



## **OpenEdge Replication:** User Guide

Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. This manual is also copyrighted and all rights are reserved. This manual may not, in whole or in part, be copied, photocopied, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from Progress Software Corporation.

The information in this manual is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear in this document.

The references in this manual to specific platforms supported are subject to change.

A (and design), Actional, Actional (and design), Affinities Server, Allegrix, Allegrix (and design), Apama, Business Empowerment, ClientBuilder, ClientSoft, ClientSoft (and Design), Clientsoft.com, DataDirect (and design), DataDirect Connect, DataDirect Connect64, DataDirect Connect OLE DB, DataDirect Technologies, DataDirect XQuery, DataXtend, Dynamic Routing Architecture, EasyAsk, EdgeXtend, Empowerment Center, eXcelon, Fathom, IntelliStream, Neon, Neon New Era of Networks, O (and design), ObjectStore, OpenEdge, PDF, PeerDirect, Persistence, Persistence (and design), POSSENET, Powered by Progress, PowerTier, ProCare, Progress, Progress DataXtend, Progress Dynamics, Progress Business Empowerment, Progress Empowerment Center, Progress Empowerment Program, Progress Fast Track, Progress OpenEdge, Progress Profiles, Progress Results, Progress Software Developers Network, ProVision, PS Select, SequeLink, Shadow, ShadowDirect, Shadow Interface, Shadow Web Interface, ShadowWeb Server, Shadow TLS, SOAPStation, Sonic ESB, SonicMQ, Sonic Orchestration Server, Sonic Software (and design), SonicSynergy, SpeedScript, Stylus Studio, Technical Empowerment, Voice of Experience, WebSpeed, and Your Software, Our Technology—Experience the Connection are registered trademarks of Progress Software Corporation or one of its subsidiaries or affiliates in the U.S. and/or other countries. AccelEvent, Apama Dashboard Studio, Apama Event Manager, Apama Event Modeler, Apama Event Store, AppsAlive, AppServer, ASPen, ASP-in-a-Box, BusinessEdge, Cache-Forward, DataDirect Spy, DataDirect SupportLink, DataDirect XML Converters, Future Proof, Ghost Agents, GVAC, Looking Glass, ObjectCache, ObjectStore Inspector, ObjectStore Performance Expert, Panthero, POSSE, ProDataSet, Progress ESP Event Manager, Progress ESP Event Modeler, Progress Event Engine, Progress RFID, PSE Pro, SectorAlliance, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, Sonic, Sonic Business Integration Suite, Sonic Process Manager, Sonic Collaboration Server, Sonic Continuous Availability Architecture, Sonic Database Service, Sonic Workbench, Sonic XML Server, The Brains Behind BAM, WebClient, and Who Makes Progress are trademarks or service marks of Progress Software Corporation or one of its subsidiaries or affiliates in the U.S. and other countries. Vermont Views is a registered trademark of Vermont Creative Software in the U.S. and other countries. IBM is a registered trademark of IBM Corporation. JMX and JMX-based marks and Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. Any other trademarks or service marks contained herein are the property of their respective owners.

**Third party acknowledgements** — See the “Third party acknowledgements” section on page Preface–8.



February 2008

Last updated with new content: Release 10.1C

Product Code: 4496; R10.1C

For the latest documentation updates see the [OpenEdge Product Documentation category](http://www.psdn.com/library/kbcategory.jspa?categoryID=129) on PSDN (<http://www.psdn.com/library/kbcategory.jspa?categoryID=129>).

---

# Contents

---

<b>Preface</b> .....	<b>Preface-1</b>
<b>1. Overview of OpenEdge Replication</b> .....	<b>1-1</b>
What is OpenEdge Replication .....	1-2
Primary benefits. ....	1-2
Key features .....	1-2
OpenEdge Replication terminology and architecture .....	1-3
Primary and secondary databases .....	1-3
Source and target databases .....	1-3
OpenEdge Replication property files .....	1-5
Source and target architecture requirements .....	1-6
OpenEdge Replication server .....	1-6
OpenEdge Replication agent. ....	1-7
Using synchronous or asynchronous replication .....	1-10
OpenEdge Replication failure processing .....	1-12
Synchronization. ....	1-12
Server failure recovery .....	1-12
Agent failure recovery .....	1-12
Special database considerations .....	1-13
Database integrity .....	1-13
Database availability .....	1-13
Enhanced Read-Only mode .....	1-13
Schema lock .....	1-14
Forced truncation of the source database before-image file .....	1-14
Target quiet points. ....	1-15
Online backup of the target database .....	1-15
<b>2. Planning for OpenEdge Replication</b> .....	<b>2-1</b>
Preliminary planning tasks .....	2-2
Creating a database backup plan .....	2-2
Evaluating your production database .....	2-2
OpenEdge Replication and after-imaging .....	2-3
After-image extents .....	2-3
Calculating current after-image volume. ....	2-5
Replication and after-image extents .....	2-8

After-image extent sizing on a source database .....	2-10
Calculating fixed after-image extent size .....	2-10
Using variable extents .....	2-12
Determining OpenEdge Replication network bandwidth .....	2-14
Estimating network bandwidth .....	2-14
Anticipating additional network overhead .....	2-15
Additional references .....	2-16
Additional business considerations .....	2-17
Latency .....	2-17
Acceptable target database downtime .....	2-18
Appropriate failover behavior .....	2-18
Limitations and restrictions .....	2-19
Ensuring success when using after-imaging .....	2-19
A summary of OpenEdge Replication and after-imaging .....	2-20
<b>3. Implementing OpenEdge Replication .....</b>	<b>3-1</b>
Choosing the implementation method .....	3-3
Using the default implementation .....	3-3
Using the deferred agent startup implementation .....	3-4
Setting up the source database for OpenEdge Replication .....	3-7
Setting up the source database with an offline backup .....	3-7
Setting up the source database with an online backup .....	3-9
Setting up the target database .....	3-11
Configuring the OpenEdge Replication property files .....	3-12
Configuring the source database property file .....	3-12
Configuring the target database property file .....	3-14
Starting OpenEdge Replication .....	3-15
Starting the source database .....	3-16
Starting the target database .....	3-16
Message logging during startup .....	3-17
Restoring source and target databases .....	3-18
OpenEdge Replication startup and initialization process .....	3-19
Database connection considerations .....	3-20
Database access once OpenEdge Replication is running .....	3-21
Target database access .....	3-21
Source database access .....	3-22
Stopping OpenEdge Replication .....	3-23
Shutting down the source database .....	3-23
Shutting down the target database .....	3-24
Shutting down the OpenEdge Replication server .....	3-24
Terminating the OpenEdge Replication agent .....	3-25
Emptying AI extents on the source database .....	3-26
Latency reporting .....	3-27
OpenEdge Replication server perspective .....	3-27
OpenEdge Replication agent perspective .....	3-27
OpenEdge Replication utilities and commands .....	3-28
Normal OpenEdge Replication activity .....	3-29
Normal source database activity .....	3-29
Normal target database activity .....	3-29
Choosing a hot standby database .....	3-31
Handling OpenEdge Replication failure conditions .....	3-32
Database crash .....	3-32
Lost connection .....	3-32

Transitioning to the target database .....	3-35
Configuring transition .....	3-35
Transition processing .....	3-37
Manually applying after-image extents .....	3-38
Re-enabling OpenEdge Replication after transition .....	3-40
<b>4. OpenEdge Replication: From Failure to Recovery .....</b>	<b>4-1</b>
Overview of database failure recovery .....	4-2
OpenEdge Replication failure recovery terminology .....	4-2
The OpenEdge Replication property files .....	4-3
Transition .....	4-4
Enhancements to transition functionality .....	4-4
Planning for transition .....	4-5
How OpenEdge Replication works .....	4-7
Step 1: Primary replication before a failure .....	4-8
Step 2: Primary machine failure .....	4-8
Step 3: Entering pretransition .....	4-9
Step 4: Transitioning the target database to a source database .....	4-10
Step 5: Failover .....	4-11
Step 6: Primary machine repair complete .....	4-12
Step 7: Initiating primary database transition .....	4-13
Step 8: Secondary replication is performed .....	4-15
Step 9: The Replication fallback process .....	4-16
Recovery from transition failures .....	4-28
Transition logging .....	4-30
Transition properties .....	4-32
Setting transition properties .....	4-33
Sample startup parameter file .....	4-38
Source database startup parameter file .....	4-38
Target database startup parameter file .....	4-38
Normal database startup parameter file .....	4-38
Reference .....	4-39
Transition command actions .....	4-39
Transition properties summary .....	4-40
<b>5. Reference .....</b>	<b>5-1</b>
OpenEdge Replication property files .....	5-3
Sample OpenEdge Replication source properties file .....	5-3
Sample OpenEdge Replication target properties file .....	5-4
Sample OpenEdge Replication combined properties file .....	5-4
OpenEdge Replication properties .....	5-5
Server properties .....	5-5
Control agent properties .....	5-7
Agent properties .....	5-9
DSRUTIL utility .....	5-11
DSRUTIL applyextent qualifier .....	5-13
DSRUTIL canceldefer server qualifier .....	5-16
DSRUTIL connectagent database qualifier .....	5-17
DSRUTIL disablesitereplication qualifier .....	5-18
DSRUTIL monitor qualifier .....	5-19
DSRUTIL recovery qualifier .....	5-20
DSRUTIL relwaits qualifier .....	5-22
DSRUTIL restart server qualifier .....	5-23
DSRUTIL startagent database qualifier .....	5-24
DSRUTIL status qualifier .....	5-25
When the detail argument is not used .....	5-26

DSRUTIL terminate qualifier .....	5-28
DSRUTIL transition qualifier .....	5-29
DSRUTIL triggertransition qualifier .....	5-30
OpenEdge Replication DSRUTIL MONITOR .....	5-31
Startup menu .....	5-31
Replication server status .....	5-32
Replication server remote agents .....	5-35
Replication remote agents status .....	5-36
Replication agent status .....	5-40
Virtual system tables for OpenEdge Replication .....	5-44
Virtual system table field descriptions .....	5-44
Utilities and OpenEdge Replication .....	5-51
Starting OpenEdge Replication with the Progress Explorer and dbman .....	5-55
OpenEdge Replication and database management systems .....	5-56
<b>6. OpenEdge Replication Quick Command Summary .....</b>	<b>6-1</b>
Configuring the OpenEdge Replication properties files .....	6-2
Setting up the source and target databases .....	6-3
Configuring OpenEdge Replication with deferred agent startup .....	6-6
Configuring OpenEdge Replication for one agent .....	6-7
Configuring OpenEdge Replication for two agents .....	6-8
Starting OpenEdge Replication .....	6-9
Stopping OpenEdge Replication .....	6-10
Terminating the OpenEdge Replication server and agent .....	6-11
Configuring for automatic transition .....	6-12
Configuring for manual transition .....	6-13
Using manual transition .....	6-14
Re-enabling OpenEdge Replication after transition .....	6-15
Restarting OpenEdge Replication server after target shutdown .....	6-16
Monitoring an OpenEdge Replication database .....	6-17
<b>Index .....</b>	<b>Index-1</b>

## Figures

Figure 1-1:	OpenEdge Replication from one site to another .....	1-3
Figure 1-2:	Source and target databases .....	1-5
Figure 1-3:	OpenEdge Replication server .....	1-7
Figure 1-4:	OpenEdge Replication agent .....	1-8
Figure 1-5:	OpenEdge Replication model .....	1-9
Figure 1-6:	Synchronous and asynchronous configurations .....	1-10
Figure 1-7:	Asynchronous operation .....	1-11
Figure 1-8:	Synchronous operation .....	1-11
Figure 3-1:	AI blocks sent from server to agent .....	3-29
Figure 3-2:	Continuous roll forward on agent .....	3-30
Figure 3-3:	Server properties file with automatic transition .....	3-36
Figure 3-4:	Server properties file with manual transition .....	3-37
Figure 4-1:	Primary replication before a failure .....	4-8
Figure 4-2:	Primary machine failure .....	4-8
Figure 4-3:	Entering pretransition—applying all unapplied source database AI extents .....	4-9
Figure 4-4:	Transitioning the target to a source database .....	4-10
Figure 4-5:	Secondary database transitioned to source database .....	4-11
Figure 4-6:	Secondary database activity .....	4-12
Figure 4-7:	Primary machine repair complete .....	4-12
Figure 4-8:	Performing an online backup of secondary database .....	4-13
Figure 4-9:	Restoring backup of secondary database .....	4-13
Figure 4-10:	Transitioning source database to target .....	4-14
Figure 4-11:	Transition completes .....	4-15
Figure 4-12:	Secondary replication occurs .....	4-16
Figure 4-13:	Secondary replication continues (before transition failover) .....	4-17
Figure 4-14:	Scheduling downtime to perform failback .....	4-17
Figure 4-15:	Transitioning the primary database .....	4-18
Figure 4-16:	Transitioning the primary database .....	4-19
Figure 4-17:	Databases started in new roles .....	4-20
Figure 4-18:	Primary replication activity is occurring .....	4-21
Figure 4-19:	Secondary replication continues (before controlled transition) .....	4-22
Figure 4-20:	Scheduling downtime to perform failback .....	4-23
Figure 4-21:	Primary replication activity resumes .....	4-26
Figure 5-1:	DSRUTIL Monitor Startup menu screen .....	5-31
Figure 5-2:	OpenEdge Replication server status screen .....	5-32
Figure 5-3:	OpenEdge Replication server remote agents screen .....	5-35
Figure 5-4:	OpenEdge Replication Remote Agents Status screen .....	5-36
Figure 5-5:	OpenEdge Replication agent status screen .....	5-40

**Tables**

Table 2–1:	After-image extent types . . . . .	2–4
Table 2–2:	After-image extent states . . . . .	2–4
Table 2–3:	After-image extent sizing calculations (hourly data) . . . . .	2–11
Table 2–4:	After-image extent sizing calculations (weekly data) . . . . .	2–12
Table 3–1:	Connection results during startup and initialization . . . . .	3–20
Table 3–2:	Standard system-level access to target database . . . . .	3–21
Table 3–3:	System-level access to target database during transition . . . . .	3–22
Table 4–1:	Setting transition properties . . . . .	4–34
Table 4–2:	Transition command actions . . . . .	4–39
Table 4–3:	Transition properties . . . . .	4–40
Table 5–1:	Server properties . . . . .	5–5
Table 5–2:	Control agent properties . . . . .	5–8
Table 5–3:	Agent properties . . . . .	5–9
Table 5–4:	DSRUTIL utility qualifiers . . . . .	5–11
Table 5–5:	Replication status return codes . . . . .	5–25
Table 5–6:	Return code zero status code . . . . .	5–26
Table 5–7:	Status code values . . . . .	5–26
Table 5–8:	DSRUTIL monitor startup . . . . .	5–31
Table 5–9:	Replication server status . . . . .	5–33
Table 5–10:	Replication Server Remote Agents details . . . . .	5–35
Table 5–11:	Replication Remote Agents Status details . . . . .	5–37
Table 5–12:	Replication Agent Status details . . . . .	5–41
Table 5–13:	OpenEdge Replication virtual system tables . . . . .	5–44
Table 5–14:	Virtual system table _Repl-Server field description . . . . .	5–44
Table 5–15:	Virtual system table _Repl-AgentControl field description . . . . .	5–45
Table 5–16:	Virtual system table _Repl-Agent field descriptions . . . . .	5–48
Table 5–17:	Utility support for OpenEdge Replication . . . . .	5–51



---

# Preface

---

This Preface contains the following sections:

- Purpose
- Audience
- Organization
- Using this manual
- Typographical conventions
- Examples of syntax descriptions
- OpenEdge messages
- Third party acknowledgements

## Purpose

This book describes how to configure and use OpenEdge® Replication. In addition, it describes the underlying architecture of OpenEdge Replication.

## Audience

This book is intended for anyone familiar with OpenEdge® database administration who plans to set up and use OpenEdge Replication.

## Organization

### Chapter 1, “Overview of OpenEdge Replication”

Provides an overview of the functionality and architecture of OpenEdge Replication.

### Chapter 2, “Planning for OpenEdge Replication”

Provides a description of database considerations when using OpenEdge Replication.

### Chapter 3, “Implementing OpenEdge Replication”

Provides instructions on how to use OpenEdge Replication and describes the procedures it automatically performs.

### Chapter 4, “OpenEdge Replication: From Failure to Recovery”

Provides details about how to handle failure conditions and database failure on source or target machines, including information about transition and the failback process.

### Chapter 5, “Reference”

Provides reference information for OpenEdge Replication, including descriptions of properties, utilities, and virtual system tables.

### Chapter 6, “OpenEdge Replication Quick Command Summary”

Provides a quick command summary for setting up and using OpenEdge Replication.

## Using this manual

Use this manual to obtain an overview of OpenEdge Replication and its architecture, plan for and implement OpenEdge Replication, handle replication database recovery, and review replication reference and command summary information.

OpenEdge provides a special purpose programming language for building business applications. In the documentation, the formal name for this language is *ABL (Advanced Business Language)*. With few exceptions, all keywords of the language appear in all UPPERCASE, using a font that is appropriate to the context. All other alphabetic language content appears in mixed case.

For the latest documentation updates see the OpenEdge Product Documentation category on PSDN <http://www.psdn.com/library/kbcategory.jsps?categoryID=129>.

## References to ABL compiler and run-time features

ABL is both a compiled and an interpreted language that executes in a run-time engine. The documentation refers to this run-time engine as the *ABL Virtual Machine (AVM)*. When the documentation refers to ABL source code compilation, it specifies *ABL* or *the compiler* as the actor that manages compile-time features of the language. When the documentation refers to run-time behavior in an executing ABL program, it specifies *the AVM* as the actor that manages the specified run-time behavior in the program.

For example, these sentences refer to the ABL compiler's allowance for parameter passing and the AVM's possible response to that parameter passing at run time: "ABL allows you to pass a dynamic temp-table handle as a static temp-table parameter of a method. However, if at run time the passed dynamic temp-table schema does not match the schema of the static temp-table parameter, the AVM raises an error." The following sentence refers to run-time actions that the AVM can perform using a particular ABL feature: "The ABL socket object handle allows the AVM to connect with other ABL and non-ABL sessions using TCP/IP sockets."

## References to ABL data types



ABL provides built-in data types, built-in class data types, and user-defined class data types. References to built-in data types follow these rules:

- Like most other keywords, references to specific built-in data types appear in all UPPERCASE, using a font that is appropriate to the context. No uppercase reference ever includes or implies any data type other than itself.
- Wherever *integer* appears, this is a reference to the INTEGER or INT64 data type.
- Wherever *character* appears, this is a reference to the CHARACTER, LONGCHAR, or CLOB data type.
- Wherever *decimal* appears, this is a reference to the DECIMAL data type.
- Wherever *numeric* appears, this is a reference to the INTEGER, INT64, or DECIMAL data type.

References to built-in class data types appear in mixed case with initial caps, for example, `Progress.Lang.Object`. References to user-defined class data types appear in mixed case, as specified for a given application example.

## Typographical conventions

This manual uses the following typographical conventions:

Convention	Description
<b>Bold</b>	Bold typeface indicates commands or characters the user types, provides emphasis, or the names of user interface elements.
<i>Italic</i>	Italic typeface indicates the title of a document, or signifies new terms.
<b>SMALL, BOLD CAPITAL LETTERS</b>	Small, bold capital letters indicate OpenEdge key functions and generic keyboard keys; for example, <b>GET</b> and <b>CTRL</b> .
<b>KEY1+KEY2</b>	A plus sign between key names indicates a <b>simultaneous</b> key sequence: you press and hold down the first key while pressing the second key. For example, <b>CTRL+X</b> .
<b>KEY1 KEY2</b>	A space between key names indicates a <b>sequential</b> key sequence: you press and release the first key, then press another key. For example, <b>ESCAPE H</b> .
<b>Syntax:</b>	
Fixed width	A fixed-width font is used in syntax statements, code examples, system output, and filenames.
<i>Fixed-width italics</i>	Fixed-width italics indicate variables in syntax statements.
<b>Fixed-width bold</b>	Fixed-width bold indicates variables with special emphasis.
UPPERCASE fixed width	Uppercase words are ABL keywords. Although these are always shown in uppercase, you can type them in either uppercase or lowercase in a procedure.
	This icon (three arrows) introduces a multi-step procedure.
	This icon (one arrow) introduces a single-step procedure.
Period (.) or colon (:)	All statements except DO, FOR, FUNCTION, PROCEDURE, and REPEAT end with a period. DO, FOR, FUNCTION, PROCEDURE, and REPEAT statements can end with either a period or a colon.
[ ]	Large brackets indicate the items within them are optional.
[ ]	Small brackets are part of ABL.
{ }	Large braces indicate the items within them are required. They are used to simplify complex syntax diagrams.
{ }	Small braces are part of ABL. For example, a called external procedure must use braces when referencing arguments passed by a calling procedure.

Convention	Description
	A vertical bar indicates a choice.
...	Ellipses indicate repetition: you can choose one or more of the preceding items.

## Examples of syntax descriptions

In this example, `ACCUM` is a keyword, and *aggregate* and *expression* are variables:

### Syntax

```
ACCUM aggregate expression
```

`FOR` is one of the statements that can end with either a period or a colon, as in this example:

```
FOR EACH Customer:
  DISPLAY Name.
END.
```

In this example, `STREAM` *stream*, `UNLESS-HIDDEN`, and `NO-ERROR` are optional:

### Syntax

```
DISPLAY [ STREAM stream ] [ UNLESS-HIDDEN ] [ NO-ERROR ]
```

In this example, the outer (small) brackets are part of the language, and the inner (large) brackets denote an optional item:

### Syntax

```
INITIAL [ constant [ , constant ] ]
```

A called external procedure must use braces when referencing compile-time arguments passed by a calling procedure, as shown in this example:

### Syntax

```
{ &argument-name }
```

In this example, `EACH`, `FIRST`, and `LAST` are optional, but you can choose only one of them:

### Syntax

```
PRESELECT [ EACH | FIRST | LAST ] record-phrase
```

In this example, you must include two expressions, and optionally you can include more. Multiple expressions are separated by commas:

### Syntax

```
MAXIMUM ( expression , expression [ , expression ] ... )
```

In this example, you must specify MESSAGE and at least one *expression* or SKIP [ ( *n* ) ], and any number of additional *expression* or SKIP [ ( *n* ) ] is allowed:

### Syntax

```
MESSAGE { expression | SKIP [ ( n ) ] } ...
```

In this example, you must specify { *include-file*, then optionally any number of *argument* or &*argument-name* = "*argument-value*", and then terminate with }:

### Syntax

```
{ include-file  
  [ argument | &argument-name = "argument-value" ] ... }
```

## Long syntax descriptions split across lines

Some syntax descriptions are too long to fit on one line. When syntax descriptions are split across multiple lines, groups of optional and groups of required items are kept together in the required order.

In this example, WITH is followed by six optional items:

### Syntax

```
WITH [ ACCUM max-length ] [ expression DOWN ]  
    [ CENTERED ] [ n COLUMNS ] [ SIDE-LABELS ]  
    [ STREAM-IO ]
```

## Complex syntax descriptions with both required and optional elements

Some syntax descriptions are too complex to distinguish required and optional elements by bracketing only the optional elements. For such syntax, the descriptions include both braces (for required elements) and brackets (for optional elements).

In this example, ASSIGN requires either one or more *field* entries or one *record*. Options available with *field* or *record* are grouped with braces and brackets:

### Syntax

```
ASSIGN  { [ FRAME frame ] { field [ = expression ] }
        [ WHEN expression ] } ...
      | { record [ EXCEPT field ... ] }
```

## OpenEdge messages

OpenEdge displays several types of messages to inform you of routine and unusual occurrences:

- **Execution messages** inform you of errors encountered while OpenEdge is running a procedure; for example, if OpenEdge cannot find a record with a specified index field value.
- **Compile messages** inform you of errors found while OpenEdge is reading and analyzing a procedure before running it; for example, if a procedure references a table name that is not defined in the database.
- **Startup messages** inform you of unusual conditions detected while OpenEdge is getting ready to execute; for example, if you entered an invalid startup parameter.

After displaying a message, OpenEdge proceeds in one of several ways:

- Continues execution, subject to the error-processing actions that you specify or that are assumed as part of the procedure. This is the most common action taken after execution messages.
- Returns to the Procedure Editor, so you can correct an error in a procedure. This is the usual action taken after compiler messages.
- Halts processing of a procedure and returns immediately to the Procedure Editor. This does not happen often.
- Terminates the current session.

OpenEdge messages end with a message number in parentheses. In this example, the message number is 200:

```
** Unknown table name table. (200)
```

If you encounter an error that terminates OpenEdge, note the message number before restarting.

## Obtaining more information about OpenEdge messages

In Windows platforms, use OpenEdge online help to obtain more information about OpenEdge messages. Many OpenEdge tools include the following Help menu options to provide information about messages:

- Choose **Help**→**Recent Messages** to display detailed descriptions of the most recent OpenEdge message and all other messages returned in the current session.
- Choose **Help**→**Messages** and then type the message number to display a description of a specific OpenEdge message.
- In the Procedure Editor, press the **HELP** key or **F1**.

On UNIX platforms, use the OpenEdge pro command to start a single-user mode character OpenEdge client session and view a brief description of a message by providing its number.



**To use the pro command to obtain a message description by message number:**

1. Start the Procedure Editor:

`OpenEdge-install-dir/bin/pro`

2. Press **F3** to access the menu bar, then choose **Help**→**Messages**.
3. Type the message number and press **ENTER**. Details about that message number appear.
4. Press **F4** to close the message, press **F3** to access the Procedure Editor menu, and choose **File**→**Exit**.

## Third party acknowledgements

OpenEdge includes Imaging Technology copyrighted by Snowbound Software 1993-2003.  
[www.snowbound.com](http://www.snowbound.com).

OpenEdge includes software developed by the Apache Software Foundation (<http://www.apache.org/>). Copyright © 1999 The Apache Software Foundation. All rights reserved (Xerces C++ Parser (XML) and Xerces2 Java Parser (XML)); Copyright © 1999-2002 The Apache Software Foundation. All rights reserved (Xerces Parser (XML)); and Copyright © 2000-2003 The Apache Software Foundation. All rights reserved (Ant). The names “Apache,” “Xerces,” “ANT,” and “Apache Software Foundation” must not be used to endorse or promote products derived from this software without prior written permission. Products derived from this software may not be called “Apache”, nor may “Apache” appear in their name, without prior written permission of the Apache Software Foundation. For written permission, please contact [apache@apache.org](mailto:apache@apache.org). Software distributed on an “AS IS” basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License agreement that accompanies the product.

OpenEdge includes software copyrighted by DataDirect Technologies Corp., 1991-2007.



OpenEdge includes software developed by Vermont Creative Software. Copyright © 1988-1991 by Vermont Creative Software.

OpenEdge includes code licensed from RSA Security, Inc. Some portions licensed from IBM are available at <http://oss.software.ibm.com/icu4j/>.

OpenEdge includes the UnixWare platform of Perl Runtime authored by Kiem-Phong Vo and David Korn. Copyright © 1991, 1996 by AT&T Labs. Permission to use, copy, modify, and distribute this software for any purpose without fee is hereby granted, provided that this entire notice is included in all copies of any software which is or includes a copy or modification of this software and in all copies of the supporting documentation for such software. THIS SOFTWARE IS BEING PROVIDED “AS IS”, WITHOUT ANY EXPRESS OR IMPLIED WARRANTY. IN PARTICULAR, NEITHER THE AUTHORS NOR AT&T LABS MAKE ANY REPRESENTATION OR WARRANTY OF ANY KIND CONCERNING THE MERCHANTABILITY OF THIS SOFTWARE OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.

OpenEdge includes the RSA Data Security, Inc. MD5 Message-Digest Algorithm. Copyright ©1991-2, RSA Data Security, Inc. Created 1991. All rights reserved.

OpenEdge includes software developed by the World Wide Web Consortium. Copyright © 1994-2002 World Wide Web Consortium, (Massachusetts Institute of Technology, European Research Consortium for Informatics and Mathematics, Keio University). All rights reserved. This work is distributed under the W3C® Software License [<http://www.w3.org/Consortium/Legal/2002/copyright-software-20021231>] in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

OpenEdge includes Sonic software, which includes software developed by Apache Software Foundation (<http://www.apache.org/>). Copyright © 1999-2000 The Apache Software Foundation. All rights reserved. The names “Ant”, “Axis”, “Xalan”, “FOP”, “The Jakarta Project”, “Tomcat”, “Xerces” and/or “Apache Software Foundation” must not be used to endorse or promote products derived from the Product without prior written permission. Any product derived from the Product may not be called “Apache”, nor may “Apache” appear in their name, without prior written permission. For written permission, please contact [apache@apache.org](mailto:apache@apache.org).

OpenEdge includes Sonic software, which includes the JMX Technology from Sun Microsystems, Inc. Use and Distribution is subject to the Sun Community Source License available at <http://sun.com/software/communitysource>.

OpenEdge includes Sonic software, which includes software developed by the ModelObjects Group (<http://www.modelobjects.com>). Copyright © 2000-2001 ModelObjects Group. All rights reserved. The name “ModelObjects” must not be used to endorse or promote products derived from this software without prior written permission. Products derived from this software may not be called “ModelObjects”, nor may “ModelObjects” appear in their name, without prior written permission. For written permission, please contact [djacobs@modelobjects.com](mailto:djacobs@modelobjects.com).

OpenEdge includes Sonic software, which includes files that are subject to the Netscape Public License Version 1.1 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.mozilla.org/NPL/>. Software distributed under the License is distributed on an “AS IS” basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing

rights and limitations under the License. The Original Code is Mozilla Communicator client code, released March 31, 1998. The Initial Developer of the Original Code is Netscape Communications Corporation. Portions created by Netscape are Copyright © 1998-1999 Netscape Communications Corporation. All Rights Reserved.

OpenEdge includes software Copyright © 2003-2006, Terence Parr All rights reserved. Neither the name of the author nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. Software distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License agreement that accompanies the product.

OpenEdge includes ICU software 1.8 and later - Copyright © 1995-2003 International Business Machines Corporation and others All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

OpenEdge includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (<http://www.openssl.org/>). Copyright © 1998-2007 The OpenSSL Project. All rights reserved. This product includes cryptographic software written by Eric Young ([eay@cryptsoft.com](mailto:eay@cryptsoft.com)). This product includes software written by Tim Hudson ([tjh@cryptsoft.com](mailto:tjh@cryptsoft.com)). Copyright © 1995-1998 Eric Young ([eay@cryptsoft.com](mailto:eay@cryptsoft.com)) All rights reserved. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact [openssl-core@openssl.org](mailto:openssl-core@openssl.org). Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project. Software distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License agreement that accompanies the product.

OpenEdge includes Sonic software which includes a version of the Saxon XSLT and XQuery Processor from Saxonica Limited that has been modified by Progress Software Corporation. The contents of the Saxon source code and the modified source code file (Configuration.java) are subject to the Mozilla Public License Version 1.0 (the "License"); you may not use these files except in compliance with the License. You may obtain a copy of the License at <http://www.mozilla.org/MPL/> and a copy of the license (MPL-1.0.html) can also be found in the installation directory, in the Docs7.5/third\_party\_licenses folder, along with a copy of the modified code (Configuration.java); and a description of the modifications can be found in the Progress SonicMQ and Progress Sonic ESB v7.5 README files. Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License. The Original Code is The SAXON XSLT and XQuery Processor from Saxonica Limited. The Initial Developer of the Original Code is Michael Kay (<http://www.saxonica.com/products.html>). Portions created by Michael Kay are Copyright © 2001-2005. All rights reserved. Portions created by Progress Software Corporation are Copyright © 2007. All rights reserved.

OpenEdge includes software developed by IBM. Copyright © 1995-2003 International Business Machines Corporation and others. All rights reserved. Permission is hereby granted,

free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation. Software distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License agreement that accompanies the product. Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder. OpenEdge includes Sonic software, which includes software developed by ExoLab Project (<http://www.exolab.org/>). Copyright © 2000 Intalio Inc. All rights reserved. The names "Castor" and/or "ExoLab" must not be used to endorse or promote products derived from the Products without prior written permission. For written permission, please contact [info@exolab.org](mailto:info@exolab.org). Exolab, Castor and Intalio are trademarks of Intalio Inc.

OpenEdge includes Sonic software, which includes software Copyright © 1999 CERN - European Organization for Nuclear Research. Permission to use, copy, modify, distribute and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. CERN makes no representations about the suitability of this software for any purpose. It is provided "as is" without expressed or implied warranty.

OpenEdge includes Sonic software, which includes software developed by the University Corporation for Advanced Internet Development <http://www.ucaid.edu> Internet2 Project. Copyright © 2002 University Corporation for Advanced Internet Development, Inc. All rights reserved. Neither the name of OpenSAML nor the names of its contributors, nor Internet2, nor the University Corporation for Advanced Internet Development, Inc., nor UCAID may be used to endorse or promote products derived from this software and products derived from this software may not be called OpenSAML, Internet2, UCAID, or the University Corporation for Advanced Internet Development, nor may OpenSAML appear in their name without prior written permission of the University Corporation for Advanced Internet Development. For written permission, please contact [opensaml@opensaml.org](mailto:opensaml@opensaml.org).

OpenEdge includes DataDirect products for the Microsoft SQL Server database which contains a licensed implementation of the Microsoft TDS Protocol.

OpenEdge includes Sonic software, which includes code licensed from Mort Bay Consulting Pty. Ltd. The Jetty Package is Copyright © 1998 Mort Bay Consulting Pty. Ltd. (Australia) and others.



# Overview of OpenEdge Replication

---

This chapter provides an introduction to OpenEdge® Replication, as described in the following sections:

- [What is OpenEdge Replication](#)
- [OpenEdge Replication terminology and architecture](#)
- [Using synchronous or asynchronous replication](#)
- [OpenEdge Replication failure processing](#)
- [Special database considerations](#)

## What is OpenEdge Replication

Data replication has two major real-time functions:

- To distribute copies of information to one or more sites
- To provide failure recovery to keep data constantly available to customers

OpenEdge Replication automatically replicates a local OpenEdge® database to remote OpenEdge databases running on one or more machines. Once OpenEdge Replication is installed, configured, and started, replication happens automatically.

OpenEdge Replication offers users the ability to keep OpenEdge databases identical while also providing a hot standby in case a database fails. When a database fails, another becomes active. Therefore, mission-critical data is always available to your users.

### Primary benefits

OpenEdge Replication provides the following benefits:

- Availability of mission-critical data 24 hours a day, seven days a week
- Minimal or no disruption in the event of unplanned downtime or disaster

### Key features

OpenEdge Replication provides the following features:

- Automated, real-time replication of databases for failover or disaster recovery
- Failback functionality
- A single source database and one or two target database configurations
- Data integrity between source and target databases
- Continued source database activity while administration tasks are being performed
- Replication activity reporting
- Online backup of source and target databases

# OpenEdge Replication terminology and architecture

The following sections define OpenEdge Replication terminology and describe the application's supporting architecture:

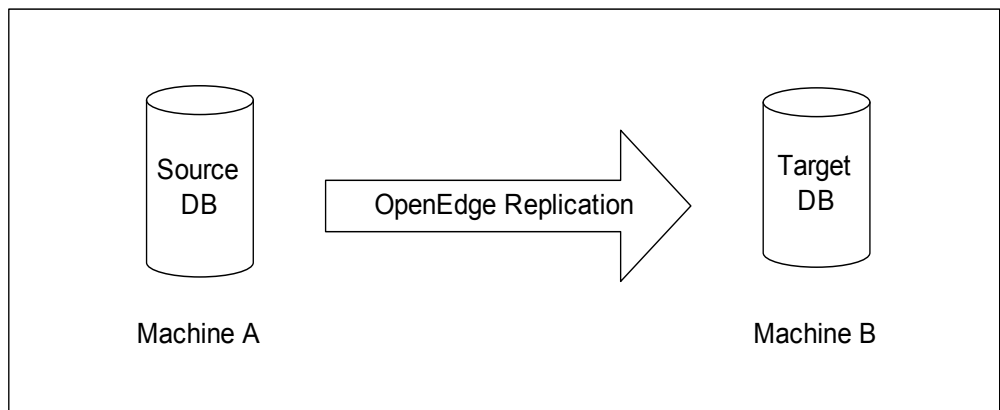
- [Primary and secondary databases](#)
- [Source and target databases](#)
- [OpenEdge Replication property files](#)
- [Source and target architecture requirements](#)
- [OpenEdge Replication server](#)
- [OpenEdge Replication agent](#)

## Primary and secondary databases

A *primary database* is the database that is updated from your application; the primary database is the one you initially enable as the OpenEdge Replication source database. A *secondary database* is the initial replica of the primary database; the secondary database is the one you initially enable as the OpenEdge Replication target database.

## Source and target databases

OpenEdge Replication makes a copy of one database onto another and keeps those two databases identical. As shown in [Figure 1–1](#), one database must be considered the source and the other the target.



**Figure 1–1: OpenEdge Replication from one site to another**

### The source database

The source database is the database that can be updated by users. The source database:

- Is where users do their work and make their database updates.
- Has both write access and read access.
- Is not considered a source database until it is enabled as an OpenEdge Replication source database.
- Is the database from which the OpenEdge Replication server replicates data to the target database.
- Must have after-imaging (AI) activated. The after-imaging feature lets you recover a database that was damaged when a failure caused the loss of the database or primary recovery (before-image) area.

When you enable after-imaging, the database engine writes notes containing a description of all database changes to the after-image files.

- Uses after-imaging to capture all database activity performed. This activity is then sent to the target database.

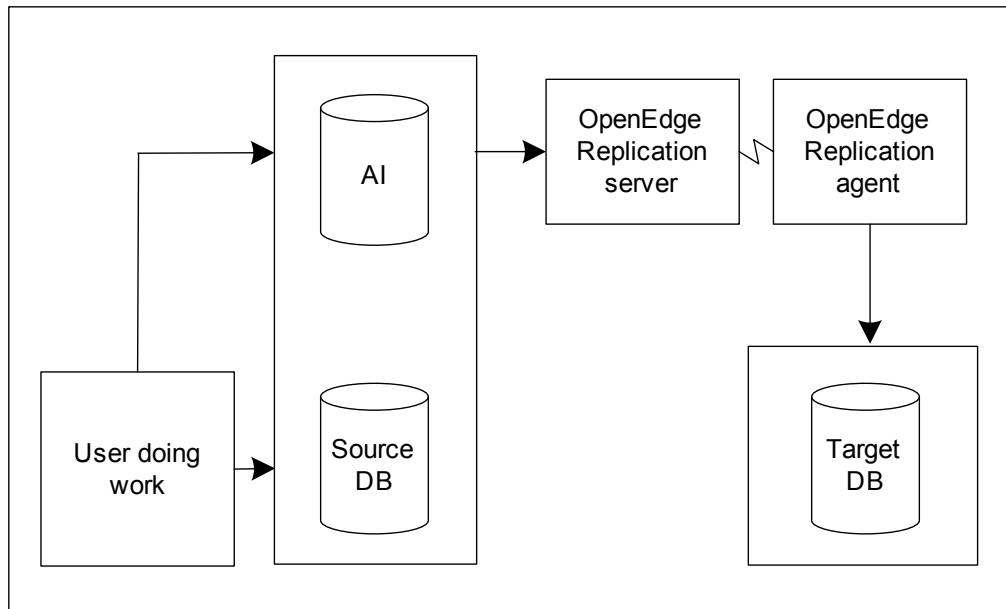
### The target database

The target database is an identical copy of the source database. A second target database can be set up for additional redundancy. The target database:

- Originates from the source database and contains the same data, schema, and logical structure as the source database.
- Is updated solely by the OpenEdge Replication agent.
- Does not allow database updates by anything other than OpenEdge Replication.
- Permits queries and reports by users, as well as any non-database write activity (database utilities, for example) if OpenEdge Replication Plus is installed. For information on what your OpenEdge Replication installation supports, see the [“Database access once OpenEdge Replication is running”](#) section on page 3–21. See the [“Utilities and OpenEdge Replication”](#) section on page 5–51 for supported non-database write activity.



Figure 1–2 shows the source and target databases in the OpenEdge Replication model.



**Figure 1–2: Source and target databases**

---

**Note:** The source and target databases should reside on separate machines so the target can run if the source machine fails.

---

## OpenEdge Replication property files

The OpenEdge Replication source database and target database properties are stored in property files. Each property file contains information specific to the source or target database. For example, the source properties file contains details including, among other things, the name of the source database, the agent or agents the replication server controls, what type of transition the agent will perform, and how long a replication agent will wait after losing contact with the replication server before performing automatic transition. The target properties file contains details such as the name of the agent and the name of the target database.

OpenEdge Replication provides two separate **sample** property files: the `source.rep1.properties` file and the `target.rep1.properties` file. You can copy each of these files to modify and use as your replication property files.

If you want, you can also combine the files into one property file.

## Source and target architecture requirements

Before OpenEdge Replication starts, the source and target databases are automatically checked to ensure that the databases are identical in the following ways:

- The logical structure—but not necessarily the physical structure—of the databases. (All user-defined areas must be identical except the AI areas.)
- The versions of the databases.
- The database block sizes.
- If large file support is enabled, it must be enabled on both databases.
- Before-Image (BI) block sizes.

Once OpenEdge Replication is configured and running, it will propagate any source database changes to the target database. For OpenEdge Replication to function properly, follow these source and target database guidelines:

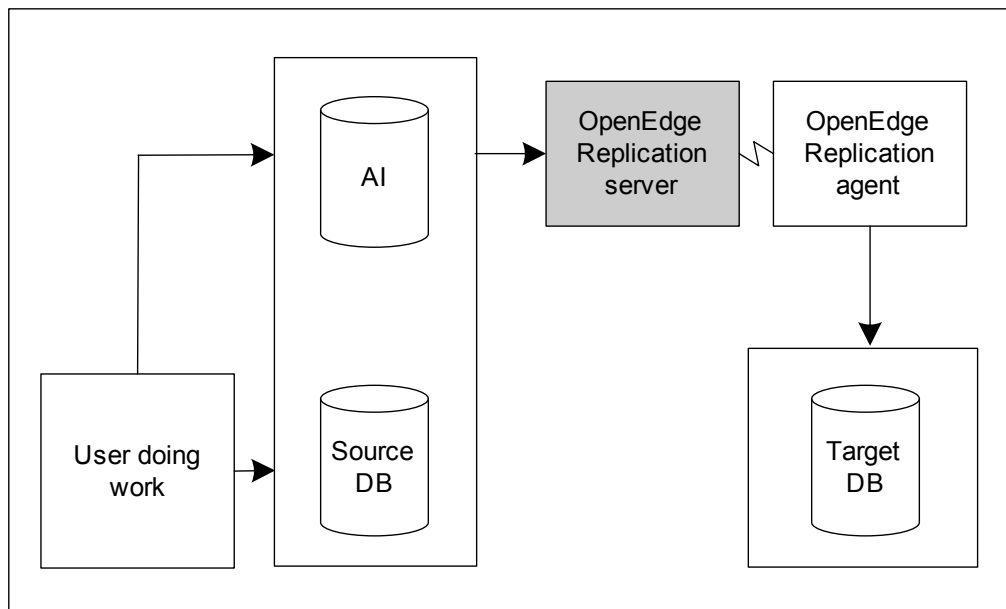
- For safety, the target database should reside on a different machine than the source.
- Both the source and the target machines must have the same *endian ordering*, which is the way the multiple byte integers are stored in memory—either by MSB (most-significant byte) or LSB (least-significant byte). Those systems storing by MSB are called Big Endian, and those storing by LSB are called Little Endian.

The term *endianess* is used in general to describe when binary files are portable between platforms; those platforms with the same endianess may use binary data transparently. Typically, UNIX machines and Windows machines use different endian ordering for storage. Therefore, a Windows source database can be replicated to another Windows machine, but not to an HPUX machine. An HPUX source database can be replicated to another HPUX machine.

## OpenEdge Replication server

For OpenEdge Replication to succeed in keeping the source and target databases identical, communication has to take place so that transaction log records from the source are propagated to the target. The OpenEdge Replication server connects to the source database and sends any updates made there to the OpenEdge Replication agent on the target machine or machines. Also, the OpenEdge Replication server process provides communications for startup, schema locks, and server-side recovery if a failure occurs.

Figure 1–3 shows the OpenEdge Replication server in the OpenEdge Replication model.



**Figure 1–3: OpenEdge Replication server**

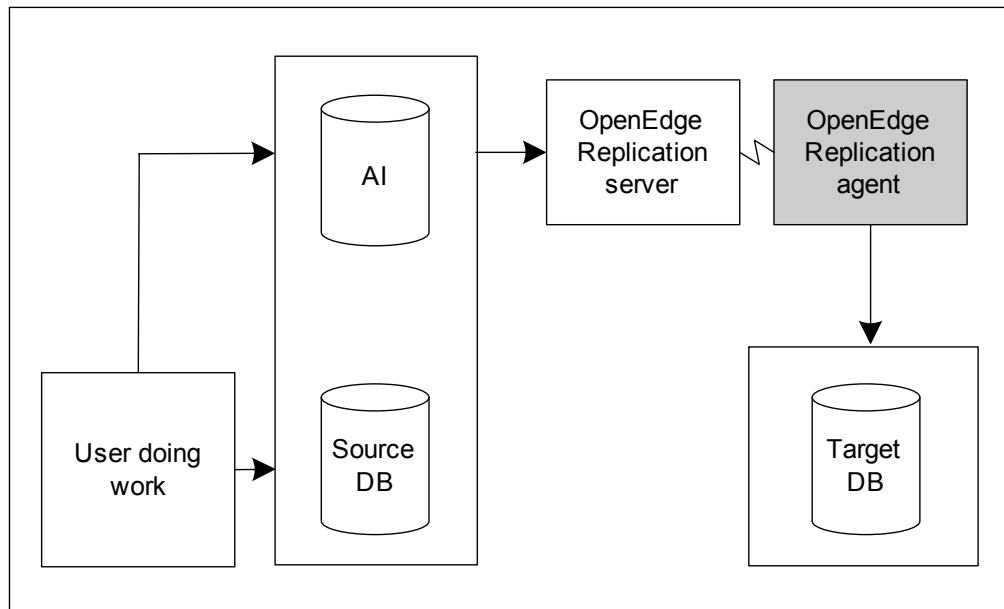
The OpenEdge Replication server must reside on the source database machine. The OpenEdge Replication server:

- Connects to the source database and establishes and maintains communications with the database server.
- Establishes, maintains, and controls communication with the OpenEdge Replication agents.
- Sends source database updates to the target database to keep them identical. This is achieved by using the AI transaction log where AI blocks of information are sent to the target.

## OpenEdge Replication agent

For the OpenEdge Replication server to succeed in keeping the source and target databases identical, it needs an OpenEdge Replication agent to receive information and perform updates to the target database. The OpenEdge Replication agent process receives configuration and operating instructions from the OpenEdge Replication server, including what actions to follow if connection to the OpenEdge Replication server is lost.

Figure 1–4 shows the OpenEdge Replication agent process in the OpenEdge Replication model.



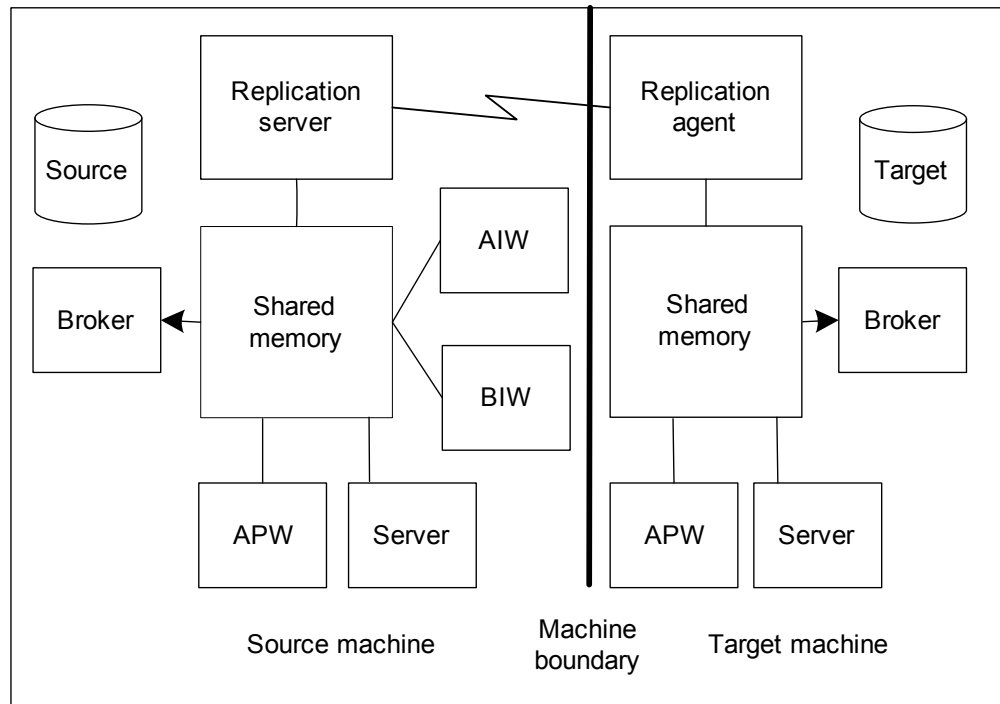
**Figure 1–4: OpenEdge Replication agent**

The OpenEdge Replication agent performs the actual process of updating the target database. The agent must reside on the target database machine and provides the following services:

- Updates the target database to keep it identical with the source database by using the AI blocks sent from the OpenEdge Replication server.
- Performs a continuous roll forward of the source database activity to the target database.
- Places the target database into Enhanced Read-Only mode. For more information about Enhanced Read-Only mode, see the [“Enhanced Read-Only mode”](#) section on page 1–13. For information about what your OpenEdge Replication installation supports, see the [“Database access once OpenEdge Replication is running”](#) section on page 3–21.

The OpenEdge Replication model with source and target databases, OpenEdge Replication server, and OpenEdge Replication agent coexists with a standard database model with its servers, brokers, and other processes.

Figure 1–5 illustrates the OpenEdge Replication model coexisting in a standard database environment.



**Figure 1–5: OpenEdge Replication model**

As shown in Figure 1–5, the source and target databases **can** be on the same machine. However, the source and target database **should** reside on separate machines so the target can run if the source machine fails.

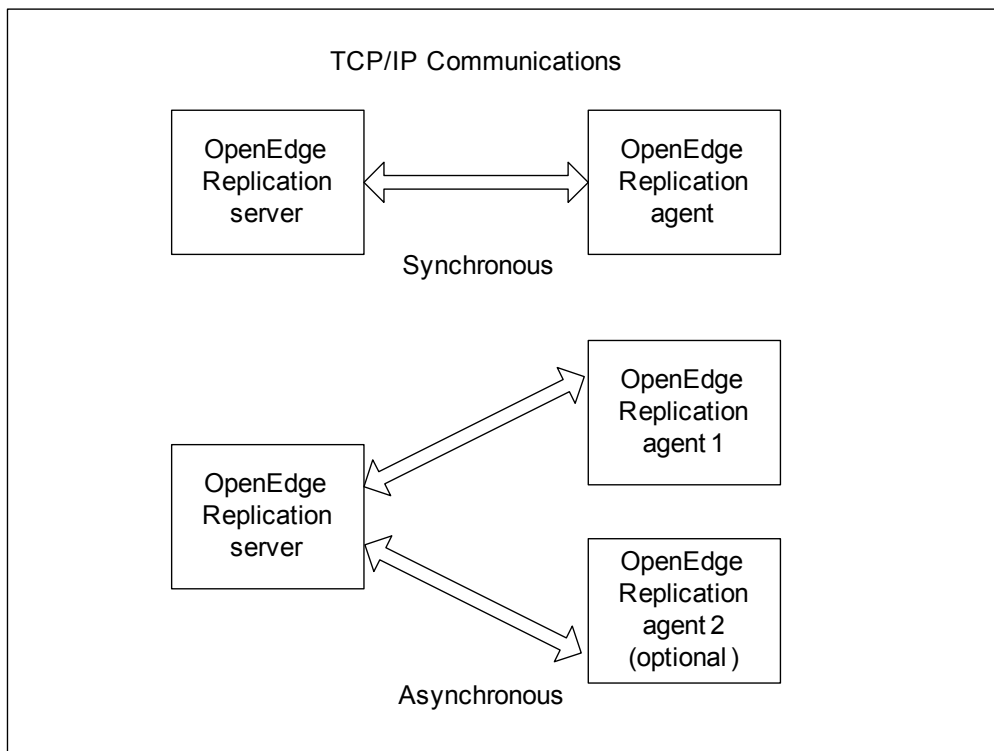
### Designating a critical agent

OpenEdge Replication allows a maximum of two agents per server. You can designate one agent as critical when you want to automatically transition to that agent's target database during failure processing. For example, if you have two target databases, only the database managed by a critical agent (with the `transition` property set to `auto`) will automatically transition during failure processing.

There can be, at most, one critical agent defined. If a second critical agent is configured, it will be changed to a noncritical agent. All agents by default are noncritical. To change an agent to critical, the `critical` property must be set to 1 in the `source-db-name.rep1.properties` file. For more information, see the "OpenEdge Replication properties" section on page 5–5.

## Using synchronous or asynchronous replication

OpenEdge Replication supports two methods of replication: synchronous and asynchronous. Figure 1–6 shows both synchronous and asynchronous methods of replication.

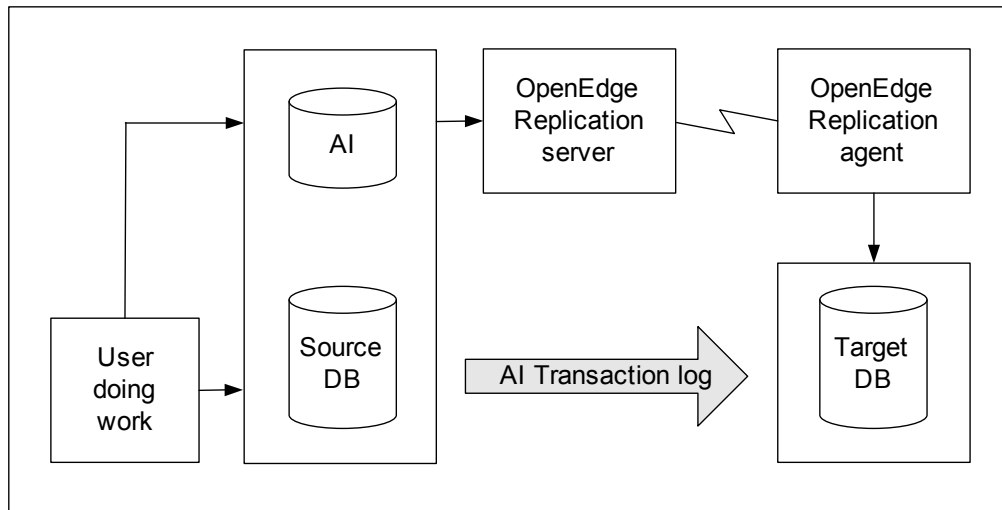


**Figure 1–6: Synchronous and asynchronous configurations**

In Figure 1–6, the asynchronous replication supports a maximum of two OpenEdge Replication agents and the synchronous replication supports only one OpenEdge Replication agent. There is one OpenEdge Replication agent for every target database.

During asynchronous operation, the user changes records and the transactions are committed without acknowledgement and sent back to the OpenEdge Replication server. Without waiting, the OpenEdge Replication server sends more AI blocks from the AI transaction log to the OpenEdge Replication agent, and the OpenEdge Replication agent applies these changes to the target database. Of the two configurations (synchronous and asynchronous), asynchronous performs better.

Figure 1–7 shows asynchronous operation in the OpenEdge Replication model.

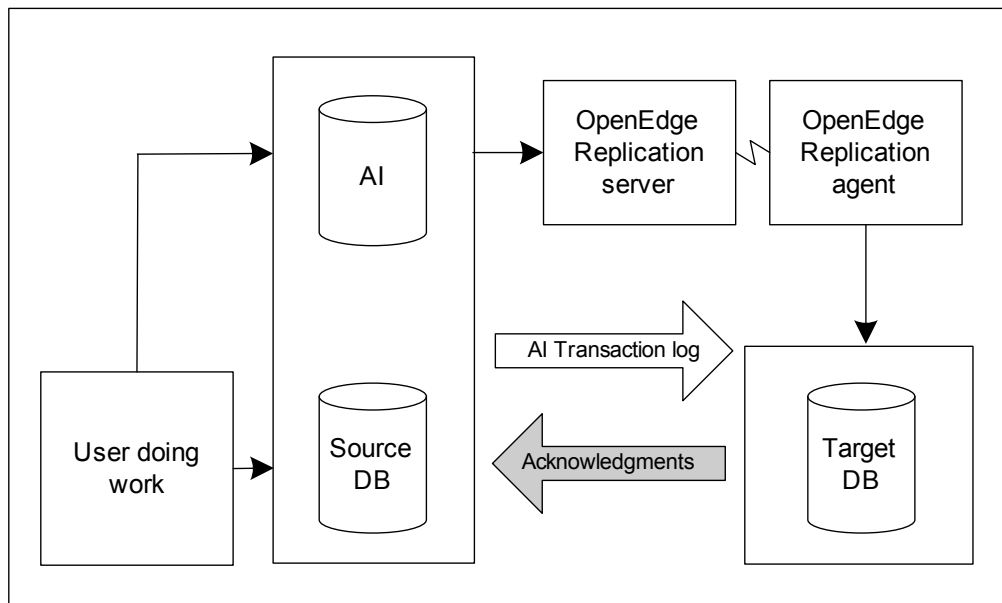


**Figure 1–7: Asynchronous operation**

During synchronous connection, the user changes records and the transactions are committed. When the OpenEdge Replication agent encounters a transaction end, it sends an acknowledgment back to the OpenEdge Replication server. The committing user will block (wait) until the transaction is fully applied to the target database. Other users are not blocked during this activity.

Of the two configurations, synchronous is the safest; however, it is also a low-performance option. For more information on choosing asynchronous versus synchronous mode, see the “Choosing a hot standby database” section on page 3–31.

Figure 1–8 shows synchronous operation in the OpenEdge Replication model.



**Figure 1–8: Synchronous operation**

Because of the user blocks in the synchronous model, performance will be much slower than in the asynchronous model.

## OpenEdge Replication failure processing

If a failure does occur, such as a lost TCP/IP connection between the OpenEdge Replication server and the OpenEdge Replication agent, failure processing starts. If a failure occurs on the OpenEdge Replication server, it starts a process known as *failure recovery*. If a failure occurs on the OpenEdge Replication agent, the agent prepares to perform *transition*, which takes place if the connection to the server remains lost. For OpenEdge Replication to continue normal operations after a failure and after establishing connection again, the OpenEdge Replication server and the OpenEdge Replication agent must perform *synchronization*.

The following sections document synchronization and server and agent failure recovery.

### Synchronization

Synchronization is the process performed by the OpenEdge Replication server and the OpenEdge Replication agent to ensure that the source database and target database are identical. Synchronization is performed during database and replication startup and during failure recovery processing.

### Server failure recovery

Failure recovery is the process that the OpenEdge Replication server performs after a communication failure with the OpenEdge Replication agent. During failure recovery, the OpenEdge Replication server attempts to reconnect to an OpenEdge Replication agent or agents that have lost connection with the OpenEdge Replication server. After connection is re-established, the OpenEdge Replication server and OpenEdge Replication agent attempt to synchronize. Once synchronization is complete, normal operations resume.

If the operation is unsuccessful, the OpenEdge Replication server will remove the failed agent from its list of OpenEdge Replication agents to replicate to. If all OpenEdge Replication agents fail and connection cannot be established, the OpenEdge Replication server will stop. If the server stops, source databases activities continue and AI extents continue to fill.

### Agent failure recovery

*Transition* is the action that the OpenEdge Replication agent performs after a communication failure has occurred with the OpenEdge Replication server. Within the `transition-timeout` value (as defined in `source-db-name.rep1.properties`, the OpenEdge Replication server properties file), the OpenEdge Replication agent listens for connection requests from the OpenEdge Replication server. After connection is established, the OpenEdge Replication server and OpenEdge Replication agent perform synchronization. Once synchronization is complete, normal operations resume.

If the OpenEdge Replication agent does not receive a communication attempt within the `transition-timeout` value, and `auto-transition` has been configured, the target database is transitioned into a normal OpenEdge database. If manual transition has been configured, the OpenEdge Replication agent will continue to wait for connection until either manual transition is performed using the DSRUTIL utility or the database is shut down. For more information about the DSRUTIL utility, see the “[DSRUTIL utility](#)” section on page 5–11.



## Special database considerations

The following sections describe database considerations related to using OpenEdge Replication.

### Database integrity

OpenEdge Replication provides failure processing to help ensure database integrity, which is a major concern of all database users. The need for database integrity is compounded because when OpenEdge Replication is in use, two or more databases must remain identical.

OpenEdge Replication achieves database integrity by performing synchronization. However, OpenEdge Replication is not a substitute for a comprehensive backup and recovery plan.

When a source database is started, various operations are performed by the database manager to guarantee database resiliency. Some of the operations change the database. When a source database is started, the database changes must be replicated to any and all target databases. OpenEdge Replication achieves this by performing synchronization during startup. During the startup synchronization process, all database activity that might have occurred during database startup is replicated to the target database.

### Database availability

The primary function of OpenEdge Replication is to provide database availability in the event of a failure. Since OpenEdge Replication replicates a source database to one or more target databases, either target database can be used as a hot standby should the source database become unavailable. This is accomplished through the process of transition.

For more information about transition, see [Chapter 4, “OpenEdge Replication: From Failure to Recovery.”](#)

### Enhanced Read-Only mode

When the OpenEdge Replication agent is running, the database is placed into an *Enhanced Read-Only* mode (ERO). ERO mode enforces user read-only functionality while providing the benefits of OpenEdge Replication multi-user access to a database. For information on what your OpenEdge Replication installation supports, see the “[Database access once OpenEdge Replication is running](#)” section on page 3–21.

When the target database is opened and the OpenEdge Replication agent is started, ERO is implicitly set. Any process that connects to the target database cannot make database updates, with the exception of the OpenEdge Replication agent. Any process connected to a target database cannot lock records.

ERO mode, unlike a client’s read-only functionality, is a database server concept. ERO mode offers full database capabilities with a buffer pool, shared buffers, and read-only private buffers. The read-only (-RO) restriction is set at the client side. ERO is a database restriction.

When a failure occurs and the target database transitions to a normal OpenEdge database, all future connections to the target database are full access. All current connections are disconnected; when they reconnect, they are full access. See the [“OpenEdge Replication failure processing”](#) section on page 1–12 for more information about the transitioning of a target database to a normal OpenEdge database.

### Schema lock

Whenever the schema is changed on the source database, a schema lock is required. The OpenEdge Replication server communicates a schema lock request to the OpenEdge Replication agent, causing a schema lock on the target database as well. The OpenEdge Replication agent will write a message to the target database log indicating that there is a schema lock requested.

By default, this lock will be held until the schema changes are completed on both the source and target databases. If a user in read-only mode on the target database is accessing tables, the schema lock cannot occur until the user releases the tables. The user process requesting the schema lock will block until it gets the schema lock. The DBA must ensure that users on the read-only target database do not prevent a client on the source database from making a schema change.

Alternatively, you can set the Schema-Lock-Action server property to force. The agent will attempt to acquire the exclusive schema lock five times. If the fifth attempt fails, the agent disconnects all users from the target and makes another attempt. If the last attempt fails, the server and all agents terminate. This allows the source database to resume normal activity. When schema update activity completes, the server and target can be restarted.

For more information, see the [“OpenEdge Replication properties”](#) section on page 5–5.

### Forced truncation of the source database before-image file

You can truncate the source database BI by using the following command (where *db-name* is the name of your database):

```
proutil db-name -C truncate bi -F
```

When you execute the command, after-imaging is disabled. When the source database is restarted, the following actions occur:

1. A message is logged to indicate that after-imaging is no longer enabled.
2. OpenEdge Replication is disabled.
3. The database is shut down.

Disabling OpenEdge Replication allows all OpenEdge database utilities to be used to aid in database recovery. To use OpenEdge Replication again on this database, the database must be recovered and then re-enabled for OpenEdge Replication.

## Target quiet points

If there is a quiet point enabled on the target, the source database will block until the target quiet point is disabled.

## Online backup of the target database

In previous releases of OpenEdge Replication, you could perform an online backup of the target database only during transition. Beginning with this release, you can perform an online backup of the target database outside of transition, while OpenEdge Replication is running. The OpenEdge Replication agent is the only process that can update the target database; in the case of the online backup, however, no changes to the database itself occur. The backup process locks buffers and blocks but not database records.

When an online backup of the target database is underway, activity continues on the source database as long as:

- Asynchronous replication is being performed.
- The Replication server is able to acquire a shared schema lock on the source database. The server must acquire the schema lock to block updates to the source database schema during the synchronization process.

Source database activity continues while the online backup is being performed on the target database, as long as there is enough available AI extent space.

### To begin

You begin the online backup process by using the PROBKUP command:

```
probkup online db-name [incremental] device-name [parameters]
```

In which:

*online*

Indicates the backup is taking place online.

*db-name*

Specifies the database you want to back up.

*incremental*

Indicates that the backup is an incremental backup.

### *device-name*

Identifies a special device (for example, a tape drive) or a standard file. If *device-name* identifies a special device, PROBKUP assumes the device has removable media, such as a tape or a floppy diskette. For Windows, use `\\.\tape0` for the device name if you are backing up to a tape drive.

### *parameters*

Indicates any additional parameters you want to use with PROBKUP.

For more general details about database backup, see *OpenEdge Data Management: Data Administration*.

## What happens during an online backup of the target database

For the online backup of the target database to be successful, coordination between the backup process and OpenEdge Replication is necessary. The following is a summary of how the backup process and OpenEdge Replication work together to ensure that the backup proceeds properly once you enter the PROBKUP command:

1. If the Replication agent is active:
  - a. The online backup utility sends a message to the Replication agent indicating that an online backup is about to begin.
  - b. The Replication agent informs the Replication server that an online backup of the target database is about to start, provided that:
    - The target database is not currently blocked (due to a quiet point, BI stall, AI stall, or another online backup).
    - The Replication agent is connected to the Replication server.
    - Synchronization is not currently being performed.
2. One of the following actions then occurs:
  - If the Replication server **can** acquire a shared schema lock on the source database and asynchronous replication is being performed, the Replication server will indicate to the RDBMS that it is busy. This allows activity on the source database to continue.  
  
The Replication server then sends a positive response to the Replication agent, and activity continues as described in [Step 3](#).
  - If the Replication server **cannot** acquire the shared schema lock, the server notifies the Replication agent, and the Replication agent then notifies the user who started the backup, that the backup cannot be performed at this time.

3. If the Replication agent receives a positive response from the Replication server (as described in [Step 2](#)), the online backup is allowed to continue.
4. The online backup utility then performs the online backup of the target database.
5. When the backup finishes, a completion message is sent to the Replication agent.
6. The Replication agent then sends a completion message to the Replication server.
7. The Replication server enters recovery and begins synchronization, which updates the target database with all activity that occurred on the source database while the online backup was running on the target.

Once recovery synchronization completes, the Replication server returns to normal processing.

8. Online backup completes processing.



## Planning for OpenEdge Replication

---

This chapter describes how to plan for OpenEdge Replication, as outlined in the following sections:

- Preliminary planning tasks
- OpenEdge Replication and after-imaging
- Replication and after-image extents
- After-image extent sizing on a source database
- Determining OpenEdge Replication network bandwidth
- Additional business considerations
- A summary of OpenEdge Replication and after-imaging

## Preliminary planning tasks

Before you begin using OpenEdge Replication, you must perform some preliminary tasks. Specifically, you must:

- Create a database backup plan.
- Evaluate your production database.

---

**Note:** OpenEdge Replication requires activation of after-imaging on the source database. If you are unfamiliar with after-imaging, see *OpenEdge Data Management: Database Administration*.

---

### Creating a database backup plan

Prior to planning for OpenEdge Replication, be sure that you have a fully functional backup plan in place. For more information about how best to consider and implement a proper backup plan and schedule for your database, see *OpenEdge Data Management: Database Administration*.

### Evaluating your production database

With a good database backup plan in place, consider the following factors related to your database environment before running OpenEdge Replication:

- **The database you will replicate** — Choose a database that your users expect to have constant access to.
- **System resources** — Be sure you have enough resources to implement after-image processing on the source database. When you turn on AI, the transaction logs generated could consume significant disk space.
- **The machine the source database will reside on** — Choose reliable hardware for your source database so that machine downtime does not interfere with OpenEdge Replication and user access to the database. The machine you choose should have enough CPU and memory to support the addition of OpenEdge Replication.
- **The machine the target database will reside on** — Typically, the target database is on a different machine than the source database. This configuration is advantageous since a failure of the source machine would not prevent users from using the target.
- **The logical structure of the source and target databases** — Modify the source copy of the structure file on your target machine, if the logical structure of both the source and target databases is not identical.
- **Reliable TCP/IP communications between the source and target database** — This is a key element in keeping your source and target databases up to date. Without reliable communications between the OpenEdge Replication server and the OpenEdge Replication agent, OpenEdge Replication must spend time in failure recovery, which will cause interrupted user access to your databases.



## OpenEdge Replication and after-imaging

OpenEdge Replication requires that the after-imaging feature be activated on the source database. The after-imaging feature allows you to recover a database that was damaged when a failure caused the loss of the database or primary recovery (before-image) area. When you enable after-imaging, the database engine writes notes containing a description of all database changes to the after-image logs.

Before you implement OpenEdge Replication, it is recommended that you calculate, on average, how much after-image volume your database currently generates. You can then use this average to estimate the network bandwidth that establishing replication for the database will require.

After-image data plays the largest part in determining how much bandwidth will be needed. Although exact data volume cannot be accurately calculated, as it depends on the type of transactions within the application and how much activity is going on against the database, it can be estimated if you know the average amount of after-image volume on your system.

The information in this section provides details related to using after-imaging with OpenEdge Replication and calculating its effects on database volume and network bandwidth. For more general information about database after-imaging, see *OpenEdge Data Management: Data Administration*.

### After-image extents

*After-image extents* contain a series of notes grouped together into after-image blocks. You can use the after-image extents with the roll-forward recovery process to restore the database to the condition it was in before you lost the database, without losing completed transactions that occurred since the last backup.

#### After-image extent types

There are two types of after-image extents: *fixed-length* and *variable-length*. As a general rule, fixed-length extents are preferable to variable-length extents. There are, however, circumstances in which variable-length extents may be appropriate.

When determining which extent type to use, take into account your business requirements, as well as the benefits and drawbacks of each extent type, as described in [Table 2-1](#).

**Table 2-1: After-image extent types**

Type	Benefit	Drawback
Fixed-length	<b>Performance</b> — Fixed extents incur the performance impact for allocating and formatting the blocks at extent creation time. This causes minimal performance impact to the database during normal operations.	<b>Full extent management</b> — It is possible for multiple extents to fill within a cycle of extent management. This must be accounted for in the after-image management.
Variable-length	<b>Performance loss</b> — When a new after-image block is required and there are no empty blocks, the database broker has to allocate and format additional space from the operating system.	<b>Full disks</b> — Extents could grow to take up all of the available disk space.  <b>Larger file management</b> — Management of variable extents could involve very large files since the extents can grow to any size. Backing up of these files or moving them around must be taken into account.

### After-image extent states

[Table 2-2](#) describes the five different after-image extent states that can occur when a database is enabled for OpenEdge Replication as a source database.

**Table 2-2: After-image extent states**

(1 of 2)

After-image extent state	Description
EMPTY	The after-image extent is empty and ready to be used by the RDBMS.
BUSY	The after-image extent is currently being written to by the RDBMS. Transaction-log records will accumulate until an extent switch is performed either by the RDBMS when the extent fills or by DBA action when the <code>rfutil db-name -C aimage new</code> command is issued.
LOCKED	The after-image extent is FULL but has not been replicated in its entirety to target database(s). The extent cannot be emptied by the <code>rfutil db-name -C aimage empty</code> command until all transaction-log records have been sent to and applied by the Replication agent. The after-image blocks in an extent in this state can be extracted and written to a new file using the <code>rfutil db-name -C aimage extract</code> command.

**Table 2–2: After-image extent states**

(2 of 2)

After-image extent state	Description
FULL	The after-image extent is FULL and is ready to be archived then emptied.
ARCHIVED	The after-image extent has been archived by the After-image Management Utility. For details about the utility, see <i>OpenEdge Data Management: Database Administration</i> .

## Calculating current after-image volume

There are several ways to determine how much after-image data your database is currently generating. To determine the highest volume at any one time, you must sample at a small interval, such as one minute, for several days or over a month to arrive at average usage per month. Then, take the maximum value of your sample. If your network is close to capacity, the difference between your average load and your maximum load may be enough to saturate your network, causing additional network problems.

The following are four typical methods you can use to calculate after-image volume:

- Use after-image virtual system table (VST) information.
- Use the RFUTIL command.
- Use the PROMON command.
- Use before-image VST information.

It is important to note that none of these methods take into account spikes in after-image activity.

### Using after-image VST information

You can use VSTs to determine how many after-image blocks have been written. The VST table used for this purpose is `_ActAiLog`. The fields with the appropriate information are `_AiLog-TotWrites`, and `_AiLog-BytesWritn`. The `_AiLog-TotWrites` field is in the form of after-image blocks.

Refer to your after-image block size to see how much data is involved. Take samples at the start of the time period and at the end of the time period. The difference between these fields results in the count of after-image blocks that have been written.

Using this method to determine after-image volume is preferable to other methods, as it is more flexible, easier to maintain, and less intrusive on database performance.

### Using the RFUTIL command

To determine the after-image volume from the command line, you must run the following command at the start and at the end of your desired time frame:

```
rfutil db-name -C aimage extent list
```

The after-image extent size is shown in 1K blocks. Subtract the end value from the start value to determine how many blocks (count) have been written. Use this value to calculate the after-image generation rate per period depending on the network rate desired. It is important to note that emptying the after-image extents negates this method, as it changes the after-image block counts within the extent to zero.

### Using the PROMON command

Start PROMON for the database that will become the source database by using the following command:

```
promon db-name
```

Once PROMON is running, do the following:

1. Type **R&D** in the **Enter your selection** field, and press **ENTER**.
2. Type **5** to select **Adjust Monitor Options**, and press **ENTER**.
3. Type **3** to select **Monitor sampling interval**, and press **ENTER**.
4. Type **3600** in the **Enter sample interval <1 to 3600>** field, and press **ENTER** twice.
5. Type **2** to select **Activity Displays**, and press **ENTER**.
6. Type **6** to select **AI Log**, and press **ENTER**.

The **Activity: after-image Log** appears, as shown:

```
10/17/05 Activity: after-image Log
10:38:13      10/17/05 10:32 to 10/17/05 10:38 (5 min 16 sec)
```

	Total	Per Min	Per Sec	Per Tx
Total after-image writes	<b>4121</b>	782	13.04	0.31
after-image Writer writes	0	0	0.00	0.00
Records written	272040	51653	860.89	20.68
Bytes written	33906016	6437851	107297.52	2577.82
Busy buffer waits	0	0	0.00	0.00
Buffer not avail	2271	431	7.19	0.17
Partial writes	1841	350	5.83	0.14
Log force waits	0	0	0.00	0.00

Notice the highlighted elapsed time on the second line of the sample screen.

To gather a sample for 60 minutes, type **S** and press **ENTER**. The message “Sampling for 3600 . . .” appears. When the 60 minutes have elapsed, the numbers shown on the rest of the screen reflect running totals for that elapsed time. Record the number from the **Total after-image writes** row in the **Total** column. This is the total number of after-image blocks written in one hour. You may repeat this process to gather additional samples. Calculating a realistic average number of blocks written in one hour requires several samples. The more samples gathered, the more accurate the average is.

Now divide the average number of blocks generated in one hour by after-image block size. The after-image block size can be found from the main menu of **R&D** by selecting option **1** (Status Displays) and then option **10** (after-image Log). The after-image block size is the second-to-last entry. Take this number and divide it by 1024 to arrive at the block size ( $8192 / 1024 = 8$ ). Multiply the value from the activity menu by the value calculated. The result is the number of 1K after-image blocks written during a typical hour period.

### Using before-image VST information

You can also use before-image VSTs to determine approximately how many after-image blocks will be written. The VST table used for this is `_ActBILog`. The fields with the appropriate information are `_BiLog-TotWrites` and `_BiLog-BytesWritn`. `_BiLog-TotWrites` is in the form of before-image blocks. Refer to your before-image block size to see how much data is involved. You must also take into account that the before-image block size must be the same as your after-image block size. Take samples at the start of the time period and at the end of the time period; the difference between these samples results in the count of after-image blocks that would have been written.

If you do not have after-imaging enabled against the database, this method is the only choice in calculating after-image volume.

## Replication and after-image extents

As database changes are performed, the actions required to make those changes are recorded as transaction-log records. The OpenEdge Replication server reads after-image blocks then sends them using TCP/IP to the OpenEdge Replication agent. The OpenEdge Replication agent then applies the transaction-log records contained in the after-image block to the target database.

When a database is enabled for Replication and database activity continues without OpenEdge Replication being active, the after-image extents provide a persistent store for all of the source database activity. Once Replication is restarted, all non-replicated database activity is replicated using the data in the after-image extents. Given this, it is essential that all after-image data remain accessible so that OpenEdge Replication can properly process it. Once the after-image data is replicated, the space it occupies can be reused by the RDBMS. Reusing after-image space is accomplished by emptying FULL-UNLOCKED after-image extents using the following command:

```
rfutil db-name -C aimage empty
```

The source and target databases are interdependent in two ways. First, the target database requires after-image blocks and transaction-log records generated by the source database in order to remain synchronized with the source database. All source database update operations generate transaction-log records (notes) that are written into after-image extents. The second and subtler dependency is the fact that the state of the source database after-image extents depends on normal replication activity between the source and target databases. As long as normal replication occurs between the source and its target databases, filled source after-image extents can be archived and emptied. However, if normal replication is not occurring, source database after-image extents continue to fill but they will remain in a LOCKED state until they are replicated in their entirety. The fact that source database activity can occur without replication actively being performed is reliant upon the total capacity of source after-image extents being greater than that of non-replicated databases.

In addition to the increased capacity of source after-image extents, there is another factor involved when sizing after-image extents for a source database. When a database is enabled as a source database, the RDBMS generates additional transaction-log records that are written into the recovery log (BI) and the after-image log. The overhead generated by these additional transaction-log records must be considered when developing successful after-image sizing and management schemes when using OpenEdge Replication. The replication-specific transaction-log records group logical-record operations and consist of a begin operation and an end operation.

The OpenEdge RDBMS protects low-level record integrity using a special type of operation called a *logical operation*. A logical operation prevents records from being accessed until all updates for that record are complete.

A typical logical operation may include the following database actions:

- New data block(s) may be created when a record is extended.
- Existing data block(s) may be cleared when a record is deleted.
- The record's data must be written into each of the data blocks it occupies.
- The records indexes are updated.

While a logical operation is being performed, access to the record being added, updated, or deleted is blocked until the logical operation is complete.

## After-image extent sizing on a source database

After you estimate the volume of after-image data your database generates, you can resize the after-image extents for use with OpenEdge Replication. Progress Software Corporation recommends that the size of one after-image extent be capable of storing four hours of typical database activity when running OpenEdge Replication. Additionally, Progress Software Corporation further recommends that the total size of all after-image extents have the capacity to store one week of typical database activity. These recommendations are based on the following:

- The total capacity of your source database after-image extents must include additional capacity to allow for some target database downtime.
- Most simple hardware failures can be resolved within a four-hour period. For example, if a drive controller or network card fails, it is a simple operation to have another installed.
- In certain situations, it may be necessary to have your target machine out of service for an extended length of time. The total after-image storage allocated must have the capacity to store large numbers of transaction-log records.

There are additional transaction-log records written by the RDBMS when a database is enabled as a source database. These additional transaction-log records require additional after-image space, referred to as *replication overhead*. In order to calculate the replication overhead, the estimated after-image size information (obtained in the “[Calculating current after-image volume](#)” section on page 2–5) must be multiplied by 1.5.

The examples shown in [Table 2–3](#) and [Table 2–4](#) demonstrate how to calculate the source database after-image extent sizes and total capacity when using fixed after-image extents. The calculated after-image extent size and total after-image capacity are shown in bold at the bottom of each of the tables.

### Calculating fixed after-image extent size

Following are two examples that illustrate how to calculate the source after-image extent sizes and total capacity when using fixed after-image extents. The calculated after-image extent size and total after-image capacity are shown in bold in the last two rows of [Table 2–3](#) and [Table 2–4](#).



**After-image extent sizing based on hourly data**

After the total after-image data written for four, separate one-hour periods are determined, the sizing calculations described in [Table 2–3](#) are performed.

**Table 2–3: After-image extent sizing calculations (hourly data)**

Description	Measurement/estimate
Average after-image data generated in one typical work hour	72.5MB = average of the following: <ul style="list-style-type: none"> <li>• 70MB in hour 1</li> <li>• 67MB in hour 2</li> <li>• 79MB in hour 3</li> <li>• 74MB in hour 4</li> </ul>
OpenEdge Replication after-image overhead	50%, or 1.5 multiplier
Number of processing hours during a typical workday	16 hours per day
Number of processing hours during the typical workweek	80 (16 hours per day x 5 days)
Estimate of after-image data generated in one day	1,160MB, or 1.13GB (72.5MB x 16 hours per day)
Estimate of after-image data generated in one workweek	5,800MB, or 5.66GB (1,160MB x 5 days)
Estimate of after-image data generated in one hour with OpenEdge Replication overhead	109MB (72.5MB x 1.5)
Estimate of after-image data generated in one day with OpenEdge Replication overhead	1,740MB (1,160MB x 1.5)
Estimate of after-image data generated in one workweek with OpenEdge Replication overhead	8,700MB (1,740MB x 5 days)
<b>Single fixed after-image extent size</b>	<b>450MB (109MB x 4 hours)</b>
<b>Total number of after-image extents necessary to handle one week of processing</b>	<b>19 extents (8,700MB / 450MB)</b>

### After-image extent sizing based on weekly data

Using the total size of after-image extents for one full workweek, the sizing calculations described in [Table 2–4](#) are performed.

**Table 2–4: After-image extent sizing calculations (weekly data)**

Description	Measurement/estimate
Total after-image data generated in one typical week	15,000MB, or 14.6 GB
OpenEdge Replication after-image overhead	50%, or 1.5 multiplier
Number of processing hours during a typical workday	16 hours per day
Number of processing hours during the typical workweek	112 hours (16 hours per day x 7 days)
After-image data generated in one hour	134MB (15,000MB / 112 hrs in week)
After-image data generated in one day	2,144MB, or 2.09GB (134MB x 16 hours)
Estimate of after-image data generated in one hour with OpenEdge Replication overhead	201MB (134MB x 1.5)
Estimate of after-image data generated in one day with OpenEdge Replication overhead	3,216MB, or 3.14GB (2,144GB x 1.5)
Estimate of after-image data generated in one workweek with OpenEdge Replication overhead	22,512MB (3,216MB x 7 days)
<b>Single fixed after-image extent size</b>	<b>804MB (201MB x 4 hours)</b>
<b>Total number of after-image extents necessary to handle one week of processing</b>	<b>28 extents (22,512MB / 804MB)</b>

### Using variable extents

Estimating the volume of after-image extent size needed does not apply when using variable extents. The amount of after-image data that is written to the extents is no different when using variable extents as opposed to fixed extents. When you are using variable extents, the following factors should be taken into account:

- **No limits** — The size of the variable length extent is limited only to the size of the file system (provided large files are enabled on your system).
- **Fills to largest size** — The variable extents will fill to the largest size available for the files.

- **Easier after-image management** — Not having to deal with multiple extents becoming full and not having to change extents and do extent management more often can save work.
- **No sizing rules** — Since the after-image extents are not pre-allocated and pre-formatted, the guidelines for creating extents do not apply.

## Determining OpenEdge Replication network bandwidth

There is additional replication overhead that is required to transfer the after-image blocks over the wire to the target system. This additional overhead is estimated at ten percent and consists of header and control information for replication.

Items in this calculation include but are not limited to:

- Packaging of blocks. Replication adds overhead to each AI block.
- The database default after-image block size of 8K. Changing the after-image block size to 16K would mean fewer messages are sent, which would lower the network activity by approximately five percent.
- Standard communication between the OpenEdge Replication agent and the OpenEdge Replication server.

Determining network bandwidth can be difficult. Consider that a T1 line provides approximately 1.544 megabits per second of throughput. This value represents the theoretical limit; however, the value is typically somewhat less than this due to routers, hubs, and switches.

## Estimating network bandwidth

Calculating OpenEdge Replication's effect on the network involves many different factors.

After-image data is the largest part in determining how much bandwidth will be needed. Exact data volume cannot be accurately calculated, as it depends on the type of transactions within the application and how much activity is going on against the database. However, it can be estimated if you know the average amount of after-image volume on your system. Use the following formula to help you calculate the values for capacity planning:

$\begin{aligned} \text{AID} \times \text{AIRM} &= \text{AIDR} \\ \text{AIDR} \times \text{AIRO} &= \text{AIDRO} \\ \text{AIDRO} \times (\text{PRD}) &= \text{AIDROP} \end{aligned}$
---

AID

The total size of all the after-image blocks

AIRM

The value 1.5, which is the multiplier for additional after-image notes due to Replication

AIDR

The number of after-image blocks with Replication

AIRO

The value 1.1, which is the multiplier for Replication overhead

AIDROP

The number of after-image blocks with Replication and overhead

(PRD)

The period of time, as follows:

- **1**, if samples are 60 minutes and you want per-hour values
- **1/60**, if samples are 60 minutes and you want per-minute values
- **24**, if samples are 1 hour and you want per-day values
- **60**, if samples are 1 minute and you want per-hour values
- **1/24**, if samples are 24 hours and you want per-hour values
- **1/1440** (which is  $24 * 60$ ), if samples are 24 hours and you want per-minute values

## Examples

Consider the following example, in which:

- AID is 89MB.
- Sample length is 60 minutes.
- PRD is 1.

$$\begin{aligned} 89\text{MB} \times 1.5 &= 133.5\text{MB} \\ 133.5\text{MB} \times 1.1 &= 146.85\text{MB} \\ 146.85\text{MB} \times 1 &= 146.85\text{MB} \end{aligned}$$

In this case, this customer would send approximately 146.85 megabytes of data to the target database per hour.

In the example that follows:

- AID is 360 MB.
- Sample length is 24 hours.
- PRD is /1440.

$$\begin{aligned} 360\text{MB} \times 1.5 &= 540\text{MB} \text{ (566, 231, 040 bytes)} \\ 540\text{MB} \times 1.1 &= 594\text{MB} \text{ (622, 854, 144 bytes)} \\ 594\text{MB}/1440 &= 422 \text{ KB (432, 537)} \end{aligned}$$

This customer would send approximately 422 kilobytes of data to the target database per minute.

## Anticipating additional network overhead

TCP/IP default packet sizes are typically 1564 bytes. 64 bytes of this information (approximately four percent) is TCP and IP header information. You might need to add this amount to the final calculation for a more accurate representation of network usage.

## **Additional references**

For additional information about estimating network bandwidth, refer to the following sources:

- Unix Network Programming, Prentice Hall Software Series, By W. Richard Stevens, Copyright 1990 ISBN 0-13-949876
- <http://www.stallion.com/html/support/glossary.html#T> - Definition of T1
- <http://www.strategicwebventures.com/definitions/Glossary/T1>

## Additional business considerations

Configuring after-imaging with OpenEdge Replication requires that you understand and consider your business requirements prior to beginning work. This section discusses the following issues that you must consider when running after-imaging with OpenEdge Replication:

- [Latency](#)
- [Acceptable target database downtime](#)
- [Appropriate failover behavior](#)

### Latency

*Latency* within OpenEdge Replication is defined as the time between the update being performed on the source database and the update occurring on the target database. OpenEdge Replication latency depends on the following factors, all of which should be taken into consideration:

- **The number and frequency of updates to the source database** — If the database is frequently updated and the after-image blocks are frequently filled, the latency will be much shorter. If there is little uncommitted activity on the source database, the latency may be longer.
- **Network volume availability and bandwidth** — If the network is slow or near capacity, the latency between the source and target database increases.
- **Target database/machine downtime** — The longer the target database or machine is unavailable, the higher the latency will be. If the downtime is expected to be a multi-day event, you should consider disabling OpenEdge Replication on the source database.

During downtime of the target database or machine, the source/production database is unprotected. If the target is used for read-only access, a down target would mean no read-only access.

Extents must not be deleted if they must be used to roll forward in the event of a failure.

For more information about latency, see [Chapter 5, “Reference.”](#)

## Acceptable target database downtime

There are several factors that help you determine your acceptable target database downtime, including:

- **How the target database is being used** — If the target database is being used for reporting, then allowing the target database to be down for longer periods (upwards of one day) might not be an issue. However, if the target database serves as a disaster recovery database, the acceptable downtime is much less (possibly five to ten minutes or less).
- **Whether the target database is out of date** — The determination of a database as out of date could be based on a specific time frame, or on the volume of data that would need to be applied to the database to synchronize the source and target databases. Once this threshold is reached and you decide to disable replication, you will have to begin the replication enablement processes to restart the replication process.

## Appropriate failover behavior

Determining when to failover is in direct correlation with determining latency and acceptable target down time. Once either of these criteria has failed, a decision as to whether to fail over to the target database must be made.

Another scenario where failover is more appropriate is loss of the source database. Loss of the source database can be due to something as simple as the database shutting down and needing to be restarted. A more severe example is the loss of hardware, making access to the database impossible.

In this instance, you must determine whether the database can be recovered in an acceptable amount of time. The acceptable amount of time is determined by your business requirements for having the database and application available. If you can afford 15 minutes of database unavailability, then this is your measurement for failover.

Keep in mind that in the event of a failure, data can be lost. For example, if the source database goes down between BI write and AI write and the currently busy extent is applied to the target, it will have the data written to AI but nothing else. The source and target databases are then no longer synchronized.



## Limitations and restrictions

Before you begin implementing OpenEdge Replication, you should be aware of the following:

- If you perform a BI truncate on your database and the BI truncate alters the database in any way—for example, BI truncate after abnormal database end—there is a possibility that active transactions will be undone. This process generates AI transaction log records. Since OpenEdge Replication is not running at the time the BI truncate is run, all AI areas could potentially fill up. Therefore, be sure to have enough available space in your AI areas to handle this event.
- A database that is enabled for OpenEdge Replication cannot be restored unless OpenEdge Replication is first disabled. For more information on restoring a database, see the [“Restoring source and target databases”](#) section on page 3–18.
- In general, a database enabled for OpenEdge Replication cannot be modified in structure or data when OpenEdge Replication is not running.

When OpenEdge Replication is running, there are specific activities allowed on the source and target databases. If an attempt is made to perform unauthorized activity, an error message is logged and the activity is disallowed.

For specific details on activities that are allowed and disallowed, see [Table 5–17](#).

- OpenEdge Replication does not support two-phase commit enabled databases.
- OpenEdge Replication requires at least one ABL broker in order to function. This ABL broker must be started before any SQL broker. Specifically, the ABL broker must be the first broker started. When SQL brokers are started, you must not specify the `-DBService` startup argument.

## Ensuring success when using after-imaging

Keep the following in mind when you are using after-imaging:

- If the disk runs out of space and no empty after-image extent is available (even if you are using variable extents), you must perform emergency maintenance; otherwise, the database is forced to shut down. To prevent the database engine from shutting down when it exhausts after-image disk space, you must start your database with the after-image stall (`-aistall`) startup parameter.
- You must archive and manage after-image extents as part of standard after-image processing. For further information on archiving your after-image extents, see *OpenEdge Data Management: Database Administration*.

## A summary of OpenEdge Replication and after-imaging

Implementing OpenEdge Replication with after-imaging requires several considerations:

- Understanding how OpenEdge Replication can be configured is important when making design and implementation decisions.
- After-image processing is critical for maintaining database performance at an acceptable level and for maintaining optimum performance of OpenEdge Replication. The after-image implementation must be resilient and stable.
- The number of after-image extents required and the volume of these extents are dependent on your business decisions and the hardware that is available for after-imaging and OpenEdge Replication.
- Business considerations must be taken into account when determining the configuration of after-image extent sizes and archiving.

Operational planning is especially important for a successful OpenEdge Replication implementation.

---

## Implementing OpenEdge Replication

---

This chapter describes how to get started with OpenEdge Replication (once you complete the planning phase), as detailed in the following sections:

- [Choosing the implementation method](#)
- [Setting up the source database for OpenEdge Replication](#)
- [Setting up the target database](#)
- [Starting OpenEdge Replication](#)
- [Starting the source database](#)
- [Restoring source and target databases](#)
- [OpenEdge Replication startup and initialization process](#)
- [Database connection considerations](#)
- [Database access once OpenEdge Replication is running](#)
- [Stopping OpenEdge Replication](#)
- [Latency reporting](#)
- [OpenEdge Replication utilities and commands](#)

- Normal OpenEdge Replication activity
- Choosing a hot standby database
- Handling OpenEdge Replication failure conditions
- Transitioning to the target database

## Choosing the implementation method

You can choose between two implementations of OpenEdge Replication: *default implementation* or *deferred agent startup implementation*.

The default implementation requires that the OpenEdge Replication agents start before the connection time-out expires. Also, the default configuration of OpenEdge Replication does not allow source and target database activity until both the OpenEdge Replication server and agents have completed their startup and initialization phases.

The deferred agent startup implementation allows the source database and the OpenEdge Replication server to start without connecting to an agent. This implementation is useful if you need to create a target database from an online backup of the source database. The advantage is that you can get your source database up and running much sooner.

Keep in mind that if the database is large and backups are time consuming, you may want to consider deferred agent startup rather than the default implementation. Deferred agent startup allows online backups, which will minimize the downtime of the database. For more information, see the [“Using the deferred agent startup implementation”](#) section on page 3–4 and the [“Setting up the source database with an online backup”](#) section on page 3–9.

## Using the default implementation

Following is an overview of the tasks you must perform to create the default implementation of OpenEdge Replication.



### To use the default implementation of OpenEdge Replication:

1. Execute your PROENV script every time you open a command-line window or shell, and ensure that you have \$DLC, \$PROMSGS, \$DSRHOME, and \$PROCFG environment variables set correctly. For more information about these variables, see *OpenEdge Getting Started: Installation and Configuration*.
2. Shut down the source database.
3. On the source machine, do the following:
  - a. Create an initial backup of the source database.
  - b. Create a structure file (.st) of the source database.
  - c. Ensure after-imaging is set up and running.
  - d. Enable the source database for OpenEdge Replication.
  - e. Perform an incremental backup of the source database.

For more information about these tasks, see the [“Setting up the source database for OpenEdge Replication”](#) section on page 3–7.

4. On the target machine, do the following:
  - a. Copy the source database backup to the target machine. (The source database backup is the initial copy of the target database.)
  - b. Enable the target database for OpenEdge Replication.

For more information about these tasks, see the [“Setting up the target database”](#) section on page 3–11.

5. To configure OpenEdge Replication, do the following:
  - a. Create the server properties file.
  - b. Create the agent properties file.

For more information, see the [“OpenEdge Replication property files”](#) section on page 5–3.

6. Start the databases in the following order:
  - a. The target database
  - b. The source database

Because the databases are enabled for OpenEdge Replication, the OpenEdge Replication server and agents will also start.

This default implementation does not allow source or target database activity until both the server and agents complete their startup and initialization phases.

## Using the deferred agent startup implementation

By using the deferred agent startup implementation, you can configure OpenEdge Replication to allow database activity sooner than if you use the default implementation. The deferred agent startup mode allows you to create a target database by doing an online backup of the source database. In contrast, the default implementation requires that the source database be shut down while you perform the backup.

To use deferred agent startup, you must specify the `defer-agent-startup` property for the OpenEdge Replication server in the `[server]` section of the property file. If you set `defer-agent-startup` to a valid, non-zero time-out value, the source database can be active before the OpenEdge Replication server even contacts its configured agent(s). For more information on this property and its values, see the [“OpenEdge Replication property files”](#) section on page 5–3 and the [“Server properties”](#) section on page 5–5.

The deferred agent startup implementation configures the OpenEdge Replication server to do the following:

- If the OpenEdge Replication server cannot connect to its configured agent(s) on the first connection attempt, it will go into deferred agent startup mode.
- While in deferred agent startup mode, the OpenEdge Replication server will wait five minutes and then attempt agent connection again.
- The OpenEdge Replication server will attempt to connect every five minutes until all agent(s) are connected, or until the time-out specified in the `defer-agent-startup` expires.

---

**Notes:** If you do not want to wait the five minutes for the server to attempt connection to the agents, you can force agent connection using the `connectagent` command of the DSRUTIL utility. For more information, see the “[DSRUTIL utility](#)” section on page 5–11.

To cancel `defer-agent-startup`, use the `cancelDefer` command of the DSRUTIL utility.

---

- Once the server connects to the agent(s), then startup, initialization, and synchronization are performed.
- During the entire connection process, source database activity continues to occur but will be halted briefly while the OpenEdge Replication server reinserts itself into normal RDBMS AI block processing.



**To create the deferred agent startup implementation:**

1. Execute your `PROENV` script every time you open a command-line window or shell, and ensure that you have `$DLC`, `$PROMSGS`, `$DSRHOME`, and `$PROCFG` environment variables set correctly.
2. Shut down the source database.
3. On the source machine, do the following:
  - a. Create an initial backup of your source database.

---

**Note:** [Step a](#) is not necessary if you prefer to do an online backup of the source database. Online backup is preferable in situations where the database is large, backups are time consuming, and you want to minimize the downtime of the database. For more information, see the “[Setting up the source database with an online backup](#)” section on page 3–9.

---

- b. Create a structure file (`.st`) of the source database.
- c. Ensure after-imaging is set up and running.

- d. Enable the source database for OpenEdge Replication.
- e. Perform an incremental backup of the source database.

---

**Note:** [Step e](#) is not necessary if you intend to create an online backup. See the [“Setting up the source database with an online backup”](#) section on page 3–9.

---

- 4. Configure your OpenEdge Replication server by doing the following:
  - a. Configure the OpenEdge Replication server properties file to set the `defer-agent-startup` property to a valid time-out value.
  - b. Configure the OpenEdge Replication server properties file for your other server properties.

For more information, see the [“OpenEdge Replication properties”](#) section on page 5–5.

- 5. Start the source database.
- 6. If you have not already created an offline backup, create an online backup of the source database.
- 7. On the target machine do the following:
  - a. Use the source database backup as an initial copy of the target database.
  - b. Enable the target database for OpenEdge Replication.

For more information about these tasks, see the [“Setting up the target database”](#) section on page 3–11.

- 8. Configure the OpenEdge Replication agent properties file.
- 9. Start the OpenEdge Replication target database.
- 10. Wait for the OpenEdge Replication server to connect to the agent(s), or use `DSRUTIL connectagent` to force a connection without waiting. For more information on `connectagent`, see [Table 5–4](#).

Once the synchronization between the OpenEdge Replication server and agent(s) is complete, normal OpenEdge Replication target database activity is allowed.

### Special considerations for deferred agent startup

Schema updates are not allowed while the OpenEdge Replication server is performing synchronization. If schema updates are being performed when failure recovery synchronization begins, source database updates will block until failure recovery is complete.

Source database activity cannot continue without the agent(s) connected when synchronous replication is being used.



# Setting up the source database for OpenEdge Replication

You use either an offline backup or an online backup of the source database to create the target database.

## Setting up the source database with an offline backup

This section describes some of the tasks you must perform on the source database to prepare to run OpenEdge Replication using an offline backup of the source database to create the target database.

If you are using the deferred agent startup implementation of OpenEdge Replication and want to create a target database from an online backup of the source database, see the [“Setting up the source database with an online backup”](#) section on page 3–9.

### Before you begin

Before setting up the source database, you must complete the following preliminary tasks:

1. Execute your PROENV script every time you open a command-line window or shell, and ensure that you have \$DLC, \$PROMSGS, \$DSRHOME, and \$PROCFCG environment variables set correctly. For more information about these variables, see *OpenEdge Getting Started: Installation and Configuration*.
2. Shut down the source database.
3. Back up the source database.

You can use either PROBKUP or `os -copy` to back up the database. However, keep in mind that if your source database has after-image files and you use `os -copy`, after-imaging will also be enabled on the target database. After-imaging is not necessary on the target database. If it is enabled, however, the management of the after-image files on the target must be handled separately from the source.

For more information about how to perform these tasks, see *OpenEdge Data Management: Database Administration*.

### Creating a structure file

Once you have backed up your source database, you need to build a structure file (.st). Use the following command to build this file:

```
prostrct list source-db-name source-db-name.st
```

(Later, you will move the structure file that originates from the source to the target machine.)

## Enabling after-imaging

To use OpenEdge Replication, you must enable after-imaging on your source database. If AI is already enabled on your machine, skip this section and go to the [“Enabling the source database for OpenEdge Replication”](#) section on page 3–8.



### To enable AI on your source database:

1. Add AI areas by entering the following command:

```
prostrct add source-db-name addai.st
```

Where `addai.st` is the name of your structure file. The structure file defines AI areas to be added to the database.

You might need to build a structure file that contains the AI area specifications.

2. Back up the database using the following command:

```
probkup source-db-name source-db-name.bak
```

3. Turn on AI with the RFUTIL utility, as shown:

```
rfutil db-name -C aimage begin
```

For more information about using the RFUTIL utility, see *OpenEdge Database Management: Database Administration*.

## Enabling the source database for OpenEdge Replication

OpenEdge Replication requires that the source database be enabled for Replication before you start it.

To enable the source database, enter the following command:

```
proutil source-db-name -C enableSiteReplication source
```

## Performing an incremental backup of the source database

After your database has been enabled as a source database, you must perform an incremental backup. To do this, enter the following command:

```
probkup source-db-name target-db-name incremental
```

You will use the incremental backup to create the target database. For more information about incremental backups, see *OpenEdge Database Management: Database Administration*.

## Where to go next

After you complete the tasks described in this section, go to the [“Starting the target database”](#) section on page 3–16.

## Setting up the source database with an online backup

This section describes some of the tasks you must perform on the source database to prepare to run OpenEdge Replication. Refer to this section if you are using an online backup of the source database to create the target database.

To enable online backups, you need to implement the deferred agent startup mode, which is described in the [“Using the deferred agent startup implementation”](#) section on page 3–4.

You also must have enabled AI on the database.

### Before you begin

Execute your PROENV script every time you open a command-line window or shell, and ensure that you have \$DLC, \$PROMSGS, \$DSRHOME, and \$PROCFG environment variables set correctly.

For more information on how to perform these tasks, see *OpenEdge Database Management: Database Administration*.

### Creating a structure file

Once you have shut down your source database, you need to build a structure file (.st). Use the following command to build this file:

```
prostrct list source-db-name source-db-name.st
```

The structure file that originates from the source will be moved to the target machine.

### Enabling the source database for OpenEdge Replication

OpenEdge Replication requires that the source database be enabled before starting the database. To enable the source database (online or offline), enter the following command:

```
proutil source-db-name -C enableSiteReplication source
```

### Configuring the OpenEdge Replication server for deferred agent startup

Online backups require a server that is configured for deferred agent startup. To configure such a server, you must set the `defer-agent-startup` property to a valid time-out value (in minutes) in the server property file. In the following example, the time-out is set for four hours:

```
[server]
  defer-agent-startup=240
```

For more information, see the [“OpenEdge Replication property files”](#) section on page 5–3 and the [“OpenEdge Replication properties”](#) section on page 5–5.

### Performing an online backup

To perform an online backup, use the `-REPLTargetCreation` and `online` options. For example:

```
probkup online source-db-name source-db-name.bak -REPLTargetCreation
```

If you need more information about database backups, see *OpenEdge Database Management: Database Administration*.

After you complete the tasks described in this section, go to the [“Starting the target database”](#) section on page 3–16.

## Setting up the target database

You must restore your backup of the source database to use as your target database.



### To create the target database:

1. Copy the structure file (.st) from the source database to the target machine.

This structure file lists the physical structure of the source database. If the physical structure of your target database is different (for example, there are different drives, slices, or directories) you must edit the structure file to accurately describe the physical structure of the target database.

2. Restore the backup copy of the source database on the target machine by entering this command:

```
prorest target-db-name source-db-backup-name
```

For more information on backing up and restoring a database, see *OpenEdge Data Management: Database Administration*.

3. If the database is not online, restore the incremental backup of your source database on the target machine. For example:

```
prorest target-db-name source-db-incrementalbackup-name
```

For more information on restoring incremental backups, see *OpenEdge Data Management: Database Administration*.

### Enabling the target database for OpenEdge Replication

OpenEdge Replication requires that you enable your target database for OpenEdge Replication with the following command:

```
proutil target-db-name -C enableSiteReplication target
```

## Configuring the OpenEdge Replication property files

To configure your source database and target database for OpenEdge Replication, you must configure the replication property files for both the OpenEdge Replication server and the OpenEdge Replication agent.

OpenEdge Replication provides two sample property files: `source.rep1.properties` and `target.rep1.properties`. You can save and modify these sample files to use with your source and target OpenEdge Replication databases. Because the files are samples, some of the properties and values are generic; it is up to you to customize the properties to reflect the settings you want to use with your own source and target databases.

You can also combine the source and target properties into one property file with the name `db-name.replication.properties`.

### Configuring the source database property file

Before you start OpenEdge Replication on your source database, you must configure your replication server. You configure the server by saving the sample source properties file provided by OpenEdge Replication and editing it.

**To configure an OpenEdge Replication server for the source database:**

1. Make a copy of the `source.rep1.properties` file located in the `properties` subdirectory of the OpenEdge Replication installation.
2. Place this new file in the same directory as your source database with the name `db-name.rep1.properties`, where `db-name` is the name of your source database.
3. To alter the default configuration, edit the file. If you are on UNIX, you can use `vi`, `emacs`, or your favorite text editor to change the properties file.

If you are in Windows, you can use Notepad or the Progress Explorer to change your properties file. For more information about the variables you can configure in the properties file, see the [“OpenEdge Replication properties”](#) section on page 5–5. For more information about changing the properties by using Progress Explorer, see the Progress Explorer Help.

The following is a sample `source.rep1.properties` file for the OpenEdge Replication server with two configured agents:

```
# OpenEdge Replication properties file for a database that will be used
# as a source database for OpenEdge Replication. This is a two agent
# configuration.

[server]
    control-agents=agent1, agent2
    database=source
    transition>manual
    transition-timeout=1200

[control-agent.agent1]
    name=agent1
    database=your target name
    host=yourhost
    port=your port or service name
    connect-timeout=120
    replication-method=async
    critical=0

[control-agent.agent2]
    name=agent2
    database=your target name
    host=yourhost
    port=your port or service name
    connect-timeout=120
    replication-method=async
    critical=0
```

In this example, `agent1` in the `control-agents=agent1` of the `[server]` section must match the control agent name specified in the `[control-agent.agent1]` section head. Also, `agent2` in the `control-agents=agent2` of the `[server]` section must match the control agent name specified in the `[control-agent.agent2]` section head. `ALL` is not allowed as an agent name. Each agent must have a unique name.

When you finish editing the file, save it.

### Agent rules

The following rules apply to the OpenEdge Replication agent:

- Once an OpenEdge Replication agent is removed from the properties file, it should not be added again unless the target database is recreated from the source database.
- Do not rename an OpenEdge Replication agent after OpenEdge Replication has been started once.
- You can add an OpenEdge Replication agent (maximum is two agents) as long as the target databases are created as instructed.

## Configuring the target database property file

Before you start OpenEdge Replication on your target database, you must configure your agent. You configure your agent by saving the sample target properties file provided by OpenEdge Replication and editing it.



### To configure an OpenEdge Replication agent for the target database:

1. Make a copy of the `target.rep1.properties` file located in the `properties` subdirectory of the OpenEdge Replication installation.
2. Place this new file in the same directory as your target database with the name `db-name.rep1.properties`, where `db-name` is the name of your target database.
3. To alter the default configuration, edit the file. If you are on UNIX, you can use `vi`, `emacs`, or your favorite text editor to change the properties file. If you are in Windows, you can use Notepad or the Progress Explorer to change your properties file.

For more information about the variables you can configure in the properties file, see the [“OpenEdge Replication properties”](#) section on page 5–5. For more information about changing the properties by using Progress Explorer, see the Progress Explorer Help.

The following is a sample properties file for the OpenEdge Replication agent:

```
[agent]
  name=agent1
  database=your target name
  listener-minport=4387
  listener-maxport=4500

[transition]
  database-role=normal
  auto-begin-ai=0
  auto-add-ai-areas=0
  ai-structure-file=rep11_ai.st
```



## Starting OpenEdge Replication

To run OpenEdge Replication, you simply start your source and target databases with an OpenEdge Replication qualifier (`-DBService replserv` or `-DBService replagent`).

The startup order does not matter. It is recommended, however, that if you did not set the `defer-agent-startup` property to a non-zero value in your server properties file, you start your target database first so that the OpenEdge Replication server does not time out.

If you do start your source database first, the target database must be started before the `connect-timeout` property in the `[control-agent.name]` section of the server properties file expires.

## Starting the source database

Use the following command to start the source database:

```
proserve -db source-db-name -DBService replserv
```

The `-DBService` argument instructs the broker to start the OpenEdge Replication server. Database activity cannot begin until OpenEdge Replication has completed its startup and initialization, unless you have used the `defer-agent-startup` property.

---

**Note:** Any valid PROSERVE argument can be used to start the source database. When using arguments that affect shared memory (number of users, clients per server, maximum number of servers, etc.), you should be careful to use the same arguments and values when you start the target database.

---

If there is a configuration error, the OpenEdge Replication server will be brought down even if there is no critical agent. The following are examples of a configuration error:

- Missing AI extents
- Improperly created target
- Large file mismatch

## Starting the target database

Use the following command to start the target database:

```
proserve -db target-db-name -DBService replagent -S [port | service name ]
```

The database broker will monitor the *port* or *service name* specified using the `-S` argument; therefore, inclusion of the argument is required. The port or service name must be the same as the port or service name parameter you specified in the OpenEdge Replication server's `repl.properties` file.

The port or service name you specify with the `-S` argument can also be used by ABL clients and the Replication server. If the broker started is configured for both ABL and SQL connections, the SQL clients can use the port and service name as well.

The `-DBService` parameter instructs the broker to start the OpenEdge Replication agent. Only limited database connections are allowed until OpenEdge Replication completes startup synchronization.

---

**Note:** You can start the target database with any valid PROSERVE argument. When using arguments that affect shared memory (number of users, clients per server, maximum number of servers, etc.), you should be careful to use the same arguments and values you used when you started the source database.

---

## Message logging during startup

While OpenEdge Replication is starting up, a log is produced that can be accessed in the event of a startup failure. This log file is named for the OpenEdge Replication server as `repl.server.startup.lg` and for the OpenEdge Replication agent as `repl.agent.startup.lg`. The location of this file is in the current working directory of the process where the OpenEdge database was started from. This special log file is used because neither database logging nor standard output is available.

## Restoring source and target databases

A database that is enabled for OpenEdge Replication cannot be restored unless OpenEdge Replication is first disabled.



### To restore a database that is enabled as a source database:

1. Use the following command to disable OpenEdge Replication on the source database:

```
proutil source-db-name -C DisableSiteReplication source
```

2. Use the following command to restore your source database:

```
prorest source-db-name [ file | device ]
```

Be sure you delete the *db-name.rep1.recovery* file any time you restore.

If the database you are restoring was previously enabled for OpenEdge Replication, the database is again enabled for OpenEdge Replication after the restore. OpenEdge Replication cannot be restarted for this database until OpenEdge Replication is disabled and then re-enabled. The target database must be resourced after the restore, disable, and enable have been performed.

---

**Note:** You can restore a database enabled as a target database once you disable OpenEdge Replication. This is not a recommended practice, however, because the target database should always be created from a source database, not from a backup of itself. If something happens to your target database, and you need a new copy of it, take the latest backup of your source database. This guarantees that the databases can be synchronized. The latest backup of your source database can be from a full online backup or a full offline backup. It cannot be an incremental backup. Be sure to delete the recovery file before restarting OpenEdge Replication.

---

Once a database is enabled for OpenEdge Replication, information about the state of OpenEdge Replication is kept in the database itself. This information is not restored when the database is restored. The only way to recover this information is to re-enable the database for OpenEdge Replication.

## OpenEdge Replication startup and initialization process

During the OpenEdge Replication startup and initialization process, the OpenEdge Replication server attempts to contact the OpenEdge Replication agent through the target database broker on the port specified in the `control-agent` section of the source database replication properties file. Once the OpenEdge Replication server makes contact with the OpenEdge Replication agent, a handshaking process takes place. During this process, OpenEdge Replication:

- Determines if the source and target databases are identical
- Verifies that the target database was created from the source database
- Performs synchronization
- Allows database connections

If the `defer-agent-startup` property is set to a valid non-zero value, source database activity is allowed once the source database is started. Source database activity will not halt until the very end of the synchronization process, when the OpenEdge Replication server completes the synchronization process. When synchronization is complete, the Replication server will reinsert itself back into the AI Block write process, where the OpenEdge RDBMS will be unblocked and normal database and OpenEdge Replication activity will continue.

Before synchronization begins, if the OpenEdge Replication server cannot connect to its configured agent(s) on the first connection attempt, it will go into a deferred agent startup. While in this state, the OpenEdge Replication server will wait five minutes, then attempt the OpenEdge Replication agent connection again. The OpenEdge Replication server will remain in this state until all agent(s) are connected or until the time-out specified in the `defer-agent-startup` property is reached. Once all the agents are connected to, startup, initialization, and synchronization are performed. If you do not want to wait for the five-minute intervals between connection attempts to the agent(s), you can force agent connection using the DSRUTIL function `connectagent`.

Schema updates are not allowed while the OpenEdge Replication server is performing synchronization. If schema updates are being performed when failure recovery synchronization begins, source database updates will block until failure recovery is complete.

Source database activity cannot continue without the agent(s) connected when synchronous replication is being used.

## Database connection considerations

As OpenEdge Replication is started, all database connections are disallowed until OpenEdge Replication startup and initialization phases are complete, unless you use the `defer-agent-startup` property in the server properties file. Disallowing database connections has different results based upon the type of client connecting. Client connections in this section are defined to be user sessions or utilities that perform actions on the database.

Table 3–1 shows the results of connecting during startup and initialization based on the type of client connecting.

**Table 3–1: Connection results during startup and initialization**

Type of client	Connection results
Self Serve Client	Blocks until database connections are allowed.
Remote Client	Ends in an error stating that the broker cannot spawn the server. A message is sent to the database log.
OpenEdge utility	Blocks until database connections are allowed.
Forced Shutdown	Always allowed.
Java ODBC Client (SQL Server access)	Ends in an error stating a network daemon error.
AppServer Self Serve Client	Blocks until database connections are allowed.
AppServer Remote Client	Ends in an error stating that the broker cannot spawn the server. A message is sent to the database log.
WebSpeed Self Serve Client	Blocks until database connections are allowed.
WebSpeed Remote Client	Ends in an error stating that the broker cannot spawn the server. A message is sent to the database log.

## Database access once OpenEdge Replication is running

Once OpenEdge Replication is running, database access is controlled differently on the source and target, and also depends on the type of product installed. (For example, the Workgroup database does not allow background processes such as APW, BIW, or AIW.)

The following sections document the different types of connections allowed.

### Target database access

The type of access allowed on the target database depends on which OpenEdge Replication product you have installed and the activity being performed by the OpenEdge Replication agent, as follows:

- If the OpenEdge Replication Plus product is installed, any process can access the target database for read-only use. If updates are attempted, an error occurs.
- If the OpenEdge Replication Plus product is not installed, access to the target database is limited to the system-level tools shown in [Table 3–2](#).

Limited logins are allowed immediately after the agent completes its startup. Normal logins that allow the access listed in [Table 3–2](#) are allowed after synchronization completes.

**Table 3–2: Standard system-level access to target database**

Tool	Description
Shut down	PROSHUT command to shut down the database
Monitor	PROMON utility that displays database information
AIW	A background process that writes AI buffers to disk soon after they are filled
BIW	A background process that continually writes filled BI buffers to disk
WDOG	The watchdog that cleans up after improperly terminated processes
OpenEdge Replication agent	Receives information from the OpenEdge Replication server and updates the target database
OpenEdge Replication Utility DSRUTIL	Allows you to perform specific OpenEdge Replication agent and target database commands
Database agent	Provides information to OpenEdge® Management about the status of the database

If the OpenEdge Replication agent is transitioning the target database to a normal OpenEdge database, critical system-level commands are allowed to access the database, as shown in [Table 3–3](#).

**Table 3–3: System-level access to target database during transition**

<b>System-level activity</b>	<b>Description</b>
Shut down	PROSHUT command to shut down the database.
Monitor	PROMON that displays database information.
WDOG	The watchdog cleans up after improperly terminated processes.

When transition is complete and the target database becomes a normal OpenEdge database, all traditional database activity is allowed.

Connecting a client to a target database in read-only mode (-R0) is never allowed. If the target database transitions to a normal OpenEdge database, read-only mode is allowed.

## Source database access

All normal database access is allowed on the source database whether the OpenEdge Replication product or the OpenEdge Replication Plus product is installed.



# Stopping OpenEdge Replication

To shut down and stop database activity, use the PROSHUT or PROMON utility first on the source database and then the target database. For more information on how to use the PROSHUT and PROMON utilities, see *OpenEdge Database Management: Database Administration*.

## Shutting down the source database

This section describes how to perform both a non-forced and a forced shutdown of your source database.



To specify a non-forced explicit shutdown, enter the following command:

```
proshut source-db-name -by
```

The following actions occur:

1. All processes other than the broker and the OpenEdge Replication server are shut down.
2. The broker flushes all database activity.
3. After the AI buffers are flushed, the OpenEdge Replication server processes them.
4. When the OpenEdge Replication server completes processing, it instructs the OpenEdge Replication agent to end processing.
5. When the OpenEdge Replication server completes its shutdown process, shutdown of the database completes.



To specify a forced shutdown, enter the following command:

```
proshut source-db-name -byF
```

The database will immediately shut down with a warning placed in the log file. With a forced shutdown, the AI buffers are not flushed. After a forced shutdown, the source and target databases are not considered identical. At this point, OpenEdge Replication performs the steps in the [“OpenEdge Replication startup and initialization process”](#) section on page 3–19 to restart it.

## Shutting down the target database

You can perform either a non-forced or a forced shutdown of your target database.

- To perform a non-forced explicit shutdown, enter the following command:

```
proshut target-db-name -by
```

The following events occur:

1. All processes other than the broker and the OpenEdge Replication agent are shut down.
2. The broker flushes all database activity.
3. After database activity has been flushed to disk, the OpenEdge Replication agent informs the OpenEdge Replication server that it has been shut down. If this is the only OpenEdge Replication agent serviced by the OpenEdge Replication server, the OpenEdge Replication server will shut down. If this OpenEdge Replication agent was configured as a critical OpenEdge Replication agent, even if there are additional OpenEdge Replication agents, the OpenEdge Replication server will shut down.
4. When the OpenEdge Replication agent has completed its processing, the target database will complete its shutdown.

- To perform a forced shutdown, enter the following command:

```
proshut target-db-name -byF
```

The database will immediately shut down with a warning placed in the log file. With a forced shutdown, the AI buffers are not flushed. After a forced shutdown, the source and target database are not considered to be identical. At this point, OpenEdge Replication stops and you must follow the steps in the [“OpenEdge Replication startup and initialization process”](#) section on page 3–19 to restart it.

## Shutting down the OpenEdge Replication server

In the event of an emergency in which you need to shut down the OpenEdge Replication server without first shutting down the source database, enter the following command:

```
dsrutil source-db-name -C TERMINATE server
```

When the OpenEdge Replication server terminates activity, it informs its connected OpenEdge Replication agents to terminate as well. The status of the database is not affected by OpenEdge Replication server termination.

You can then disable OpenEdge Replication after you have terminated the server.

**To disable OpenEdge Replication after you have terminated the server:**

1. Release any pending waits by entering the following command:

```
dsrutil source-db-name -C RELWAITS
```

2. Disable OpenEdge Replication by entering the following command:

```
dsrutil source-db-name -C DisableSiteReplication source
```

Once the source database is disabled, the only way to replicate the database again is to re-enable OpenEdge Replication for both the source and the target databases.

## Terminating the OpenEdge Replication agent

In the event of an emergency in which you need to shut down the OpenEdge Replication agent without first shutting down the target database, enter the following command:

```
dsrutil target-db-name -C TERMINATE agent
```

When the OpenEdge Replication agent terminates activity, it informs its OpenEdge Replication server that this OpenEdge Replication agent is being shut down. If this is the only OpenEdge Replication agent serviced by the OpenEdge Replication server, the OpenEdge Replication server will shut down. If this OpenEdge Replication agent was configured as a critical OpenEdge Replication agent, even if there are additional OpenEdge Replication agents, the OpenEdge Replication server will shut down.



To completely disable OpenEdge Replication after you have terminated the agent, enter the following command:

```
dsrutil target-db-name -C DisableSiteReplication target
```

## Emptying AI extents on the source database

AI extents can fill up on the source database. If you set the parameter `-ai stall` during startup, processing will pause if your AI extents become full. If you do not set the startup parameter `-ai stall`, and your AI extents fill up, processing will end. If either condition occurs, AI extents need to be emptied.

- To empty AI extents, enter the following command:

```
rfutil source-db-name -C aimage extent empty
```

AI extents cannot be emptied until they are replicated to the target. When source database activity continues during failure recovery or with the use of the `defer-agent-startup` property, the AI file can grow rapidly. You need to provide enough space in your AI file to accommodate this; otherwise, processing stalls if the AI file becomes full. If processing stalls and OpenEdge Replication cannot continue, OpenEdge Replication can be disabled online.

- To disable OpenEdge Replication online, enter the following command:

```
dsrutil source-db-name -C DisableSiteReplication source
```

## Latency reporting

If the target database transitions to a normal database, latency reporting allows you to determine how far behind the target database is from the source database. When the target transitions to a normal database, there will be some number of AI blocks that have been written to the source database that have not been applied by the OpenEdge Replication agent to the target database. To know how much, if any, information has not been applied to the target database, DSRUTIL MONITOR provides latency information from both from the server's perspective and agent's perspective.

### OpenEdge Replication server perspective

From the server perspective, DSRUTIL MONITOR provides the following latency details:

- The current RDBMS AI block
- The AI block that the OpenEdge Replication server last read
- The time the last AI block was read
- The current transaction ID assigned to RDBMS

### OpenEdge Replication agent perspective

From the agent perspective, DSRUTIL MONITOR provides the following latency details:

- The current source database AI block.
- The AI block that the OpenEdge Replication agent last processed.
- The last transaction ID just applied.
- The source database machine time that the last transaction was applied to the source database. This time stamp is critical, as it will tell you that transactions committed before that date are in the target database.

For more information on the DSRUTIL MONITOR reports on latency from the OpenEdge Replication server and agent perspectives, see [Table 5–8](#).

## OpenEdge Replication utilities and commands

You can use OpenEdge Replication utilities and commands to manage your processes and databases. Specifically, you can use:

- The DSRUTIL utility, which performs specific OpenEdge Replication administration tasks. See the [“DSRUTIL utility”](#) section on page 5–11.
- The DSRUTIL utility monitor capabilities, which allow you to monitor OpenEdge Replication. See the [“OpenEdge Replication DSRUTIL MONITOR”](#) section on page 5–31.
- Virtual System Tables, which allow you to access OpenEdge Replication through ABL or SQL. See the [“Virtual system tables for OpenEdge Replication”](#) section on page 5–44.

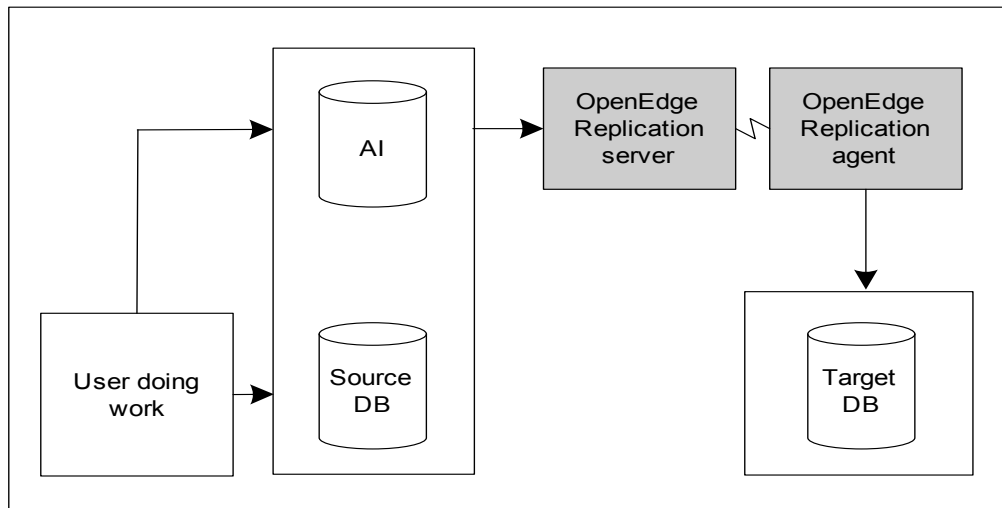
## Normal OpenEdge Replication activity

OpenEdge Replication begins when the source and target databases are started. While the OpenEdge Replication server and OpenEdge Replication agent are performing startup and initialization, other database activity is not allowed, unless you specified a non-zero value for the `defer-agent-startup` property. If a process attempts to connect to either database during this phase, the process will block until OpenEdge Replication has completed its work and normal database activity starts. See the [“Database access once OpenEdge Replication is running”](#) section on page 3–21 for further details.

### Normal source database activity

During normal source database activity with OpenEdge Replication running, all AI blocks written to an AI extent are sent to the OpenEdge Replication server. The OpenEdge Replication server then sends the AI blocks to all configured OpenEdge Replication agents. This process continues until the source database is shut down.

Figure 3–1 shows AI blocks sent from the server to the agent.

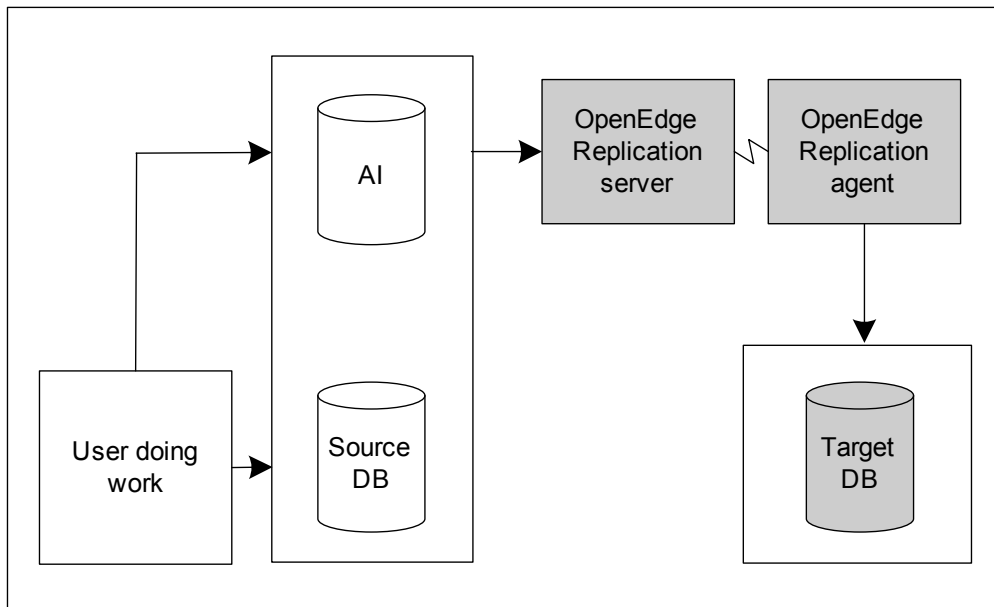


**Figure 3–1: AI blocks sent from server to agent**

### Normal target database activity

During normal operations, the OpenEdge Replication agent is performing a continuous roll forward. All database activity performed on the source database is applied to the target database in this manner. The OpenEdge Replication agent receives an AI block from the OpenEdge Replication server and applies the AI transaction log records to the target database. This process continues until the source or target database is shut down.

This continuous roll forward is shown in [Figure 3–2](#).



**Figure 3–2: Continuous roll forward on agent**

For the most part, the OpenEdge Replication agent performs no specific action for individual transaction log records other than applying them to the target database, as [Figure 3–2](#) shows. However, there are exceptions. The OpenEdge Replication agent does special processing for the following transaction log records:

- Checkpoint
- AI extent switch
- Transaction end

Both a checkpoint and an AI extent switch cause the OpenEdge Replication agent to inform the OpenEdge Replication server that a synchronization point, commonly known as a *sync point*, has been encountered. A sync point instructs the OpenEdge Replication server that this is the point to begin synchronization during database startup or failure recovery.

When synchronous replication is in effect, any process ending a transaction either by committing or rolling back will block or wait until the transaction end is processed by the OpenEdge Replication agent. After the OpenEdge Replication agent applies the transaction end, it informs the OpenEdge Replication server, and the OpenEdge Replication server unblocks the waiting process.

The sync point and all other information needed for synchronization are stored in a file called *db-name.rep1.recovery*, where *db-name* is the name of your source or target database. Do not edit this file. This file must be deleted whenever the source or target database is restored, or when OpenEdge Replication is re-enabled.



## Choosing a hot standby database

If failover processing occurs, the target database is available as a *hot standby*. A hot standby database is a database that is updated and ready to go immediately. (Replicated databases are hot standby databases.) You can have up to two target databases. Only one of the target databases can be set to transition automatically to a normal OpenEdge database by setting its agent to `critical` and transition to `auto`. Only one target database can be critical because you do not want users to be updating two different databases after automatic transition has occurred. The second target database is not transitioned unless you manually transition it. This provides you with options for your hot standby choice.

If you choose the target database that automatically transitions to a normal OpenEdge database as your hot standby, it is possible that transactions could be lost when the failure occurs.

If you are running in synchronous mode and the target database transitions to a normal OpenEdge database, a maximum of one transaction per client is lost. If you are running in asynchronous mode, some number of transactions can be lost per client, depending on how far behind the OpenEdge Replication server was when the connection was lost. It also depends on how far behind your source database and server are.

To assist you in determining the number of transactions lost, OpenEdge Replication provides latency reporting. For more information about latency reporting, see the [“Latency reporting”](#) section on page 3–27 and the [“DSRUTIL utility”](#) section on page 5–11.

The advantage to transitioning is that your users will have full access to an alternate database as a hot standby during a failure condition. If you cannot allow for the chance that a transaction is lost during failure processing, the target database with manual transition set will synchronize with your source database once the connection is re-established, so long as you do not transition it. In this scenario there is no risk to transactions being lost.

## Handling OpenEdge Replication failure conditions

When OpenEdge Replication is running, normal database activity occurs. If OpenEdge Replication encounters a failure, failure processing occurs to try to recover. OpenEdge Replication handles a variety of failure conditions. These failure conditions are divided into two general categories, which are described in the following sections:

- [Database crash](#)
- [Lost connection](#)

---

**Note:** Shutting down either the source or target database using PROSHUT without a forced shutdown is considered a normal event and not a failure by either the OpenEdge Replication server or agent. Additionally, if the OpenEdge Replication server is terminated with `DSRUTIL db-name -C terminate Server` or the OpenEdge Replication agent is terminated with `DSRUTIL db-name -C terminate Agent`, this is also not considered a failure.

---

### Database crash

OpenEdge Replication responds if either the source or the target database crashes.

If the target database crashes, the OpenEdge Replication agent fails. When this happens, the OpenEdge Replication server performs failure recovery because it loses its connection with the OpenEdge Replication agent.

If the source database crashes, the OpenEdge Replication server fails. When this happens, the OpenEdge Replication agent goes into failure processing because it has lost its connection with the server. See the [“Lost connection”](#) section on page 3–32 for more information.

### Lost connection

A lost connection, in which a OpenEdge Replication server loses contact with its agents, can occur for a variety of reasons, including:

- Abnormal termination of the OpenEdge Replication server
- A system crash on the machine where the OpenEdge Replication server is running
- A break in the TCP/IP connection between the OpenEdge Replication server and its agents

When a lost connection occurs, the source goes into failure recovery and the target goes into transition.

## Detecting TCP/IP communications failures

It is possible that a break in the TCP/IP connection between an OpenEdge Replication server and its agents can go undetected. For example, in a large, complex network with a number of bridges and routers, a segment of the network could go down, interrupting the communications between the server host machine and the agent host machine. However, TCP/IP would still be running in other segments of the network and the server or agent might be unaware of the break.

You can ensure that TCP/IP failures are detected by having the server and agent ping each other. If there is no response to the ping, the connection is assumed to be broken and failure recovery begins.

Use the server `Rep1-Keep-Alive` property to enable pinging between the server and the agent. A ping is sent every thirty seconds. The `Rep1-Keep-Alive` property allows you to specify the number of seconds to wait for a response to the ping. If there is no response for the specified period (the default is 300 seconds), a connection failure condition is set and failure recovery begins.

For more information about configuring `Rep1-Keep-Alive`, see the “[Server properties](#)” section on page 5-5.

## Source failure recovery after losing connection

When the OpenEdge Replication server loses connection with one or more OpenEdge Replication agents, the OpenEdge Replication server tries to contact the OpenEdge Replication agent and establish connection for an amount of time determined by the `connect-timeout` value set in the OpenEdge Replication server properties file.

The OpenEdge Replication server does the following:

1. The OpenEdge Replication server recognizes that there has been an agent failure. The server places itself into a state that allows continuous RDBMS activity, as if the OpenEdge Replication server is not running.
2. The OpenEdge Replication server tries to reconnect to OpenEdge Replication agents for a set amount of time.

Source database activity by clients is still allowed unless synchronous replication is being used or schema updates are being performed by a process.

3. If the OpenEdge Replication server is able to reconnect to the OpenEdge Replication agent, it again begins processing AI blocks from the RDBMS. When it gets within ten AI blocks of the RDBMS, the OpenEdge Replication server halts normal database activity and completes the synchronization process.

Schema updates are not allowed while the OpenEdge Replication server is performing synchronization. If schema updates are being performed when failure recovery synchronization begins, source database updates will block until failure recovery completes.

Source database activity cannot continue without the agent connected when synchronous replication is being used.

4. When synchronization is completed, the OpenEdge Replication server reinserts itself back into the AI block write process and the RDBMS will be unlocked, allowing normal database activity and replication activity to continue.

If the OpenEdge Replication server is unable to reconnect to all agents or to a critical agent in the configured `connect-timeout` period, the OpenEdge Replication server will terminate and source database activity will continue. In other words, if there are no critical agents, the server must be able to reconnect to all agents or it will terminate. If one agent is a critical agent, the server will continue if it can reconnect to the single critical agent. When source database activity continues while the OpenEdge Replication server is not running, be sure that there is enough AI extent space to handle all database activity until the OpenEdge Replication server is restarted and replication continues.

There is a possibility when failure recovery is being performed and synchronization takes place that the OpenEdge Replication server might not catch up to the RDBMS. During this time, all target databases are not up to date with the source.

### **Target transition after losing connection**

When the OpenEdge Replication agent loses contact with the OpenEdge Replication server, the OpenEdge Replication agent goes into transition. During transition after a lost connection, the OpenEdge Replication agent listens for the OpenEdge Replication server in order to re-establish connection, if auto transition is configured, for a set amount of time determined by the `transition-timeout` value in the OpenEdge Replication agent properties file. The OpenEdge Replication agent does the following:

1. When the OpenEdge Replication agent first loses contact with the OpenEdge Replication server, it goes into a pre-transition state where it listens for the OpenEdge Replication server.
2. If contact is not established and the agent is configured to perform auto transition, the target database is transitioned to a normal OpenEdge database. A normal OpenEdge database means that all standard client connections and updates can be performed on it.
3. If manual transition is configured, the OpenEdge Replication agent continues waiting until the database administrator initiates a change. Until the administrator initiates a change using the DSRUTIL utility, the database will remain in an unknown state.

For more information about transition, see the [“Transitioning to the target database”](#) section on page 3–35.

## Transitioning to the target database

Transitioning is the process by which the target database becomes a normal database after a failure. See the “[Handling OpenEdge Replication failure conditions](#)” section on page 3–32 for more information about the types of failure that OpenEdge Replication responds to. The following sections describe:

- [Configuring transition](#)
- [Transition processing](#)
- [Manually applying after-image extents](#)
- [Re-enabling OpenEdge Replication after transition](#)

### Configuring transition

For transitioning to operate properly, you must configure it. The following section describes choosing one or two target databases and automatic or manual transition processing.

#### Choosing one or two target databases

OpenEdge Replication allows for up to two target databases, though only one of the target databases can be used to automatically transition into a normal OpenEdge database. The target database that will automatically transition to a normal OpenEdge database in a failure condition is the first target database whose `[control-agent.agent]` properties in the `server.rep1.properties` file have `critical` set to 1 and `transition` set to `auto`. If you set more than one target database to `critical=1` and `transition=auto`, OpenEdge Replication will only recognize the first target database specified in the `server.rep1.properties` file for automatic transition. The second database will have to be transitioned manually.

The target database you choose as critical is the database you can use as a hot standby should the source database become unavailable. This target database should be on a machine that has reliable TCP/IP connectivity and has the resources for clients to connect to and perform database updates should your source database become unavailable.

Only one target database should be designated to transition. If you transition two target databases and users make updates to both databases, you will not have a single target database with which to resource your source database.

#### Setting up automatic transition

When the OpenEdge Replication agent loses contact with the OpenEdge Replication server, the agent will wait for a configured amount of time, known as `transition-timeout`, for the server to reconnect. If the OpenEdge Replication server does not reconnect before the `transition-timeout` expires, the target database will be transitioned to a normal database by the agent.

For automatic transition to be performed by the OpenEdge Replication agent, the following must be true:

- The OpenEdge Replication server property `transition` must be set to `auto`.
- The control-agent property `critical` must be set to 1.
- The server property `transition-timeout` must be set to a reasonable value.

You can use the sample properties file in [Figure 3–3](#) as a guide.

```
# OpenEdge Replication properties file for a database that will be used
# as both a source and target database for OpenEdge Replication.
#
[server]
    control-agents=agent1
    database=source
    transition=auto
    transition-timeout=1200

[control-agent.agent1]
    name=agent1
    database=target
    host=localhost
    port=4502
    connect-timeout=120
    replication-method=async
    critical=1
```

**Figure 3–3: Server properties file with automatic transition**

As [Figure 3–3](#) shows, the OpenEdge Replication agent, `agent1`, waits for connection from the OpenEdge Replication server for 1200 seconds, or 20 minutes, before it performs transition.

### Setting up manual transition

With manual transition, when the OpenEdge Replication agent loses contact with the OpenEdge Replication server, it waits indefinitely for a transition to be performed by the DBA. If the OpenEdge Replication server reconnects any time before transition is performed, normal processing resumes.

In order for manual transition to be performed by the OpenEdge Replication agent, the following must be configured:

- The server property `transition` must be set to `manual`.
- The control-agent property `critical` can be set to 0, for noncritical.
- The server property `transition-timeout` should be set to a reasonable value. Even though this value is not used for manual transition, it should be set in case transition is changed to `auto`.

The sample properties file in [Figure 3–4](#) can be used as a guide.

```
# OpenEdge Replication properties file for a database that will be used
# as both a source and target database for OpenEdge Replication.
#
[server]
    control-agents=agent1
    database=source
    transition>manual
    transition-timeout=1200

[control-agent.agent1]
    name=agent1
    database=target
    host=localhost
    port=4502
    connect-timeout=120
    replication-method=async
    critical=0
```

**Figure 3–4: Server properties file with manual transition**

As [Figure 3–4](#) shows, the OpenEdge Replication agent, agent1, will wait for connection from the OpenEdge Replication server indefinitely for the DBA to perform transition.

Before performing a manual transition, be sure to refer to the “[Manually applying after-image extents](#)” section on page 3–38.

To perform transition manually, use the following command:

```
DSRUTIL target-db-name -C transition Agent
```

## Transition processing

The OpenEdge Replication agent transitions a target database to a normal database that can be used as a hot standby in the event of a failure of the source database; therefore, the database type specified in the property file can be source or normal. The target database can be accessed for all database activity.

The actual process the OpenEdge Replication agent performs is as follows:

1. All database connections are disallowed.
2. All user processes connected to the target database are shut down to prevent locks on critical resources required by the database manager.
3. All active transactions are undone or rolled back.
4. OpenEdge Replication is disabled for the target database. Specifically, OpenEdge Replication can no longer be performed for this database.

5. Database connections are allowed.
6. The now-normal database is shut down.

Once these steps are complete, the now-normal database can be restarted using production arguments. When the source machine again becomes available, all activity performed on the database that was once the target database must be backed up and then restored to the source machine.

## Manually applying after-image extents

During normal conditions, the target database is automatically updated with data from the source database. The updates are transmitted as blocks of data from the source's AI transaction log. After a failure condition, it is possible that the most recent blocks were not able to reach the target. For example, a TCP/IP failure could lose packets that were waiting for transmission.

You can recover the missing data by manually applying the after-image extents that contain the data that has not already been applied to the target. However, you can do so only under the following conditions:

- The transition property (under the [server] directive in the properties file) must be set to `manual`.
- The agent must be in a pre-transition state.
- The storage device that contains the source database AI extents must be accessible by the target machine. You cannot save AI extents on the same system where the source is running because the extents would not be accessible after a system crash. Network area storage (NAS) or storage area network (SAN) devices are ideal for storing AI extents.



### To apply AI extents:

1. Display failure recovery information using the following command:

```
dsrutil target-db-name -C RECOVERY Agent
```

2. Determine the AI extent number from the command output. Select the last-applied AI extent or, if the last-applied extent was completely processed, select the next available extent.



The following is an example of the relevant section of the command output:

```

      .
      .
      .
Last AI Extent processed
AIMAGE BEGIN date:      Tue Oct 18 13:33:31 2005
AIMAGE NEW date:        Tue Oct 18 13:48:55 2005
After Image File Number:      3
File Last Opened:       Tue Oct 18 13:48:55 2005
Completely Applied to Target:   No

```

Determine the file number of the last-applied AI extent. In this example the last-applied extent is 3. Since the value after Completely Applied to Target is No, you will use 3 as the *extent-name* in [Step 3](#).

If the value after Completely Applied to Target were Yes, you would use 4 as the *extent-name* in [Step 3](#), since extent 3 would have already been completely applied.

---

**Note:** The After Image File Number corresponds to the Seqno that appears in the extent list when you use the `applyextent` qualifier with DSRUTIL. For more information, see the “[DSRUTIL applyextent qualifier](#)” section on page 5–13.

---

3. Use the appropriate AI extent file number in the following command:

```
dsrutil target-db-name -C ApplyExtent extent-name
```

As the command executes, it performs the following validation:

- The extent must exist and be valid.
- The status of the extent must be FULL, BUSY, or LOCKED.
- The AI extent file number either must be the same as the AI extent file number for the last AI block processed by the agent, or it must be the next extent file number (if the previous AI extent was completely processed).

After you apply the AI extent, you can begin a manual transition of the target. For more information, see the “[Setting up manual transition](#)” section on page 3–36.

## Re-enabling OpenEdge Replication after transition

After transition, you can re-enable what was once the target database on the source machine. To accomplish this, you can perform the following steps, or you can use the failback and transition procedures described in [Chapter 4, “OpenEdge Replication: From Failure to Recovery.”](#)



### To re-enable what was once the target database on the source machine after transition:

1. Shut down the former target database from the target database directory using the following command:

```
proshut target-db-name -by
```

2. Back up the former target database from the target database directory using the following command:

```
probkup target-db-name target-db-name.bak
```

3. Copy or move *target-db-name.bak* to the source machine using FTP or Remote Copy (RCP). Be sure to use binary transfer.
4. Disable OpenEdge Replication for the source database from the source database directory using the following command:

```
proutil source-db-name -C disablesitereplication source
```

5. Restore the source database from the target database backup in the source database directory using the following command:

```
prorest source-db-name target-db-name.bak
```

The structure file that existed prior to transition will be used by PROREST, so the physical structure of the newly restored source database matches the structure file from the restore. If PROREST does not find a structure file, the structure of the newly restored source database will not match what it was before the restore.

For more details about backup and recovery strategies, see *OpenEdge Data Management: Database Administration*.

6. Enable after-imaging on the source database from the source database directory using the following command:

```
rfutil source-db-name -C aimage begin
```

7. Enable the source database for OpenEdge Replication in the source database directory using the following command:

```
proutil source-db-name -C enablesitereplication source
```

8. Delete the file `source-db-name.repl.recovery`.
9. Enable the target database for OpenEdge Replication in the target database directory using the following command:

```
proutil target-db-name -C enablesitereplication target
```

10. Delete the file `target-db-name .repl.recovery`.

The source machine now has the source database, and the target database has been re-enabled. OpenEdge Replication can again be used on both databases. To start OpenEdge Replication, see the [“OpenEdge Replication startup and initialization process”](#) section on page 3–19.

### Re-enabling a second target database

If you have a second target database that was not designated as the transition database, you can re-enable OpenEdge Replication for the second target database.



#### To re-enable OpenEdge Replication for the second target database directory:

1. Disable OpenEdge Replication for the second target database using the following command:

```
proutil target2-db-name -C disablesitereplication target
```

2. Restore the second target database from the target database using the following command:

```
prorest target2-db-name target-db-name.bak
```

The structure file that existed prior to transition will be used by PROREST, so the physical structure of the newly restored source database matches the structure file from the restore. If PROREST does not find a structure file, the structure of the newly restored source database will not match what it was before the restore. For more details about backup and recovery strategies, see *OpenEdge Data Management: Database Administration*.

3. Enable the second target database for OpenEdge Replication using the following command:

```
proutil target2-db-name -C enablesitereplication target
```

4. Delete the file *target2-db-name.rep1.recovery*.

The source machine now has the source database, and the target databases have been re-enabled. OpenEdge Replication can again be used on the three databases. For details about starting OpenEdge Replication, see the [“OpenEdge Replication startup and initialization process”](#) section on page 3–19.

---

## OpenEdge Replication: From Failure to Recovery

---

This chapter describes the OpenEdge Replication failover and failback process, which moves secondary database activity on one machine back to the primary database on another machine.

The chapter provides information about the following topics:

- [Overview of database failure recovery](#)
- [Transition](#)
- [How OpenEdge Replication works](#)
- [Recovery from transition failures](#)
- [Transition logging](#)
- [Transition properties](#)
- [Sample startup parameter file](#)
- [Reference](#)

## Overview of database failure recovery

When you enable OpenEdge Replication, you replicate a database on a primary machine to one or two (at most) secondary databases on a different machine or machines. In the event of a failure of either the primary database or the machine on which it runs, database activity can be failed over to one of the secondary databases. Activity continues on that second database and machine until the primary database or machine is again available.

Typically, a failure is indicated by a loss of communications between the Replication server on the primary database and the Replication agent on the secondary database. In this situation, the Replication agent enters a state known as *pretransition*, in which the agent waits until either the Replication server reconnects or transition is performed (automatically or manually).

Once you transition the secondary database to a source database, all database update activity is *failed over* to the secondary database. Once the primary machine is again available, all database update activity can be failed back to it. This process is known as *failback*. Failback is essentially a failover from the secondary to the primary machine.

Once the failback process completes, all database update operations are again performed on the primary database, and the secondary database returns to read-only access.

## OpenEdge Replication failure recovery terminology

With the introduction of failback processing to OpenEdge Replication, it is important to further clarify some existing terminology, specifically as it relates to primary and secondary databases as well as source and target databases.

### Primary database

A *primary database* is the database that is updated from your application; the primary database is the one you initially enable as the OpenEdge Replication *source database*. During a failure recovery, the role of this database can switch from source to target and then back again to source. However, while the role of the database might change, the database itself remains the primary database. (See the “[Source database](#)” section on page 4–3 for details about the source database.)

### Secondary database

A *secondary database* is the initial replica of the primary database; the secondary database is the one you initially enable as the OpenEdge Replication *target database*. During a failure recovery, the role of the secondary database can switch from target to source and then back again to target. (See the “[Target database](#)” section on page 4–3 for details about the target database.)

## Source database

The source database is the database that can be updated by users. The source database:

- Is where users do their work and make their database updates
- Has both write access and read access
- Is not considered a source database until it is enabled as an OpenEdge Replication source database
- Is the database from which the OpenEdge Replication server replicates data to the target database

## Target database

The target database is an identical copy of the source database. A second target database can be set up for additional redundancy. The target database:

- Originates from the source database and contains the same data, schema, and logical structure as the source database
- Is updated solely by the OpenEdge Replication agent

## The OpenEdge Replication property files

Each OpenEdge Replication source database and target database has a property file, which contains information specific to that database. For example, the source properties file contains details including, among other things, the name of the source database, the name of the agent or agents the Replication server controls, what type of transition the agent will perform, and how long a Replication agent will wait after losing contact with the Replication server before performing automatic transition. The target properties file contains details such as the name of the agent and the name of the target database.

OpenEdge Replication provides two separate, **sample** property files: the `source.rep1.properties` file and the `target.rep1.properties` file. You can copy each of these files to modify and use as your replication property files, saving each with the correct name for your source database and target database (*source-db-name.rep1.properties* and *target-db-name.rep1.properties*).

The Replication failover and failback processes perform successfully only if you set all the properties appropriately for your database environment.

---

**Note:** Progress Software Corporation recommends that you test OpenEdge Replication (with the properties you have set) in a nonproduction environment before implementing it in a full production setting.

---

## Transition

A key component of replication failover and failback is the process known as transition. *Transition* is the changing of the database role to perform failover either when a failure occurs on the primary machine (which is hosting the source database) or from the secondary machine once the primary machine again becomes available.

### Enhancements to transition functionality

With the introduction of failback to OpenEdge Replication, both the scope and functionality of transition have increased in the following areas:

- Support for additional transition scenarios
- Intuitive transition processing
- Automated transition functions

#### Support for additional transition scenarios

Transition encompasses the following activities:

- The role of a **target** database can be transitioned to that of a **normal** or a **source** database. The database can be either online or offline when the transition is performed.

If the **target** database is transitioned to a **source** database, AI areas can be added to the target database if they are not present and AI can be started for the database.

If the **target** database is transitioned to a **normal** database, the database is available for database update operations, but it can never be transitioned to a source database.

- A **source** database can be transitioned to a **target** database. The database can be either online or offline during the transition. If transition is performed on the source and the source is online, a transition failover must be performed.
- A **source** database can be transitioned to a **normal** database, provided you first disable replication.
- You can configure transition to restart the transitioned database automatically after the transition process successfully completes.

#### Intuitive transition processing

The transition process is intuitive; it chooses the appropriate type of transition to perform based on the following criteria:

- Is the database a source or a target database?
- Is the database online or offline?
- Has a prior transition for the database completed successfully?
- Is the Replication server or agent running?
- Has a replication failure just occurred?



## Automated transition functions

All transition operations can now perform automated processing. Some of the automated processing is controlled by new transition properties in the replication property files. Other automated processing is based on the state of the database that is being transitioned.

The transition operation includes these (and other) automated functions:

- Shutdown of the transitioned database
- Restart of the transitioned database
- Addition of AI areas
- Starting of AI
- Notification of the peer to implicitly perform transition

## Planning for transition

To use transition effectively, you must plan for it in advance by setting **all** the transition properties. If you do not set the transition properties, full transition processing does not occur. Instead, the extent of transition is limited to changing the role of a database from target to normal.

Plan for transition by making all of the following decisions:

- **Which is the primary and which is the secondary database?**

To begin transition planning, decide which database is your primary database and which is your secondary database. The primary database is the one on which users initially perform updates.

- **What type of transition will occur?**

Consider whether the target database will be transitioned to a source database or normal database.

- **What type of after-imaging functionality should be implemented?**

You can choose to start AI automatically on the transitioned database. You can also add AI areas.

- **What types of backup do you want to perform during transition?**

Consider what can you accomplish with each backup method, and why it might be advantageous to use one method instead of another method.

- **Do you want the database to restart automatically after transition?**

Think about whether you want to control database startup or have the transition process start the database.

- **Will transition automatically attempt to recover in the event of a failure?**

A backup must exist in order for automatic recovery to occur.

Once you make each of these decisions, you can set the transition properties accordingly, as described in the [“Setting transition properties”](#) section on page 4–33.

## How OpenEdge Replication works

OpenEdge Replication typically occurs with activity between the Replication server on the source database on the primary machine and the Replication agent on the target database on the secondary machine. If the Replication agent loses communication contact with the Replication server, you perform failover to move all database update activity from the source database to the target database. Once the primary machine becomes available again, you can move, or fail back, all the database update activity to the primary database or machine. To minimize downtime to your application, you can schedule the failback process to run when you want.

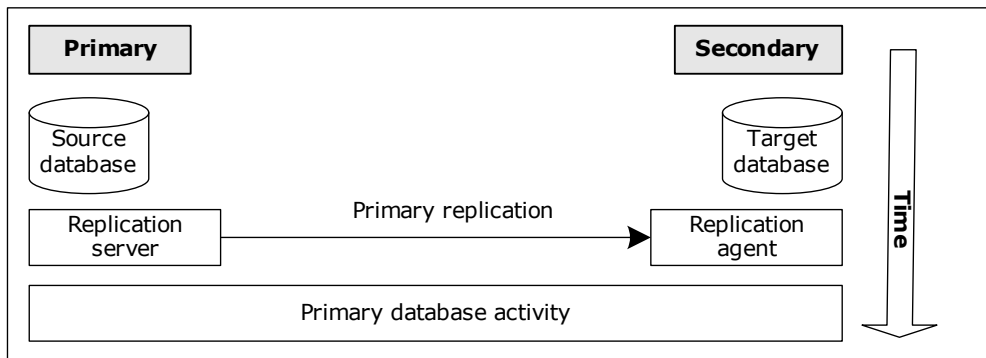
The steps in the OpenEdge Replication process are as follows:

1. During primary (normal) replication, the primary database has the role of source database and the secondary database has the role of target database. The Replication server exists on the source database and the Replication agent exists on the target database.
2. If there is a failure on the machine hosting the primary database, the Replication agent on the target database loses communication contact with the source database's Replication server.
3. The Replication agent on the target database enters pretransition.
4. Transition then occurs, making the secondary database (formerly a target) a source database.
5. All database activity is failed over to the secondary database, which is now functioning as a source database. The secondary database now becomes the production database.
6. At some point, the machine hosting the primary database is fixed.
7. Transition is initiated, and the primary database on the repaired machine changes its role from source to target. At this point, the secondary database is the source and the primary database is the target.
8. Secondary replication, which is essentially primary replication in reverse, is performed.
9. At a convenient time, the failback process can be performed to return the databases to the roles originally established, with the primary database as the source and the secondary database as the target. This can be done in either of the following ways:
  - Transition the secondary database using the failover command.
  - Transition the primary database and then do a separate transition of the secondary database. This is known as controlled transition.

Each of these steps is described in the following sections.

## Step 1: Primary replication before a failure

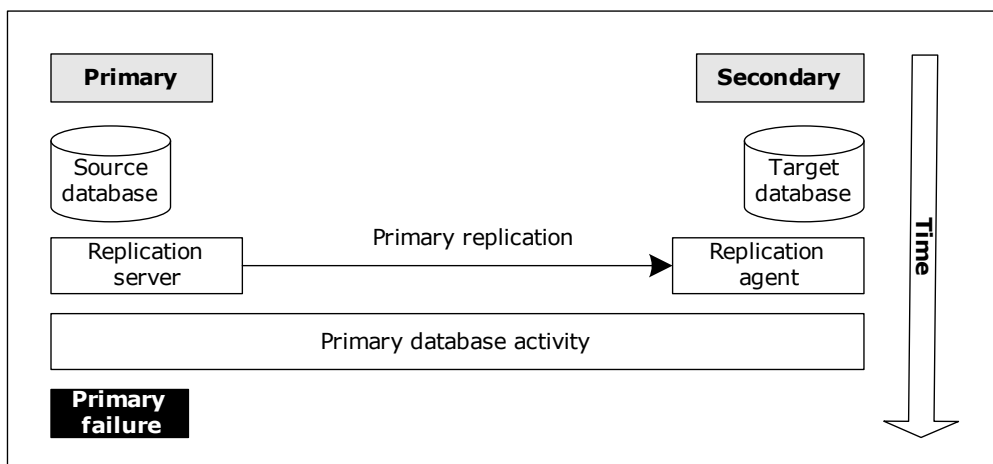
During typical OpenEdge Replication operation, activity occurs between the Replication server on the source database on the primary machine and the Replication agent on the target database on the secondary machine, as shown in Figure 4-1.



**Figure 4-1: Primary replication before a failure**

## Step 2: Primary machine failure

Replication activity continues until a failure occurs on the primary machine, as shown in Figure 4-2.

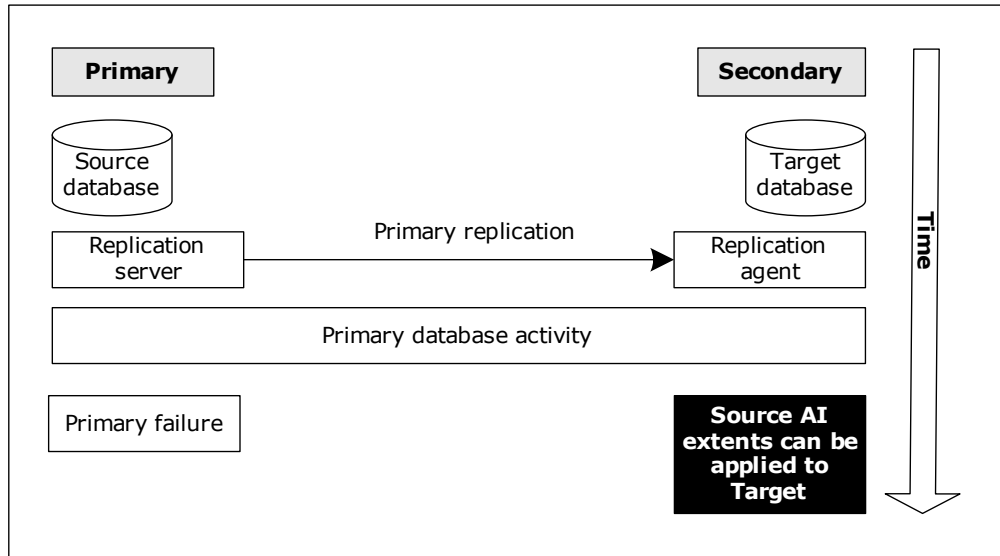


**Figure 4-2: Primary machine failure**

The Replication agent recognizes the failure as a TCP/IP communication error. Typical failures might be due to a network outage between the primary and secondary machines, a primary database crash, or a primary hardware crash.

### Step 3: Entering pretransition

Replication attempts to recover from a failure; however, if it is unable to do so, all database activity can be failed over, or *transitioned*, to the secondary database and machine. The first task in readying the secondary database for transition is to apply all unapplied source database after-image extents, as shown in Figure 4-3.



**Figure 4-3: Entering pretransition—applying all unapplied source database AI extents**

The application of the extents can be accomplished only if the secondary machine has access to these extents. For example, if the extents are stored on a storage area network (SAN) device, the secondary machine must have access to the device.



#### To apply all unapplied source database after-image extents:

1. Use the following command to determine the first after-image extent to apply:

```
dsrutil target -C recovery
```

The output from this command shows the source database after-image extent information.

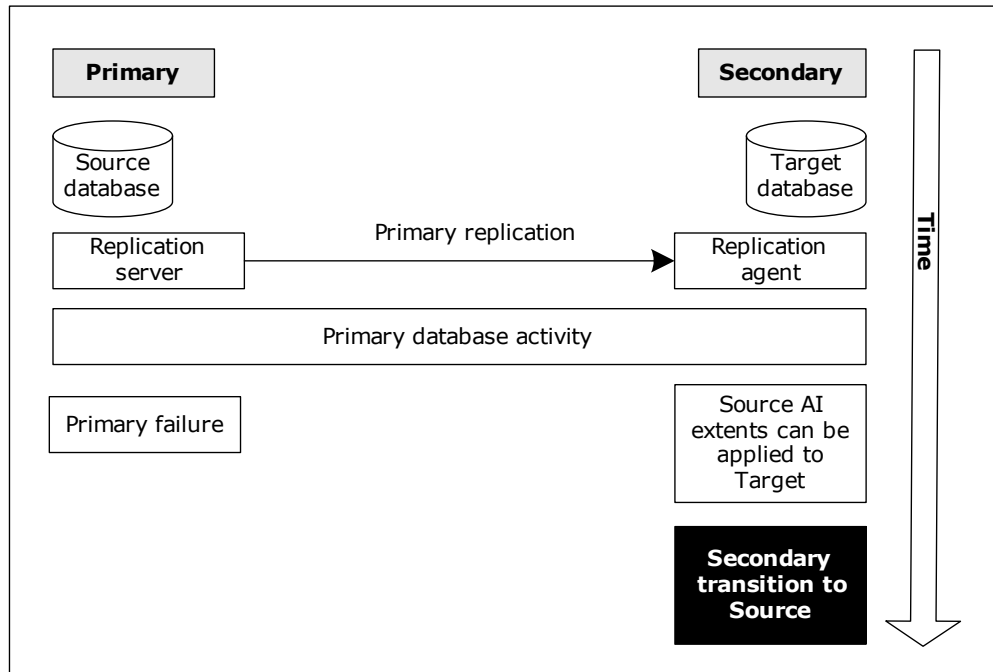
2. Beginning with the after-image extent listed, apply each source database after-image extent using the following command in succession until you have applied all of them:

```
dsrutil target -C applyExtent qualified-extent-name
```

If the source database after-image extents cannot be applied to the target database before transition, the target database will not contain any of the unapplied source database transactions. In this case, unless the transactions can be recreated and re-entered after transition, you **will** lose an indeterminate number of transactions.

## Step 4: Transitioning the target database to a source database

The next step in the failover process is to transition the target database to a source database, as shown in [Figure 4-4](#).



**Figure 4-4: Transitioning the target to a source database**



**To transition the database successfully:**

1. Properly configure the transition properties. See the [“Setting transition properties”](#) section on page 4-33 for details.
2. Use the following command to perform transition:

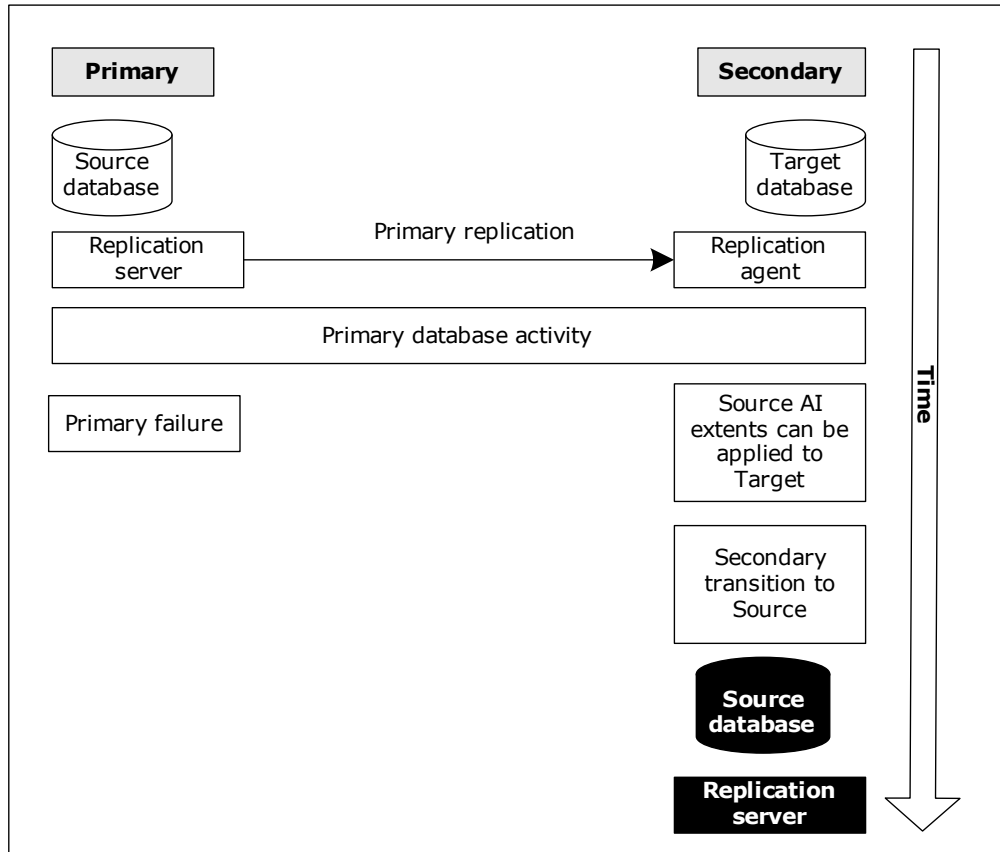
```
dsrutil target -C transition
```

You can also configure transition to occur automatically, so that neither you nor another DBA needs to explicitly execute the command to initiate it.

If you do configure automatic transition, however, be aware that source after-image extents cannot be applied before the target database is transitioned. You **will** need to redo all work that was not applied.

## Step 5: Failover

Once the secondary database is transitioned to a source database, all activity that was formerly performed on the primary machine can now be performed on the secondary machine. (This assumes, of course, that your application is installed and accessible on the secondary machine.)



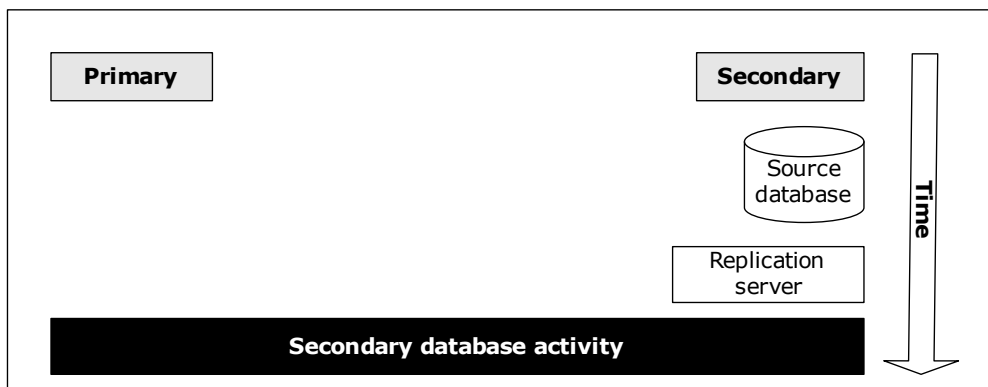
**Figure 4–5: Secondary database transitioned to source database**

You can configure transition to start the secondary database automatically once the transition process completes, or you can start the database manually. The decision is entirely up to you.

Since the secondary database is now a source database (as shown in [Figure 4–5](#)), the Replication server is started at the same time the secondary database is started. The Replication server will continue to make connection attempts to any configured agent. To increase the number of connection attempts that will occur, use the `defer-agent-startup` property. For more information about setting this property, see the [“Server properties”](#) section on page 5–5.

If you suspect that the primary machine will be down for an extended period of time, you can allow the Replication server to terminate due to its inability to connect to the configured agents. The Replication server can then be restarted on the primary machine when the machine is again up and running.

Once you successfully transition the secondary database from a target to a source database, all activity that was previously performed on the primary machine can be performed on the secondary machine, as shown in [Figure 4-6](#).

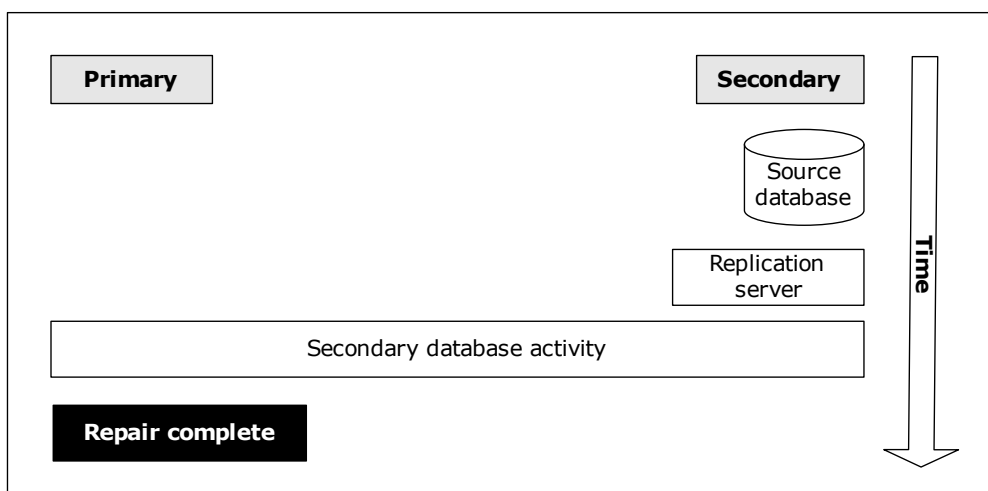


**Figure 4-6: Secondary database activity**

Activity can continue for as long as you find necessary. You should, however, begin to consider when to initiate the process required to move production processing back to your primary computer.

## Step 6: Primary machine repair complete

Once the repair of the primary computer is complete, as shown in [Figure 4-7](#), and the machine is again up and running, you can begin the initial processes involved with failing back production processing to the primary computer.



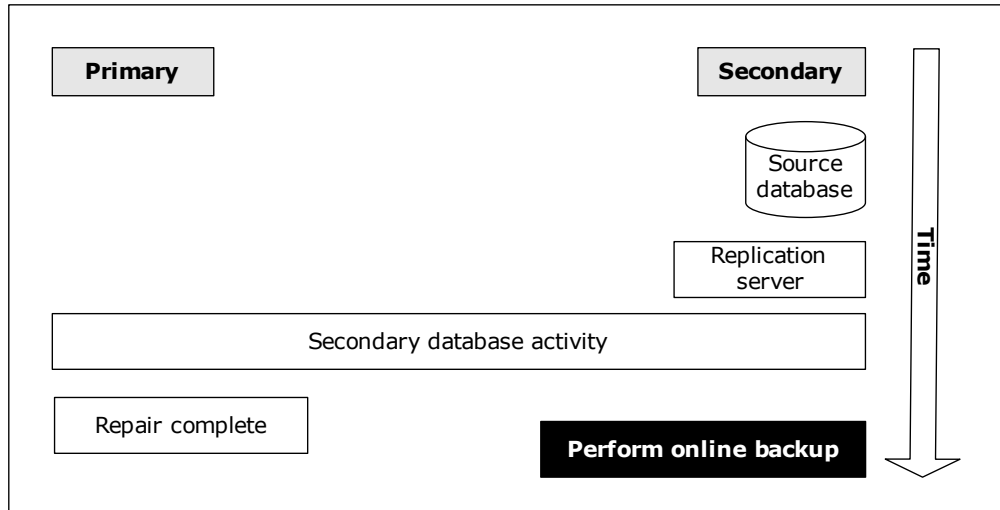
**Figure 4-7: Primary machine repair complete**



## Step 7: Initiating primary database transition

The first step in getting ready to fail back processing to the primary computer is to begin replication from the secondary database to the primary database on the primary machine. This process is known as *secondary replication*.

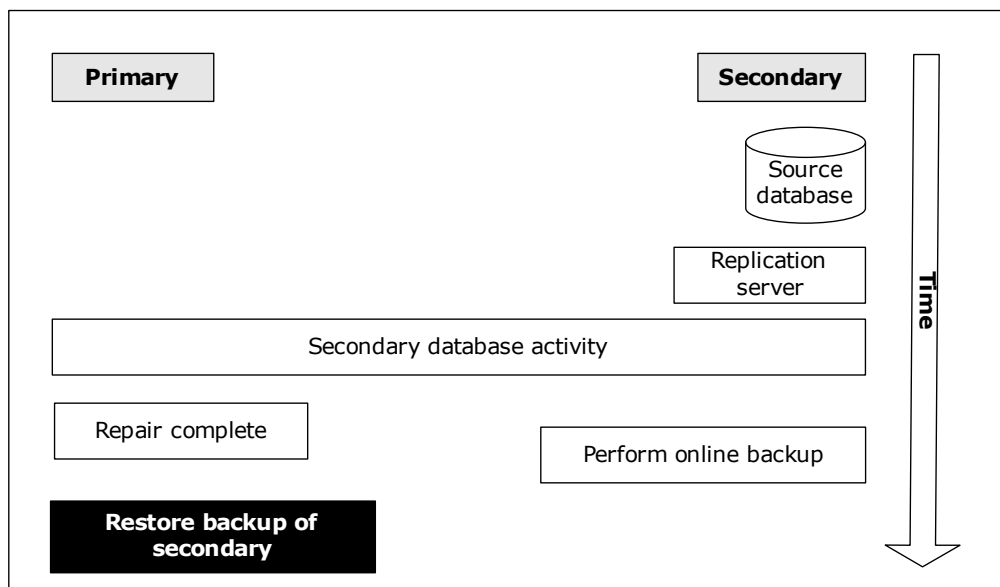
As shown in [Figure 4–8](#), setting up secondary replication begins with a backup of the secondary database; using an online backup limits the amount of downtime required.



**Figure 4–8: Performing an online backup of secondary database**

The backup files must be either sent to the primary computer or stored on a shared device that both the primary and secondary machines can access.

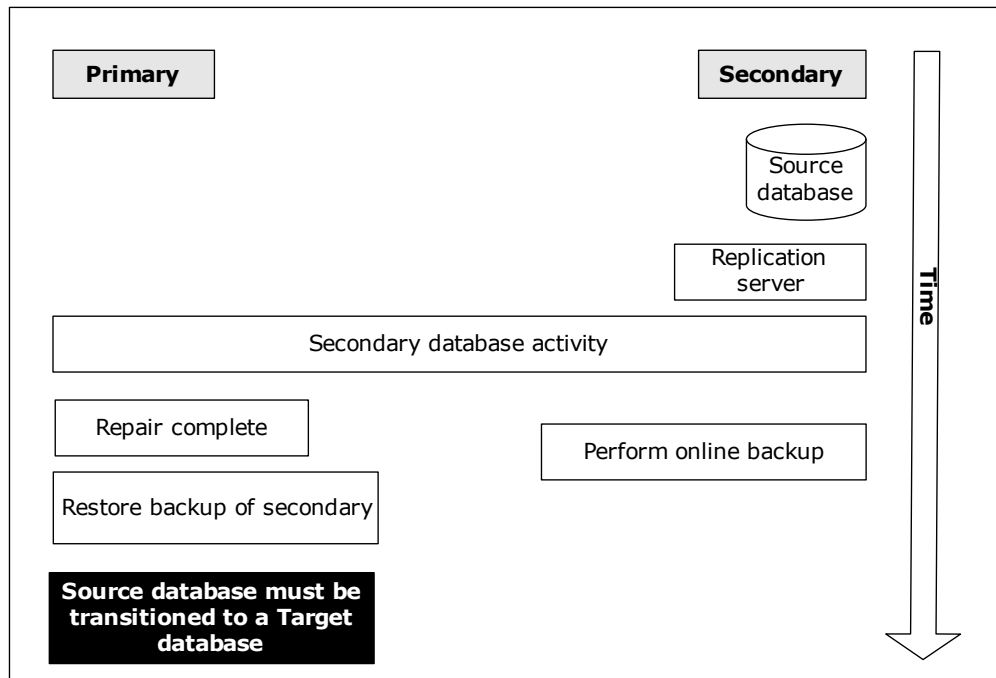
Once the backup files are available on the primary machine, you can restore the new primary database using the files, as shown in [Figure 4–9](#).



**Figure 4–9: Restoring backup of secondary database**

Before you restore the database, ensure that you have set up a structure file that the PROREST utility can use. For details about creating a structure file, see the [“Creating a structure file”](#) section on page 3–7.

You must now transition the primary database from a source to a target, as shown in [Figure 4–10](#).



**Figure 4–10: Transitioning source database to target**

You can use either of the following methods to transition the primary database:

- To transition the database **as it is restored**, use the following command:

```
prorest primary backup-file -REPLTransition
```

- To transition the database **after it has been restored**, first use the following command to restore it:

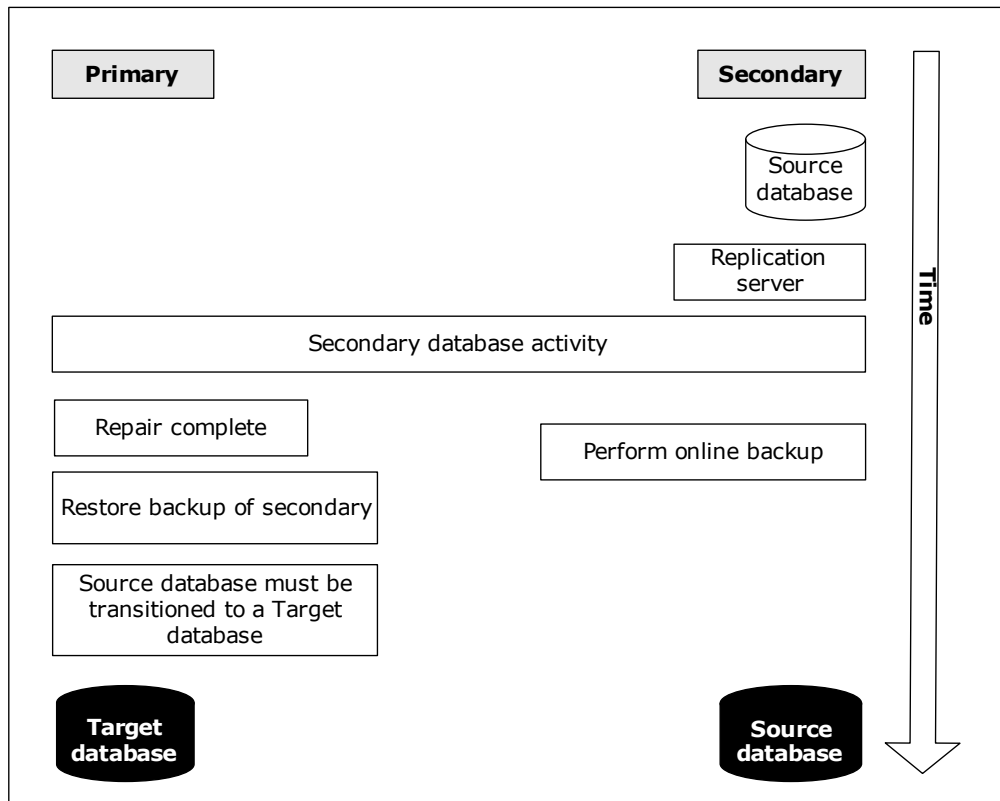
```
prorest primary backup-file
```

Then use the following command for transition:

```
proutil primary -C enableSiteReplication target
```

Regardless of which method you choose, the primary database will become a target database ready to be the replica of the production database on the secondary machine.

At this point, the secondary database is a source database and the primary database is a target database, as shown in [Figure 4–11](#).



**Figure 4–11: Transition completes**

You can now perform secondary replication.

## Step 8: Secondary replication is performed

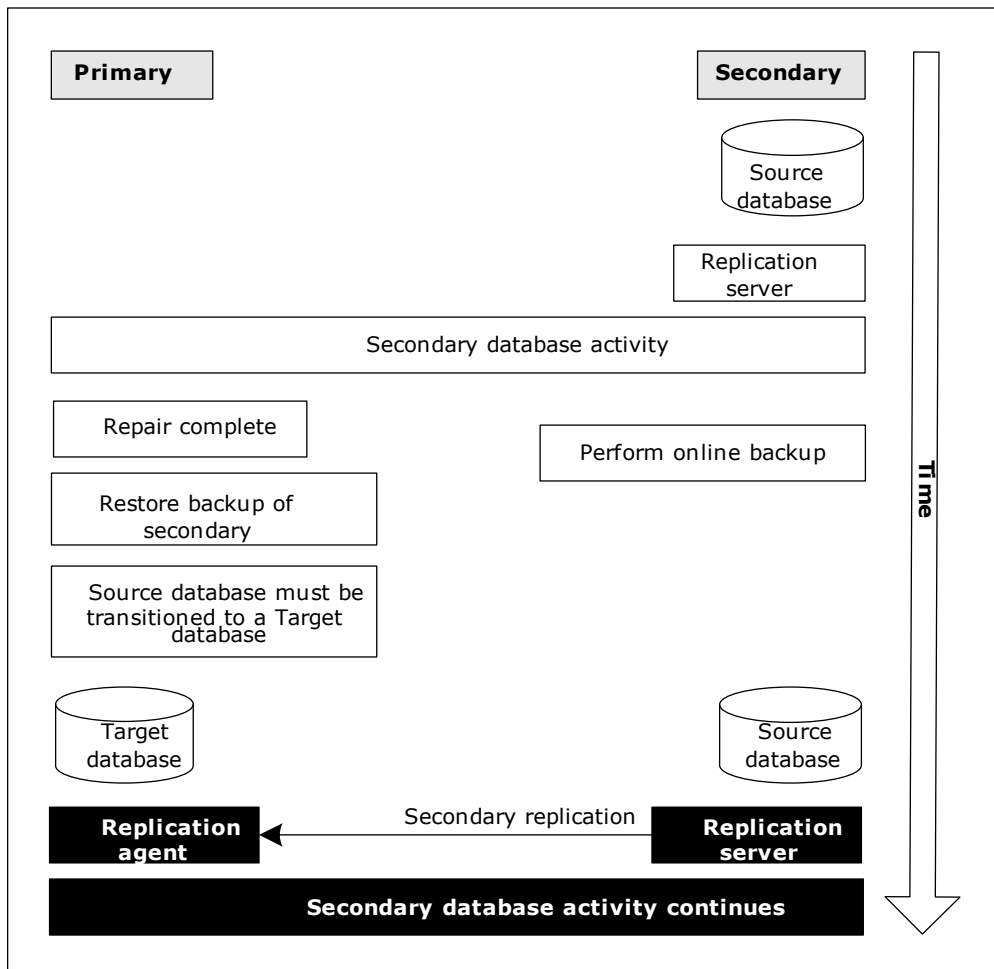
Once you start the primary database, the Replication agent starts as well. If you have configured the Replication server on the target machine with the `defer-agent-startup` property and it has not terminated, it will connect to this agent during its next connection attempt. (The Replication server might terminate if the amount of time you specified in the `defer-agent-startup` property or the `connect-timeout` property has expired.)

To restart the Replication server if it has terminated, use the following command:

```
dsrutil secondary -C restart server
```

Once the Replication server and agent begin communicating, secondary replication will begin.

As shown in [Figure 4–12](#), all secondary database activity that has taken place since the online backup was performed will be replicated to the primary database, as it is now the target database.



**Figure 4–12: Secondary replication occurs**

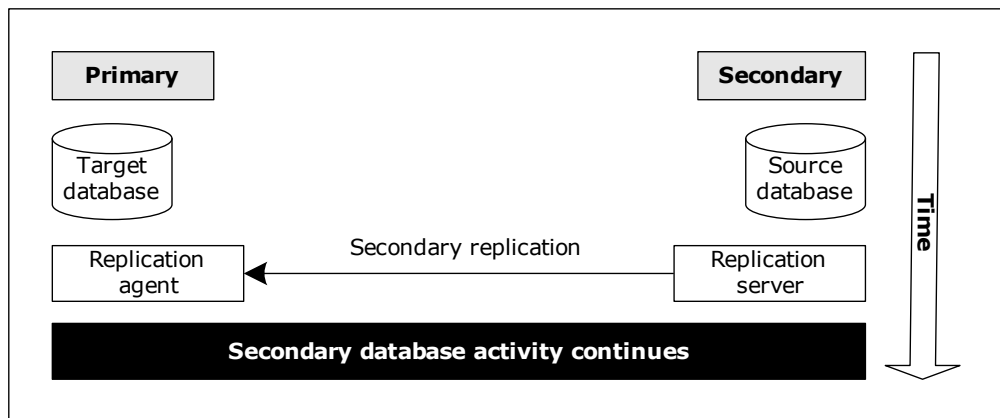
## Step 9: The Replication failback process

You can perform OpenEdge Replication failback production processing to the primary computer by using either of the following methods:

- Transitioning the secondary database using the `failover` command modifier
- Performing a controlled transition, in which you transition the primary database and then perform a separate transition of the secondary database

### Failback processing using transition failover

When secondary replication is being performed, as shown in [Figure 4-13](#), both databases are up and running and all secondary transactions are being replicated to the primary database.



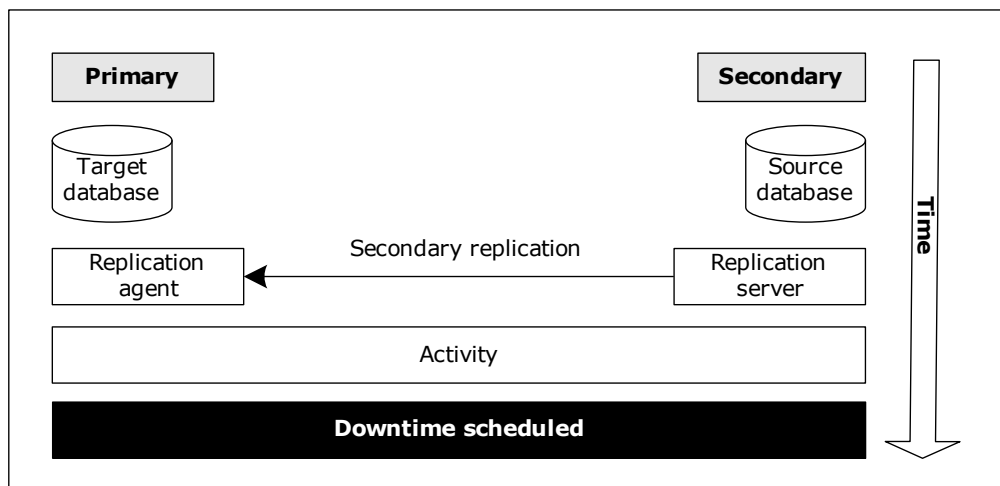
**Figure 4-13: Secondary replication continues (before transition failover)**

At this point, the secondary is considered the production database and the primary is considered the replica.

While secondary replication is occurring and the Replication server and agent are actively performing Replication, you must determine when the best time is to fail back production processing to the primary computer.

When you begin failback processing, it is critical that no users be connected to either the primary or the secondary database. You can quickly ensure this by shutting down and restarting both databases. Once you verify that no users are connected to either database, you can start the failback process.

[Figure 4-14](#) illustrates the scheduling of database downtime.



**Figure 4-14: Scheduling downtime to perform failback**

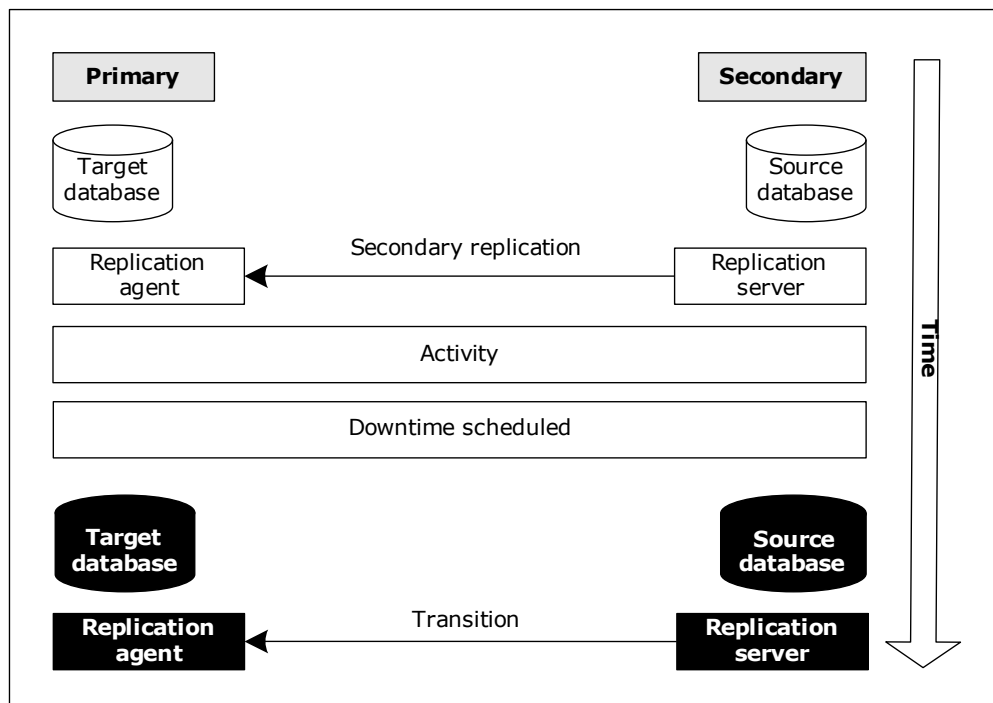
To initiate the failback process using transition failover, issue the following command on the second machine:

```
dsrutil secondary -C transition failover
```

This command instructs the secondary database to begin transition. The failover command modifier instructs OpenEdge Replication that this is a failover transition and causes Replication to transition both the primary and secondary databases.

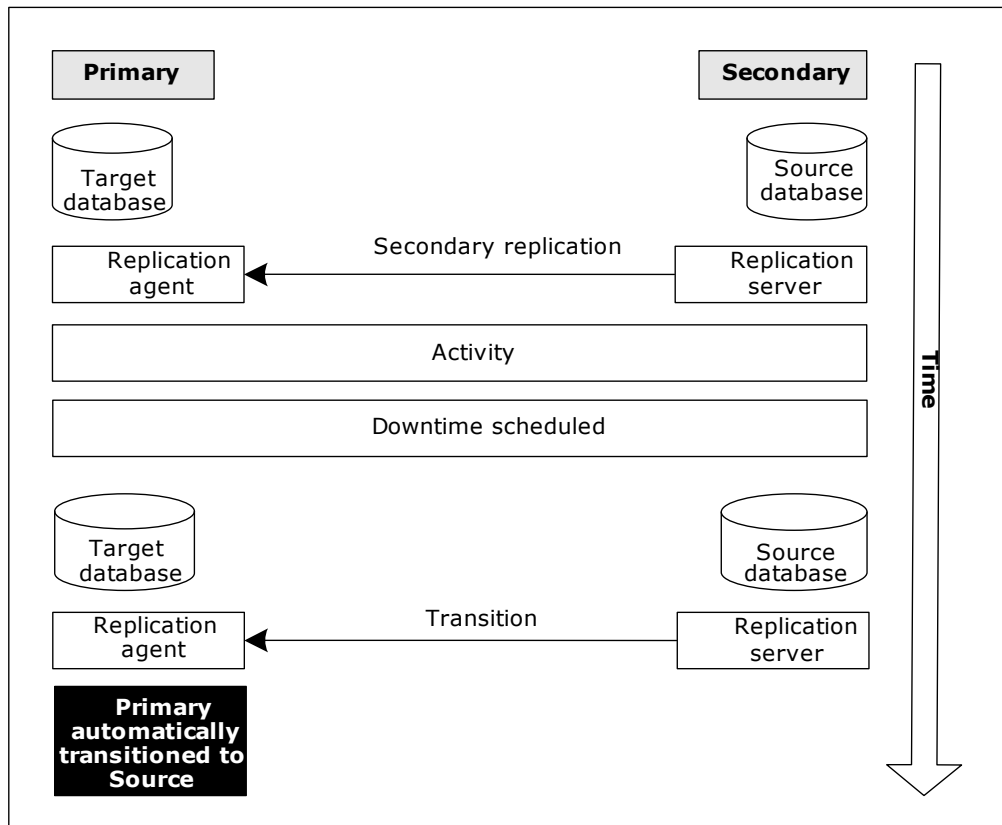
When the transition process begins, the Replication server informs the Replication agent on the primary machine to begin preparing the primary database for transition. At this point, the secondary database is shut down and then restarted.

Once the startup synchronization process is complete, the actual transition process can begin, as shown in [Figure 4-15](#).



**Figure 4-15: Transitioning the primary database**

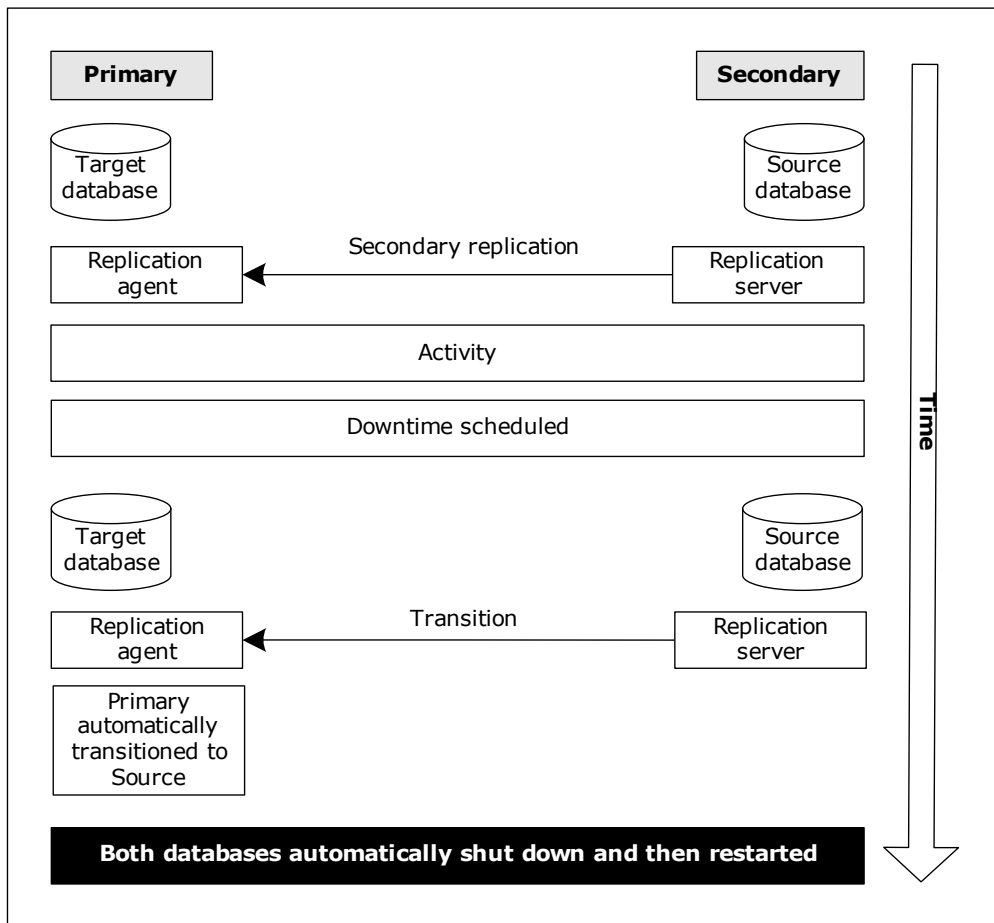
Once the startup synchronization process completes normally, the transition of the primary database is performed as configured, as shown in [Figure 4–16](#).



**Figure 4–16: Transitioning the primary database**

Once the transition of the primary database reaches a critical point (that is, immediately before the database is to be restarted), the transition of the secondary database is performed. Once the transition of the secondary database completes normally, the completion of the transition of the primary database is started.

Once transition completes normally for both databases, the databases will be restarted in their new roles, as shown in [Figure 4–17](#).



**Figure 4–17: Databases started in new roles**



As shown in [Figure 4–18](#), the roles of both databases have been reversed. The primary database is again the production database and the secondary database is again the replica. Primary replication is again being performed as it was before the initial failure occurred. The Replication server is replicating all primary transactions to the secondary database.

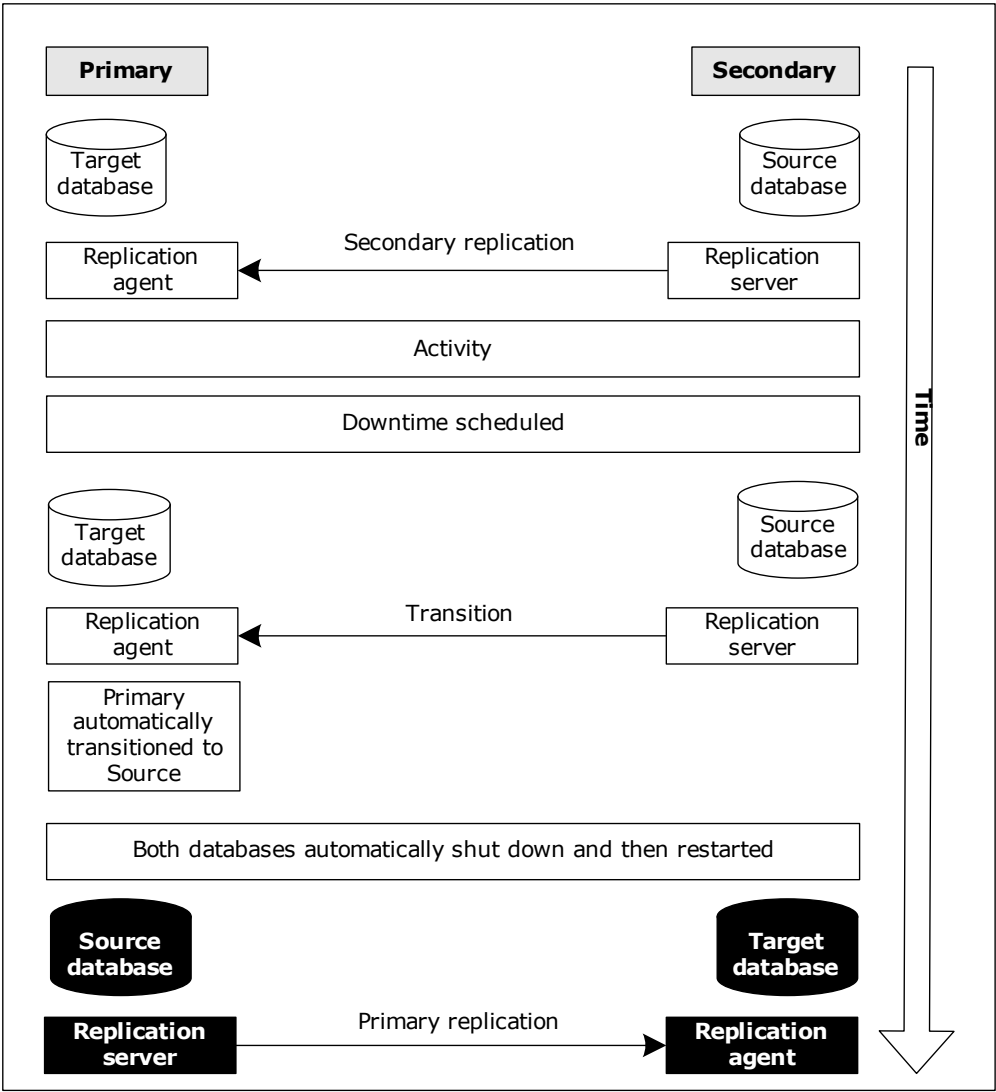


Figure 4–18: Primary replication activity is occurring

## Advantages and disadvantages of using transition failover to perform failback

There are both advantages and disadvantages to using transition failover in failback processing.

The advantages are as follows:

- As long as you have configured transition properly, you can initiate the entire failback process by executing one command. There are no manual steps required, and you can accomplish everything from one machine.
- When the transition process completes, the roles of both databases have been reversed and the databases are restarted and resume activity.

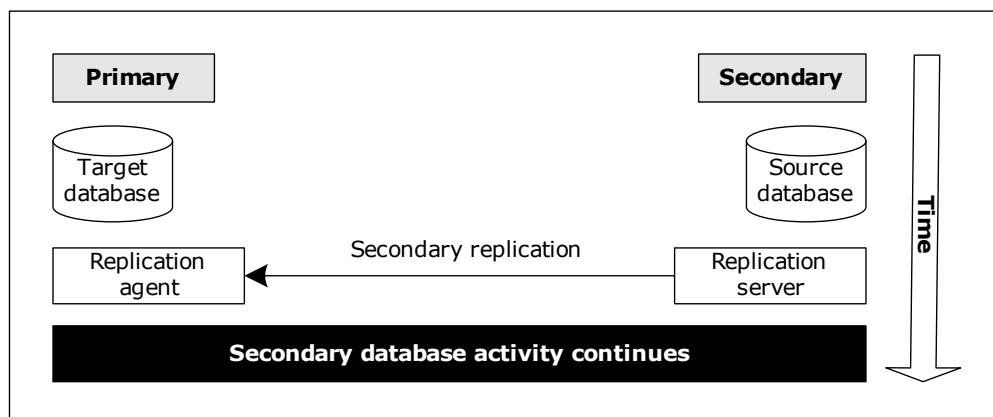
The disadvantages are as follows:

- The transition process is an all-or-nothing event. If transition fails for either database, both databases are restored to their original state, provided recovery backups were properly configured. The result is that the primary database remains the target and the secondary database remains the source.
- Both the primary and secondary databases are offline during the entire transition process. If a failure occurs during the transition of the primary database, that database must be restored in order to return it to its original state. The restoration of this database might be a time-consuming operation during which both the primary and secondary databases are down.

A solution to the potentially lengthy downtime in the event of a transition failure involves performing failback by using controlled transition. Note, however, that this process requires a greater level of DBA intervention and access to both the primary and secondary machines.

## Failback processing using controlled transition

When secondary replication is being performed, as shown in [Figure 4–19](#), both databases are up and running and all secondary transactions are being replicated to the primary database.

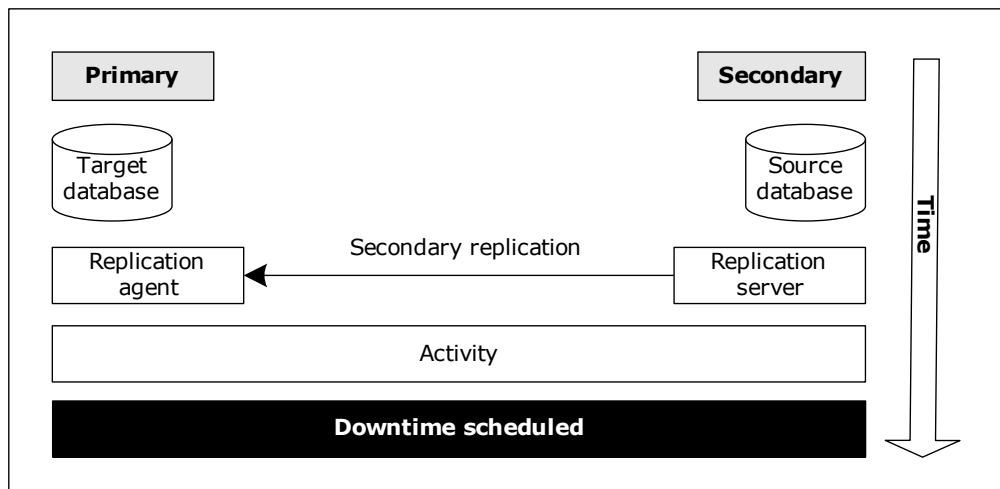


**Figure 4–19: Secondary replication continues (before controlled transition)**

At this point, the secondary is considered the production database and the primary is considered the replica.

While secondary replication is occurring and the Replication server and agent are actively performing Replication, you must determine when the best time is to fail back production processing to the primary computer.

Figure 4–20 illustrates the scheduling of database downtime.



**Figure 4–20: Scheduling downtime to perform failback**

When you begin failback processing, it is critical that no users be connected to either the primary or the secondary database. Both databases must be shut down to transition them.

It is recommended that the transition configuration for both the primary and the secondary databases be checked and modified if needed at this time.

Do not restart the databases after transition, so that you can perform special actions in the event of a transition failure. Once you verify that no users are connected to either database, you can start the failback process.



#### To initiate the failback process using controlled transition:

1. Shut down and restart both databases. Doing so ensures that all source activity is flushed and, in turn, replicated to the target database.
2. Verify the synchronization of the databases by performing one of the following:
  - Examining the database log file
  - Using the following command:

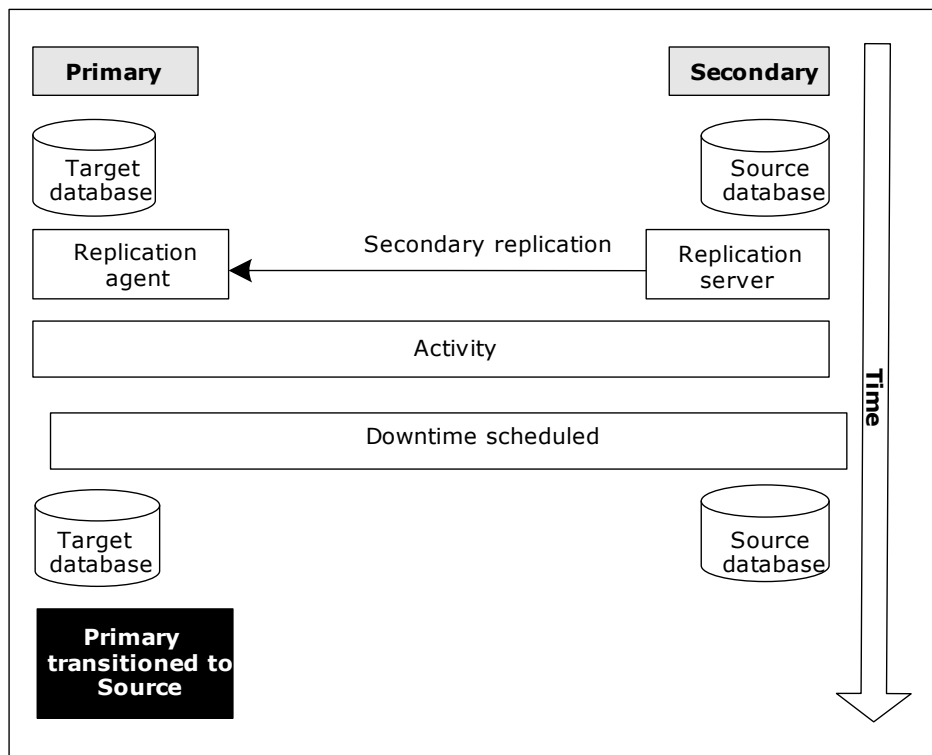
```
dsrutil source -C status -detail
```

When a status of 3049 is returned, both databases are synchronized.

3. Shut down the databases again.
4. Issue the following command on the primary machine:

```
dsrutil primary -C transition
```

This command transitions the primary database into a source, as shown:



If the transition fails, you can restart the secondary database to allow production work to continue. You can then attempt the failback operation again when you can once more schedule downtime.

After the transition of the primary database to source database completes, transition of the secondary database can begin.

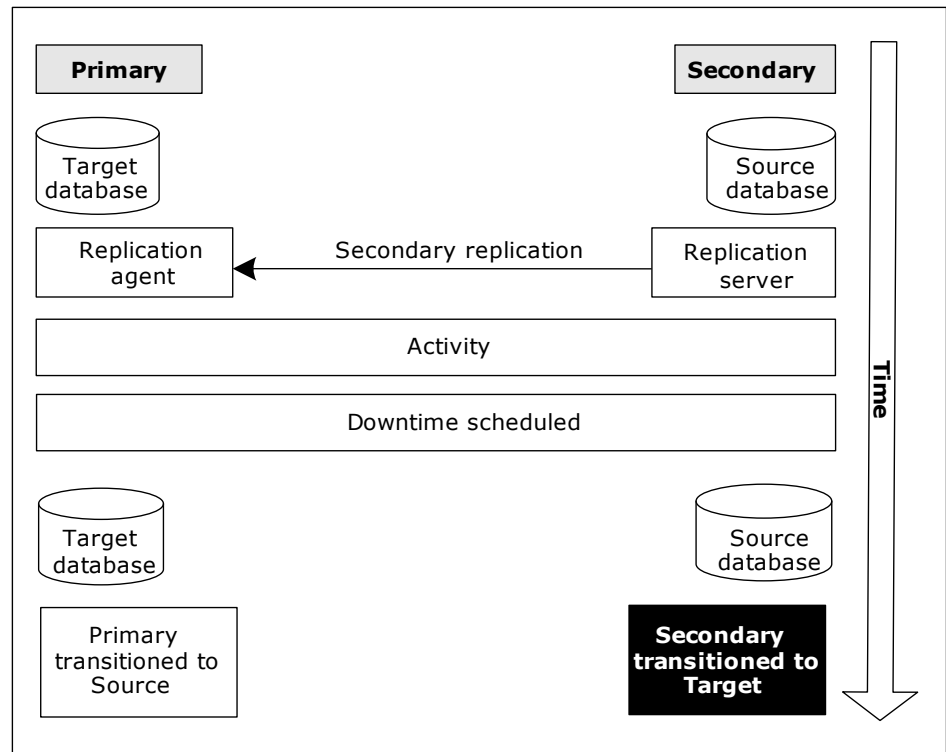
5. On the secondary machine, issue the following command to transition the secondary database to a target database:

```
dsrutil secondary -C transition
```

If the transition of the secondary database fails, you can start the primary database as the source database. This allows production activity again to proceed normally.

If a transition failure does occur and you do start the primary database, you must still complete the transition of this secondary database to the role of a target database. You can do this by again executing the command provided earlier in this step.

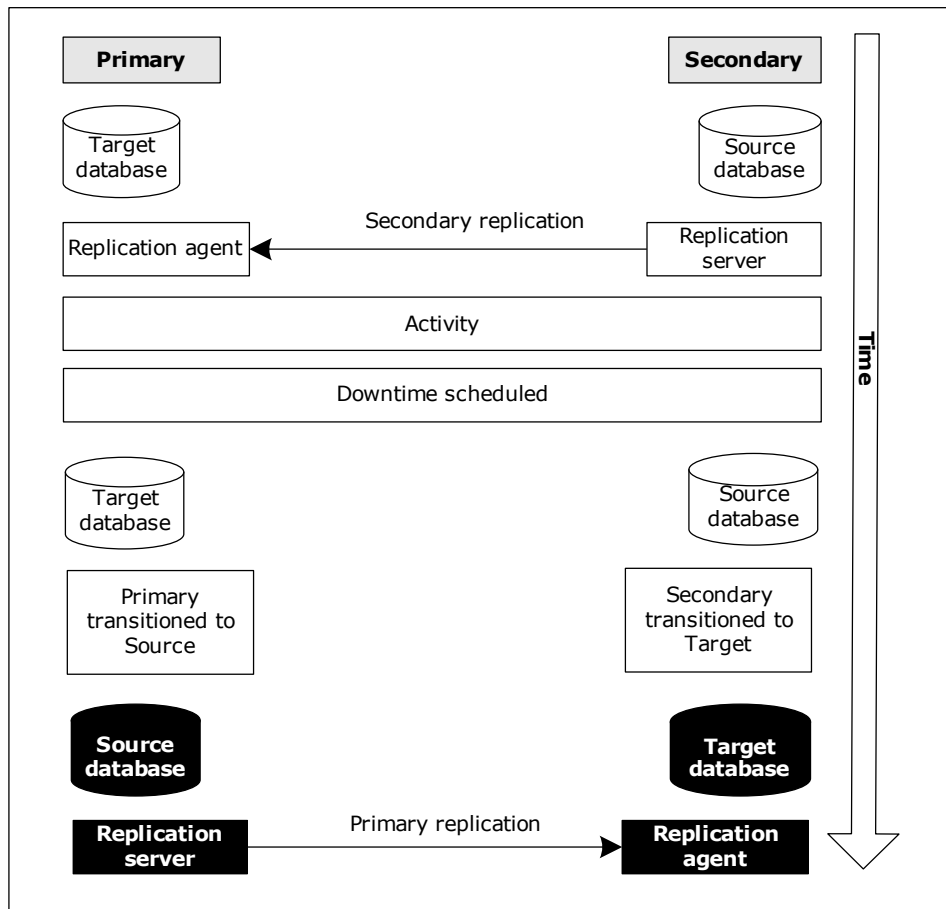
After the command completes normally, the transition of the secondary database to target is complete, as shown:



At this point the roles that both databases had during secondary replication have been reversed:

- The primary database is again the production database.
- The secondary database is again the replica.

Primary replication is again being performed as it was before the initial failure occurred. The Replication server is replicating all primary transactions to the secondary database, as shown in Figure 4-21.



**Figure 4-21: Primary replication activity resumes**

### **Advantages and disadvantages of performing failback with controlled transition**

There are both advantages and disadvantages to this method of failback processing.

The advantages are as follows:

- Downtime in the event of a failure is minimized.
- You maintain complete control over when and how the transition of both databases is performed.
- In the event of a primary transition failure, you can still use the secondary database for production activity.
- In the event of a secondary transition failure, you can use the primary database for production activity.

The disadvantages are as follows:

- You must perform additional steps to complete failback.
- You must have access to both the primary and secondary machines.
- Multiple database shutdowns are required.
- You must restart both databases after they successfully transition.

## Recovery from transition failures

In most cases, if a failure occurs during transition of either a source or target database, recovery will be performed and the databases will be returned to their original states (that is, before the transition was performed). In order to perform recovery, transition must perform a recovery backup immediately before the database is irreversibly changed. For example, during the transition of a target database, all live transactions must be rolled back. This is considered an irreversible change unless the database is backed up before the database change is performed.

The transition property `recovery-backup-arguments` allow you to specify the backup arguments used when transition performs its recovery backup. This property not only sets up the recovery backup arguments, it also instructs transition to perform a recovery backup. If this property is not specified, a transition will not perform a recovery backup. The recovery backup is performed immediately before the first irreversible operation is performed on the database.

Transition prepares as follows for a possible recovery before transitioning the database:

1. A new file named `database.transition.recovery` is created in the database directory. This file contains control and critical database information that must be saved in case a recovery is required.
2. If the recovery backup arguments are specified, transition performs a backup as follows:
  - If the database is a source, an offline backup of the database is performed before any additional operations are performed on the database. If the database is online at the time of transition, it is shut down before the backup is performed.
  - If the database is a target, an online backup is performed if the database is online; or an offline backup is performed if the database is offline. In both cases, the recovery backup is performed before any other transition operation is performed.

If an error occurs during the transition of a source database, the control and critical database information is overwritten with the information saved before transition began. The database should now be in the same state it was before transition was started, but it will be offline. Transition does not need to restore the recovery backup made.



If an error occurs during the transition of a target database, the following recovery actions are performed:

- If the database has been irreversibly changed and a recovery backup **was** performed:
  - a. The current database log file, `database.log`, is renamed `database.log.save`.
  - b. The database is deleted.
  - c. The database is restored from the recovery backup. Ensure that you have the latest database structure file, `database.st`, in the database directory.
  - d. The saved database log, `database.log.save`, is renamed `database.log`.
  - e. The control and critical database information is overwritten with the information saved before transition began.

The database should now be in the same state it was before transition was started, but it will be offline.

- If the database has been irreversibly changed and a recovery backup **was not** performed, the recovery process ends with an error indicating that it cannot continue with the database recovery.

## Transition logging

The transition of any database is a very complicated process. The transition process does provide status and progress messages; in addition, you can activate detailed transition logging.

Progression information similar to the following is presented while the transition is being performed:

```
Transitioning database /dir/srcdb
-----
13:28:51 Opening database                : Succeeded
13:28:51 Setting up transition           : Succeeded
13:28:53 Shutting down database          : Succeeded
13:29:25 Truncating BI                   : Succeeded
13:29:28 Starting database in Cur Role   : Succeeded
13:29:44 Synchronization in process     : Succeeded
13:29:49 Replication Server processing  : Succeeded
13:29:54 Preparing to transition Target DB : Succeeded
13:30:05 Shutting down database          : Succeeded
13:30:32 Target transition being performed : Succeeded
13:31:14 Switching AI Extents            : Succeeded
13:31:20 Switching database role         : Succeeded
13:31:20 Updating database master block  : Succeeded
13:31:20 Comparing databases             : Succeeded
13:31:20 Completing Target transition    : Succeeded
13:31:47 Backing up database            : Succeeded
13:31:52 Starting database in New Role   : Succeeded

The Transition of this database has completed normally.
```

This information is sent to stdout by the rprep1[.exe] program. The information shown in the previous sample is representative of a successful transition.

If transition does not complete normally, information similar to the following will be sent to stdout:

```
Transitioning database /dir/srcdb
-----
08:12:00 Opening database                : Succeeded
08:12:00 Setting up transition           : Succeeded
08:12:03 Shutting down database          : Succeeded
08:12:37 Truncating BI                   : Succeeded
08:12:40 Starting database in Cur Role   : Succeeded
08:12:56 Synchronization in process     : Succeeded
08:13:01 Replication Server processing  : Succeeded
08:13:06 Preparing to transition Target DB : Succeeded
08:13:16 Shutting down database          : Succeeded
08:13:44 Target transition being performed : Failed with -241

The transition of this database failed.  Attempting recovery.

08:14:22 Retrieving prior Recovery Control : Succeeded
08:14:22 Opening database                 : Succeeded
08:14:22 Restoring prior Recovery Control  : Succeeded
08:14:22 Updating Replication Control Info : Succeeded

The Transition of this database failed, but recovery was successful.
```

In addition to the information above, transition will output information to the database log file `database.lg` and, if instructed to do so, to a separate transition log. Transition will perform additional logging to a separate log file named `database.repl.util.lg` when the `-logging` argument is supplied to DSRUTIL as follows:

```
dsrutil database -C transition [failover] [-logging 2]
```

This log is formatted as the database log, but it contains much more diagnostic information. (The suggested minimum logging level for diagnosing the problem is 2.)

## Transition properties

You must group all transition properties into the transition section of the *db-name.rep1.properties* file. The properties file must be located in the same directory as the *db-name.db* file.

The following is a sample of properties for a primary database:

```
[server]
  control-agents=agent1
  database=ks1
  transition=manual
  transition-timeout=600
  agent-shutdown-action=recovery
  repl-keep-alive=0

[control-agent.agent1]
  name=agent1
  database=target
  host=localhost
  port=6931
  connect-timeout=120
  replication-method=async
  critical=0

[agent]
  name=agent1
  database=ks1
  listener-minport=4387
  listener-maxport=4500
  repl-keep-alive=0

[transition]
  database-role=reverse
  responsibility=primary
  auto-begin-ai=1
  auto-add-ai-areas=1
  transition-to-agents=agent1
  ai-structure-file=ks1.addai.st
  restart-after-transition=1
  source-startup-arguments=-DBService replserv
  target-startup-arguments=-S 6931 -DBService replagent
  recovery-backup-arguments=primary.recovery.bak
```

The following is a sample of properties for a secondary database:

```
#
[agent]
  name=agent1
  database=target
  listener-minport=4387
  listener-maxport=4500
  repl-keep-alive=0

[server]
  control-agents=agent1
  database=ks1
  transition>manual
  transition-timeout=600
  defer-agent-startup=60
  agent-shutdown-action=recovery
  repl-keep-alive=0

[control-agent.agent1]
  name=agent1
  database=target
  host=localhost
  port=6931
  connect-timeout=120
  replication-method=async
  critical=0

[transition]
  transition-to-agents=agent1
  responsibility=secondary
  database-role=reverse
  auto-begin-ai=1
  auto-add-ai-areas=1
  ai-structure-file=ks2.addai.st
  restart-after-transition=1
  source-startup-arguments=-DBService replserv
  target-startup-arguments=-S 6931 -DBService replagent
  backup-method=full-offline
  backup-arguments=ks2.sav
  incremental-backup-arguments=ks2.sav.inc
  recovery-backup-arguments=!secondary.recovery.bak
```

See [Table 4–3](#) for a description of each transition property. For details about setting the properties, see the [“Setting transition properties”](#) section on page 4–33.

## Setting transition properties

Once you make the decisions outlined in the [“Planning for transition”](#) section on page 4–5, you must set the transition properties that support those decisions, as listed in [Table 4–1](#). (See [Table 4–3](#) for a complete list of all transition properties.)

Table 4-1: Setting transition properties

(1 of 4)

To implement this transition planning decision . . .	Do this . . .
Which is the primary and which is the secondary database?	<p>Modify the <code>responsibility</code> property (currently used for informational purposes only) to identify the type of database. Possible values are:</p> <ul style="list-style-type: none"> <li>• <b>Primary</b> — This database is the primary database.</li> <li>• <b>Secondary</b> — This database is the secondary database.</li> </ul>
What type of transition will occur?	<p>Modify the <code>database-role</code> property to identify the new role of the database once it is transitioned. Possible values are:</p> <ul style="list-style-type: none"> <li>• <b>Reverse</b> — The role of the database is reversed: A source database becomes a target database, and a target database becomes a source database.</li> <li>• <b>Normal</b> — The role of the database becomes that of a normal database; the database is no longer enabled for replication once the transition is performed. This is the default value.</li> </ul>
What type of after-imaging operations should transition perform?	<p>Modify the after-imaging properties <code>auto-begin-ai</code>, <code>auto-add-ai-areas</code>, and <code>ai-structure-file</code> to set AI behavior.</p> <p>Specify either of these values for the <code>auto-begin-ai</code> property:</p> <ul style="list-style-type: none"> <li>• <b>0</b> — Do not begin AI automatically after a target-to-source transition.</li> <li>• <b>1</b> — Begin AI automatically after a target-to-source transition.</li> </ul> <p>Specify either of these values for the <code>auto-add-ai-areas</code> property:</p> <ul style="list-style-type: none"> <li>• <b>0</b> — Do not add AI areas to the database automatically (when a database is transitioned to a source database or there are currently no AI areas for the database).</li> <li>• <b>1</b> — Add AI areas to the database automatically by using the structure file specified in the <code>ai-structure-file</code> property (when a database is transitioned to a source database or there are currently no AI areas for the database).</li> </ul> <p>Specify the name of the structure file (which contains the list of AI areas to add) in the <code>ai-structure-file</code> property.</p>

Table 4–1: Setting transition properties

(2 of 4)

To implement this transition planning decision . . .	Do this . . .
<p>What types of backup do you want to perform during transition?</p>	<p>Modify the backup properties <code>backup-method</code>, <code>backup-arguments</code>, and <code>incremental-backup-arguments</code> to set the type of backup you want to perform during transition and before AI is enabled.</p> <p>Specify one of these values for the <code>backup-method</code> property:</p> <ul style="list-style-type: none"> <li> <b>mark</b> — Mark the database as backed up by using the following command:           <pre>rfutil db-name -C mark backedup</pre> <p>Marking the database as backed up does not allow future AI extents to be used when recovering from a disaster.</p> </li> <li> <b>full-offline</b> — Back up the database offline by using the OpenEdge PROBKUP utility.           <p>The backup is performed in two steps. The first backup is a full backup, which is performed before AI is enabled for the database. The second backup is an incremental backup, which is performed after AI is enabled and after the role of the database is changed.</p> </li> <li> <b>full-online</b> — Back up the database online after the database is restarted, which occurs after the database has been transitioned.           <p>Specify the <code>backup-arguments</code> property for the arguments required for the full online and offline backups performed for the database. Keep the following in mind:</p> <ul style="list-style-type: none"> <li>You must specify, at a minimum, the target file or device in these arguments for both online and offline backups.</li> <li>To avoid overwriting a backup, do not use the same target file or device for both the backup and the incremental backup.</li> <li>Do not use backup validation parameters (such as <code>-vp</code> and <code>-vf</code>).</li> <li>Begin the arguments with <code>device-name</code>.</li> </ul> <p>Specify the <code>incremental-backup-arguments</code> property for the arguments required for the offline incremental backup performed after AI is enabled and the database's role is reversed. Begin the arguments with <code>device-name</code>.</p> </li> </ul>

**Table 4-1: Setting transition properties***(3 of 4)*

To implement this transition planning decision . . .	Do this . . .
Do you want the database to restart automatically after transition?	<p>Modify the following automatic transition properties to specify whether the database should be automatically restarted after transition:</p> <ul style="list-style-type: none"> <li>• <code>restart-after-transition</code></li> <li>• <code>source-startup-arguments</code>, <code>target-startup-arguments</code>, or <code>normal-startup-arguments</code></li> </ul> <p>Specify either of these values for the <code>restart-after-transition</code> property:</p> <ul style="list-style-type: none"> <li>• <b>0</b> — Do not automatically restart the database after transition is performed.</li> <li>• <b>1</b> — Automatically restart the database after transition is performed. When you specify this value, you must also supply the <code>*-startup-arguments</code> properties, or the database startup will fail.</li> </ul> <p>Specify the <code>*-startup-arguments</code> property or properties:</p> <ul style="list-style-type: none"> <li>• If the database is transitioned to a normal database, specify the <code>normal-startup-arguments</code> property. These arguments are used when the database is started. The arguments will be appended to the PROSERVE command. In most cases, the only argument specified here should be <code>-pf</code> followed by a parameter file name, as shown here:  <code>-pf db-name.normal.pf</code></li> <li>• If the database is transitioned to a source database, you must specify the <code>source-startup-arguments</code> property. The arguments are appended to the PROSERVE command and are used when the database is started. In most cases, the only argument specified should be <code>-pf</code> followed by a parameter file name, as shown here:  <code>-pf db-name.source.pf</code></li> </ul>



**Table 4–1: Setting transition properties**

(4 of 4)

To implement this transition planning decision . . .	Do this . . .
<p>Do you want the database to restart automatically after transition?</p> <p><i>(continued)</i></p>	<p>Because the database is a source, you must also specify the following argument as an indication to the broker to start the Replication server:</p> <pre>-DBService replserv</pre> <ul style="list-style-type: none"> <li>• If the database is transitioned to a target database, you must specify the <code>target-startup-arguments</code> property. The arguments are appended to the <code>PROSERVE</code> command and are used when the database is started. In most cases, the only argument specified here should be <code>-pf</code> followed by a parameter file name, as shown here:</li> </ul> <pre>-pf db-name.target.pf</pre> <p>Because the database is a target, you must also specify the following arguments as an indication to the broker to start the Replication agent and listen on the TCP/IP port as specified:</p> <pre>-DBService replagent -S port-number or service-name</pre> <ul style="list-style-type: none"> <li>• If the database role is reversed, you must specify the <code>source-startup-arguments</code> property and the <code>target-startup-arguments</code> property.</li> </ul> <p>For additional details about the <code>source-</code>, <code>target-</code>, or <code>normal-startup-arguments</code>, see the <a href="#">“Sample startup parameter file”</a> section on page 4–38.</p>
<p>Will transition automatically attempt to recover in the event of a failure?</p>	<p>Modify the <code>recovery-backup-arguments</code> property, which determines whether transition automatically attempts a recovery.</p> <p>Specify the device name, followed by any other backup arguments.</p>

## Sample startup parameter file

Transition uses the parameter file after a database is transitioned if transition is instructed to restart the database when it completes. Transition can potentially use a startup parameter file when:

- The database is restarted as a source.
- The database is restarted as a target.
- The database is restarted as a normal database. (This file is optional.)

### Source database startup parameter file

If you are transitioning the database to a source database, specify in the `source-startup-arguments` property the name of the `.pf` file that contains this data:

```
-DBService replserv [This line is required]
-B 10000 -n 1000
```

See [Table 4–3](#) for details about the `source-startup-arguments` property.

### Target database startup parameter file

If you are transitioning the database to a target database, specify in the `target-startup-arguments` property the name of the `.pf` file that contains the following data:

```
-DBService replagent -S 6931 [This line is required]
-B 10000 -n 1000
```

See [Table 4–3](#) for details about the `target-startup-arguments` property.

### Normal database startup parameter file

If you are transitioning the database to a normal database, specify in the `normal-startup-arguments` property the name of the `.pf` file that contains this data, as shown:

```
-B 10000 -n 1000
```

See [Table 4–3](#) for details about the `normal-startup-arguments` property.

## Reference

The following sections provide reference information related to OpenEdge Replication transition and transition properties.

### Transition command actions

Table 4–2 describes the action taken by the `dsrutil db-name -C transition` command when it is executed for databases in varying states.

**Table 4–2: Transition command actions**

Database type	Database state	Online or offline	Description of transition performed
Source	Database is newly enabled	Offline	An error occurs because the database is newly enabled.
Source	Database is newly enabled	Online	An error occurs because the database is online and has never been replicated.
Source	Database has been replicated	Offline	The database is transitioned as instructed in the <code>repl.properties</code> file.
Source	Database has been replicated	Online	In order to transition a source database in this state, the Replication server must be communicating with the Replication agent to convey a transition and failover request to the Replication agent.
Target	Database has been replicated	Offline	In order to transition a target database in this state, the database cannot be newly transitioned.
Target	Database has been replicated	Online	<p>In order to transition a target database in this state, the following conditions must be met:</p> <ul style="list-style-type: none"> <li>• The Replication agent cannot be communicating with the Replication server.</li> <li>• The Replication agent must be in pretransition.</li> <li>• All source AI extents must be applied to the target database by using <code>dsrutil db-name -C applyExtent</code>.</li> </ul> <p>You must explicitly perform this step, possibly multiple times depending on the number of source AI extents.</p>
Normal	–	Online Offline	An error occurs because the database is not enabled for replication.

## Transition properties summary

Table 4–3 provides a summary of the transition properties. The table lists each property in the `repl.properties` file, identifies the property type, and provides a property description.

**Table 4–3: Transition properties**

(1 of 3)

Property name	Type and length	Description
database-role	character[15]	<p>The new role of the database once it is transitioned. The possible values for this property are as follows:</p> <ul style="list-style-type: none"> <li><b>reverse</b> — The role of the database is reversed: A source database becomes a target database, and a target database becomes a source database.</li> <li><b>normal</b> — The role of the database becomes that of a normal database; the database is no longer enabled for replication once the transition is performed. This is the default value.</li> </ul>
responsibility	character[15]	<p>This property must contain one of the following values:</p> <ul style="list-style-type: none"> <li><b>primary</b> — This database is the primary database.</li> <li><b>secondary</b> — This database is the secondary database.</li> </ul> <p>This property is currently informational.</p>
restart-after-transition	integer	<p>The database can be automatically restarted after transition is performed.</p> <p>Valid values for the property are <b>0</b> and <b>1</b>. When the property is set to <b>1</b>, the following <code>*-startup-arguments</code> properties must be supplied, or the database startup will fail:</p> <ul style="list-style-type: none"> <li>If the database role is normal, you must specify the <code>normal-startup-arguments</code>.</li> <li>If the database role is reversed, you must specify the <code>source-startup-arguments</code> and the <code>target-startup-arguments</code>.</li> </ul>
source-startup-arguments	character[256]	<p>If the database is transitioned to a source database, these arguments are used when the database is started. The arguments will be appended to the <code>PROSERVE</code> command (used to start the database).</p> <p>In most cases, the only argument specified here should be <code>-pf</code> followed by a parameter file name. For example:</p> <pre>-pf db-name.source.pf</pre> <p>Because the database is a source, you must also specify the following argument, as an indication to the broker to start the replication server:</p> <pre>-DBService replserv</pre>

Table 4–3: Transition properties

(2 of 3)

Property name	Type and length	Description
target-startup-arguments	character[256]	<p>If the database is transitioned to a target database, these arguments are used when the database is started. The arguments will be appended to the PROSERVE command.</p> <p>In most cases, the only argument specified here should be <code>-pf</code> followed by a parameter file name. For example:</p> <pre>-pf db-name.target.pf</pre> <p>Because the database is a target, you must also specify the following arguments, as an indication to the broker to start the replication agent and to listen on the TCP/IP port specified with <code>-S port-number</code> or <code>service-name</code>:</p> <pre>-DBService replagent -S port-number or service-name</pre>
normal-startup-arguments	character[256]	<p>If the database is transitioned to a normal database, these arguments are used when the database is started. The arguments will be appended to the PROSERVE command.</p> <p>In most cases, the only argument specified here should be <code>-pf</code> followed by a parameter file name. For example:</p> <pre>-pf db-name.normal.pf</pre>
auto-begin-ai	integer	Specify <b>1</b> to automatically begin AI after a target-to-source database transition. After-imaging can be started for a database that has AI areas.
transition-to-agents	character[256]	<p>Transition to the first agent in the list when a failure occurs. If the first agent is not available, transition to the second agent in the list.</p> <p>A valid value for this property is any configured agent name; separate the names by a comma if you are listing more than one. For example:</p> <pre>agent1,agent2</pre>
auto-add-ai-areas	integer	<p>This property instructs the transition process to automatically add AI areas to the database if:</p> <ul style="list-style-type: none"> <li>It is transitioned to a source database.</li> <li>There are currently no AI areas for the database.</li> </ul> <p>If the value specified for the property is <b>0</b>, AI areas are not added to the database. If the value specified is <b>1</b>, AI areas are automatically added to the database by using the structure file specified in the <code>ai-structure-file</code> property.</p>
ai-structure-file	character[256]	Name of the structure file that contains the list of AI areas to add.

Table 4-3: Transition properties

(3 of 3)

Property name	Type and length	Description
backup-method	character[15]	<p>Specifies the backup method performed before AI is enabled. The following values are valid:</p> <ul style="list-style-type: none"> <li> <b>mark</b> — Mark the database backed up by using the following command:  <code>rfutil db-name -C mark backedup</code>  Marking the database as backed up does not allow future AI extents to be used when recovering from a disaster. </li> <li> <b>full-offline</b> — The database will be backed up offline by using the PROBKUP utility.   The backup is performed in two steps: The first backup is a full backup, which is performed before AI is enabled for the database. The second backup is an incremental backup, which is performed after AI is enabled and after the role of the database is changed. </li> <li> <b>full-online</b> — An online backup is performed after the database is restarted, which occurs after the database has been transitioned. </li> </ul>
backup-arguments	character[256]	<p>The arguments required for the full online and offline backups that are performed for the database.</p> <p>At a minimum, the target file or device must be specified in these arguments for both online and offline backups. To avoid overwriting a backup, do not use the same target file or device for both the backup and the incremental backup.</p> <p>Do not use backup validation parameters (such as <code>-vp</code> and <code>-vf</code>).</p> <p>The arguments should begin with <i>device-name</i>.</p>
incremental-backup-arguments	character[256]	<p>The arguments required for the offline incremental backup performed after AI is enabled and the database's role is reversed.</p> <p>The arguments should begin with <i>device-name</i>.</p>
recovery-backup-arguments	character[256]	<p>The arguments required for the full online and offline backups that are performed for the database.</p> <p>At a minimum, the target file or device must be specified in these arguments for both online and offline backups. To avoid overwriting a backup, do not use the same target file or device for both the backup and the incremental backup.</p> <p>Do not use backup validation parameters (such as <code>-vp</code> and <code>-vf</code>).</p> <p>The arguments should begin with <i>device-name</i>.</p>

---

## Reference

---

This chapter provides OpenEdge Replication reference information, as described in the following sections:

- [OpenEdge Replication property files](#)
- [OpenEdge Replication properties](#)
- [DSRUTIL utility](#)
- [DSRUTIL applyextent qualifier](#)
- [DSRUTIL canceldefer server qualifier](#)
- [DSRUTIL connectagent database qualifier](#)
- [DSRUTIL disablesitereplication qualifier](#)
- [DSRUTIL monitor qualifier](#)
- [DSRUTIL recovery qualifier](#)
- [DSRUTIL relwaits qualifier](#)
- [DSRUTIL restart server qualifier](#)
- [DSRUTIL startagent database qualifier](#)
- [DSRUTIL status qualifier](#)
- [DSRUTIL terminate qualifier](#)
- [DSRUTIL transition qualifier](#)
- [DSRUTIL triggertransition qualifier](#)

- [OpenEdge Replication DSRUTIL MONITOR](#)
- [Virtual system tables for OpenEdge Replication](#)
- [Utilities and OpenEdge Replication](#)
- [Starting OpenEdge Replication with the Progress Explorer and dbman](#)
- [OpenEdge Replication and database management systems](#)



## OpenEdge Replication property files

The properties files used by OpenEdge Replication control how OpenEdge Replication operates. There is a properties file for the source database and a properties file for each target database. Each properties file is named *db-name.rep1.properties*, where *db-name* is the name of the source or target database. Each properties file must be stored in the directory where the database resides.

If you use the same name for your source and target database, restoring the source database to the target will result in the target properties file being overwritten. To prevent lost information, you can combine your source and target properties file into one file as shown in the [“Sample OpenEdge Replication combined properties file”](#) section on page 5–4.

The following sections show the sample properties files that are shipped with the product for the source and target databases as well as a definition of the property file directives.

### Sample OpenEdge Replication source properties file

The following is an excerpt from the sample source properties file that is shipped with the OpenEdge Replication product:

```
[server]
  control-agents=agent1
  database=source
  transition=manual
  transition-timeout=600

[control-agent.agent1]
  name=agent1
  database=target
  host=your host
  port=your port
  connect-timeout=120
  replication-method=async
  critical=0

[transition]
  database-role=reverse
  auto-begin-ai=0
  auto-add-ai-areas=0
  ai-structure-file={!{value-of:server.database}.addai.st
```

The properties files are located in `<OpenEdgeReplication-install-dir>\properties`.

For information about these properties and their values, see the [“Server properties”](#) section on page 5–5 and the [“Control agent properties”](#) section on page 5–7.

## Sample OpenEdge Replication target properties file

The following is an excerpt from the sample target properties file that is shipped with the OpenEdge Replication product:

```
[agent]
  name=agent1
  database=target
  listener-minport=4387
  listener-maxport=4500

[transition]
  database-role=normal
  auto-begin-ai=0
  auto-add-ai-areas=0
  ai-structure-file={!{value-of:agent.database}.addai.st
```

For more information on these properties and their values, see the “[Agent properties](#)” section on page 5–9.

## Sample OpenEdge Replication combined properties file

The following is a sample combined properties file; you can use the combined file if your source and target database names are the same. This file must reside in both the source database directory and the target database directory:

```
[server]
  control-agents=agent1
  database=source
  transition=manual
  transition-timeout=600
[control-agent.agent1]
  name=agent1
  database=target
  host=your host
  port=your port
  maximum-message=32
  connect-timeout=120
  replication-method=async
  critical=0
[agent]
  name=agent1
  database=target
  listener-minport=4387
  listener-maxport=4500
```

# OpenEdge Replication properties

This section describes the values that you can set in the source and target properties files.

## Server properties

Table 5–1 is a list of server properties and their values.

Server properties are found after the [server] directive in the OpenEdge Replication source properties file. The server properties file must be stored in the directory where the source database resides.

Property names and values are separated by an equal sign. For example: `transition=auto`.

**Table 5–1: Server properties**

(1 of 3)

Property name	Value	Description
control-agents	<i>agent_name</i> [, <i>agent_name</i> ]	Specifies a comma-separated list of OpenEdge Replication agent names. The comma must be followed by a space. The agent name must match the control agent name specified in the [control-agent] properties section head.  ALL is not allowed as an agent name. Each agent must have a unique name.
database	<i>db_name</i>	Specifies the source database name.
defer-agent-startup	<i>minutes</i>	Specifies for how long, in minutes, the server attempts to connect to an agent if the first connection attempt is unsuccessful.  <i>minutes</i> is a value $\geq 0$ and $\leq 10080$ .
maximum-polling-delay	<i>milliseconds</i>	Specifies the maximum value, in milliseconds, for a polling delay.  By default, the polling delay starts at 5 ms and automatically increases during periods of inactivity to a maximum of 500 ms.  <i>milliseconds</i> is a value $>500$ and $<1000$ .
minimum-polling-delay	<i>milliseconds</i>	Specifies the minimum value, in milliseconds, for a polling delay.  By default, the polling delay starts at 5 ms and automatically increases during periods of inactivity to a maximum of 500 ms.  <i>milliseconds</i> is a value $\geq 1$ and $\leq 10$ .

Table 5–1: Server properties

(2 of 3)

Property name	Value	Description
repl-Keep-Alive	<i>seconds</i>	<p>Specifies a time-out period for communications between a server and its agents. If a connection between the server and agent is not verified before the time-out expires, failure recovery begins.</p> <p>By default, this property is enabled and has a value of 300 seconds. The minimum value is 60 seconds; there is no maximum value.</p>
schema-Lock-Action	wait   force	<p>Specifies the action an agent takes if an exclusive schema lock is not granted.</p> <p>Possible actions are:</p> <ul style="list-style-type: none"> <li>• <b>wait</b> — The agent waits until the exclusive schema lock is granted. The server blocks until the exclusive schema lock is granted.</li> <li>• <b>force</b> — The agent attempts to acquire the exclusive schema lock five times. If the fifth attempt fails, the agent disconnects all users from the target and makes another attempt. If the last attempt fails, the server and all agents terminate. When schema update activity completes, the server and target can be restarted.</li> </ul> <p><b>Note:</b> Values are case-sensitive. Use <code>wait</code> or <code>force</code>, but not <code>Wait</code> or <code>Force</code>.</p>
agent-shutdown-action	recovery   normal	<p>Specifies the action an agent takes during a shutdown when the replication server ends.</p> <p>Possible actions are:</p> <ul style="list-style-type: none"> <li>• <b>recovery</b> — The agent will remain active but in a standby state waiting for the replication server to reconnect.</li> <li>• <b>normal</b> — The agent will terminate; the target database will stay up.</li> </ul>

Table 5–1: Server properties

(3 of 3)

Property name	Value	Description
transition	auto   manual	<p>Specifies how to transition the target database to a normal database.</p> <p>If a synchronous agent or an asynchronous critical agent cannot reconnect with the source database within the time specified in the <code>transition-timeout</code> property, the following will occur:</p> <ul style="list-style-type: none"> <li>• <b>auto</b> —The agent automatically transitions the target database.</li> <li>• <b>manual</b> —The agent listens for the server until it reconnects with the source database or the DSRUTIL command is executed. The DSRUTIL <code>db-name -C transition</code> command transitions the target database.</li> </ul> <p><b>Note:</b> Values are case-sensitive. Use <code>auto</code> or <code>manual</code>, but not <code>Auto</code> or <code>Manual</code>.</p>
transition-timeout	seconds	<p>Specifies the number of seconds the target database waits before it performs auto-transition. This property is ignored when <code>transition&gt;manual</code>.</p> <p>The value is incremented by the sum of the <code>connect-timeout</code> for all configured agents.</p>

## Control agent properties

Table 5–2 is a list of control agent properties and their values. Control agent properties define to the server which agents it can contact, where it can contact them, and how the agents should perform.

The properties are specified after the `[control-agent.name]` directive in the source properties file. The value of `name` must exactly match one of the names specified for the control-agents property under the `[server]` directive. If you run two agents, you need a `[control.agent.name]` directive for each of the agents.

Property names and values are separated by an equal sign. For example:

```
name=agent1
```

Table 5–2: Control agent properties

Property name	Value	Description
connect-timeout	<i>seconds</i>	<p>Specifies for how long, in seconds, the server will attempt to connect to its configured agents. This property is also used by the server while reconnecting to the agent after communication has been lost.</p> <p>Specifies how many seconds the Replication agent will wait for connection from the OpenEdge Replication server before the Replication agent shuts itself down.</p> <p><i>seconds</i> is an integer <math>\geq 120</math> and <math>\leq 86,400</math>.</p> <p>Using this property means you do not have to do a forced shutdown on your target database. If the OpenEdge Replication agent does not receive a connection attempt from the OpenEdge Replication server before the number of seconds specified has elapsed, the OpenEdge Replication agent will terminate and allow some limited system-level target database connections.</p>
critical	0   1	<p>Specifies whether the agent is critical:</p> <ul style="list-style-type: none"> <li>• 1 is critical.</li> <li>• 0 is the default value of noncritical.</li> </ul> <p>A critical agent is an asynchronous agent for the target database that can become the source database if the source database becomes unavailable.</p>
database	<i>db-name</i>	Specifies the source database name.
host	<i>IP-address</i>   <i>hostname</i>	Specifies to the server which host the agent will start on.
ipver	<i>ipv4</i>   <i>ipv6</i>	Specifies whether to use TCP/IP IPv4 or TCP/IP IPv6 between the Replication server and the Replication agent. If the target database broker is using TCP/IP IPv6, OpenEdge Replication must also use IPv6.
name	<i>agent-name</i>	<p>Specifies the OpenEdge Replication agent name. By convention, the <i>agent-name</i> should match the name specified in the target database properties file. This directive is used only by Progress Explorer.</p> <p>ALL is not allowed as an agent name. Each agent must have a unique name.</p>
port	<i>service-name</i>   <i>port-num</i>	Specifies which port the server should use to connect to the target database. The port number specified must be the same port specified with the -S parameter when the target database broker was started.
replication-method	async   sync	<p>Specifies the type of replication, either asynchronous (async) or synchronous (sync).</p> <p><b>Note:</b> Values are case-sensitive. Use async or sync, but not Async or Sync.</p>

## Agent properties

Table 5–3 is a list of agent properties and their values. Agent properties define the configuration for the local agent running for the target database.

Agent properties are found after the [agent] directive in the agent properties file. The properties file must be stored in the directory where the target database resides.

Property names and values are separated by an equal sign. For example: name=agent1.

**Table 5–3: Agent properties**

(1 of 2)

Property name	Value	Description
connect-timeout	<i>seconds</i>	Specifies how many seconds the OpenEdge Replication agent will wait for connection from the OpenEdge Replication server before the replication agent shuts itself down.  <i>seconds</i> is an integer $\geq 120$ and $\leq 86,400$ .  Using this property means you do not have to do a forced shutdown on your target database. If the OpenEdge Replication agent does not receive a connection attempt from the OpenEdge Replication server before the number of seconds specified have elapsed, the OpenEdge Replication agent will terminate and allow some limited system-level target database connections.
database	<i>db_name</i>	Specifies the source database name.
host	<i>IP-address</i>   <i>hostname</i>	Specifies to the server which host the agent will start on.
ipver	<i>ipv4</i>   <i>ipv6</i>	Specifies whether to use TCP/IP IPv4 or TCP/IP IPv6 between the Replication server and the Replication agent. If the target database broker is using TCP/IP IPv6, OpenEdge Replication must also use IPv6.
listener-maxport	<i>port-number</i>	Specifies the maximum TCP port number for the agent.  <i>port-number</i> must be $> \text{listener-minport} + 1$ and $<$ the maximum allowable port number on the system.  For UNIX the maximum port number is 65534. For Windows the maximum port number is 32765.
listener-minport	<i>port-number</i>	Specifies the minimum TCP port number. The agent selects a port in a range between the values specified by <i>listener-minport</i> and <i>listener-maxport</i> .  <i>port-number</i> must be $> 1024$ and $< \text{listener-maxport}$ .
maximum-polling-delay	<i>milliseconds</i>	Specifies the maximum value, in milliseconds, for a polling delay.  By default, the polling delay starts at 5 ms and automatically increases during periods of inactivity to a maximum of 500 ms.  <i>milliseconds</i> is a value $> 500$ and $< 1000$ .

Table 5–3: Agent properties

(2 of 2)

Property name	Value	Description
minimum-polling-delay	<i>milliseconds</i>	<p>Specifies the minimum value, in milliseconds, for a polling delay.</p> <p>By default, the polling delay starts at 5 ms and automatically increases during periods of inactivity to a maximum of 500 ms.</p> <p><i>milliseconds</i> is a value <math>\geq 1</math> and <math>\leq 10</math>.</p>
name	<i>agent_name</i>	<p>Specifies the OpenEdge Replication agent name. By convention, the <i>agent-name</i> should match the name specified in the source database properties file. This directive is used only by Progress Explorer.</p> <p>ALL is not allowed as an agent name. Each agent must have a unique name.</p>



## DSRUTIL utility

Once you have initially set up, enabled, and started OpenEdge Replication, you can use the DSRUTIL utility to perform specific OpenEdge Replication server, OpenEdge Replication agent, source database, and target database requests.

### Syntax

Operating system	Syntax
UNIX Windows	DSRUTIL <i>db-name</i> -C <i>ACTION</i> [Server   Agent] [ <i>name</i>   ALL]

*db-name*

Database is the name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

-C *action*

The -C qualifier is used to specify the action to be performed on the database. You can supply the qualifiers described in [Table 5–4](#).

Server | Agent

The action will be performed on the OpenEdge Replication server or the OpenEdge Replication agent.

*name* | ALL

The action will be performed on one OpenEdge Replication agent (*name*) or on all OpenEdge Replication agents (ALL). Each OpenEdge Replication server maintains a list of named OpenEdge Replication agents that it is communicating with. The agent name must be valid.

**Table 5–4: DSRUTIL utility qualifiers**

(1 of 2)

Qualifier	Description
applyextent	Allows you to apply AI extents generated by the source directly to the target database.
canceldefer	Instructs the OpenEdge Replication server to stop attempting to reconnect.
connectagent	Instructs the OpenEdge Replication server to start one or both of its configured agents.
disablesitereplication	Allows you to disable OpenEdge Replication while the source database is online.
monitor	Displays a PROMON-type series of screens that show the current state of replication.
recovery	Displays the replication recovery information.

**Table 5–4: DSRUTIL utility qualifiers***(2 of 2)*

Qualifier	Description
relwaits	Frees up any pending waits that might be outstanding so that database activity can continue.
restart	Restarts the OpenEdge Replication server.
startagent	See connectagent.
terminate	Terminates the currently running OpenEdge Replication server or agent.
transition	Instructs an OpenEdge Replication agent to transition a replication-enabled database.
triggertransition	Forces the target database to go into a pre-transition state.

You can use the following command actions and modifiers with the DSRUTIL utility.

## DSRUTIL applyextent qualifier

Allows you to apply AI extents generated by the source directly to the target database. This is useful when there is a source failure and there are AI blocks in transit between the server and agents.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil <i>db-name</i> -C applyextent <i>extent-name</i></code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

*extent-name*

This extent is provided by the recovery qualifier.

The following requirements exist for using this feature:

- The agent must be in pre-transition state.
- The transition property must be set to `manual`.
- The source must save AI extents to remote storage that is accessible to the target's agent.

You can determine which extent to apply by using the following command:

```
dsrutil db-name -C recovery Agent
```

The information shown here must be used to correctly apply source after-image extents to the target database in the event of a source database failure when the following command is executed:

```
dsrutil db-name -C applyextent extent-name
```

In order to determine the after-image extent name using the After-Image File Number supplied, you must do one of the following:

- If the source database is available, use the following command to generate a list of after-image extents for the source database:

```
rfutil source-db-name -C aimage list
```

Executing this command produces the following output:

```
Extent: 1
Status: Full
  Type: Variable Length
  Path: /vobs_rep1/solaris/bin/ks1.a1
  Size: 120
  Used: 1
  Start: Wed Oct 26 08:32:12 2005
  Seqno: 1

Extent: 2
Status: Full
  Type: Variable Length
  Path: /vobs_rep1/solaris/bin/ks1.a2
  Size: 4728
  Used: 4534
  Start: Wed Oct 26 08:32:14 2005
  Seqno: 2

Extent: 3
Status: Busy
  Type: Variable Length
  Path: /vobs_rep1/solaris/bin/ks1.a3
  Size: 4728
  Used: 4456
  Start: Wed Oct 26 08:33:29 2005
  Seqno: 3

Extent: 4
Status: Empty
  Type: Variable Length
  Path: /vobs_rep1/solaris/bin/ks1.a4
  Size: 120
  Used: 0
  Start: N/A
  Seqno: 0
```

Match the Seqno from this output to the After Image File Number provided by the DSRUTIL recovery output. Apply all BUSY and FULL extents beginning with this extent.

- If the source database is unavailable but its after-image extents are available on a SAN or NAS device that the target machine has access to, you must determine the first after-image extent to apply.

Change to the directory where the source after-image extents are stored, and then execute the following command:

```
rfutil db-name -C aimage scan -a after-image-extent-name
```

The command produces the following output:

```
After-image dates for this after-image file: (1633)
  Last AIMAGE BEGIN Wed Oct 26 08:32:12 2005 (1640)
  Last AIMAGE NEW Wed Oct 26 08:33:29 2005 (1641)
  This is aimage file number 3 since the last AIMAGE BEGIN. (1642)
  This file was last opened for output on Wed Oct 26 08:33:29 2005.
(1643)

41706 notes were processed. (1634)
0 in-flight transactions. (3785)
614 transactions were started. (1635)
614 transactions were completed. (11138) At the end of the .ai file,
0 transactions were still active. (1636)
```

Match the aimage file number *n* from this output to the After-Image File Number provided by the DSRUTIL recovery output. Apply all BUSY and FULL extents beginning with this extent.

## DSRUTIL canceldefer server qualifier

Instructs the OpenEdge Replication server to stop attempting to reconnect. This action is only applicable if the `defer-agent-startup` property was set to a valid non-zero time that has not yet expired.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C canceldefer server</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

The advantage of using this command is that if you have connected to one of your required agents and you do not want to wait for the second, non-critical agent to connect, normal replication processing will begin and connection retries will stop.

The OpenEdge Replication agent that is connected and the OpenEdge Replication server will go through startup synchronization. Once synchronization is complete, normal replication activity will continue.

If you start the Replication server by using the `defer-agent-startup` parameter, you can still issue the `canceldefer` command after the Replication server has started the agent. The Replication agent status or log file contains a confirmation that the agent started successfully.

## DSRUTIL connectagent database qualifier

The OpenEdge Replication server starts one or both of its configured agents. The advantage to using this command is that you do not have to restart your server.

This qualifier is a synonym for the startagent database qualifier.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C connectagent database [name   ALL]</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

### Notes

The database must be a valid source database that is online.

The OpenEdge Replication server must be running.

If the name of the agent or names of all of the agents specified are currently active, an error will be returned.

If the restart fails, an error will be returned.

Success will be returned on successful completion.

## DSRUTIL disablesitereplication qualifier

Allows you to disable OpenEdge Replication while the source database is online. Before OpenEdge Replication is disabled, the server is terminated.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C disablesitereplication [Source   Target]</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

When the database is up and running but the OpenEdge Replication server and agent are not, for example after a crash or termination, this command makes the specified database a normal OpenEdge database. This command works only when the database is up. If it is a target database, this command does not disable ERO mode. To change this, you must shut down and restart the database.

If the agent is still running, this command will terminate the agent before disabling OpenEdge Replication.



## DSRUTIL monitor qualifier

Displays a PROMON-type series of screens that show the current state of replication. This is a useful command to see what activities are taking place while OpenEdge Replication is running.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C monitor</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

## DSRUTIL recovery qualifier

Displays the replication recovery information.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C recovery [server   agent]</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

The replication recovery information for the server looks similar to the following:

Replication version:	4.0
Date created:	Wed Oct 26 08:32:58 2005
Date last written:	Wed Oct 26 08:33:46 2005
Replication server information:	
Number of agents:	1
Number of unused agents:	1
Last modified:	Wed Oct 26 08:32:08 2005
Master block update count:	21
Remote Agent information:	
Remote Agent 1	
Identification:	1
Agent name:	agent1
Last AI block acknowledged:	area: 14, seq: 3, loc: 4259840, offset: 3189
Last modified:	Wed Oct 26 08:33:29 2005
Last AI block ACK time:	Wed Oct 26 08:33:46 2005
Remote agent host:	localhost
Remote agent database:	ks2

The replication recovery information for the agent looks similar to the following:

```
Replication version:          4.0
  Date created:              Wed Oct 26 08:32:57 2005
  Date last written:        Wed Oct 26 08:34:36 2005

Replication local agent information:
  Last Block:                Complete
  Last block received location: area: 14, seq: 3, loc: 4554752, offset: 0
  Last block processed location: area: 0, seq: 0, loc: 0, offset: 0
  Last block ACKed location:  area: 14, seq: 3, loc: 4259840, offset: 3189
  Last block received:       no date
  Last block ACKed:          no date
  ID of the last TX begin:    4345
  ID of the last TX end:      4345
  Time of last TX end:        Wed Oct 26 08:33:46 2005
  Last AI Extent processed
    AIMAGE BEGIN date:        Wed Oct 26 08:32:12 2005
    AIMAGE NEW date:          Wed Oct 26 08:33:29 2005
    After-Image File Number:   3
    File Last Opened:          Wed Oct 26 08:33:29 2005
    Completely Applied to Target: No
```

## DSRUTIL relwaits qualifier

This is useful if the server ended and the OpenEdge database is waiting for OpenEdge Replication server action. For example, if the OpenEdge Replication server is waiting for synchronous transaction acknowledgments, `relwaits` frees up any pending waits that might be outstanding so that database activity can continue.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C relwaits</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

## DSRUTIL restart server qualifier

Restarts the OpenEdge Replication server. The action will not be performed if the OpenEdge Replication server is currently running.

The advantage to using this command is that you do not have to restart your database. If the Replication server crashes, however, and you restart it by using this command on your source database, it is possible that the Replication agent will terminate. An error message will also be logged in the target database's log file. If this occurs, shut down and restart the target database and then restart the Replication server with this command for your source database.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C restart server</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

## **DSRUTIL startagent database qualifier**

This qualifier is a synonym for the connectagent qualifier. For more information, see the [“DSRUTIL connectagent database qualifier”](#) section on page 5–17.

## DSRUTIL status qualifier

Allows you to query the status of a Replication server or agent.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C status agentname -detail</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

*agentname*

The name of the agent whose status you want. This name is optional if the database is a source database.

*-detail*

Provides the detail value, as shown in [Table 5-7](#).

The return codes listed in [Table 5-5](#) are valid.

**Table 5-5: Replication status return codes**

Code	Description
0	The command completed normally, and the status code was sent to stdout.
2	There was a generic database open error.
3	The database was opened, but it is not enabled for replication.

## When the detail argument is not used

The return code equals zero when the *-detail* argument is not used, and one of the status codes in [Table 5–6](#) is returned via stdout.

**Table 5–6: Return code zero status code**

If the status code is . . .	It reflects the status for . . .
1xxx	The server
2xxx	The control agent
3xxx	The agent

See [Table 5–7](#) for a list and description of the possible status code values.

**Table 5–7: Status code values**

(1 of 2)

Status	Class	Detail status	Value
100	Connecting.	Initial connection.	1001
100	Connecting.	Initializing.	1002
100	Connecting.	Server Initialization.	6001
100	Connecting.	Connecting to Agents.	6002
100	Connecting.	Configuring Agent(s).	6003
101	Processing is taking place.	Startup Synchronization.	3048
101	Processing is taking place.	Normal Processing.	3049
101	Processing is taking place.	Recovery Synchronization.	3050
101	Processing is taking place.	Recovery Processing.	6004
101	Processing is taking place.	Startup Synchronization.	6005
101	Processing is taking place.	Normal Processing.	6021
102	Activity is halted.	Online backup of the Target Database.	3051
102	Activity is halted.	Target Database in Quiet Point.	3052
102	Activity is halted.	Target Database is in a BI stall.	3053
102	Activity is halted.	Target Database is in an AI stall.	3054
103	Pre-transition state.	Pre-transition.	2080



**Table 5–7: Status code values***(2 of 2)*

Status	Class	Detail status	Value
103	Pre-transition state.	Applying After-image Extent.	2081
104	Agent is listening.	Listening.	2083
105	Transition.	Transitioning.	2082
199	Inactive.	Initial Connection Failed.	1032
199	Inactive.	Recovery Failed.	1033
199	Inactive.	Invalid Target Database Configuration.	1034
199	Inactive.	Agent Failed.	1035
199	Inactive.	Agent is Ignored.	1036
199	Inactive.	Agent is Stopped.	1037
199	Inactive.	Agent is Terminated.	1038
199	Inactive.	Agent is Ended.	1063
199	Inactive.	Server is ended.	6060
255	Unknown.	Unknown.	0

For example, for a source database whose Replication server is connecting to its configured agents, the status returned would be 1100.

## DSRUTIL terminate qualifier

Terminates the currently running OpenEdge Replication server or agent.

The advantage to using this command is that the database stays up and running, whereas PROSHUT would shut the database down.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C [terminate server   agent]</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

## DSRUTIL transition qualifier

Instructs an OpenEdge Replication agent to transition this target database to a normal OpenEdge database. This command can be used in either of the following situations:

- To force a transition when the server is in contact with the agent
- If the server and agent have lost connection and you want to perform a manual transition

### Syntax

Operating system	Syntax
UNIX Windows	<code>dshrutil <i>db-name</i> -C transition [failover] -logging [ 1   2 ]</code>

*db-name*

The name of the source database to perform the action on.

failover

Causes the Replication server to tell the Replication agent to start transition. When transition on the target database completes successfully, transition begins for the source database.

-logging

Turns on transition logging.

1

Produces summary logging.

2

Produces detailed logging.

## DSRUTIL triggertransition qualifier

Forces the target database to go into a pre-transition state.

The command cannot be used if the Replication agent is communicating with the Replication server. The target database can be started, which will start the Replication agent.

You can then execute `rprep1 db-name -C triggertransition`. The trigger transition command then places the agent into pretransition.

At that point, any available source AI extents can be applied to the target database; then the target database can be transitioned to a normal database.

### Syntax

Operating system	Syntax
UNIX Windows	<code>dsrutil db-name -C triggertransition</code>

*db-name*

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

# OpenEdge Replication DSRUTIL MONITOR

The DSRUTIL utility monitor allows you to monitor OpenEdge Replication and provides the display options described in the following sections.

## Startup menu

When you use the DSRUTIL MONITOR command, the screen shown in [Figure 5–1](#) appears.

Site Replication Monitor Version 2	Page 1
Database: /twoagent/tdba	
S. Replication server status	
R. Replication server remote agents	
A. Replication agent status	
M. Modify display defaults	
Q. Quit	
Enter your selection:	

**Figure 5–1: DSRUTIL Monitor Startup menu screen**

As shown in [Figure 5–1](#), the first two lines of the Startup Menu are the utility title line and the fully qualified name of the database specified by the user. The remaining lines are defined in [Table 5–8](#).

**Table 5–8: DSRUTIL monitor startup**

Field	Description
Replication server status	Instructs the user to enter the letter <b>S</b> to display the Replication server Status display screen
Replication server remote agents	Instructs the user to enter the letter <b>R</b> to display the Replication remote agent Selection menu
Replication agent status	Instructs the user to enter the letter <b>A</b> to display the Replication local agent Status display screen
Modify display defaults	Instructs the user to enter the letter <b>M</b> to display the current utility display settings and prompts the user for any desired changes
Quit	Instructs the user to enter the letter <b>Q</b> to exit the utility
Enter your selection	Instructs the user to enter a selection for the action the utility is to perform

## Replication server status

If you select **Replication server status** from the Startup Menu of the DSRUTIL Monitor Utility, the screen shown in [Figure 5–2](#) appears.

OpenEdge Replication Monitor		Page 1
Database: /vobs_repl/solaris/bin/ks1		
Database is enabled as OpenEdge Replication: Source		
Server is:	Connecting to Agent(s)	
Number of configured agents:	1	
Defer Agent Startup :		
Continue connection attempts until:	Tue Nov 27 01:18:27 2007	
Deferred Agent startup will expire in :	9 Hr 58 Min 37 Sec	
Next connection attempt in :	4 Min 41 Sec	
Connection attempts performed :	1	
Agent(s) currently connected :	0	
Delay Interval (current / min / max):	5 / 5 / 500	
Recovery information:		
State:	No recovery being performed	
Agents needing recovery:	0	
Agents connected:	0	
Agents in synchronization:	0	
Transition information:		
Type:	Manual	

**Figure 5–2: OpenEdge Replication server status screen**

As shown in [Figure 5–2](#), the first two lines of the Replication server status screen are the utility title line and the fully qualified name of the database specified by the user. The remaining lines are defined in [Table 5–9](#).

**Table 5–9: Replication server status***(1 of 2)*

Field	Description
Database is enabled as OpenEdge Replication	Indicates whether the database is enabled as a replication source or target.
Server is	<p>Describes how the server is processing information:</p> <ul style="list-style-type: none"> <li>• <b>Normal processing</b> — The server is processing information in the normal fashion.</li> <li>• <b>Performing initialization</b> — The server is performing initialization.</li> <li>• <b>Performing startup synchronization</b> — The server is in the process of synchronizing the target databases with the source database.</li> <li>• <b>Connection</b> — The server connection to agent(s).</li> <li>• <b>Configuring connected agents</b> — The server is performing handshaking with the agent.</li> <li>• <b>Performing failure recovery</b> — The server is attempting failure recovery from a connection failure.</li> <li>• <b>Unknown</b> — The server is in an unknown state.</li> </ul>
Number of configured agents	Shows the number of agents currently configured to operate with the server.
Defer Agent Startup	Shows information related to deferred agent startup.
Continue connection attempts until	Shows the time, day, and date when connection attempts will stop.
Deferred Agent Startup will expire in	Shows the remaining duration of deferred agent startup.
Next connection attempt in	Shows when the next connection attempt starts.

**Table 5–9: Replication server status**

(2 of 2)

Field	Description
Connection attempts performed	Shows how many connection attempts have occurred.
Agent(s) currently connected	Shows the number of connected agents.
Delay Interval (current / min / max)	Shows the amount of time, in milliseconds, the server will wait between polling the database for information to be replicated. <b>Current</b> is the current value, <b>min</b> is the minimum value, and <b>max</b> is the maximum value.  Polling is used to increase the performance of the server, and the wait is used to limit the amount of overhead when no data is available to be replicated.
Recovery information	Shows failure-recovery-related details.
Agents needing recovery	Shows the number of remote agents requiring failure recovery.
Agents connected	Shows the number of remote agents currently connected to the server.
Agents in synchronization	Shows the number of remote agents in the process of being brought up to date.
Transition information	Shows Replication transition details.
Type	Shows the type of transition to be performed: <ul style="list-style-type: none"> <li>• <b>Manual</b> — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database</li> <li>• <b>Automatic</b> — Indicates a transition from a target database to a normal OpenEdge database will take place without intervention</li> </ul>
Transition timeout limit	Shows the maximum amount of time that will elapse before the transition of a target database to a normal OpenEdge database.  <b>Note:</b> This line does not appear in <a href="#">Figure 5–2</a> , as the limit is shown for automatic transition only.



## Replication server remote agents

If you select **Replication server remote agents** from the Startup Menu of the DSRUTIL Monitor Utility (shown in [Figure 5-1](#)), the screen shown in [Figure 5-3](#) appears.

```

Site Replication Monitor Version 2 Page 1

Database: /twoagent/sdb

Database is enabled as OpenEdge Replication: Source

Remote Agents Configured:

ID   Name      Host Name   Database
1.   agent1    localhost   tdba
2.   agent2    localhost   tdbb

Q. Quit

Enter your selection:

```

**Figure 5-3: OpenEdge Replication server remote agents screen**

As shown in [Figure 5-3](#), the first two lines of the Replication Server Remote Agents screen are the utility title line and the fully qualified name of the database specified by the user.

The remaining lines of the Replication Server Remote Agents screen are defined in [Table 5-10](#).

**Table 5-10: Replication Server Remote Agents details**

Field	Description
Database is enabled as OpenEdge Replication	Indicates whether the database is enabled as a replication source or target.
Remote Agents Configured	Displays a list of the currently configured remote agents. Each remote agent is identified by an <b>ID</b> , the agent's <b>Name</b> , the <b>Host Name</b> , and the target <b>Database</b> .
Quit	Instructs the user to type the letter <b>Q</b> to exit the <b>Replication Server Remote Agents</b> menu.
Enter your selection	Instructs the user to type the <b>ID</b> of the remote agent whose information is to be displayed.

## Replication remote agents status

If you select the **Replication remote agents status** from the Startup Menu of the DSRUTIL Monitor Utility (shown in [Figure 5-1](#)), the screen shown in [Figure 5-4](#) appears.

```

Site Replication Monitor Version 2      Page 1
Database: /twoagent/sdb
Agent:
  Name:                                agent2
  ID:                                   2
  Host name:                           localhost
  Target database:                      tdbb
  State:                               Normal processing
  Critical:                            No
  Method:                              Asynchronous
  Remote agent is:                     Connected via broker
                                     Source and target are synchronized
                                     In normal processing
Remote agent is waiting for:           Nothing
Recovery state:                       No recovery being performed
Maximum bytes in TCP/IP message:      30720
Server/Agent connection time:         Mon Oct 31 14:50:58 2005
Server/Agent connection timeout:      120.000 seconds
Transition information:
  Type:                               Manual
  Timeout limit:                      0 seconds
The last block was sent at:            Mon Oct 31 14:57:42 2005
Activity information:
  Blocks sent:                        720
  Blocks acknowledged:                28
  Synchronization points              12
AI Block Information:
  Current RDBMS Block (Seq / Block):  5 / 1923
  Last Sent Block (Seq / Block):       5 / 1923
Server to agent load check interval:   10 blocks
Time between server and agent load checks: 5.597 seconds
Time taken to respond to load check:   0.178 seconds
RETURN - repeat, U - continue uninterrupted, Q - quit:

```

**Figure 5-4: OpenEdge Replication Remote Agents Status screen**

As shown in [Figure 5-4](#), the first two lines of the Replication Remote Agents Status screen are the utility title line and the fully qualified name of the database specified by the user.

The remaining lines of the Replication Remote Agents Status screen are defined in [Table 5–11](#).

**Table 5–11: Replication Remote Agents Status details** (1 of 3)

Field	Description
Agent	Shows a selection of basic remote agent information.
Name	Shows the name of the remote agent.
ID	Shows the remote agent identification number.
Host name	Shows the host on which the target database associated with the remote agent resides.
Target database	Shows the name of the database associated with the remote agent.
State	<p>Shows information about what the server knows about the remote agent. For example:</p> <ul style="list-style-type: none"> <li>• <b>Normal processing</b> — The server and agent are performing normal processing.</li> <li>• <b>Initial connection</b> — The agent is waiting for initial connection from server.</li> <li>• <b>Startup synchronization</b> — The server and agent are synchronizing.</li> <li>• <b>Initialization</b> — The agent is being initialized by the server.</li> <li>• <b>Initial connection failed</b> — The server could never connect an agent.</li> <li>• <b>Invalid target database configuration</b> — Something in the target database does not match the source.</li> <li>• <b>Agent terminated</b> — The target database shut down or the agent terminated using <b>Terminate</b> agent.</li> <li>• <b>Online backup of Target Database</b> — An online target database backup is being performed.</li> <li>• <b>Recovery synchronization</b> — Recovery synchronization is being performed.</li> <li>• <b>Recovery failed</b> — Failure recovery failed.</li> <li>• <b>Unknown</b> — The agent is in an unknown state.</li> </ul>
Critical	Shows whether the remote agent is a critical agent ( <b>Yes</b> ) or not critical ( <b>No</b> ).
Method	Shows the replication method: <b>asynchronous</b> or <b>synchronous</b> .

**Table 5–11: Replication Remote Agents Status details***(2 of 3)*

Field	Description
Remote agent is waiting for	Shows why the agent might be waiting. The field value might be one of the following: <ul style="list-style-type: none"> <li>• <b>Nothing</b> — Indicates no waiting is taking place</li> <li>• <b>Schema lock request</b> — Indicates the server is waiting for the agent to acquire the schema lock so the database schema can be changed</li> </ul>
Recovery state	Shows where the server and the agent are in the failure recovery process. The field value can be one of the following: <ul style="list-style-type: none"> <li>• <b>Failed for the agent</b> — Failure recovery could not be completed for the agent.</li> <li>• <b>No recovery being performed</b> — There is currently no failure recovery being done.</li> <li>• <b>Just entered recovery</b> — The server has just started failure recovery and will determine which remote agents remain connected and which must be reconnected.</li> <li>• <b>Server attempting connection</b> — The server is attempting to connect to those remote agents no longer communicating to the server.</li> <li>• <b>Initialize synchronization with agents</b> — The server is initializing the agent failure recovery synchronization.</li> <li>• <b>Synchronizing agents</b> — The connected agents are in the process of being brought up to date with the database changes made to the source database.</li> <li>• <b>Recovery complete</b> — The failure recovery of all connected agents has been completed.</li> </ul>
Maximum bytes in TCP/IP message	Shows the maximum number of bytes used for the TCP/IP communication messages.
Server/Agent connection time	Shows the date and time at which the server and agent connected.
Server/Agent connection timeout	Shows the number of seconds after which the OpenEdge Replication server will stop attempting to connect to the agent.
Transition information	Shows remote agent-related transition details.
Type	Shows the type of transition to be performed: <ul style="list-style-type: none"> <li>• <b>Manual</b> — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database</li> <li>• <b>Automatic</b> — Indicates a transition from a target database to a normal OpenEdge database will take place without intervention</li> </ul>

**Table 5–11: Replication Remote Agents Status details***(3 of 3)*

Field	Description
Timeout limit	Shows the maximum amount of time that will elapse before the transition of a target database to a normal OpenEdge database; shown for automatic transition only.
The last block was sent at	Shows the date and time the last block was sent to the agent.
Activity information	Shows activity information.
Blocks sent	Shows the number of blocks sent to the agent.
Blocks acknowledged	Shows the number of blocks acknowledged by the agent.
Synchronization points	Shows the number of synchronization points that have occurred.
AI Block Information	Provides latency information that shows how far behind OpenEdge Replication is in updating the target database. This is important if the target database is transitioned due to source database failure.
Current RDBMS Block (Seq / Block)	Shows the current RDBMS AI block.  Seq is the AI extent sequence number. It can be viewed by using <code>rfutil LIST</code> .
Last Sent Block (Seq / Block)	Shows the last AI block that was sent to the OpenEdge Replication agent.
Server to agent load check interval	Shows the number of blocks the server will send to the agent, at which point the server will wait for an acknowledgment from the agent for the block just sent.
Time between server and agent load checks	Shows the average elapsed time between the load checks.
Time taken to respond to load check	Shows the average elapsed time the agent took to acknowledge the server for the block just sent.

## Replication agent status

If you select the **Replication agent status** from the Startup Menu of the DSRUTIL Monitor Utility, the screen shown in [Figure 5–5](#) appears.

```

OpenEdge Replication Monitor                                Page 1

Database:  /vobs_repl/solaris/bin/ks2

Database is enabled as OpenEdge Replication:  Target

Agent:
  Name:                                agent1
  ID:                                   1
  Host name:                           127.0.0.1
  State:                               Normal Processing
  Ready:                               Yes
  Critical:                            No
  Method:                              Asynchronous
Agent is waiting for:                      Nothing
Maximum bytes in TCP/IP message:         8500
Server/Agent connection time:           Mon Nov 26 15:10:12 2007
Delay Interval (current / min / max):    5 / 5 / 500
Transition information:
  Type:                                Manual
The last block received at:              Mon Nov 26 15:11:23 2007
Activity information:
  Blocks received:                      493
  Blocks processed:                     493
  Blocks acknowledged:                 19
  Notes processed:                      32972
  Transactions started:                 505
  Transactions ended:                   504
  Synchronization points:               8
AI Block Information:
  Source RDBMS Block (Seq / Block):     2 / 5778
  Last Processed Block (Seq / Block):    2 / 5734
Latency Information:
  Repl Server behind Source DB by:      0      second(s)
  Current Source Database Transaction:   25426
  Last Transaction Applied to Target:    25405
  Target Current as of (Target, Source): Mon Nov 26 15:11:23 2007, Mon
Nov 26 15:11:23 2007 with delta of 000:00:00

```

**Figure 5–5: OpenEdge Replication agent status screen**

As shown in [Figure 5–5](#), the first two lines of the Replication Agent Status screen are the utility title line and the fully qualified name of the database specified by the user.

The remaining lines of the Replication Agent Status screen are defined in [Table 5–12](#).

**Table 5–12: Replication Agent Status details**

(1 of 3)

Field	Description
Database is enabled as OpenEdge Replication	Indicates whether the database is enabled as a replication source or target.
Agent	Lists basic remote agent information.
Name	Lists the name of the remote agent.
ID	Lists the remote agent identification number.
Host name	Lists the host on which the target database associated with the remote agent resides.
State	Shows agent processing information: <ul style="list-style-type: none"> <li>• <b>Normal processing</b> — The server and agent are performing normal processing.</li> <li>• <b>Initial connection</b> — The agent is waiting for initial connection from server.</li> <li>• <b>Startup synchronization</b> — The server and agent are synchronizing.</li> <li>• <b>Initialization</b> — The agent is being initialized by the server.</li> <li>• <b>Initial connection failed</b> — The server could never connect an agent.</li> <li>• <b>Invalid target database configuration</b> — Something in the target database does not match the source.</li> <li>• <b>Agent terminated</b> — The target database shut down or agent terminated using <code>Terminate agent</code>.</li> <li>• <b>Online backup of Target Database</b> — An online target database backup is being performed.</li> <li>• <b>Recovery synchronization</b> — Recovery synchronization is being performed.</li> <li>• <b>Recovery failed</b> — Failure recovery was unsuccessful.</li> <li>• <b>Unknown</b> — The agent is in an unknown state.</li> </ul>
Critical	Lists whether the remote agent is a critical agent ( <b>Yes</b> ) or not ( <b>No</b> ).
Method	Lists the replication method: <b>asynchronous</b> or <b>synchronous</b> .

Table 5–12: Replication Agent Status details

(2 of 3)

Field	Description
Agent is waiting for	Shows why the agent might be waiting. The field value might be one of the following: <ul style="list-style-type: none"> <li>• <b>Nothing</b> — Indicates no waiting is taking place</li> <li>• <b>Schema lock request</b> — Indicates the server is waiting for the agent to acquire the schema lock so the database schema can be changed</li> </ul>
Maximum bytes in TCP/IP message	Shows the maximum number of bytes used for the TCP/IP communication messages.
Server/Agent connection time	Shows the date and time at which the server and agent connected.
Delay Interval (current / min / max)	Shows the amount of time, in milliseconds, the agent will wait between polling the TCP/IP connection for information to be replicated. <b>Current</b> is the current value, <b>min</b> is the minimum value, and <b>max</b> is the maximum value.  Polling is used to increase the performance of the server, and the wait is used to limit the amount of overhead when no data is available to be replicated.
Transition information	Shows a group of remote agent-related transition information.
Type	Shows the type of transition to be performed: <ul style="list-style-type: none"> <li>• <b>Manual</b> — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database</li> <li>• <b>Automatic</b> — Indicates a transition from a target database to a normal OpenEdge database will take place without intervention</li> </ul>
The last block received at	Shows the date and time the last block was sent to the agent.
Activity information	Shows a selection of activity information.
Blocks received	Shows the number of blocks received from the server.
Blocks processed	Shows the number of blocks processed.
Blocks acknowledged	Shows the number of blocks acknowledged.
Notes processed	Shows the number of AI transaction notes processed by the agent.
Transactions started	Shows the number of transactions started on the agent.
Transactions ended	Shows the number of transactions completed on the agent.
Synchronization points	Shows the number of synchronization points that have occurred.



**Table 5–12: Replication Agent Status details***(3 of 3)*

Field	Description
AI Block Information	Identifies the current AI block that is being written by the source database and provides the last AI block that was sent by Replication.
Source RDBMS Block (Seq / Block)	Provides the current source database block that the OpenEdge Replication agent is processing.
Last Processed Block (Seq / Block)	Provides the last source database block that the OpenEdge Replication agent processed.
Latency Information	Provides latency information that shows how far behind OpenEdge Replication is in updating the target database. This is important if the target database is transitioned due to source database failure.
Repl Server behind Source DB by	Provides the number of seconds the OpenEdge Replication server is behind the source database.
Current Source Database Transaction	Provides the current source database transaction that the OpenEdge Replication agent is processing.
Last Transaction Applied to Target	Provides the last source database transaction that was applied to the target database.
Target Current As Of	Provides the date of the last transaction applied to the target database by the OpenEdge Replication agent based on source database time.

## Virtual system tables for OpenEdge Replication

Virtual system tables are available with OpenEdge Replication, as shown in [Table 5–13](#).

**Table 5–13: OpenEdge Replication virtual system tables**

Virtual system table	Description
OpenEdge Replication Server _Repl-Server	Provides detailed OpenEdge Replication server information
OpenEdge Replication Control Agent _Repl-AgentControl	Provides detailed information about the OpenEdge Replication agents this OpenEdge Replication server is controlling
OpenEdge Replication Agent _Repl-Agent	Provides detailed OpenEdge Replication agent information

For more information on using virtual system tables, see *OpenEdge Data Management: Database Administration*.

### Virtual system table field descriptions

This section describes the field descriptions for the OpenEdge Replication virtual system tables. [Table 5–14](#) shows the field descriptions for \_Repl-Server.

**Table 5–14: Virtual system table \_Repl-Server field description** (1 of 2)

Field name	Field number	Data type	Description
_ReplSrv-AgentCount	2	INTFLD	Agents Count Format ->>>>>>>>>9 Number of agents that the OpenEdge Replication server is controlling
_ReplSrv-BlocksSent	3	INTFLD	AI Blocks Sent Format ->>>>>>>>>9 Number of AI Blocks the OpenEdge Replication server has sent to connected OpenEdge Replication agents

**Table 5–14: Virtual system table \_Repl-Server field description** (2 of 2)

Field name	Field number	Data type	Description
_ReplSrv-StartTime	4	CHRFLD	Server Start Time Format X(24) Date and time when the OpenEdge Replication server was started
_ReplSrv-LastBlockSentAt	5	CHRFLD	Last Block Sent At Format X(24) Date and time when the OpenEdge Replication server was started

Table 5–15 shows the field descriptions for \_Repl-AgentControl.

**Table 5–15: Virtual system table \_Repl-AgentControl field description** (1 of 3)

Field name	Field number	Data type	Description
_ReplAgtCtl-AgentID	2	INTFLD	Agent ID Format ->>>>>>>>>9 The ID of the OpenEdge Replication agent
_ReplAgtCtl-AgentName	3	CHRFLD	Agent Name Format X(32) Name of the OpenEdge Replication agent as configured in the repl.properties file
_ReplAgtCtl-ConnectTime	4	CHRFLD	Agent Start Time Format X(24) Date and time when the OpenEdge Replication server successfully connected to the OpenEdge Replication agent
_ReplAgtCtl-RemoteDBName	5	CHRFLD	Remote Database Name Format X(256) Fully qualified database name to which the OpenEdge Replication agent is connected

**Table 5–15: Virtual system table \_Repl-AgentControl field description (2 of 3)**

Field name	Field number	Data type	Description
_ReplAgtCtl-RemoteHost	6	CHRFLD	Remote Host Name Format X(128) Name of the host where the OpenEdge Replication agent is running
_ReplAgtCtl-Port	7	INTFLD	Connected to Agent on Port Number Format ->>>>>>>>>9 The port number the OpenEdge Replication server uses to connect to this OpenEdge Replication agent
_ReplAgtCtl-BlockSent	8	INDFLD	Blocks Sent to this Agent Format ->>>>>>>>>9 The number of AI blocks the OpenEdge Replication server has sent to this OpenEdge Replication agent
_ReplAgtCtl-BlocksACK	9	INTFLD	Number of block acknowledgments received for this agent Format ->>>>>>>>>9 The number of block acknowledgements this OpenEdge Replication agent has received from its remote counterpart
_ReplAgtCtl-LastBlockSentAt	10	CHRFLD	Last Block Sent At Format X(24) The data and time the last block was sent to this OpenEdge Replication agent

**Table 5–15: Virtual system table \_Repl-AgentControl field description** (3 of 3)

Field name	Field number	Data type	Description
_ReplAgtCtl-Method	11	CHRFLD	<p>Replication Method</p> <p>Format X(1)</p> <p>The type of replication that is being performed with this OpenEdge Replication agent:</p> <ul style="list-style-type: none"> <li>• <b>A</b> — Asynchronous (mode &amp; RP_MODE_ASYNC)</li> <li>• <b>S</b> — Synchronous (mode &amp; RP_MODE_SYNC)</li> </ul>
_ReplAgtCtl-Status	12	INTFLD	<p>Agent's Status</p> <p>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</p> <p>The OpenEdge Replication agent's status:</p> <ul style="list-style-type: none"> <li>• <b>1</b> — Normal (Processing as expected)</li> <li>• <b>2</b> — Recovery (This OpenEdge Replication agent is in recovery mode)</li> </ul>
_ReplAgtCtl-CommStatus	13	INTFLD	<p>Agent's Communication Status</p> <p>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</p> <p>The OpenEdge Replication agent's TCP communication status:</p> <ul style="list-style-type: none"> <li>• <b>1</b> — Connected</li> <li>• <b>2</b> — Disconnected</li> </ul>

Table 5–16 shows the field descriptions for \_Repl-Agent.

**Table 5–16: Virtual system table \_Repl-Agent field descriptions** *(1 of 3)*

Field name	Field number	Data type	Description
_ReplAgt-AgentID	2	INTFLD	Agent ID Format ->>>>>>>>>9 The ID of the OpenEdge Replication agent
_ReplAgt-AgentName	3	CHRFLD	Agent Name Format X(32) Name of the OpenEdge Replication agent as configured in the repl.properties file
_ReplAgt-StartTime	4	CHRFLD	Agent Start Time Format X(24) Date and time when the OpenEdge Replication agent was started
_ReplAgt-ConnectTime	5	CHRFLD	Agent Connect Time Format X(24) Date and time when the OpenEdge Replication server successfully connected to the OpenEdge Replication agent
_ReplAgt-DBName	6	CHRFLD	Database Name Format X(128) Fully qualified database name that the OpenEdge Replication agent is connected to
_ReplAgt-ServerHost	7	CHRFLD	Server Host Name Format X(128) Name of the host where the OpenEdge Replication server is running
_ReplAgt-Port	8	INTFLD	The agent is connected to this port number Format ->>>>>>>>>9 The port number the OpenEdge Replication server uses to connect to this OpenEdge Replication agent

**Table 5–16: Virtual system table \_ReplAgent field descriptions** (2 of 3)

Field name	Field number	Data type	Description
_ReplAgt-BlocksReceived	9	INTFLD	Number of blocks received Format ->>>>>>>>>9 The number of AI blocks this OpenEdge Replication server has received
_ReplAgt-BlocksProcessed	10	INTFLD	Number of blocks processed. The number of AI blocks this OpenEdge Replication agent has processed
_ReplAgt-BlocksACK	11	INTFLD	Number of block acknowledgements sent by this agent Format ->>>>>>>>>9 The number of block acknowledgements this OpenEdge Replication agent has sent to the OpenEdge Replication server
_ReplAgt-NotesProcessed	12	INTFLD	Number of Notes Processed Format ->>>>>>>>>9 The number of AI notes this OpenEdge Replication agent has processed
_ReplAgt-LastTRID	13	INTFLD	Transaction ID of Last Transaction began Format ->>>>>>>>>9 The Transaction ID of the last Transaction Begin encountered
_ReplAgt-LastEND	14	INDFLD	Transaction ID of last Transaction End Format ->>>>>>>>>9 The Transaction ID of the last Transaction End encountered
_ReplAgt-Method	15	CHRFLD	Replication Method Format X(1) The type of replication that is being performed with this OpenEdge Replication agent: <ul style="list-style-type: none"> <li>• A — Asynchronous</li> <li>• S — Synchronous</li> </ul>

**Table 5–16: Virtual system table \_Repl-Agent field descriptions** *(3 of 3)*

Field name	Field number	Data type	Description
_ReplAgt-Status	16	INTFLD	<p>Agent's Status</p> <p>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</p> <p>The OpenEdge Replication agent's status:</p> <ul style="list-style-type: none"> <li>• <b>1</b> — Normal (Processing as expected)</li> <li>• <b>2</b> — Recovery (This OpenEdge Replication agent is in failure recover mode)</li> </ul>
_ReplAgt-CommStatus	17	INTFLD	<p>Agent's Communication Status</p> <p>The OpenEdge Replication agent's TCP communication status:</p> <ul style="list-style-type: none"> <li>• <b>1</b> — Connected</li> <li>• <b>2</b> — Disconnected</li> </ul>



## Utilities and OpenEdge Replication

Many of the OpenEdge database utilities are used to change a database. [Table 5–17](#) lists the utilities and indicates whether they are allowed to operate on a source or target database.

**Table 5–17: Utility support for OpenEdge Replication**

(1 of 4)

Utility	Source database	Target database
PROBKUP		
Online	Yes	Yes
Incremental	Yes	Yes
Normal	Yes	Yes
PROCOPY	Yes	Yes
PRODB	Yes	Yes
PRODEL	Yes	Yes
Progress Explorer	NA	NA
PROLOG	Yes	Yes
PROMON	Yes	Yes
PROREST	No	No
PROSTRCT		
Add	Yes	Yes
Buildddb	Yes	Yes
Create	NA	NA
List	Yes	Yes
Remove	Yes	No
Repair	Yes	Yes
Statistics	Yes	Yes
Unlock	No	No

**Table 5–17: Utility support for OpenEdge Replication***(2 of 4)*

Utility	Source database	Target database
PROUTIL		
2phase begin	No	No
2phase commit	No	No
2phase end	No	No
2phase modify	No	No
2phase recover	No	No
bigrow	Yes	No
busy	Yes	Yes
bulkload	No	No
chanalys	Yes	Yes
cmpdb	Yes	Yes
codepage-compiler	Yes	Yes
conv89	NA	NA
convchar	Yes	No
convfile	No	No
dbanalys		
online	Yes	Yes
offline	Yes	Yes
dbauthkey	No	No
dbipcs	Yes	Yes
dbrpr	Yes	No
dump	Yes	Yes
dumpspecified		
online	Yes	Yes
offline	Yes	Yes
enablelargfiles	Yes	Yes
enablesitereplication	Yes	Yes
disablesitereplication	Yes	Yes
holder	Yes	Yes

**Table 5–17: Utility support for OpenEdge Replication***(3 of 4)*

Utility	Source database	Target database
PROUTIL		
idxanalys		
online	Yes	Yes
offline	Yes	Yes
idxblockreport		
online	Yes	Yes
offline	Yes	Yes
idxbuild	No	No
idxcheck	Yes	Yes
idxcompact		
online	Yes	No
offline	No	No
idxfix		
online	Yes	No
offline	No	No
idxmove		
online	Yes	No
offline	No	No
iostats		
online	Yes	Yes
offline	Yes	Yes
load	No	No
mvsch	No	No
rckey	No	No
tabanalys		
online	Yes	Yes
offline	Yes	Yes
tablemove		
online	Yes	No
offline	No	No
truncate area	No	No
truncate bi	Yes	No
updatevst	No	No
wordbreak-compiler	NA	NA
word-rules	No	No

**Table 5–17: Utility support for OpenEdge Replication***(4 of 4)*

Utility	Source database	Target database
RFUTIL		
aimage begin	Yes	Yes
aimage end	No	Yes
aimage extent empty	Yes	Yes
aimage extent full	Yes	Yes
aimage extent list	Yes	Yes
aimage new	Yes	Yes
aimage scan	Yes	Yes
aimage truncate	No	No
mark backup	Yes	Yes
roll forward	No	No
roll forward retry	No	No
SQLDUMP	Yes	Yes
SQLLOAD	Yes	No
SQLSCHEMA	Yes	Yes

## Starting OpenEdge Replication with the Progress Explorer and dbman

Once you enable OpenEdge Replication and set the properties for both the source and the target databases, you can start OpenEdge Replication automatically with either the Progress Explorer or the dbman utility, instead of the PROSERVE command. This is possible because the database configurations that make OpenEdge Replication possible are saved in the `conmgr.properties` file by the Progress Explorer.

After you install OpenEdge Replication and the AdminServer is already running, the AdminServer will not be able to perform OpenEdge Replication configuration until you stop and restart it. This is because the AdminServer does not consider a database enabled for OpenEdge Replication until the next time the AdminServer starts. To force the AdminServer to consider the database enabled for OpenEdge Replication, simply stop the AdminServer and then restart it.

To start OpenEdge Replication with the dbman utility, enter the following command:

```
dbman -database db-name -start
```

Where *db-name* is the name of either the source or the target database.



### To start your database with the Progress Explorer:

1. In Progress Explorer, select either your source or target database icon.
2. Right-click and choose **Properties**.
3. Select the **Automatically Start this Database when the Admin Server is Started** option, and click **OK**.

## OpenEdge Replication and database management systems

OpenEdge Replication is easily integrated into existing DBMS control methodologies in the following ways:

- You can manage OpenEdge Replication through the AdminServer and administer it through the Progress Explorer. For more information about the Progress Explorer, see the [“Starting OpenEdge Replication with the Progress Explorer and dbman”](#) section on page 5–55.
- You can manage OpenEdge Replication by scripts through the command-line interface using startup arguments or parameter files.
- OpenEdge Management is able to manage and monitor OpenEdge Replication-enabled databases that are either managed or script-based. For more information, see *OpenEdge Management Database Management Guide* and *OpenEdge Management Reporting Guide*.

---

## OpenEdge Replication Quick Command Summary

---

This chapter provides command summary information for the following tasks:

- [Configuring the OpenEdge Replication properties files](#)
- [Setting up the source and target databases](#)
- [Configuring OpenEdge Replication with deferred agent startup](#)
- [Configuring OpenEdge Replication for one agent](#)
- [Configuring OpenEdge Replication for two agents](#)
- [Starting OpenEdge Replication](#)
- [Stopping OpenEdge Replication](#)
- [Terminating the OpenEdge Replication server and agent](#)
- [Configuring for automatic transition](#)
- [Configuring for manual transition](#)
- [Using manual transition](#)
- [Re-enabling OpenEdge Replication after transition](#)
- [Restarting OpenEdge Replication server after target shutdown](#)
- [Monitoring an OpenEdge Replication database](#)

## Configuring the OpenEdge Replication properties files

Before you perform any of the commands described in this summary, you must have configured your *source-db-name.repl.properties* file and your *target-db-name.repl.properties* file. For more information, including details about how to use the sample properties files supplied by OpenEdge to create your properties files, see the [“Configuring the OpenEdge Replication property files”](#) section on page 3–12.



## Setting up the source and target databases

This section describes how to set up the source and target databases.



### To set up the source database for offline backup:

1. Back up the source database, as shown:

```
probkup source-db-name device-name
```

2. Create a structure file. For example:

```
prostrct list source-db-name source-db-name.st
```

3. If AI is not enabled:

- a. Create and edit a structure file (*source-db-name\_ai.st*) to add AI.
- b. Apply *source-db-name\_ai.st* to the source database, as shown:

```
prostrct add source-db-name source-db-name_ai.st
```

- c. Back up the database, as shown:

```
probkup source-db-name source-db-name.bak
```

- d. Begin AI, as shown:

```
rfutil source-db-name -C aimage begin
```

4. Enable Replication, as shown:

```
proutil source-db-name -C enableSiteReplication source
```

5. Perform an incremental backup, as shown:

```
probackup source-db-name incremental device-name2
```

**To set up the source database with deferred agent startup for online backup:**

1. Create a structure file. For example:

```
prostrct list source-db-name source-db-name.st
```

2. If AI is not enabled:

- a. Create and edit a structure file (*source-db-name-ai.st*) to add AI.
- b. Apply *source-db-name-ai.st* to the source database, as shown:

```
prostrct add source-db-name source-db-name-ai.st
```

- c. Begin AI, as shown:

```
rfutil source-db-name -C aimage begin
```

3. Set defer-agent-startup in the server properties file, as shown:

```
[server]
  control-agents=agent-name
  database=source-db-name
  transition>manual
  transition-timeout=1200
  defer-agent-startup=1400
```

4. Enable Replication, as shown:

```
proutil source-db-name -C enableSiteReplication source
```

5. Restart the source database, as shown:

```
PROSERVE -db source-db-name -DBService replserv
-S [ port | service name ]
```

**To set up the target database:**

1. Move the *source-db-name.st*, the source backup file, and the incremental backup from the source machine to the target machine directory.
2. Edit *source-db-name.st* on your target machine to match the target physical structure.
3. Do a restore from both the backup and the incremental backup of the source database, as shown:

```
prorest target-db-name device-name  
prorest target-db-name device-name2
```

4. Enable Replication, as shown:

```
proutil target-db-name -C enableSiteReplication target
```

5. Start the server for the target, as shown:

```
proserve -db target-db-name -DBService replagent  
-S [port | service name]
```

For more information about setting up the source and target databases, see the [“Choosing the implementation method”](#) section on page 3–3.

## Configuring OpenEdge Replication with deferred agent startup

You configure OpenEdge Replication by copying and editing the sample source properties file.



### To configure OpenEdge Replication with deferred agent startup:

1. Copy the source properties file from `$DSRHOME/properties/source.rep1.properties` to your source database directory, and rename the copy with your source database name.
2. Edit the `[server]` section by setting `defer-agent-startup` to an appropriate value, as shown:

```
[server]
control-agents=agent-name
database=source-db-name
transition=manual
transition-timeout=1200
defer-agent-startup=1400
```

For complete information about `defer-agent-startup`, see the [“Choosing the implementation method”](#) section on page 3–3.

## Configuring OpenEdge Replication for one agent

You configure OpenEdge Replication for one agent by copying and editing the sample properties file.



### To configure OpenEdge Replication for one agent:

1. Copy the source properties file from `$DSRHOME/properties/source.rep1.properties` to your source database directory, and rename the copy with your source database name.
2. Edit the `[server]` and `[control.agent]` sections, similar to the following:

```
[server]
  control-agents=agent-name
  database=source-db-name
  transition=manual
  transition-timeout=1200

[control.agent.agent-name]
  name=agent-name
  database=target-db-name
  host=yourhost
  port=your port or service name
  connect-timeout=120
  replication-method=async
  critical=0
```

3. Copy the target properties file from `$DSRHOME/properties/target.rep1.properties` to your target database directory, and rename the copy with your target database name.
4. Edit the `[agent]` section, similar to the following:

```
[agent]
  name=agent-name
  database=target-db-name
  listener-minport=1500
  listener-maxport=4500
```

## Configuring OpenEdge Replication for two agents

You configure OpenEdge Replication for two agents by copying and editing the sample properties file.



### To configure OpenEdge Replication for two agents:

1. Copy the source properties file from `$DSRHOME/properties/source.rep1.properties` to your source database directory, and rename the copy with your source database name.
2. Edit the `[server]` and `[control.agent]` sections, similar to the following:

```
[server]
  control-agents=agent1-name, agent2-name
  database=source-db-name
  transition=manual
  transition-timeout=1200

[control-agent.agent1-name]
  name=agent1-name
  database=target-db-name
  host=yourhost
  port=your port or service name
  connect-timeout=120
  replication-method=async
  critical=0

[control-agent.agent2-name]
  name=agent2-name
  database=target-db2-name
  host=yourhost
  port=your port or service name
  connect-timeout=120
  replication-method=async
  critical=0
```

3. Copy the target properties file from `$DSRHOME/properties/target.rep1.properties` to the database directories for each target database, and rename the copies with the corresponding target database name.
4. Edit the `[agent]` section in each file, similar to the following:

```
[agent]
  name=agent-name
  database=target-db-name
  listener-minport=1500
  listener-maxport=4500
```

## Starting OpenEdge Replication

After you have set up the databases and configured replication, you can start OpenEdge Replication by starting both the source and target databases as services.



### To start OpenEdge Replication:

1. On the source machine, start the source database, as shown:

```
proserve -db source-db-name -DBService replserv -S[ port | service name]
```

2. On the target machine, start the target database, as shown:

```
proserve -db target-db-name -DBService replagent  
-S[ port | service name]
```

---

**Note:** The *port* or *service name* must match the entries in the control agent sections of the target properties file.

---

For more information about these steps, see the “[Starting the source database](#)” section on page 3–16 and the “[Starting the target database](#)” section on page 3–16.

## Stopping OpenEdge Replication

You can use the PROSHUT command on the source and target databases to stop OpenEdge Replication.



### To stop OpenEdge Replication:

1. On the source machine, enter the following:

```
proshut source-db-name -by
```

2. On the target machine, enter the following:

```
proshut target-db-name -by
```

---

**Note:** You can also force a shutdown using `proshut db-name -byF`. However, this is not recommended under normal circumstances.

---

For more information about these steps, see the [“Shutting down the source database”](#) section on page 3–23 and the [“Shutting down the target database”](#) section on page 3–24.



## Terminating the OpenEdge Replication server and agent

You can use the DSRUTIL utility to terminate an OpenEdge Replication server and agent.



### To terminate OpenEdge Replication servers and agents:

1. On the source machine, enter the following:

```
dsrutil source-db-name -C terminate server
```

2. On the target machine, enter the following:

```
dsrutil target-db-name -C terminate agent
```

For more information about these steps, see the [“Shutting down the OpenEdge Replication server”](#) section on page 3–24 and the [“Terminating the OpenEdge Replication agent”](#) section on page 3–25.

## Configuring for automatic transition

You configure OpenEdge Replication for automatic transition by editing the source properties file.



### To configure OpenEdge Replication for automatic transition:

1. Set `transition=auto` in the properties file in the source database directory.
2. Set `critical=1` in the properties file in the source database directory. For example:

```
[server]
control-agents=agent-name1
database=source-db-name
transition=auto
transition-timeout=1200

[control-agent.agent-name1]
name=agent1
database=target-db-name
host=yourhost
port=port or service name
connect-timeout=120
replication-method=async
critical=1
```

For more information, see the [“Setting up the target database”](#) section on page 3–11.

## Configuring for manual transition

You configure OpenEdge Replication for manual transition by editing the source properties file.



### To configure OpenEdge Replication for manual transition:

1. Set `transition=manual` in the properties file in the source database directory.
2. Set `critical=0` in the properties file in the source database directory. For example:

```
[server]
  control-agents=agent-name1
  database=source-db-name
  transition=manual
  transition-timeout=1200

[control-agent.agent-name1]
  name=agent-name1
  database=target-db-name
  host=yourhost
  port=port or service name
  connect-timeout=120
  replication-method=async
  critical=0
```

For more information, see the [“Setting up the target database”](#) section on page 3–11.

## Using manual transition

Use the DSRUTIL utility anytime after OpenEdge Replication is running to transition a target database to a normal database.

To manually transition a target database, enter the following command:

```
dsrutil target-db-name -C transition agent
```

## Re-enabling OpenEdge Replication after transition

Once the target database has transitioned to be the new source database, you can re-enable OpenEdge Replication.



### To re-enable OpenEdge Replication after transition:

1. In the target database directory, enter the following commands:

```
proshut target-db-name -by  
dbutil probkup target-db-name target-db-name.bak
```

2. In the source database directory, enter the following commands:

```
proutil source-db-name -C disablesitereplication source  
prorest source-db-name target-db-name.bak  
rfutil source-db-name -C aimage begin  
proutil source-db-name -C enablesitereplication source
```

3. In the target database directory, enter the following command:

```
proutil target-db-name -C enablesitereplication target
```

For more information about these steps, see the [“Re-enabling OpenEdge Replication after transition”](#) section on page 3–40.

## Restarting OpenEdge Replication server after target shutdown

Restart the OpenEdge Replication server with the `dsrutil start server` command. You can use this process after the target database has been shut down with a `proshut database -by` command. In this case, after database activity has been flushed to disk on the target database, the OpenEdge Replication agent informs the OpenEdge Replication server that it has been shut down. If this is the only OpenEdge Replication agent serviced by the OpenEdge Replication server, the OpenEdge Replication server shuts down. If the OpenEdge Replication agent was configured as a critical OpenEdge Replication agent, even if there are additional OpenEdge Replication agents, the OpenEdge Replication server shuts down.



### To restart OpenEdge Replication after a target shutdown:

1. In the target database directory, enter the following command:

```
proserve -db target-db-name -DBService replagent -S [port | service-name]
```

2. In the source database directory, enter the following command:

```
dsrutil source-db-name RESTART Server
```

## Monitoring an OpenEdge Replication database

You can monitor an OpenEdge Replication source or target database once it is up and running.

To monitor the source or target database, enter the following command:

```
dsrutil db-name -C monitor
```





---

# Index

---

## A

Administration 3–1

After-Imaging

- emptying extents 3–26
- enabling 3–8
- extent blocks 3–29
- extent sizing on source database 2–10
- extent states 2–4
- extent types 2–4
- extents 2–3
- fixed-length extents 2–3
- full extents 3–26
- switching extents 3–30
- variable-length extents 2–3

Agent 1–7

- configuration 3–14, 5–33
- connection timeout 5–9, 5–38
- critical 5–8
- host 5–8
- latency report 3–27
- listener-maxport 5–9
- listener-minport 5–9
- polling 5–5, 5–9
- port 5–8
- properties 5–9
- rules 3–13
- single agent configuration 6–7
- status 5–31, 5–40
- terminating 3–25, 5–12, 5–28, 6–11
- two agent configuration 6–8

ApplyExtent command action 5–11, 5–13

Architecture 1–3

Asynchronous

- configuration 1–10
- operation 1–11

Automatic transition 3–35, 5–34, 6–12

## B

Before-Image truncate 1–14

Benefits of OpenEdge Replication 1–2

## C

Checkpoint 3–30

Commands 3–28

Configuring

- agent 3–14, 5–33, 6–7, 6–8
- errors 3–16
- OpenEdge Replication 3–12

Connection 5–33

connect-timeout property 5–8, 5–9

Control agent properties 5–5, 5–7

Creating the target database 3–11

Critical agent 1–9

critical property 5–8

**D**

Data replication 1–2

Database

- access 3–21

- availability 1–13

- Before Image truncate 1–14

- connection considerations 3–20

- crash 3–32

- Enhanced Read-only mode

  - defined 1–13

- integrity 1–13

- schema locks 1–14

database property 5–5, 5–8, 5–9

database.repl.recovery 3–30

database.repl.recovery file 3–18

db.repl.properties 5–3

-DBService argument 3–16

Default implementation 3–3

defer-agent-startup property 3–4, 3–10,  
3–19, 5–5

Deferred agent startup 3–4, 3–6, 3–10, 6–6

disablesitereplication argument 3–18, 5–11,  
5–18

Disabling OpenEdge Replication 3–18

DSRUTIL

- MONITOR

  - latency details 3–27

DSRUTIL utility 3–28, 3–36, 5–11

- agent option 5–11

- MONITOR 5–31

- server option 5–11

- startup menu 5–31

**E**

Emptying AI extents 3–26

Enable target 3–11

enablesitereplication argument 3–11

Endian ordering

- defined 1–6

Enhanced Read-only mode (ERO)

- defined 1–13

Estimating network bandwidth 2–14

**F**

Failover

- defined 2–18

Failure

- conditions 3–32

- processing 1–12

- recovery 1–12, 3–31

Fixed-length after-image extents 2–3

Full AI extents 3–26

Functional definitions 1–3

**H**

Hot standby

- defined 3–31

**I**

Implementation 3–3

- default 3–3

- deferred agent startup 3–4

Incremental backup 3–8

Initialization process 3–19

ipver property 5–8, 5–9

**L**

Latency 3–27, 3–31, 5–40

- details provided by DSRUTIL

- MONITOR 3–27

Limitations and restrictions 2–19

listener-maxport property 5–9

listener-minport property 5–9

Logical operation 2–8

Lost connection 3–32

**M**

Manual transition 3–36, 5–34, 6–13

maximum-polling-delay property 5–5, 5–9

Message logging 3–17

minimum-polling-delay property 5–5, 5–10

Monitoring 6–17

**N**

name property 5–8, 5–10

Normal

- database activity 3–29
- processing 5–33
- target activity 3–29

**O**

Offline backup 3–7

Online backup of source database 3–9,  
3–10, 6–4

Online backup of target database 1–15

OpenEdge Replication

- administration 3–1
- agent 1–7
- agent properties 5–9
- architecture 1–3
- automatic transition 6–12
- benefits 1–2
- commands 3–28
- configuring 3–12
- control agent properties 5–7
- critical agent 1–9
- defined 1–2
- disabling 3–18
- enabling 3–11
- failover 2–18
- feature 1–2
- functional definitions 1–3
- implementation 3–3
- limitations and restrictions 2–19
- manual transition 6–13
- model 1–9
- monitoring 6–17
- network bandwidth estimation 2–14
- overhead 2–10, 2–14
- planning 2–2
- property files 5–3
- re-enabling after transition 6–14
- server 1–6
- server properties 5–5
- starting 6–9
- startup 3–15
- stopping 3–23, 6–10
- utilities 3–28, 5–51

**P**

Performing failure recovery 5–33

Performing initialization 5–33

Planning 2–2

port property 5–8

Pre-transition state 3–34

Primary database  
defined 1–3

PROBKUP 1–15, 5–51

PROCOPY 5–51

PRODB 5–51

PRODEL 5–51

Progress Explorer 5–51

PROLOG 5–51

PROMON 5–51

Properties 5–5

- agent 5–9
- control agent 5–7

Property files 5–3

- sample combined 5–4
- sample source 5–3
- sample target 5–4
- source 5–3
- target 5–4

PROREST 3–18, 5–51

PROSERVE 3–16

PROSHUT 3–23, 3–24

PROSTRCT 5–51

PROUTIL 3–11, 3–18, 5–52

**R**

\_Repl-Server VST 5–44

\_ReplAgt-AgentID 5–48

\_ReplAgt-AgentName 5–48

\_ReplAgt-BlocksACK 5–49

\_ReplAgt-BlocksProcessed 5–49

\_ReplAgt-BlocksReceived 5–49

\_ReplAgt-ConnectTime 5–48

\_ReplAgt-DBName 5–48

\_ReplAgt-LastEND 5–49

\_ReplAgt-LastTRID 5–49

\_ReplAgt-Method 5–49

\_ReplAgt-NotesProcessed 5–49

- `_ReplAgt-Port` 5–48
- `_ReplAgt-ServerHost` 5–48
- `_ReplAgt-StartTime` 5–48
- `_ReplAgtCtl-BlocksACK` 5–46
- `_ReplAgtCtl-AgentID` 5–45
- `_ReplAgtCtl-BlockSent` 5–46
- `_ReplAgtCtl-CommonStatus` 5–47
- `_ReplAgtCtl-ConnectTime` 5–45
- `_ReplAgtCtl-LastBlockSentAt` 5–46
- `_ReplAgtCtl-Method` 5–47
- `_ReplAgtCtl-Port` 5–46
- `_ReplAgtCtl-RemoteDBName` 5–45
- `_ReplAgtCtl-RemoteHost` 5–46
- `_ReplAgtCtl-Status` 5–47
- `_ReplSrv-AgentCount` 5–44
- `_ReplSrv-BlockSent` 5–44
- `_ReplSrv-LastBlockSentAt` 5–45
- `_ReplSrv-StartTime` 5–45
- Recovery 1–12
  - lost connection 3–33
- RECOVERY command action 5–20
- Re-enabling OpenEdge Replication after transition 6–15
- RELWAITS command action 5–22
- Remote agent status 5–35, 5–36
- `repl.server.startup.lg` 3–17
- Replication agent. *See* Agent
- Replication overhead 2–10
- Replication server. *See* Server
- `replication-method` property 5–8
- `Repl-Keep-Alive` property 5–6
- RESTART command action 5–12, 5–23
- Restarting server 6–16
- Restoring
  - databases 3–18
  - source database 3–18

- RFUTIL 5–54

- Roll forward 3–30

## S

- Schema lock 1–14
  - request 5–38

- Schema-Lock-Action property 5–6

- Secondary database 1–3, 4–2
  - defined 1–3

- Server 1–6
  - latency report 3–27
  - properties 5–5
  - recovery 5–20
  - remote agents details 5–35
  - remote agents status 5–36
  - restarting after target shutdown 6–16
  - status 5–31, 5–32, 5–33
  - terminating 3–24, 6–11

- Setup
  - source 3–7, 3–9, 6–3
  - target 3–11, 6–3, 6–5

- Shutdown
  - source 3–23
  - target 3–24

- Single agent configuration 6–7

- Source
  - access 3–22
  - combined properties file 5–4
  - database
    - defined 1–3
  - enabling 3–8, 3–9
  - incremental backup 3–8
  - machine 2–2
  - normal activity 3–29
  - offline backup 3–7
  - online backup 3–9, 3–10, 6–4
  - properties file 5–3
  - PROSERVE startup 3–16
  - requirements 1–6
  - restoring 3–18
  - setup 3–7, 3–9, 6–3
  - shutdown 3–23
  - startup 3–16
  - structure files 3–7, 3–9

- SQLDUMP 5–54

- SQLLOAD 5–54

- SQLSCHEMA 5–54

- `startAgent` command action 5–12, 5–24

- Starting OpenEdge Replication 5–55, 6–9

- Startup 3–15
  - message logging 3–17
  - process 3–19
  - source 3–16
  - target 3–16
- Stopping OpenEdge Replication 3–23, 6–10
- Structure files 3–7, 3–9
- Supported utilities 5–51
- Sync point
  - defined 3–30
- Synchronization
  - defined 1–12
  - processing 5–33
- Synchronous
  - configuration 1–10
  - operation 1–11
- T**
- Target
  - access 3–21
  - combined properties file 5–4
  - create 3–11
  - database
    - defined 1–3
  - enabling 3–11
  - machine 2–2
  - normal activity 3–29
  - online backup 1–15
  - properties file 5–4
  - PROSERVE startup 3–16
  - quiet points 1–15
  - requirements 1–6
  - restoring 3–18
  - setup 3–11, 6–3, 6–5
  - shutdown 3–24
  - startup 3–16
  - transition 3–35
- target.repl.properties 3–14
- TCP/IP communications 1–10
- TERMINATE command action 3–24, 3–25, 5–28
- Terminating
  - agent 3–25, 6–11
  - server 3–24, 6–11
- Transaction end 3–30
- Transition 4–4
  - automatic 3–35
  - configuration 3–35
  - defined 3–35
  - lost connection 3–34
  - manual 3–36
  - processing 3–37
  - re-enabling OpenEdge Replication after 6–15
  - timeout limit 5–34
- Transition command action 5–29
- transition property 5–7
- transition-timeout property 5–7
- TriggerTransition command action 5–12, 5–30
- U**
- Unknown state 5–33
- Unsupported utilities 5–51
- Utilities 3–28, 5–51
- V**
- Variable-length after-image extents 2–3
- Virtual system tables 5–44
- VST field descriptions 5–44

