objects.
6. Identify changes to ACL Permissions in this release, and upgrade your ACLs accordingly.

For detailed information on configuring and starting the Integrity server, or for information on administering the Integrity server, see the *PTC Integrity Server Administration Guide*.

Server Installation Type

The Integrity server installer prompts you to specify an installation type:

- If you are installing a new server, click **New server**.
- If you are upgrading an existing server, click **Upgrade of an existing server**.

To upgrade an existing server

Note

If you do not follow this procedure, you must manually put the IPTs into the database after the server starts. If you have RM 07 IPTs, none of the existing RM field names that are built-in fields in Integrity are mapped to the appropriate fields in the IPTs. This requires that you manually update the IPTs.

1. Run the Integrity installation.
2. When the installer prompts you for the type of installation, click **Upgrade of an existing server**, and then click **Next**.
3. Type the path or browse to the location of the existing server install directory, and then click **Next**.

   The installer copies any existing IPTs and stores them in the Integrity server database.
4. Continue the server installation.

Server Repository Type

Integrity supports the database repository only. The selected server repository is the location where configuration management information (including revision and project information) is stored.

Installation Directory

If you are migrating server files, you must install the Integrity server to a different directory than the previous installation.

Set Server Hostname and Port Number

The Integrity server uses the fully qualified host name of the server machine. However, when upgrading to this release, you want to retain the same host name and port number used by the previous Integrity server installation.

Upgrade Your Database

During the installation, you select your database type and database connection information. You are then prompted to migrate the database schema you used with the existing Integrity server for use in Integrity 10.9. For more information, see the *PTC Integrity Server Administration Guide*.

Note

- The server installer does not permit you to create new tables if it detects any existing Integrity tables. If there are existing data, your choices are to migrate the data or exit the installation.
- For Oracle databases, ORA-01555 errors can occur if you do not have sufficiently large rollback segments. Consult your Oracle product documentation for more information.

If the migration fails, you must resolve the error, restore the database, and then attempt the migration again.

Note

If the migration fails due to a Database Transaction Log Size Exceeded error, you can correct the problem and then rerun the migration without first restoring the database.

Upgrade ACL Permissions

The Access Control Lists (ACLs) use a set of database tables to store security, configuration, and administrative information for the Integrity.

Permissions added since the last release are set to deny. For more information, see the *PTC Integrity Server Administration Guide*.

Step 3: Migrate Server Configuration Files

In this release, certain properties are now contained in the database. If you have existing properties you want to retain, Integrity provides the means to migrate server properties and configuration files automatically. However, the source files you are migrating must be located in the original directory structure.

If you intend to manually transfer server configuration file information instead of using the utility provided, see the section entitled “Manually Transferring Server Configuration Files”.

To migrate server settings and configuration files, use the `migrateServerConfig` command for the `isutil` utility.

Once the properties are successfully migrated to the database, they can be further modified in the Integrity Administration Client GUI or from the CLI using the `integrity setproperty`, `im setproperty`, and `si setproperty` commands; and the `im diag` and `si diag` commands.

Migrating Using the `isutil` Utility

The `migrateServerConfig` command for the `isutil` utility migrates:

- properties

  Note

  The `migrateServerConfig` command converts security settings as part of the properties file migration.

- policies
- trigger scripts

  Script files that do not exist in the destination location are copied to the destination location and retain their original file names. All other script file names have an appropriate file extension appended and are copied to the destination location.

  - A file extension of `.10` is appended for Integrity versions 10.0-10.4.
  - A file extension of `.11` is appended for Integrity 10.5.
  - A file extension of `.13` is appended for Integrity 10.6.

  No merging of script contents is performed by the utility. You must manually modify script file contents.

  Similarly, the `global.events` file is copied and renamed.

  - It is renamed to `global.events.10` on Integrity versions 10.0-10.4.
  - It is renamed to `global.events.11` on Integrity 10.5.
  - It is renamed to `global.events.13` on Integrity 10.6.

> **Note**
>
> If the scripts are located outside of the standard directory, the utility does not copy them. You must manually move them to the desired location.

> **Note**
>
> As of Integrity 10.6, Japanese versions of the trigger scripts are no longer installed with Integrity server.

- reports and report resources

> **Note**
>
> In the event of an upgrade conflict (a report file was edited by the user and was also updated by PTC since the last release), the existing reports in the destination location (`*installdir*/data/reports`)are retained with their original filenames, and the new files have a number appended to their filenames. For Integrity 10.6 and later, the included report files support localization (see the documentation for localizing reports in the *PTC Integrity Server Administration Guide*). If you are using included reports from an Integrity release 10.5 and earlier, you can continue to use those reports if there is no localization requirement. To make use of the localization features of reports, manually update the reports as needed.
>
> After you run this utility to upgrade to Integrity 10.6 and later, the `*installdir*/data/ reports/ja` folder will be present on the upgraded server. However, PTC Integrity no longer supports language-based folder, so you can delete this folder. Manually copy your pre-10.6 report recipes (English and non-English) to `*installdir*/data/reports/recipes`.

- Java properties from `*installdir*/config/mksservice.conf`:
  - Memory (only if they are larger in the previous release than the latest release): `-Xms`, and `-Xmx`, `-Xss`
  - Garbage collection: (if they do not currently exist): `-XX:+PrintGCTimeStamps`,

`-XX:+PrintTenuringDistribution`, and
`-XX:+PrintGCDetails`

- change package reviewer rules
- `sitenotes.html` (if the file does not exist in the target directory)
- information about shared Visual Studio solutions and projects in *`installdir`*`/data/vsi/ vsibinding.properties`

Note

As of Integrity 10.6, this utility no longer migrates files that are not part of the Integrity version to which you are updating. If you wish to use existing files on your system that were dropped or deprecated, you must manually copy them from the source server (that is, the one you are migrating from) to destination server (that is, the one you are migrating to).

The `isutil` utility is installed with Integrity 10 in the following location:

`installdir``/bin/isutil.exe`

where *installdir* is the installation directory for Integrity server 10.

The syntax for the command is:
`isutil –c migrateServerConfig` “*`installdir`*”

where *installdir* is the installation directory for the source Integrity server you are migrating from.

Caution

There is no undo mechanism in place to undo a migration. To restore Integrity server 10 files, see the section entitled “Migration Backup Files”.

For the purposes of this procedure, only the `migrateServerConfig` command is documented and intended to be used. For more information on the available commands for the `isutil` utility, see the *PTC Integrity Server Administration Guide*.

Upon completion, the command outputs a message to the console describing the success or failure of the migration. If the command fails, details are printed to the console only, requiring you to analyze and save the output. If there are script files, you need to manually merge them at this time.

Note

Passwords in migrated files are never displayed in messages to the console or the log file. Each character in the password is replaced with “X”.

Migration Backup Files

The `migrateServerConfig` command for the `isutil` utility creates a backup of every file in the Integrity server 10 installation directory that it changes. Files are backed up into the following location:

```
installdir/backup
```

where *installdir* is the Integrity server 10 installation directory.

The backup directory contents mimic the target server directory structure. The backup root directories are versioned such that rerunning the migration never overwrites existing backup files. For example, running the migration a second time creates an additional folder `backup1`, and running a third time, `backup2`.

Migration Log

The `migrateServerConfig` command for the `isutil` utility generates the following log:

```
installdir/log/serverMigration.log
```

where *installdir* is the Integrity server 10 installation directory.

The `serverMigration.log` file contains:

- The date and time the migration occurred.
- The absolute path to the source directory used, for example, the installation directory for Integrity server 10.
- The name of each file changed by the utility, and the name of the backup file created to represent the original (see the section entitled “Migration Backup Files”).
- A description of how the file was changed, stating:
  - if the file was deleted
  - if the file is an exact copy from the old installation (including the name of the source file copied, and its new name)
  - what settings were changed, moved or removed (including the setting name, its original value, and its new value)

Migrating Agent Properties

The Integrity Agent `AgentUtils.exe` utility migrates Integrity Agent properties. This utility is run on each agent and must be pointed at the previous installed version of the Integrity Agent.

The utility also migrates the following Java properties from *`installdir`*`/config/mksservice.conf`:

- Memory (only if they are larger in the previous release than in the latest release): `-Xms`, `-Xmx`, `- Xss`
- Garbage collection: (if they do not currently exist): `-XX:+PrintGCTimeStamps`, `-XX:+PrintTenuringDistribution`, and `-XX:+PrintGCDetails`
- The Integrity Agent 10 target directory must be located in a different location than the original Integrity Agent 2009 installation location.

The utility is installed with Integrity Agent 10 in the following location:
`installdir``/bin/AgentUtils.exe`

where *installdir* is the directory Integrity Agent is migrated to.

The syntax for the command is:
`agentutil –c migrateAgentConfig` *`Integrity Agent installdir`*

where *Integrity Agent installdir* is the current Integrity Agent target directory.

On completion, the command outputs a message to the console describing the success or failure of the migration. If the command fails, details are printed to the console only, requiring you to analyze and save the output.

Note

Passwords in migrated files are never displayed in messages to the console or the log file. Each character in the password is replaced with “X”.

Step 4: Start Integrity 10.9 Server

To start the new Integrity server, use the `mksis start` command.

For complete details on running the Integrity server, see the server installation documentation in the*PTC Integrity Server Administration Guide* or the Integrity Help Center.

Ensure that users can connect to the Integrity server by checking the server log file (and verifying an entry for `GENERAL(0): Listening on port *:`*`nnnn`*) or by connecting through an Integrity client.

Step 5: Uninstall Existing Integrity Server

Note

Before uninstalling the existing Integrity server, ensure that the Integrity server replacing it has been installed, configured, tested, and is performing as required.

For more information on uninstalling the existing Integrity server (the previous installation you are upgrading from), refer to the documentation that was originally issued for the release version you installed.

When uninstalling Integrity server on Windows, you should always use the following file:

`installdir/uninstall/IntegrityServerUninstall.exe`

After the server is stopped, the `IntegrityServerUninstall.exe` file removes the Integrity service and launches the application that performs the server uninstall.

If the server is running as a service, make note of the user the service is running as before uninstalling. In most cases this is “system”. If you use the operating system’s uninstall feature or the uninstall shortcut in the Integrity server program group, you must manually remove the service.

Note

When you are ready, migrate your relationship data to the new table introduced in this release. For more information, see the next section.

2

Command Line Interface Changes

This release includes changes that affect the command line interface (CLI).

Note

As part of any upgrade, PTC recommends that you review and update any scripts you use with the CLI.

This document summarizes the key changes for the command line interfaces that have changes in this release of Integrity.

Command Line Information

For detailed information on all commands and options, see the CLI man pages.

im Command Options

im Command	Changes

`im copyissue`	New command options:
	`-customFieldDefinition`
	`-customFieldValue`

`im createcontent`	New command option: `-customFieldValue=value`
`im createfield`	New value for `-type`: `ier`

`im createissue`	New command options:
	`-customFieldDefinition`
	`-customFieldValue`

`im createsegment`	New command option: `-customFieldValue=value`

`im editfield`	New command options:
	`-ierDefaultColumns`
	`-ierThingName`
	New value for `-removeOverride`
	`ierThingName`
	Deprecated value for `-removeOverride`
	`ierRelativeURI`

`im editissue`	New command options:
	`-removeCustomFieldDefinition`
	`-customFieldDefinition`
	`-customFieldValue`
	`-[no|confirm]removeCustomFieldDefinitions`
	`-[no|confirm]removeCustomFieldPickValues`

im Command	Changes
	-[no|confirm]removeCustomFieldValues
im fields	New values for -fields: ierDefaultColumns ierThingName
im importcontent	New command option: -customFieldValue
im importissue	New command options: -customFieldDefinition -customFieldValue
im importsegment	New command option: -customFieldValue
im viewissue	New command option: -[no]showIncomingExternalReferences
im viewpendingimports	New command
im viewsegment	New command option: -perspective

integrity Command Options

There are no changes to integrity commands for this release.

si Command Options

si Command	Changes
si createdevpath	Deprecated command option: -resultingSubprojectConfiguration New command options: -creationMethod -onLiveConfiguration
si extenddevpath	Deprecated command option: -projectRevision

tm Command Options

tm Command	Changes

`tm resultfields`	New values for `-fields`:
	`ierDefaultColumns`
	`ierThingName`

ACL Permissions

There are no changes to the Access Control Lists (ACLs) for this release.