



# J A D E <sup>TM</sup>

## Upgrading to JADE 7.0 Consolidated Release Information

VERSION 7.0.04



Copyright © 2012  
Jade Software Corporation Limited  
All rights reserved

Jade Software Corporation Limited cannot accept any financial or other responsibilities that may be the result of your use of this information or software material, including direct, indirect, special or consequential damages, or loss of profits. There are no warranties extended or granted by this document or software material.

You should be very careful to ensure that the use of this software material and/or information complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Jade Software Corporation Limited.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Copyright © 2012 Jade Software Corporation Limited.  
All rights reserved.

JADE is a trademark of Jade Software Corporation Limited. All trade names referenced are the service mark, trademark, or registered trademark of the respective manufacturer.

For details about other licensing agreements for third party products, you must read the JADE **ReadMe.txt** file.

# Contents

<b>JADE Release Support</b>	<b>6</b>
Deimplementations and Deprecations .....	6
ActiveX Exposure Deimplementation .....	6
Compact JADE Single User Mode .....	6
Database Rebuild Files Deimplementation.....	6
JADE Dump and Load Utility Deimplementation.....	6
JADE on Linux .....	7
JADE Portable Graphical User Interface (GUI) Client .....	7
Rational Rose Deimplementation .....	7
System: <i>getDbObjectCacheStats</i> Method.....	7
Web Service Framework Jade61 Application Type .....	7
<b>Accessing Details about Faults Fixed in Releases</b>	<b>8</b>
How to Locate PARs Fixed in a Specific Release.....	8
<b>Upgrading from an Earlier JADE 7.0 Release or JADE 6.3.05 Release or Higher</b>	<b>9</b>
Running the Upgrade.....	9
Running Two Releases of JADE on the Same Workstation.....	12
High-Availability Upgrade .....	12
JADE Thin Client Upgrade .....	13
Upgrading a 32-Bit Presentation Client Connecting to a 64-Bit Application Server .....	13
Upgrading a Synchronized Database Environment (SDE).....	13
Upgrading an RPS Node from JADE 6.3 to 7.0 .....	14
Upgrade Validation.....	14
Hot Fix Releases .....	14
<b>Changes in JADE Release 7.0.04</b>	<b>15</b>
Behavioral Change of Web Services (PAR 56921).....	15
Certificate Authentication (PARs 56904, 57638).....	15
Converting a User Database (PAR 56876).....	15
Date Parsing Routines for Two-Digit Years (PAR 56832) .....	16
Deleting and Renaming Entities .....	16
Deleting Packages (NFS 45928).....	16
Deleting a Schema (NFS 57540) .....	17
Renaming a Schema (NFS 57075).....	17
Dock Controls (PAR 56859) .....	17
Initiating an Application (PAR 57541).....	18
Iterating Backwards through a Virtual Collection (PAR 57322).....	18
JADE Unit Testing Framework.....	18
Debugging the Execution of a JADE Unit Test (NFS 52101).....	18
Initializing and Finalizing the JADE Unit Test Framework (PAR 56498).....	18

**Changes in JADE Release 7.0.04, continued**

JadeLocal Transport (PAR 57602) .....	19
JadeWebServiceProvider::getServerVariable (PAR 57163) .....	19
Java Framework Support for HugeStringArray Objects (NFS 54330) .....	19
Journal Rate Analysis Sampling (PAR 57125) .....	20
Monitoring Node Locks (NFS 54406) .....	20
Reblocking Collection Class Maps .....	20
Relational Population Service (RPS) .....	21
Adding an Existing Property or Method to an RPS Table (NFS 57061) .....	21
Converting OID Columns (PAR 57473) .....	21
Datapump Application Execution (PAR 53009) .....	22
Extracting an RPS Table (PAR 55803, PAR 56829) .....	22
Silverlight .....	22
Accessing Third-Party Controls (PAR 56871) .....	22
Extracting a Schema to Generate a XAP File (PAR 57458) .....	23
JADE Properties for XAML Controls inside DataTemplates (PAR 57187) .....	23
XAML Browser (PAR 57192) .....	23
Synchronized Database Service (SDS) (PAR 57217) .....	24
JadeDatabaseAdmin::getCurrentJournalOffset Method .....	24
Changed JadeDatabaseAdmin::sdsGetSecondaryProxy Method .....	24
[ConnectionParams] Section SocketBufferSize Parameter .....	25
System::verifyDbEncryptionMasterKey Method (NFS 57338) .....	25

---

# Upgrading to JADE Release 7.0.04

---

This document covers the following topics.

- [JADE Release Support](#)
  - [Deimplementations and Deprecations](#)
- [Accessing Details about Faults Fixed in Releases](#)
- [Upgrading from an Earlier JADE 7.0 Release or JADE 6.3.05 Release or Higher](#)
  - [Running the Upgrade](#)
  - [High-Availability Upgrade](#)
  - [JADE Thin Client Upgrade](#)
  - [Upgrading a Synchronized Database Environment \(SDE\)](#)
  - [Upgrading an RPS Node from JADE 6.3 to 7.0](#)
  - [Upgrade Validation](#)
  - [Hot Fix Releases](#)
- [Changes in JADE Release 7.0.04](#)

Refer to **RelInfo7003.pdf** in your JADE **documentation** directory for details about JADE release 7.0.03 changes that may affect your JADE 6.3 existing schemas and changes in JADE release 7.0.03.

---

**Tip** For details about using Acrobat Reader to view JADE documents, see “[JADE Product Information Library in Portable Document Format](#)”, in Chapter 2 of the *JADE Development Environment User's Guide*.

The *JADE Product Information Library* document (**JADE.pdf**) provides a summary of contents of documents in the JADE product information library and navigation to the documents.

---

If you want to develop your own installation process for Windows, the JADE install and upgrade steps are documented in the **ReadmeInstallSteps.doc** file in the **\documentation** directory.

---

**Note** To customize the deployment upgrade on Windows, see “[Customizing the Deployment Upgrade Process](#)”, in Appendix A of the *JADE Runtime Application Guide*.

---

# JADE Release Support

Support for JADE 6.2 will cease in October 2012. Support for JADE 6.3 continues until April 2014.

For details about the JADE release policy, go to:

[http://www.jade.co.nz/downloads/jade/JADE\\_ReleasePolicy.pdf](http://www.jade.co.nz/downloads/jade/JADE_ReleasePolicy.pdf)

For details about the JADE release schedule, go to:

<http://www.jade.co.nz/jade/updates.htm#releasesched>

As announced previously:

- JADE support on Windows 2000 ceased on July 13, 2010
- Support for JADE 6.1.15 (the final JADE 6.1 release) ceased in October 2010
- JADE 7.0 is not supported on Windows Server 2003

JADE 7.0 server and client nodes require Windows NT 6.0 or later (that is, Vista or Windows Server 2008, or later), because JADE 7.0 utilizes API calls that are present only in these more-recent operating systems.

## Deimplementations and Deprecations

This section contains the deimplementations and deprecations in JADE 7.0.

### ActiveX Exposure Deimplementation

As notified in JADE 6.3, JADE's ActiveX exposure feature is no longer available. In recent years, ActiveX technologies have been replaced by .NET. From JADE 6.3, you could generate exposures using these .NET technologies, providing a more modern, flexible, and easier to develop mechanism than that provided by ActiveX.

If you have not already done so, where required, we recommend that you re-write ActiveX exposures using the JADE .NET exposure. For details, see [Chapter 17](#) of the *JADE Development Environment User's Guide*.

### Compact JADE Single User Mode

As notified in JADE 6.3, Compact JADE single user node is no longer available in this product release; that is, support for JADE databases running on Windows Mobile devices is no longer available.

The Compact JADE thin client continues to be available on these devices in this release.

### Database Rebuild Files Deimplementation

The operation to rebuild files provided by the JADE Database utility **jdbutil** and **jdbutilb** programs is no longer available.

Use the **reindex files** operation to re-establish corrupted indexes.

### JADE Dump and Load Utility Deimplementation

As notified in JADE 6.3.08, JADE's Dump and Load utility (**jddlutil**) is no longer available.

To convert a user database, use the **JadeConvertDb** application in the **jadclient** program command line. For details, see [Chapter 4](#) of the *JADE Runtime Application Guide*.

## JADE on Linux

As notified in JADE 6.3, JADE 7.0 supports Windows only, which means that in this product release, Linux distributions (for both Red Hat and SUSE) are no longer available.

JADE will continue to support customers running JADE 6.3 on Linux until at least April 2014.

## JADE Portable Graphical User Interface (GUI) Client

As notified in JADE 6.3, JADE portable GUI client is no longer available with this release.

## Rational Rose Deimplementation

As notified in JADE 6.3, the JADE interface for the Rational Rose modeling tool is no longer available.

If you want to use third-party design tools (for example, Enterprise Architect), you can use the JADE XMI interface to import the model into your database. (For details, see [Chapter 5](#), “XML Metadata Interchange (XMI) Support”, of the *JADE External Interface Developer’s Reference*.)

## System::getDbObjectCacheStats Method

In JADE 7.0, the database engine no longer utilizes an object cache.

One consequence of this is that the statistics returned by the **System** class **getDbObjectCacheStats** method will contain zero values. This method will be deimplemented in the JADE 7.1 release.

## Web Service Framework Jade61 Application Type

As notified in JADE 6.3, the **Jade61** value for the **JadeHttp.ini** file **ApplicationType** parameter is no longer valid.

## Accessing Details about Faults Fixed in Releases

To access the complete documentation about the Product Anomaly Reports (PARs) fixed in this release, run **Parsys**, our Fault Managements and Customer Contact system. This system also enables you to view the progress of your own contacts.

If you have any queries about **Parsys**, please direct them to JADE Parsys Support in the first instance, at [parsysupport@jadeworld.com](mailto:parsysupport@jadeworld.com). You can download the install shield for **Parsys** from the following URL.

[http://www.jade.co.nz/jade/parsys\\_support.htm](http://www.jade.co.nz/jade/parsys_support.htm)

When you first run the **Parsys** application, it downloads an update via the automatic thin client download feature. When this has completed and you have the log-on form ready and waiting, please contact JADE Parsys Support, who will then send you an e-mail message with your user code and password details. **Parsys** requires you to change your password when you first log on.

---

**Note** Because the encryption of passwords is a one-way algorithm, we cannot advise you of your password should you forget it, but we can reset it to a known value again.

---

## How to Locate PARs Fixed in a Specific Release

This section describes the actions that enable you to locate Product Anomaly Reports (PARs) fixed in a specific release.

### ➤ To locate the PARs fixed in a specific release

1. Select the **Advanced Search** command from the Search menu with the following settings on the **Basic Search Criteria** sheet.
  - a. The **Latest** item is selected in the **Mode** combo box.
  - b. **All** is selected in the **Priority** list box.
  - c. The **PAR** check box is checked in the Phase group box.
  - d. The **Fault** check box is checked in the Type group box.
  - e. The **Closed** and **Patched** check boxes are checked in the Status group box.

---

**Note** If you want to restrict the search to the hot fixes that were produced, check the **Hot fix created** check box on the **Advanced Search Criteria II (Optional)** sheet.

---

2. On the **Advanced Search Criteria III (Optional)** sheet:
  - In the **Closed** list box of the Releases group box, select the release whose fixed PARs you want to locate (for example, the **7.0.3** list item).
3. Click the **Search** button.

# Upgrading from an Earlier JADE 7.0 Release or JADE 6.3.05 Release or Higher

This section covers the following topics.

- [Running the Upgrade](#)
  - [Running Two Releases of JADE on the Same Workstation](#)
- [High-Availability Upgrade](#)
- [JADE Thin Client Upgrade](#)
- [Upgrading a Synchronized Database Environment \(SDE\)](#)
- [Upgrading an RPS Node from JADE 6.3 to 7.0](#)
- [Upgrade Validation](#)
- [Hot Fix Releases](#)

---

**Caution** Before you upgrade to JADE 7.0, refer to **RelInfo7003.pdf** in your JADE **documentation** directory for details about JADE release 7.0.03 changes that may affect your JADE 6.3 existing schemas.

---

## Running the Upgrade

Server nodes (database upgrades) require the 64-bit edition. Client nodes can upgrade the 64-bit edition or the 32-bit edition.

The JADE 6.3 to 7.0 database upgrade differs from previous JADE upgrades in that it involves a migration of the data, due to the nature of the structural changes of the objects in the database.

---

**Notes** This database upgrade will require sufficient disk to accommodate an additional full copy of the database plus head-room of about 10 percent of the database size.

If you want to continue processing update transactions on a production JADE 6.3 database while a copy of that database is upgraded to the JADE 7.0 structure, see “[High-Availability Upgrade](#)”, later in this document.

---

The JADE Setup program enables you to upgrade your binary and database files to JADE 7.0, by performing the following actions.

1. Ensure that your JADE environment is an earlier JADE 7.0 release or release 6.3.05 or higher.

On the system to be upgraded, carry out the following certify operations. Proceed to the next certify operation only when any and all errors reported in the current operation are resolved.

  - a. A physical certify using JADE Database utility (**jdbutil.exe** or **jdbutilb.exe**), to ensure that the system is structurally correct. (For details, see [Chapter 1](#) of the *JADE Database Administration Guide*.)
  - b. A meta logical certify, to ensure that the meta model is clean. (For details, see “[Running a Non-GUI JADE Logical Certifier](#)”, in [Chapter 5](#) of the *JADE Object Manager Guide*.)
  - c. A logical certify, to ensure that the user data is referentially correct. (For details, see “[Running the Diagnostic Tool](#)”, in [Chapter 5](#) of the *JADE Object Manager Guide*.)

---

**Note** If you are unsure how to interpret the information output by the certify process, contact JADE Support ([jadesupport@jadeworld.com](mailto:jadesupport@jadeworld.com)) for advice.

---

2. If your database has partitioned database files, ensure that any offline partitions are brought online.
3. Take a full backup copy of your existing JADE directories, first ensuring that your database is not in recovery mode.

---

**Caution** As roll-forward recovery of the installation and upgrade process is not supported, it is important that you backup your database before starting the JADE Setup process to install JADE 7.0 and upgrade your existing data.

---

4. If upgrading from a JADE 6.3 system, uninstall any **jadrap** database services (that is, JADE Remote Node Access services) that you want to upgrade to JADE 7.0. (This is required because of changes in the underlying registry entries that are expected or created in JADE 7.0.)
5. Ensure that you have the appropriate privileges or capabilities to install applications.

---

**Note** Installing any Microsoft redistributable package requires administration privileges.

---

If you are upgrading the 64-bit edition, the **vc\_red.msi** Microsoft C++ 2010 SP1 Redistributable Package (x64) is required. If you are upgrading the 32-bit edition of JADE, the **vc\_red.msi** Microsoft Windows C++ 2010 SP1 Redistributable Package (x86) is required. This will be installed during the JADE installation and is supplied on the JADE distribution media.

6. To start the JADE Setup program, invoke the **setup.exe** program from the **Jade70** release medium or execute the executable program downloaded from the JADE Web site.
7. If not already installed, the required Microsoft Windows C++ Redistributable Packages are installed.
8. In the Welcome folder, click the **Next>** button to continue the upgrade process.
9. Read the entire software license agreement in the Software License Agreement folder and then click the **Yes** button to continue the installation.
10. In the Installation Type folder, select the **Feature Upgrade** option button, to specify that you want to upgrade an existing JADE release. By default, the **Fresh Copy** option is selected.
11. In the Setup Type folder, select the type of installation that you are upgrading. By default, the **Development** option is selected for 64-bit installation or **JADE Client** for 32-bit installation.

---

**Note** The **Custom** type applies only to a **Fresh Copy** installation type, and is not relevant when upgrading.

The **SDS/RPS Database Server** option applies only to 64-bit **Feature Upgrade** installation type (it is not available for a JADE 6.3 to 7.0 upgrade).

---

12. In the Select Installation Folders folder, specify the locations of the JADE files that are to be upgraded.

The upgrade process defaults to the most-recently used JADE files, and displays these values in the **Install Directory**, **Executable Directory**, and **Database Directory** text boxes. The installation directory is most likely to be the root directory in which you installed JADE, unless you subsequently renamed the root directory or moved the files to another location.

If the locations are not as required, click the adjacent browse buttons (indicated by the ... ellipsis symbols) to display the common File Selection dialog that enables you to select the appropriate directories and files. By default, the **jade.ini** file located in your specified database directory is used. If required, use the **JADE INI File** text box to specify a different valid fully qualified directory and name of the JADE initialization file; for example:

```
d:\mysys\jade\system\jade.ini
```

If Program Start folders are to be updated, specify the name of the folder in the **JADE Program Folder** text box. If you are unsure of the folder to be updated, click the adjacent **browse** button to display the common Folder selection dialog that enables you to select the folder.

The **Database Directory** text box enables you to explicitly specify the location in which the database (system) files are installed. When the destination folder is not **\Program Files**, the database destination defaults to **system** under the install folder (for example, if you specify **c:\Jade70** in the **Install Directory** text box, the database directory defaults to **c:\Jade70\system**).

If the installation directory is a subdirectory of the programmatically determined location of **\Program Files**, the **\Program Files** portion of the install directory is replaced with the programmatically discovered location for the common application data directory (for example, if you specify **c:\Program Files\Jade63** in the **Install Directory** text box, the default database location is **c:\ProgramData\Jade63\system**).

The process checks whether the specified database directory is a valid system and that it is the correct ANSI or Unicode type.

13. The Start Copying Files folder summarizing your upgrade options is displayed. If the selections displayed in the Start Copying Files folder are correct, click the **Next>** button.  
Alternatively, click the **<Back** button to modify your selections.
14. The Question dialog is displayed, advising you to ensure that you have taken a full backup of that database before you proceed with the upgrade process.  
When you are sure that you are upgrading the correct system (and that it has been backed up), click the **Yes** button to start the upgrade process.
15. A warning message box is then displayed, advising you that Dynamic Link Libraries (DLLs) may need to be recompiled. Click the **OK** button, to recompile any required DLLs.
16. A warning message may be displayed if the upgrade validation process has not completed. If so, check the **jadeupgrade.log** file for information about what needs to be modified in your user schemas to pass the validation and enable application execution.
17. When the upgrade is complete, the JADE Setup program informs you that the JADE Setup was successfully completed and that you can now view the **ReadMe.txt** file. To view the **ReadMe.txt** file, ensure that the check box is checked (the default).  
The **ReadMe.txt** file is then displayed in a text editor (for example, Notepad). The **ReadMe.txt** file is a read-only text file installed in your JADE root directory that you can print or delete, if required. This file contains a reference to other JADE-related documents.
18. Click the **Finish** button to end the JADE upgrade process.
19. Re-establish any control file paths set for database files and any required partition file attributes such as location, label, and frozen or offline states.

20. If upgrading from JADE 6.3, install any **jadrap** database services (that is, JADE Remote Node Access services) you had set up. (For details, see “[Running the Server Node as a Service](#)”, in the *JADE Remote Node Access Utility User’s Guide*.)

For a JADE 6.3 upgrade, after the upgrade has completed, the JADE 6.3 database files will be retained in the directory **database-directory\_63** (for example, **c:\Jade\system\_63**). The **database-directory** (for example, **c:\Jade\system**) will contain the upgraded JADE 7.0 database.

---

**Caution** As with any JADE release, you may need to recompile any external method Dynamic Link Libraries (DLLs) or external programs using the JADE Object Manager Application Programming Interfaces (APIs) with the new JADE **\Include** and **\Library** files before you attempt to run your upgraded JADE systems. (For details about the JADE Object Manager APIs, see [Chapter 3](#) of the *JADE Object Manager Guide*.)

Some obsolete files are deleted from the JADE directories when upgrading from JADE 6.3. If you require these files for your JADE system, you must save them before you upgrade or restore them from the original JADE 6.3 release medium.

Documentation and example files are not part of the installation and must be downloaded and installed from the JADE Web site or the release medium separately, if required, into the **documentation** folder and **examples** folder, respectively, of your JADE installation directory.

---

## Running Two Releases of JADE on the Same Workstation

You can have two releases of JADE installed on the same workstation, if the files are in different directories. If ODBC is installed, only the last installation of the JADE ODBC driver is available from the ODBC Data Source Administrator.

If you install multiple copies of JADE 7.0 on a machine, an initial dialog during the installation asks if the installation should install a new instance or maintain or update the installed instance. You should select the default **Install a new instance of this application** option.

## High-Availability Upgrade

When converting a JADE 6.3 database to a JADE 7.0 release, the database conversion step must be performed in-place on a production database with the database down for the duration of the conversion process.

The high-availability upgrade feature enables you to reduce the downtime required to upgrade a JADE 6.3 database to run with a JADE 7.0 release, by continuing to process update transactions on a production 6.3 version database while a copy of that database is upgraded to the 7.0 structure.

The software is installed, directories set up, files copied, and your user schema and data files converted to the new JADE 7.0 structure in a copy of the production database while the source database remains available for online transaction processes.

The required downtime is reduced to the steps executed in an offline transition phase, which are as follows.

1. Shut down the production database.
2. Execute an upgrade replay process to apply committed transactions from JADE 6.3 journals.
3. Execute standard release upgrade steps.
4. Copy or move the converted database to the production directory structure.

For details, see the **HighAvailabilityUpgradeSteps.doc** file in the **\documentation** directory.

## JADE Thin Client Upgrade

A JADE 7.0 presentation client upgrade:

- Rejects a presentation client upgrade from 5.2.08 or earlier. You must handle a presentation client download from JADE 6.1 by first upgrading JADE on the presentation client to release 6.0, 6.1, 6.2, or 6.3.
- Cannot handle a reversion to JADE 6.1 or earlier.
- Rejects a reversion to JADE 6.0, 6.1, or 6.2 if the JADE 6.0, 6.1, or 6.2 application server attempts to download files to the **DownloadDirectory2** directory. The only reversion that is guaranteed is from JADE release 7.0 to JADE release 6.3.
- If you are upgrading presentation clients to JADE release 7.0 under Vista or Windows 7, ensure that you have the appropriate privileges or capabilities to install applications.

If JADE is installed in the **\Program Files** directory (or **\Program Files (86)** directory on a 64-bit machine with 32-bit JADE binaries) when running under Vista or Windows 7:

- If the Vista machine has had UAC disabled, the thin client upgrade will fail because of lack of permissions for standard users. For administration users, the necessary privileges are automatically granted so the upgrade will succeed.
- If UAC is enabled, administrative users are prompted with an **Allow** or a **Cancel** choice but standard users must know and supply the user name and password of a user with administrative privileges to enable the upgrade to succeed.

For more details, see Appendix B, “[Upgrading Software on Presentation Clients](#)”, in the *JADE Thin Client Guide*.

## Upgrading a 32-Bit Presentation Client Connecting to a 64-Bit Application Server

When a 32-bit presentation client connects to a 64-bit application server, the application server upgrades the version of the presentation client but it does not change the 32-bit to 64-bit type of the presentation client, because:

- The presentation client does not check to see if the operating system on which it is running is 64-bit-capable (and it would have to inform the application server about this).
- Any support libraries needed by the presentation client (for example, ActiveX control and automation libraries) would also have to be downloaded or already installed in the presentation client.

By default, a 32-bit presentation client will be upgraded to a version requiring the Microsoft Windows Visual Studio 2010 C++ runtime binaries. If it is not possible for operational reasons to install the Visual Studio 2010 runtime binaries, you can configure the application server to use the presentation clients built with Visual Studio 2005 (that is, compatible with JADE 6.3 C++ runtime binaries). For details, see Appendix B, “[Upgrading Software on Presentation Clients](#)”, in the *JADE Thin Client Guide*.

## Upgrading a Synchronized Database Environment (SDE)

A JADE release 6.3 to 7.0 upgrade of a Synchronized Database Service (SDS) node is not supported.

When the upgrade to 7.0 had completed successfully, re-clone the secondary databases from the upgraded primary database.

## Upgrading an RPS Node from JADE 6.3 to 7.0

A JADE release 6.3 to 7.0 upgrade of a Relational Population Server (RPS) node is not supported.

To retain the RDBMS target database, see “[High-Availability Upgrade](#)”, earlier in this document.

Conversely, if you do *not* want to retain the RDBMS target database, when the upgrade to 7.0 has completed successfully, re-clone the RPS node from the upgraded primary database and recreate the Relational Database Management System (RDBMS) target database from the upgraded RPS node.

## Upgrade Validation

During the upgrade process, a validation script is run to check the integrity of the upgraded system. Any user schema entities that conflict with system schema entities are logged as errors in the **jommsgn.log** file.

All errors must be corrected and validation re-run before user applications can be executed on the updated system. If the system is in the un-validated state, a message box is displayed when you log on to the JADE development environment, asking if validation should be re-run.

## Hot Fix Releases

Hot fixes for JADE system files are released as binary files.

To apply the hot fix:

1. Shut down the system.
2. Copy the hot fix system files into the appropriate directory.
3. Start up the system.

---

**Caution** You must apply all of the files contained in the hot fix at the same time.

It is important to ensure that versions of JADE system files do not diverge from dependent binaries. Doing this ensures that dependent code files (JADE system files and libraries) are backed up and restored together. The default location of the JADE system files is the installation directory (that is, the **bin** directory).

When it is necessary to restore a database from backup and perform recovery, you must avoid reverting to earlier JADE system file and binary versions. When restoring the binaries directory, ensure that it is from the latest backup.

---

## Changes in JADE Release 7.0.04

This section contains details about changes in JADE release 7.0.04.

For a summary of changes and new features in JADE 7.0.03 (the first general release of JADE 7.0), see **RelInfo7003.pdf**, in your JADE **documentation** directory.

### Behavioral Change of Web Services (PAR 56921)

When using a Web service consumer or the **JadeHTTPConnection** class, the default communications protocol has changed from using the WinINet (the JADE 6.3.09 and 7.0.03 default value) to the WinHTTP library. WinHTTP is more appropriate for server-type environments; WinINet is more appropriate for low-performance client situations.

---

**Caution** A potential impact of this change is that if proxy servers are used, configuring the Web service consumer to use a proxy externally from JADE code will change.

For WinINet, you can use the Internet Options dialog accessed from the Control Panel. WinHTTP proxy settings are configured using the **netsh** tool. To obtain help, specify the following command.

```
netsh winhttp set proxy help
```

---

Use the **EnableWinHTTP** and **EnableWinINet** parameters in the [JadeEnvironment] section of the JADE initialization file to control the communications library that is used. The default values are now:

```
[JadeEnvironment]
EnableWinHTTP = true
EnableWinINet = false
```

If you set both parameters to **true**, WinHTTP is used in preference.

If you set both parameters to **false**, the support of both protocols is disabled and a Web service consumer can use only the JADE Direct scheme; that is, **jadehttp.tcp**. In JADE 6.3.09 and 7.0.03, an exception was raised resulting from the Windows ERROR\_RESOURCE\_DISABLED error (4309).

### Certificate Authentication (PARs 56904, 57638)

To ensure that the verification process correctly handles the Certificate Authority (CA) path and JADE initialization file parameters, the *Depth Zero Self Signed Cert* and *Invalid Purpose X.509* non-fatal errors are ignored.

In addition, the default values of the **SSLRemoteCertCheck** and **SSLRemoteVerify** parameters in the [JadeAppServer] and [JadeThinClient] sections of the JADE initialization file parameters are now as follows.

```
[JadeAppServer]
SSLRemoteCertCheck=false
SSLRemoteVerify=false
SSLVerifyDepth=9

[JadeThinClient]
SSLRemoteCertCheck=false
SSLRemoteVerify=false
SSLVerifyDepth=9
```

---

**Note** As the verify depth (specified in the **SSLVerifyDepth** parameter) is now honored, you should not set the value of this parameter to zero (0).

---

The following example of these values allows authority certificates to work.

```
[JadeAppServer]
SSLRemoteCertCheck=false
SSLRemoteVerify=true

[JadeThinClient]
SSLRemoteCertCheck=true
SSLRemoteVerify=true
;SSLVerifyDepth=0
```

## Converting a User Database (PAR 56876)

When converting a user database, the **JadeConvertDb** application converts up to 1,000 files.

If more than 1,000 files are being converted, an error message is output and the application will not convert any files.

## Date Parsing Routines for Two-Digit Years (PAR 56832)

The **JadeEditMask** class now uses the Windows Control Panel setting to convert a two-digit year into a four-digit year for a two-digit edit mask year of 'yy' when the value of the **EnhancedLocaleSupport** parameter in the [JadeEnvironment] section of the JADE initialization file is set to **true**.

By default, years:

- 00 through 29 become 2000 through 2029
- 30 through 99 become 1930 through 1999

If the value of the **EnhancedLocaleSupport** parameter is **false** (the default), the year is calculated using the current century.

## Deleting and Renaming Entities

The following subsections contain the commands that you can now include in a JADE command **.jcf** text file that is specified in the **jadloadb** batch JADE Load utility **commandFile** parameter.

### Deleting Packages (NFS 45928)

The **jadloadb** batch JADE Load utility **commandFile** parameter now enables you to delete both imported and exported packages from a schema, by using the following command syntax.

```
Delete Package schema-name::package-name
```

The following **jadloadb** example contains the fully qualified name of a JADE command **.jcf** text file that deletes a package.

```
jadloadb path=d:\jade\system commandFile=d:\temp\DeletePackages.jcf
ini=d:\jade\myjade.ini loadStyle=currentSchemaVersion
```

The **DeletePackages** command file in the previous example deletes imported package **P99** from the **Par9999Importer** schema, as follows.

```
JadeCommandFile
JadeVersionNumber 7.0.04
Commands
AbortOnError True
Delete Package Par9999Importer::P99
```

An exported package cannot be deleted if it is imported by another schema. An imported package can not be deleted if it imports a class or interface that is used as the type of a property.

## Deleting a Schema (NFS 57540)

The **jadloadb** batch JADE Load utility **commandFile** parameter now enables you to delete a specified schema, by using the following command syntax.

```
Delete Schema schema-name
```

The following **jadloadb** example contains the fully qualified name of a JADE command **.jcf** text file that deletes a schema.

```
jadloadb path=d:\jade\system commandFile=d:\temp\DeleteSchema.jcf  
ini=d:\jade\myjade.ini loadStyle=currentSchemaVersion
```

The **DeleteSchema** command file in the previous example deletes schema **Par9999Importer**, as follows.

```
JadeCommandFile  
JadeVersionNumber 7.0.04  
Commands  
AbortOnError True  
Delete Schema Par9999Importer
```

A schema cannot be deleted if it has subschemas, it is versioned, or it exports a package that is imported by another schema.

## Renaming a Schema (NFS 57075)

The **jadloadb** batch JADE Load utility **commandFile** parameter now enables you to rename an existing schema, by using the following syntax.

```
Rename Schema existing-schema-name new-schema-name
```

The following **jadloadb** example contains the fully qualified name of a JADE command **.jcf** text file that renames a schema.

```
jadloadb path=d:\jade\system commandFile=d:\temp\RenameSchema.jcf  
ini=d:\jade\myjade.ini loadStyle=latestSchemaVersion
```

The **RenameSchema** command file in the previous example renames the **ExampleSchema** to **TestSchema**, as follows.

```
JadeCommandFile  
JadeVersionNumber 7.0.04  
Commands  
AbortOnError True  
Rename Schema ExampleSchema TestSchema
```

You cannot rename the current version of a schema.

## Dock Controls (PAR 56859)

The drawing of dock control drag rectangles was changed from drawing on the desktop because such drawing under Windows 7 was slow and problematic (Windows can erase the drawing without warning).

If a dock container and the MDI client window share the same parent and overlap, when drawing on the dock container, the MDI client window causes that drawing to clip out its own window area.

JADE now recognizes the situation where the drawing is over a sibling MDI client window and it reverts to drawing on the desktop in that situation. This may then result in the drawing flickering and leaving drawing remnants, because of Windows interference.

## Initiating an Application (PAR 57541)

A call to the **Application** class **setApplicationSkin** or **setSkin** (the old style of skin) method from the **Global** class **getAndValidateUser** method now results in the initiating application being told that the new application has been successfully initiated.

In earlier releases, this occurred only when the **signOn** process completed or when the **getAndValidateUser** method created a form.

## Iterating Backwards through a Virtual Collection (PAR 57322)

The JADE compiler now allows the **reversed** option in the **foreach** instruction to be used in virtual collections.

## JADE Unit Testing Framework

The following subsections contain changes to the JADE unit testing framework.

### Debugging the Execution of a JADE Unit Test (NFS 52101)

You can now debug the execution of a JADE unit test.

When you select the **JadeTestCase** class in the Class Browser, the Jade menu now includes the **Unit Test Debug** command, which initiates the JADE unit test framework in JADE debug mode for the selected **JadeTestCase** class or selected method of the class if a method is selected.

In addition, the Jade menu in the Schema Browser contains the **Unit Test Debug** command if the schema contains a subclass of the **JadeTestCase** class. When you select this command from the Schema Browser, the JADE unit test framework is initiated in JADE debug mode for all **JadeTestCase** class subclasses.

---

**Tip** You can use the SHIFT+F9 shortcut instead of the **Unit Test Debug** command in the Jade menu.

---

### Initializing and Finalizing the JADE Unit Test Framework (PAR 56498)

When you run the JADE unit testing framework by using the F9 shortcut key to run the **JadeUnitTest** application from the JADE development environment or by running the **JadeUnitTestBatch**, **JadeUnitTestRun**, or **JadeUnitTestRunner** non-GUI client application using the **jadclient** executable, your **Application::finalize** method was called when the application finished. However, the **Application::initialize** method was not called.

It was inconsistent to execute the **finalize** method and not the **initialize** method, so from this release, neither the **finalize** nor the **initialize** method is called when a JADE unit test framework is used.

If your tests must initialize and finalize the application, they should do this explicitly, by adding a **unitTestBeforeClass** method and **unitTestAfterClass** method, which in turn could call the **app.initialize** and **app.finalize** methods, if appropriate, as shown in the following examples.

```
initialize() unitTestBeforeClass, updating;
begin
    app.initialize;
    // ... any other initialization
end;

finalize() unitTestAfterClass, updating;
begin
    app.finalize;
    // ... any other finalization
end;
```

## JadeLocal Transport (PAR 57602)

The JADE initialization file parameters required to use the **JadeLocal** transport, documented in the “JadeLocal Transport” section in Chapter 3 of the *JADE Installation and Configuration Guide*, stated that **JobLocal** was required for both the server and client specifications.

The parameter should be **JadeLocal**; that is:

```
[JadeServer]
NetworkSpecification<n>=JadeLocal,enabled|disabled,[Global\]base-name

[JadeClient]
ServerNodeSpecifications=JadeLocal,[Global\]base-name
```

## JadeWebServiceProvider::getServerVariable (PAR 57163)

The Web service provider requires the name of the method to invoke. In order to obtain this, the **getServerVariable** method is called. When the name that is retrieved is longer than 30 characters, the name is now truncated to 30 characters. In addition, if the first character of the name is uppercase, it is changed to lowercase.

This name is used to determine the method to invoke when using non-wrapped document literal format messages. When the name does not meet the method-naming requirements of JADE, the method invoked will likely fail and a SOAP fault will be returned to the Web service consumer.

## Java Framework Support for HugeStringArray Objects (NFS 54330)

The JADE Java framework now supports **HugeStringArray** objects in addition to the existing support for **StringArray** objects. This allows arrays to contain strings that are longer than 62 characters, as shown in the following code fragments.

```
import com.jadeworld.jade.rootschema.HugeStringArray;
...
@Entity
public class Entity {
    @DbProperty(name="fred", type="HugeStringArray")
    public HugeStringArray getFred() {
        return (HugeStringArray)EntityAccess.getReferenceProperty(this,
            "fred");
    }
}
```

```

public void setFred(HugeStringArray strArray) {
    EntityAccess.setReferenceProperty(this, "fred", strArray);
}

public void addFred(String str) {
    HugeStringArray hugeSA = getFred();
    hugeSA.add(str);
}

...
HugeStringArray hsa = new HugeStringArray();
em.persist(hsa);
...
myClass.setFred(hsa);
myClass.addFred("A String");

```

## Journal Rate Analysis Sampling (PAR 57125)

The sampling interval granularity has been changed from seconds to milliseconds. The default sampling interval remains one minute, expressed as **60000** milliseconds.

Analysis, generation of summary and total values, and interval snapshot data have been enhanced, by the transaction count now being the number of active transactions seen derived from the transaction IDs of the update records processed, rather than the number of **BEGIN\_TRANSACTION** records seen.

An additional sample column containing the number of **COMMIT\_TRANSACTION** records seen has been added. The file summary and totals summary have also been enhanced to report on the transaction and commit activity.

## Monitoring Node Locks (NFS 54406)

The **Locks** view in the JADE Monitor now provides the **Exclude node locks** check box, which you can check if you want to exclude node locks from the persistent locks list (for example, if you want to avoid filling the lock table with large numbers of stable object node locks when investigating locking issues).

## Reblocking Collection Class Maps

You can use the reblocking collection class maps facility after upgrading an environment from release 6.3 to 7.0, to reblock collections for the best fit to the new file structure.

You can reblock all collections in one or more map files, by using the **JadeCollectionReblocker** application in the non-GUI client (**jadclient**) program in single user or multiuser mode; that is:

```

jadclient.exe path=database-path
              ini=jade-initialization-file
              server=SingleUser|MultiUser
              schema=user-schema-name
              app=JadeCollectionReblocker
              File=file-name|mask
              [File2=file-name [File3=file-name]...]
              [workers=number-of-worker-threads]

```

The *file-name* value is a complete map file name or a partial map file name with a trailing asterisk; for example, **cust\*** means all map files that begin with the letters **cust**. Specify the *file-name* mask **\*** (a single asterisk) to indicate reblocking of all user files, including **rootdef**. Specify the mask **\_\*** (underscore asterisk) to indicate the reblocking of all system files such as **\_userscm**.

The following are examples of the reblocking action.

```
jadclient path=d:\jadeuser ini=d:\salesdb\jade.ini schema=Sales
server=singleUser app=JadeCollectionReblocker File=*

jadclient path=d:\jadeuser ini=d:\salesdb\jade.ini schema=Sales
server=singleUser app=JadeCollectionReblocker File1=testdb File2=banking
File3=_userdev workers=5
```

The optional **workers** parameter enables you to specify the number of multiple concurrent worker threads that are used. The default value of **1** is used if you do not specify the **workers** parameter. You can specify a value in the range 1 through 16. If you specify a value greater than **16**, the maximum number of 16 worker threads is used.

Reblocking is done in normal update transactions, which are committed every 1,000 collections or 30 seconds.

Reblocking skips collections that do not require reblocking. The **JadeCollectionReblocker** application can be terminated part way through a file, in which case only uncommitted changes are discarded. When reblocking is restarted for the same file, collections that have already been reblocked are skipped.

## Relational Population Service (RPS)

The following sections describe the RPS changes in this release.

### Adding an Existing Property or Method to an RPS Table (NFS 57061)

The new **DropHistoricalTableOnAddExisting** parameter in the [JadeRps] section of the JADE initialization file controls the behavior when you add an existing property or method to a historical table in an RPS mapping. The default value of this parameter is **true**; that is, the historical table is dropped if an existing property or method is added.

To modify the historical table (rather than drop the table) when an existing property or method is added, set the value of this parameter to **false**; which will then result in an **ALTER TABLE <> ADD** being used to modify the table when an existing property or method is added. Any existing rows will have a column value of **NULL**.

This parameter is read when the alter table script is created.

### Converting OID Columns (PAR 57473)

The conversion of an OID column type from **String** to **INTEGER/INTEGER** is now supported.

In addition, the following JADE schema changes are achieved using **ALTER TABLE** commands.

- Changing a JADE primitive type from **Byte** to **Integer** or **Integer64**
- Changing a JADE primitive type from **Integer** to **Integer64**

## Datapump Application Execution (PAR 53009)

The **Datapump** application must always be in a state in which it can be executed without error. If schema changes put the **Datapump** application into a non-executable state, the RPS node cannot continue replay and must be recreated from the primary.

The **Datapump** application can be put into a non-executable state in various ways, including:

- Calling uncompiled methods during application startup.
- Additions or changes to imported packages in the **Datapump** application schema.

To avoid re-creation of the RPS node from the primary, when making schema changes on the primary:

- Avoid schema changes in the **Current** version but using the **Latest Schema Version** when loading schemas.
- Initiate the transition only when a consistent set of changes has been loaded.

## Extracting an RPS Table (PAR 55803, PAR 56829)

When extracting data for an RPS mapping, the data is now buffered to improve performance.

The [JadeRps] section of the JADE initialization file can now contain the **ExtractBufferSize** parameter, which enables you to set the buffer size when extracting RPS files. A buffer of the specified size is allocated for each concurrent file being written.

The default value is **1M** and the minimum value is **8K**.

The parameter is read when the RPS node is initialized.

## Silverlight

The following sections describe Silverlight XAML changes in this release.

### Accessing Third-Party Controls (PAR 56871)

The **XamlDocument** class now provides the **registerCallback** method, which has the following signature.

```
registerCallback(callback: JadeMethod): XamlObject;
```

This method registers a JADE method for calling back and returning a **XamlObject** instance that can then be passed as a parameter to a method or property accessed by the **XamlObject** class **setXamlProperty** or **invokeXamlMethod** method.

---

**Note** The JADE method must have parameters that are compatible with the .NET method expected.

---

The following method defined in a **XamlObject** subclass is an example of the **registerCallback** method.

```
layoutRoot_loaded(control: XamlGrid input; originalSource:
                    XamlObject input) updating, browserExecution;
vars
    dto      : DataDTO;
    callback : XamlObject;
begin
    // Set dragDrop callback
    callback := registerCallback(DrapDrop::onDragQuery);
    rightBox.invokeXamlMethod(DragDropManagerType &
        ";AddDragQueryHandler", root, callback);
    callback := registerCallback(DrapDrop::onDropQuery);
    rightBox.invokeXamlMethod(DragDropManagerType &
        ";AddDropQueryHandler", root, callback);
    callback := registerCallback(DrapDrop::onDragInfo);
    rightBox.invokeXamlMethod(DragDropManagerType &
        ";AddDragInfoHandler", root, callback);
    callback := registerCallback(DrapDrop::onDropInfo);
    rightBox.invokeXamlMethod(DragDropManagerType &
        ";AddDropInfoHandler", root, callback);
    create dto transient;
    rightBox.itemsSource := dto.getData();
    delete dto;
end;
```

## Extracting a Schema to Generate a XAP File (PAR 57458)

When a Silverlight schema is extracted to generate a XAML Application (XAP) file, properties of type **MemoryAddress** are now dropped.

## JADE Properties for XAML Controls inside DataTemplates (PAR 57187)

When you save XAML in JADE, any elements with an **x:Name** attribute have a corresponding control property representing the XAML document created in the **XamlDocument** subclass. These names can then be used in JADE logic to access properties of the controls.

During the C# and Silverlight project generation, this JADE logic is converted to C# and adds logic to call methods in a utility assembly to map the JADE property that corresponds to the control on to the actual Silverlight object.

However, when elements are part of a **DataTemplate**, they are not created on document load and the C# generated from your JADE logic is therefore unable to find the real Silverlight object. Accessing such properties caused exceptions to be raised. To avoid this, JADE does not create JADE properties for XAML controls that are inside **DataTemplates**, even if they do have an **x:Name** attribute.

Because subclasses of **DataTemplate** will not be detected when a document is loaded, existing XAML documents may raise exception 14451 if they have elements with a name attribute that are subelements of a **DataTemplate** element (or a **DataTemplate** subclass).

## XAML Browser (PAR 57192)

The following changes have been made to the XAML Browser.

- The XAML menu now provides the **Find Document** command, which accesses the Find Document dialog, to enable you to select from a list of XAML documents.

- The Edit menu is now displayed when the XAML Browser has focus. (This modified Edit menu does not provide the **Global Find** or the **Macro** command.)

This Edit menu is also displayed as the context (popup) menu for the XAML editor pane in the XAML Browser.

## Synchronized Database Service (SDS) (PAR 57217)

The following subsections describe the changes made to improve the utilization of the links between a primary database and its secondaries.

- Setting the socket buffer sizes (the default value is 128K bytes, compared to the Windows operating system default value of 8K bytes).
- Consolidating multiple small journal blocks into a single message.
- Changing secondary journal writing to use buffered I/O.

### JadeDatabaseAdmin::getCurrentJournalOffset Method

The **JadeDatabaseAdmin** class now provides the **getCurrentJournalOffset** method, which has the following signature.

```
getCurrentJournalOffset (currentJournal: Integer64 output;
                        currentOffset: Integer64 output;
                        lastSwitchJournal: Integer64 output;
                        lastSwitchOffset: Integer64 output;
                        nominalSize: Integer64 output);
```

The **getCurrentJournalOffset** method retrieves the journal number and byte offset of the last record written to the journal in the respective **currentJournal** and **currentOffset** parameters.

The **lastSwitchJournal** and **lastSwitchOffset** parameters retrieve the respective journal number and byte offset of the last record written to the penultimate journal.

The **nominalSize** parameter returns the nominal size of a journal file (that is, the value of the **JournalMaxSize** parameter in the [PersistentDb] section of the JADE initialization file).

These values enable you to calculate amounts and rates of journal output.

Use this method in conjunction with the **JadeDatabaseAdmin** class **sdsGetSecondaryProxy** method to determine the amount of journal data that has not been sent to the secondary.

### Changed JadeDatabaseAdmin::sdsGetSecondaryProxy Method

The following Integer64 attributes are now added to the returned **JadeDynamicObject** by the **JadeDatabaseAdmin** class **sdsGetSecondaryProxy** method.

- **totalSends**, which is the count of messages sent to the secondary.
- **totalBlocksSent**, which is the count of journal blocks sent to the secondary. There can be from 1 through 16 blocks per message.
- **totalBytesSent**, which is the count of bytes sent to the secondary; that is, the total size of all messages sent.
- **totalUncompressedBytes**, which is the count of bytes sent to the secondary if compression was disabled.

- `lastRecordSentJournal`, which is the journal number of the last journal record sent.
- `lastRecordSentOffset`, which is the byte offset of the last journal record sent.

## [ConnectionParams] Section `SocketBufferSize` Parameter

The [ConnectionParams] section of the JADE initialization file can now contain the `SocketBufferSize` parameter, which enables you to customize TCP/IP buffers to match the link characteristics.

The default value is 128K bytes, the minimum value is 8K bytes, and the maximum value is 16M bytes.

You can also set this parameter to zero (0), in which case the operating system default value is used (for example, Windows 7 has a default value of 8K bytes).

This parameter is read when the secondary opens a connection to the primary and when the primary accepts a connection from a secondary.

## **System::verifyDbEncryptionMasterKey Method (NFS 57338)**

The `System` class now provides the `verifyDbEncryptionMasterKey` method, which specifies whether the database encryption master key is present and correct; otherwise it returns `false`.