# Contents

Preface ................................................................. Preface–1

1. **Overview of OpenEdge Replication** ................................... 1–1
   - 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
     - Using the passphrase for encrypted databases .................... 1–13
     - Enhanced Read-Only mode ............................................. 1–14
     - Schema lock ................................................................. 1–14
     - Forced truncation of the source database before-image file .... 1–15
     - Target quiet points ....................................................... 1–15
     - Online backup of the target database ................................ 1–15
2. Planning for OpenEdge Replication .................................................. 2–1
   Preliminary planning tasks .......................................................... 2–2
      Creating a database backup plan. .............................................. 2–2
      Evaluating your production database ...................................... 2–2
   OpenEdge Replication and transparent data encryption .................. 2–3
      Databases enabled for both replication and encryption ............... 2–3
   OpenEdge Replication and after-imaging .................................... 2–4
      After-image extents ............................................................... 2–4
      Calculating current after-image volume .................................. 2–6
   Replication and after-image extents .......................................... 2–9
      After-image extent sizing on a source database ......................... 2–11
      Calculating fixed after-image extent size ............................... 2–11
      Using variable extents ......................................................... 2–14
   Determining OpenEdge Replication network bandwidth ................... 2–15
      Estimating network bandwidth ............................................... 2–15
      Anticipating additional network overhead ............................... 2–17
      Additional references ......................................................... 2–17
   Additional business considerations ............................................ 2–18
      Latency ............................................................................. 2–18
      Acceptable target database downtime ..................................... 2–19
      Appropriate failover behavior .............................................. 2–19
      Limitations and restrictions .................................................. 2–19
      Ensuring success when using after-imaging ............................... 2–20
   A summary of OpenEdge Replication and after-imaging .................. 2–21

3. Implementing OpenEdge Replication ............................................. 3–1
   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 .................................................. 3–12
   Enabling before-image encryption for a Replication-enabled target database .......................... 3–13
   Configuring the OpenEdge Replication property files ..................... 3–15
      Configuring the source database property file ......................... 3–15
      Configuring the target database property file ......................... 3–17
   Starting OpenEdge Replication .................................................. 3–18
      Starting the source database ............................................... 3–18
      Starting the target database ............................................... 3–19
      Message logging during startup ............................................ 3–19
   Restoring source and target databases ....................................... 3–20
   OpenEdge Replication startup and initialization process ................. 3–21
   Database connection considerations ......................................... 3–22
   Database access once OpenEdge Replication is running .................. 3–23
      Target database access ....................................................... 3–23
      Source database access ..................................................... 3–24
   Stopping OpenEdge Replication ............................................... 3–25
      Shutting down the source database ...................................... 3–25
      Shutting down the target database ...................................... 3–26
      Shutting down the OpenEdge Replication server ....................... 3–26
      Terminating the OpenEdge Replication agent .......................... 3–27
      Emptying AI extents on the source database ............................ 3–28
   Latency reporting ..................................................................... 3–29
      OpenEdge Replication server perspective ................................ 3–29
      OpenEdge Replication agent perspective ................................. 3–29
### Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1–1</td>
<td>OpenEdge Replication from one site to another</td>
<td>1–3</td>
</tr>
<tr>
<td>Figure 1–2</td>
<td>Source and target databases</td>
<td>1–5</td>
</tr>
<tr>
<td>Figure 1–3</td>
<td>OpenEdge Replication server</td>
<td>1–7</td>
</tr>
<tr>
<td>Figure 1–4</td>
<td>OpenEdge Replication agent</td>
<td>1–8</td>
</tr>
<tr>
<td>Figure 1–5</td>
<td>OpenEdge Replication model</td>
<td>1–9</td>
</tr>
<tr>
<td>Figure 1–6</td>
<td>Synchronous and asynchronous configurations</td>
<td>1–10</td>
</tr>
<tr>
<td>Figure 1–7</td>
<td>Asynchronous operation</td>
<td>1–11</td>
</tr>
<tr>
<td>Figure 1–8</td>
<td>Synchronous operation</td>
<td>1–11</td>
</tr>
<tr>
<td>Figure 3–1</td>
<td>AI blocks sent from server to agent</td>
<td>3–31</td>
</tr>
<tr>
<td>Figure 3–2</td>
<td>Continuous roll forward on agent</td>
<td>3–32</td>
</tr>
<tr>
<td>Figure 3–3</td>
<td>Server properties file with automatic transition</td>
<td>3–39</td>
</tr>
<tr>
<td>Figure 3–4</td>
<td>Server properties file with manual transition</td>
<td>3–40</td>
</tr>
<tr>
<td>Figure 4–1</td>
<td>Primary replication before a failure</td>
<td>4–8</td>
</tr>
<tr>
<td>Figure 4–2</td>
<td>Primary machine failure</td>
<td>4–8</td>
</tr>
<tr>
<td>Figure 4–3</td>
<td>Entering pretransition—applying all unapplied source database AI extents</td>
<td>4–9</td>
</tr>
<tr>
<td>Figure 4–4</td>
<td>Transitioning the target to a source database</td>
<td>4–10</td>
</tr>
<tr>
<td>Figure 4–5</td>
<td>Secondary database transitioned to source database</td>
<td>4–11</td>
</tr>
<tr>
<td>Figure 4–6</td>
<td>Secondary database activity</td>
<td>4–12</td>
</tr>
<tr>
<td>Figure 4–7</td>
<td>Primary machine repair complete</td>
<td>4–12</td>
</tr>
<tr>
<td>Figure 4–8</td>
<td>Performing an online backup of secondary database</td>
<td>4–13</td>
</tr>
<tr>
<td>Figure 4–9</td>
<td>Restoring backup of secondary database</td>
<td>4–13</td>
</tr>
<tr>
<td>Figure 4–10</td>
<td>Transitioning source database to target</td>
<td>4–14</td>
</tr>
<tr>
<td>Figure 4–11</td>
<td>Transition completes</td>
<td>4–15</td>
</tr>
<tr>
<td>Figure 4–12</td>
<td>Secondary replication occurs</td>
<td>4–16</td>
</tr>
<tr>
<td>Figure 4–13</td>
<td>Secondary replication continues (before transition failover)</td>
<td>4–17</td>
</tr>
<tr>
<td>Figure 4–14</td>
<td>Scheduling downtime to perform failback</td>
<td>4–17</td>
</tr>
<tr>
<td>Figure 4–15</td>
<td>Initiating transition of the primary database</td>
<td>4–18</td>
</tr>
<tr>
<td>Figure 4–16</td>
<td>Continuing transition of the primary database</td>
<td>4–19</td>
</tr>
<tr>
<td>Figure 4–17</td>
<td>Databases started in new roles</td>
<td>4–20</td>
</tr>
<tr>
<td>Figure 4–18</td>
<td>Primary replication activity is occurring</td>
<td>4–21</td>
</tr>
<tr>
<td>Figure 4–19</td>
<td>Secondary replication continues (before controlled transition)</td>
<td>4–22</td>
</tr>
<tr>
<td>Figure 4–20</td>
<td>Scheduling downtime to perform failback</td>
<td>4–23</td>
</tr>
<tr>
<td>Figure 4–21</td>
<td>Primary replication activity resumes</td>
<td>4–26</td>
</tr>
<tr>
<td>Figure 5–1</td>
<td>DSRUTIL Monitor Startup menu screen</td>
<td>5–31</td>
</tr>
<tr>
<td>Figure 5–2</td>
<td>OpenEdge Replication server status screen</td>
<td>5–32</td>
</tr>
<tr>
<td>Figure 5–3</td>
<td>OpenEdge Replication server remote agents screen</td>
<td>5–35</td>
</tr>
<tr>
<td>Figure 5–4</td>
<td>OpenEdge Replication Remote Agents Status screen</td>
<td>5–36</td>
</tr>
<tr>
<td>Figure 5–5</td>
<td>OpenEdge Replication agent status screen</td>
<td>5–40</td>
</tr>
</tbody>
</table>
Tables

Table 2–1: After-image extent types .................................................. 2–5
Table 2–2: After-image extent states .................................................. 2–5
Table 2–3: After-image extent sizing calculations (hourly data) .................. 2–12
Table 2–4: After-image extent sizing calculations (weekly data) .................. 2–13
Table 3–1: Connection results during startup and initialization ...................... 3–22
Table 3–2: Standard system-level access to target database ......................... 3–23
Table 3–3: System-level access to target database during transition ................ 3–24
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 ............................................... 5–12
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 users familiar with OpenEdge® database administration and who plan 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.
Preface

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 Overview page on PSDN: [http://communities.progress.com/pcom/docs/DOC-16074](http://communities.progress.com/pcom/docs/DOC-16074).

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:

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

Syntax

| 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 |

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A vertical bar indicates a choice.</td>
</tr>
<tr>
<td></td>
<td>A vertical bar indicates a choice.</td>
</tr>
<tr>
<td></td>
<td>Ellipses indicate repetition: you can choose one or more of the preceding items.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A vertical bar indicates a choice.</td>
</tr>
<tr>
<td></td>
<td>A vertical bar indicates a choice.</td>
</tr>
<tr>
<td></td>
<td>Ellipses indicate repetition: you can choose one or more of the preceding items.</td>
</tr>
</tbody>
</table>
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 AdventNet - Agent Toolkit licensed from AdventNet, Inc. http://www.adventnet.com. All rights to such copyright material rest with AdventNet.

OpenEdge includes ANTLR (Another Tool for Language Recognition) 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 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. 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 Concurrent Java software Copyright 1994-2000 Sun Microsystems, Inc. All Rights Reserved. -Neither the name of or trademarks of Sun may be used to endorse or promote products including or derived from the Java Software technology without specific prior written permission; and Redistributions of source or binary code must contain the above copyright notice, this notice and the following disclaimers: This software is provided “AS IS,” without a warranty of any kind. ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN MICROSYSTEMS, INC. AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THE SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN MICROSYSTEMS, INC. OR ITS LICENSORS BE LIABLE FOR ANY LOSS REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEOREY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE SOFTWARE, EVEN IF SUN MICROSYSTEMS, INC. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

OpenEdge includes DataDirect software Copyright © 1991-2007 Progress Software Corporation and/or its subsidiaries or affiliates. All Rights Reserved. (DataDirect Connect for JDBC Type 4 driver); Copyright © 1993-2009 Progress Software Corporation and/or its subsidiaries or affiliates. All Rights Reserved. (DataDirect Connect for JDBC); Copyright © 1988-2007 Progress Software Corporation and/or its subsidiaries or affiliates. All Rights Reserved. (DataDirect Connect for ODBC); and Copyright © 1988-2007 Progress Software Corporation and/or its subsidiaries or affiliates. All Rights Reserved. (DataDirect Connect64 for ODBC).

OpenEdge includes DataDirect Connect for ODBC and DataDirect Connect64 for ODBC software, which include 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 DataDirect Connect for ODBC and DataDirect Connect64 for ODBC software, which include software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/). Copyright © 1998-2006 The OpenSSL Project. All rights reserved. And Copyright © 1995-1998 Eric Young (eay@cryptsoft.com). All rights reserved.

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

OpenEdge includes software authored by David M. Gay. Copyright © 1991, 2000, 2001 by Lucent Technologies (dtoa.c); Copyright © 1991, 1996 by Lucent Technologies (g_fmt.c); and
Copyright © 1991 by Lucent Technologies (rnd_prod.s). 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 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 AUTHOR NOR LUCENT MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND CONCERNING THE MERCHANTABILITY OF THIS SOFTWARE OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.

OpenEdge includes software authored by David M. Gay. Copyright © 1998-2001 by Lucent Technologies All Rights Reserved (decsrtdo.c; srtodg.c); Copyright © 1998, 2000 by Lucent Technologies All Rights Reserved (decsrtorf.c; srtord.c); Copyright © 1998 by Lucent Technologies All Rights Reserved (dmisc.c; gdtoa.h; gethex.c; gmisc.c; sum.c); Copyright © 1998, 1999 by Lucent Technologies All Rights Reserved (gdtoa.c; misc.c; smisc.c; ulp.c); Copyright © 1998-2000 by Lucent Technologies All Rights Reserved (gdtoaimp.h); Copyright © 2000 by Lucent Technologies All Rights Reserved (hd_init.c). Full copies of these licenses can be found in the installation directory, in the c:/OpenEdge/licenses folder. Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that the copyright notice and this permission notice and warranty disclaimer appear in supporting documentation, and that the name of Lucent or any of its entities not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. LUCENT DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL LUCENT OR ANY OF ITS ENTITIES BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

OpenEdge includes http package 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 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 Infragistics NetAdvantage for .NET v2009 Vol 2 Copyright © 1996-2009 Infragistics, Inc. All rights reserved.
OpenEdge includes JSTL software Copyright 1994-2006 Sun Microsystems, Inc. All Rights Reserved. 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 OpenSSL 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). This product includes software written by Tim Hudson (tjh@cryptsoft.com). Copyright © 1995-1998 Eric Young (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. 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 Quartz Enterprise Job Scheduler software Copyright © 2001-2003 James House. All rights reserved. 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. This product uses and includes within its distribution, software developed by the Apache Software Foundation (http://www.apache.org/).

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

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.

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 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. Exolab, Castor and Intalio are trademarks of Intalio Inc.
OpenEdge includes Sonic software, which 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 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.

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.

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 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.

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 Vermont Views Terminal Handling Package software developed by Vermont Creative Software. Copyright © 1988-1991 by Vermont Creative Software.

OpenEdge includes XML Tools, which includes versions 8.9 of the Saxon XSLT and XQuery Processor from Saxonica Limited (http://www.saxonica.com/) which are available from SourceForge (http://sourceforge.net/projects/saxon/). The Original Code of Saxon comprises all those components which are not explicitly attributed to other parties. The Initial Developer of the Original Code is Michael Kay. Until February 2001 Michael Kay was an employee of International Computers Limited (now part of Fujitsu Limited), and original code developed during that time was released under this license by permission from International Computers Limited. From February 2001 until February 2004 Michael Kay was an employee of Software AG, and code developed during that time was released under this license by permission from Software AG, acting as a "Contributor". Subsequent code has been developed by Saxonica Limited, of which Michael Kay is a Director, again acting as a "Contributor". A small number of modules, or enhancements to modules, have been developed by other individuals (either written especially for Saxon, or incorporated into Saxon having initially been released as part of another open source product). Such contributions are acknowledged individually in comments attached to the relevant code modules. All Rights Reserved. The contents of the Saxon files 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 can also be found in the installation directory, in the c:/OpenEdge/licenses folder. 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.

OpenEdge includes XML Tools, which includes Xs3P v1.1.3. The contents of this file are subject to the DSTC Public License (DPL) Version 1.1 (the "License"); you may not use this file except in compliance with the License. A copy of the license can be found in the installation directory, in the c:/OpenEdge/licenses folder. 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 xs3p. The Initial Developer of the Original Code is DSTC. Portions created by DSTC are Copyright © 2001, 2002 DSTC Pty Ltd. All rights reserved.

OpenEdge includes YAJL software Copyright 2007, Lloyd Hilaiel. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of Lloyd Hilaiel nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE AUTHOR `AS IS' AND ANY EXPRESS OR IMPLIED
WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT,
INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
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
- Support for TCP/IP version 6
- Support for enabling replication on a database that is also enabled for transparent data encryption
- Support for enabling transparent data encryption on a database that is also enabled for replication
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](image)
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, logical structure, and some of the same startup arguments 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 3–23. 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.

**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.repl.properties` file and the `target.repl.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.

• The values for the following database startup parameters: Lock Table Entries (-L), Number of Users (-n), Maximum Clients per Server (-Ma), and Maximum Servers (-Mn). For details about any of these parameters, see OpenEdge Data Management: Database Administration.

Once OpenEdge Replication is configured and running, it propagates any source database changes to the target database. For OpenEdge Replication to function properly, it is important that you 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.

The OpenEdge Replication server:

- Must reside on the source database machine.
- 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.

The OpenEdge Replication agent performs the actual process of updating the target database. The agent:

- Must reside on the target database machine.
- 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 1–14. For information about what your OpenEdge Replication installation supports, see the “Database access once OpenEdge Replication is running” section on page 3–23.

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.

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.repl.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.

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 acknowledgment 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.

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 blocks (waits) 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 3–34.

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

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 re-establishing a connection, the OpenEdge Replication server and the OpenEdge Replication agent must perform synchronization.

The sections that follow describe synchronization, server failure recovery, and agent failure recovery, including transition.

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 failure recovery operation is unsuccessful, the OpenEdge Replication server removes the failed agent from its list of OpenEdge Replication agents. If all OpenEdge Replication agents fail and connection cannot be established, the OpenEdge Replication server stops. 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 a set length of time (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 continues 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."

Using the passphrase for encrypted databases

If your replication-enabled database is also enabled for transparent data encryption and configured for manual start, you must specify -Passphrase every time the database is opened. Supplying the passphrase authenticates the key store.

For more information about transparent data encryption in general, see OpenEdge Data Management: Database Administration.
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 3–23.

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 writes a message to the target database log indicating that there is a schema lock requested.

By default, this lock is 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 blocks 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 blocks until the target quiet point is disabled.

Online backup of the target database

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]
```

Where:

- **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 portable hard drive or a USB memory stick. 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: Database 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, a BI stall, an 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.
This chapter describes how to plan for OpenEdge Replication, as outlined in the following sections:

- Preliminary planning tasks
- OpenEdge Replication and transparent data encryption
- 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

For details about configuring replication, see Chapter 3, “Implementing OpenEdge Replication.”
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, and note whether the database is encrypted. There are some additional factors to consider when you enable Replication on an encrypted database, as described in the “OpenEdge Replication and transparent data encryption” section on page 2–3.

- **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 transparent data encryption

OpenEdge allows you to enable transparent data encryption for a database that is enabled for replication, and also enable replication for a database that is enabled for transparent data encryption.

For replication and encryption to work together smoothly, the following requirements exist:

- Transparent data encryption must be enabled for both the source and target databases.
  
  The enablement of transparent data encryption on the target database is driven by the source database, either by the normal replication process or by your creating or recreating the target database from a transparent data encryption-enabled source database.

- You cannot enable before-image encryption on the target database without recreating the target database from the source database.

- Before-image encryption is not automatically enabled when transparent data encryption is enabled online for a source database; however, it is automatically enabled when transparent data encryption is enabled offline.

- After-image encryption is automatically enabled for a source database when transparent data encryption is enabled either online or offline. If after-imaging is active on the target, after-image encryption will automatically be enabled on the target by the Replication agent when transparent data encryption is enabled.

- The before-image and after-image encryption policies can differ between the source and the target databases. For example, the source can be enabled for before-image encryption, but the target is not required to be enabled.

Databases enabled for both replication and encryption

The interaction between replication and encryption on a database that has been enabled for both is essentially transparent. However, there are some changes to the overall replication process due to the presence of encryption. Because of these changes, keep the following in mind:

- Both the source and the target databases must be enabled for encryption and replication.

- The Encryption Policy Area number for both databases must be identical.

- BI and AI encryption for either database is optional.

- If your replication-enabled database is also enabled for transparent data encryption and configured for manual start, you must specify -Passphrase every time the database is opened. Supplying the passphrase authenticates the key store.

For more information about transparent data encryption in general, see *OpenEdge Data Management: Database Administration*. 
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: Database 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 its condition before it was lost, 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

<table>
<thead>
<tr>
<th>Type</th>
<th>Benefit</th>
<th>Drawback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed-length</td>
<td><strong>Performance</strong> — 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.</td>
<td><strong>Full extent management</strong> — It is possible for multiple extents to fill within a cycle of extent management. This must be accounted for in the after-image management.</td>
</tr>
<tr>
<td>Variable-length</td>
<td><strong>Performance loss</strong> — When a new after-image block is required and there are no empty blocks, the database broker must allocate and format additional space from the operating system.</td>
<td><strong>Full disks</strong> — Extents could grow to take up all of the available disk space. <strong>Larger file management</strong> — Management of variable extents can 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.</td>
</tr>
</tbody>
</table>

### 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)

<table>
<thead>
<tr>
<th>After-image extent state</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMPTY</td>
<td>The after-image extent is empty and ready to be used by the RDBMS.</td>
</tr>
<tr>
<td>BUSY</td>
<td>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 rfut1 db-name -C aimage new command is issued.</td>
</tr>
<tr>
<td>LOCKED</td>
<td>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 rfut1 db-name -C aimage empty 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 rfut1 db-name -C aimage extract command.</td>
</tr>
</tbody>
</table>
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 takes 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)   

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

Notice the elapsed time (in bold) 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 can 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 might 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 2–6) 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 is determined, the sizing calculations described in Table 2–3 are performed.

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

<table>
<thead>
<tr>
<th>Description</th>
<th>Measurement/estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average after-image data generated in one typical work hour</td>
<td>72.5MB = average of the following:</td>
</tr>
<tr>
<td></td>
<td>• 70MB in hour 1</td>
</tr>
<tr>
<td></td>
<td>• 67MB in hour 2</td>
</tr>
<tr>
<td></td>
<td>• 79MB in hour 3</td>
</tr>
<tr>
<td></td>
<td>• 74MB in hour 4</td>
</tr>
<tr>
<td>OpenEdge Replication after-image overhead</td>
<td>50%, or 1.5 multiplier</td>
</tr>
<tr>
<td>Number of processing hours during a typical workday</td>
<td>16 hours per day</td>
</tr>
<tr>
<td>Number of processing hours during the typical workweek</td>
<td>80 (16 hours per day x 5 days)</td>
</tr>
<tr>
<td>Estimate of after-image data generated in one day</td>
<td>1,160MB, or 1.13GB</td>
</tr>
<tr>
<td></td>
<td>(72.5MB x 16 hours per day)</td>
</tr>
<tr>
<td>Estimate of after-image data generated in one workweek</td>
<td>5,800MB, or 5.66GB</td>
</tr>
<tr>
<td></td>
<td>(1,160MB x 5 days)</td>
</tr>
<tr>
<td>Estimate of after-image data generated in one hour with OpenEdge Replication overhead</td>
<td>109MB (72.5MB x 1.5)</td>
</tr>
<tr>
<td>Estimate of after-image data generated in one day with OpenEdge Replication overhead</td>
<td>1,740MB (1,160MB x 1.5)</td>
</tr>
<tr>
<td>Estimate of after-image data generated in one workweek with OpenEdge Replication overhead</td>
<td>8,700MB (1,740MB x 5 days)</td>
</tr>
<tr>
<td>Single fixed after-image extent size</td>
<td>450MB (109MB x 4 hours)</td>
</tr>
<tr>
<td>Total number of after-image extents necessary to handle one week of processing</td>
<td>19 extents (8,700MB / 450MB)</td>
</tr>
</tbody>
</table>
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)

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

\[
\begin{align*}
AID & \times AIRM = AIDR \\
AIDR & \times AIRO = AIDRO \\
AIDRO & \times \left( \text{PRD} \right) = AIDROP
\end{align*}
\]

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.
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.

\[
\begin{align*}
89\text{MB} \times 1.5 &= 133.5\text{MB} \\
133.5\text{MB} \times 1.1 &= 146.85\text{MB} \\
146.85\text{MB} \times 1 &= 146.85\text{MB}
\end{align*}
\]

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 1/1440.

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

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:


- 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 fail over 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 3–20.
• 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 3–12.

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 source 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:

1. 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.

2. While in deferred agent startup mode, the OpenEdge Replication server will wait five minutes and then attempt agent connection again.

3. 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.

**Note:** 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.

4. Once the server connects to the agent(s), startup, initialization, and synchronization are performed.

5. During the entire connection process, source database activity continues to occur but is 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 3–12.

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. The following sections describe both procedures.

Setting up the source database with an offline backup

You can prepare to run OpenEdge Replication using an offline backup of the source database to create the target database. Perform the following steps on the source database:

Step 1: Back up the source database
Step 2: Create a structure file
Step 3: Enable after-imaging
Step 4: Enable the source database for OpenEdge Replication
Step 5: Perform an incremental backup of the source database
Step 6: Where to go next

For more information about transparent data encryption in general, see OpenEdge Data Management: Database Administration.

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 $PROCFG environment variables set correctly. For more information about these variables, see OpenEdge Getting Started: Installation and Configuration.

2. Be sure the source database is shut down.

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

Step 1: 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 PROBKUP, use the following command:

```
probkup source-db-name { file-name | device-name }
```

Step 2: Create 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.)

Step 3: Enable 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 “Step 4: Enable the source database for OpenEdge Replication” section on page 3–9.

To enable AI on your source database:

1. Create a new structure file—for example, source-db-name_ai.st—and edit it to add after-imaging. Be sure to give the new structure file a different name than the one you created in the “Step 2: Create a structure file” section on page 3–8.

2. Apply source-db-name_ai.st to the source database, as shown:

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

3. Back up the database using the following command:

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

4. Begin after-imaging, as shown:

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

For more information about using the RFUTIL utility, see *OpenEdge Data Management: Database Administration.*
Step 4: Enable 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
```

Step 5: Perform an incremental backup of the source database

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

```
probkup source-db-name incremental { source-db-incrementalbackup-name | device-name }
```

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

Step 6: Where to go next

After you complete the tasks described in this section, go to the “Setting up the target database” section on page 3–12.

Setting up the source database with an online backup

You can prepare to run OpenEdge Replication using an online backup of the source database to create the target database. Perform the following steps on the source database:

Step 1: Create a structure file

Step 2: Enable the database for after-imaging

Step 3: Enable the source database for OpenEdge Replication

Step 4: Perform an online backup

Step 5: Configure the OpenEdge Replication server for deferred agent startup

Step 6: Where to go next

To enable online backups, you must 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 $DLIC, $Promsgs, $Dsrhome, and $Procfg environment variables set correctly.

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

3–9
Step 1: Create 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
```

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

Step 2: Enable the database for after-imaging

If after-imaging is enabled for the database, go to “Step 3: Enable the source database for OpenEdge Replication” section on page 3–10.

If after-imaging is not already enabled for the database, follow these steps:

1. Create a new structure file—for example, `source-db-name_ai.st`—and edit it to add after-imaging. Be sure to give the new structure file a different name than the one you created in the “Step 1: Create a structure file” section on page 3–10.

2. Apply `source-db-name_ai.st` to the source database, as shown:

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

3. Back up the database, as shown:

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

4. Begin after-imaging, as shown:

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

Step 3: Enable the source database for OpenEdge Replication

OpenEdge Replication requires that the source database be enabled before starting the database. To enable the source database, enter the following command:

```
proutil source-db-name -C enableSiteReplication source
```
Step 4: Perform an online backup

To perform an online backup, use the -REPLTargetCreation and online options, as shown:

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

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

Step 5: Configure 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]
    control-agents=agent-name
    database=source-db-name
    transition=manual
    transition-timeout=1200
    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.

Step 6: Where to go next

After you complete the tasks described in this section, go to the “Setting up the target database” section on page 3–12.
Implementing OpenEdge Replication

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. Move the source database structure file (source-db-name.st), the source backup file (source-db-name.bak), and the incremental backup file from the source database machine to the target database machine.

2. 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 you are copying from the source database to accurately describe the physical structure of the target database.

3. If the source database is encrypted, copy the source database key store (source-db-name.ks) to the target machine, rename it with the name of the target database (target-db-name.ks), and place it into the target database directory.

   The source database keystore was created when you enabled transparent data encryption on the source database. For more information, see OpenEdge Data Management: Database Administration.

4. 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.

5. If the database is offline, restore the incremental backup of your source database on the target machine. For example:

   ```
   prorest target-db-name { source-db-incrementalbackup-name | device-name }
   ```

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

6. Enable your target database for OpenEdge Replication with the following command:

   ```
   proutil target-db-name -C enableSiteReplication target
   ```
Enabling before-image encryption for a Replication-enabled target database

To enable before-image encryption on the target database, you must recreate the target database from the source database.

To enable before-image encryption on the Replication-enabled target database:

1. Shut down the source database and the target database, using the following command for each database:

   ```sh
   proshut db-name
   ```

2. Use the following command to disable replication on both databases:

   ```sh
   DSRUTIL db-name -C DisableSiteReplication { source | target }
   ```

3. If before-image encryption is not already enabled on the source database, enable it using the following command:

   ```sh
   proutil source-db-name -C enableencryption -biencryption enable
   ```

   The before image is truncated.

4. Enable the database as a Replication source database using the following command:

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

5. Back up the database, as shown:

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

6. Copy the backup volume or volumes to the target computer.

7. Copy the source database structure file to the target database directory on the secondary computer. Make any modifications necessary to the file to match the configuration of the target database.

8. Copy the source database key store \((source-db-name.ks)\) to the target machine, rename it with the name of the target database \((target-db-name.ks)\), and place it into the target database directory. (The source database key store was created when you enabled the database for transparent data encryption.)
9. Restore the target database from the source backup volume or volumes:

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

10. Enable the database as a target database using the following command:

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

11. Configure the Replication property files for both the source and target databases. To do this:

   a. Locate the sample source database property file (`source.repl.properties`) and the sample target database property file (`target.repl.properties`), found in `OpenEdge-install-dir\properties`.

   b. Copy each file to the same directory as its corresponding database, and rename it to match the name of the database (for example, use the name `sports.repl.properties` for a database named `sports`).

   c. Modify the copied property files, if necessary.

      For example, you might want to configure OpenEdge Replication for one agent or two agents, and you perform this configuration by editing the copies of the source and target property files.

12. Start both databases.
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.repl.properties` and `target.repl.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.repl.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.repl.properties`, where `db-name` is the name of your source database. By default, the property file is located in `OpenEdge-install-dir\properties`.

3. To alter the default configuration, edit the file.

   If you are on UNIX, you can use vi, emacs, or your preferred text editor to change the properties file. You can also use OpenEdge Explorer.

   If you are in Windows, you can use Notepad, OpenEdge Explorer, or 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
   - Progress Explorer online help
   - OpenEdge Management and OpenEdge Explorer: Configuration
The following is a sample source.rep1.properties file for the OpenEdge Replication server with two configured agents:

```properties
# 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.
- `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 in the source replication properties file**

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.repl.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.repl.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, your favorite text editor, or OpenEdge Explorer to change the properties file. If you are in Windows, you can use Notepad, OpenEdge Explorer, or 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
- Progress Explorer online help
- OpenEdge Management and OpenEdge Explorer: Configuration

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=repl1_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). Note that -DBService is case-sensitive. If you do not enter the case correctly, an error appears.

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, which is case-sensitive, 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:** You can start the source 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 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, which is case-sensitive, 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. 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-name | device-name }
   ```

   Be sure you delete the `db-name.repl.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, then 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**

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

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

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

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 (-RO) 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 Data 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 stops and you must follow the steps in the “Starting OpenEdge Replication” section on page 3–18 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 “Starting OpenEdge Replication” section on page 3–18 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:

   \[\text{dsrutil source-db-name} \ -C \ \text{RELWAITS}\]

2. Disable OpenEdge Replication by entering the following command:

   \[\text{dsrutil source-db-name} \ -C \ \text{DisableSiteReplication} \ \text{source}\]

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:

\[\text{dsrutil target-db-name} \ -C \ \text{TERMINATE} \ \text{agent}\]

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:

\[\text{dsrutil target-db-name} \ -C \ \text{DisableSiteReplication} \ \text{target}\]
Emptying AI extents on the source database

AI extents can fill up on the source database. If you set the parameter -aistall during startup, processing will pause if your AI extents become full. If you do not set the startup parameter -aistall, 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:

```
dsruilt 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 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 3–23 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.

![Diagram](image)

**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.
Implementing OpenEdge Replication

This continuous roll forward is shown in Figure 3–2.

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*.repl.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.
Note that if an OpenEdge Replication agent failure occurs (for example, if the `rpagent.exe` process ends abnormally), the only way to restart the agent is to restart the target database. Under normal circumstances, the broker or watchdog process dynamically backs out active transactions for a process after that process unexpectedly ends. The dynamic backout process cannot be performed on behalf of the OpenEdge Replication agent because target database updates must always be driven from the source database. Dynamic backout does, however, update the database.

In the event of an unexpected OpenEdge Replication agent failure, dynamic transaction backout does not occur and the target database is left in an inconsistent state. This could cause ABL or SQL processes accessing the database to end abnormally. After the target database is restarted and synchronization is performed, all target database access should proceed normally.
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 3–29 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, it 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 3–35 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 3–38.
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 3–35 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.repl.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.repl.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`.
Implementing OpenEdge Replication

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 3–41.

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:

<table>
<thead>
<tr>
<th>Last AI Extent processed</th>
<th>Tue Oct 18 13:33:31 2005</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIMAGE BEGIN date:</td>
<td>Tue Oct 18 13:33:31 2005</td>
</tr>
<tr>
<td>AIMAGE NEW date:</td>
<td>Tue Oct 18 13:48:55 2005</td>
</tr>
<tr>
<td><strong>After Image File Number:</strong></td>
<td>3</td>
</tr>
<tr>
<td>Completely Applied to Target:</td>
<td>No</td>
</tr>
</tbody>
</table>

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 3–39.
**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:

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

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

   ```bash
   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:

   ```bash
   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:

   ```bash
   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:

   ```bash
   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. Perform an incremental backup of the source database:

```
probkup source-db-name incremental { source-db-incremental-backup-name | device-name }
```

9. Copy the source incremental backup file to the target database using FTP or NFS.

10. Restore the source incremental backup to the target database:

```
prorest target-db-name { source-db-incrementalbackup-name | device-name }
```

11. Enable the target database for OpenEdge Replication in the target database directory using the following command:

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

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 3–21.
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:

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

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

   ```bash
   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:

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

4. Delete the file `target2-db-name.repl.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 “Starting OpenEdge Replication” section on page 3–18.
Enabling transparent data encryption for an OpenEdge Replication-enabled source database

You can enable encryption for an OpenEdge Replication source database that is already configured and running in a replicated environment with at least one target database. The following procedures provide the steps you follow, first for an offline database and then for an online database.

For additional details about the Encryption Policy Area, the encryption command and passphrases, and encryption policies in general, see OpenEdge Data Management: Database Administration.

To enable encryption for a Replication-enabled source database that is offline:

1. Add the Encryption Policy Area to both the source and target databases.
2. Enable encryption using the following command:

   proutil source-db-name -C enableEncryption [-Cipher cipher-number] [-Autostart admin | user]

3. Update the Encryption Policy as required using the following command:

   dbutil db-name -C epolicy manage . . .

   Note that entering the command as shown above, without supplying any arguments, generates an error.

4. Copy the source database key store \((source-db-name.ks)\) to the target machine and place it into the target database directory. (The source database key store was created when you enabled the database for transparent data encryption.)

   Encryption is now enabled for the source database and will be enabled on the target database once both databases are started and replication activity resumes.

To enable encryption for a Replication-enabled source database that is online:

1. Add the Encryption Policy Area to both the source database and target databases.
2. Enable encryption using the following command:

   proutil source-db-name -C enableEncryption [-Cipher cipher-name] [-Autostart admin | user]
3. Update the Encryption Policy as required using the following command:

```
  dbutil db-name -C epolicy manage ...
```

Note that entering the command as shown above, without supplying any arguments, generates an error.

4. Copy the source directory key store (\texttt{source-db-name.ks}) to the target machine and place it into the target database directory. (The source database key store was created when you enabled the database for transparent data encryption.)

Once encryption is enabled on the source database and the replication process resumes, the Replication agent performs various tasks that automatically enable encryption for the target database based on the specific encryption after-image notes it processes. During this processing, the Replication agent stops reading TCP/IP messages from the Replication server, which means that the Replication server is no longer reading after-image blocks.

This pause could result in the interruption of database updates on the source database. Source database interruption can be minimized or eliminated by copying the source database key store—\texttt{db-name.ks}—to the target database directory immediately after encryption is enabled on the source database.
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

Before you begin working with the failback process, it is important to become familiar with 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. The source and target databases also need to have the settings for the following startup arguments in common: Lock Table Entries (-L), Number of Users (-n), Maximum Clients per Server (-Ma), and Maximum Servers (-Mn).
- 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.repl.properties file and the target.repl.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.repl.properties and target-db-name.repl.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

The availability of failback for OpenEdge Replication increases both the scope and the functionality of the transition process 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 answers to the following questions:

- 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 perform automated processing. Some of the automated processing is controlled by 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 (automatically or manually), 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. The primary database must be restored from a backup of the secondary database and then enabled as a target database before secondary replication can begin.

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.

Step 2: Primary machine failure

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

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.

![Diagram of OpenEdge Replication](image)

**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:

   ```bash
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:

   ```bash
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.

![Diagram of transition process]

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 “Step 2: Create a structure file” section on page 3–8.

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](image)

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.

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:

```
dsrunutil 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.

**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:

```
dsrtutil 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.
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: Continuing transition of the primary database](image)

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.
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.

![Secondary replication continues (before controlled transition)](image)

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 doing one of the following:
   - Examining the database log file
   - Using the following command:

   ```
dsutil 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:

   ```
dsutil 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\_lg, is renamed database\_lg\_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\_lg\_save, is renamed database\_lg.
  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
13:29:28 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 rprepl[.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.repl.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
```

Note that `-DBService` is case-sensitive. If you do not enter the case correctly, an error appears.

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

<table>
<thead>
<tr>
<th>To implement this transition planning decision . . .</th>
<th>Do this . . .</th>
</tr>
</thead>
</table>
| Which is the primary and which is the secondary database? | Modify the `responsibility` property (currently used for informational purposes only) to identify the type of database. Possible values are:  
  • **Primary** — This database is the primary database.  
  • **Secondary** — This database is the secondary database. |
| What type of transition will occur? | Modify the `database-role` property to identify the new role of the database once it is transitioned. Possible values are:  
  • **Reverse** — The role of the database is reversed: A source database becomes a target database, and a target database becomes a source database.  
  • **Normal** — 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. |
| What type of after-imaging operations should transition perform? | Modify the after-imaging properties `auto-begin-ai`, `auto-add-ai-areas`, and `ai-structure-file` to set AI behavior.  
  Specify either of these values for the `auto-begin-ai` property:  
  • **0** — Do not begin AI automatically after a target-to-source transition.  
  • **1** — Begin AI automatically after a target-to-source transition.  
  Specify either of these values for the `auto-add-ai-areas` property:  
  • **0** — 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).  
  • **1** — Add AI areas to the database automatically by using the structure file specified in the `ai-structure-file` property (when a database is transitioned to a source database or there are currently no AI areas for the database).  
  Specify the name of the structure file (which contains the list of AI areas to add) in the `ai-structure-file` property. |
What types of backup do you want to perform during transition?

Modify the backup properties `backup-method`, `backup-arguments`, and `incremental-backup-arguments` to set the type of backup you want to perform during transition and before AI is enabled.

Specify one of these values for the `backup-method` property:

- **mark** — Mark the database as backed up by using the following command:
  
  ```
  rfutil db-name -C mark backedup
  ```

  Marking the database as backed up does not allow future AI extents to be used when recovering from a disaster.

- **full-offline** — Back up the database offline by using the OpenEdge 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.

- **full-online** — Back up the database online after the database is restarted, which occurs after the database has been transitioned.

Specify the `backup-arguments` property for the arguments required for the full online and offline backups performed for the database. Keep the following in mind:

- You must specify, at a minimum, the target file or device 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.

- Do not use backup validation parameters (such as `-vp` and `-vf`).

- Begin the arguments with `device-name`.

Specify the `incremental-backup-arguments` 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 `device-name`.

---

Table 4–1: Setting transition properties

<table>
<thead>
<tr>
<th>To implement this transition planning decision . . .</th>
<th>Do this . . .</th>
</tr>
</thead>
<tbody>
<tr>
<td>What types of backup do you want to perform during transition?</td>
<td>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. Specify one of these values for the <code>backup-method</code> property:</td>
</tr>
<tr>
<td></td>
<td>- <strong>mark</strong> — Mark the database as backed up by using the following command:</td>
</tr>
</tbody>
</table>
| | ```
| | rfutil db-name -C mark backedup
| | ``` |
| | Marking the database as backed up does not allow future AI extents to be used when recovering from a disaster. |
| | - **full-offline** — Back up the database offline by using the OpenEdge 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. |
| | - **full-online** — Back up the database online after the database is restarted, which occurs after the database has been transitioned. |
| | Specify the `backup-arguments` property for the arguments required for the full online and offline backups performed for the database. Keep the following in mind: |
| | - You must specify, at a minimum, the target file or device 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. |
| | - Do not use backup validation parameters (such as `-vp` and `-vf`). |
| | - Begin the arguments with `device-name`. |
| | Specify the `incremental-backup-arguments` 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 `device-name`. |
Modify the following automatic transition properties to specify whether the database should be automatically restarted after transition:

- **restart-after-transition**
- **source-startup-arguments**, **target-startup-arguments**, or **normal-startup-arguments**

Specify either of these values for the **restart-after-transition** property:

- **0** — Do not automatically restart the database after transition is performed.
- **1** — Automatically restart the database after transition is performed. When you specify this value, you must also supply the ***-startup-arguments** properties, or the database startup will fail.

Specify the ***-startup-arguments** property or properties:

- If the database is transitioned to a normal database, specify the **normal-startup-arguments** 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 `-pf` followed by a parameter file name, as shown here:
  
  `-pf db-name.normal.pf`

- If the database is transitioned to a source database, you must specify the **source-startup-arguments** 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 `-pf` followed by a parameter file name, as shown here:
  
  `-pf db-name.source.pf`

Because the database is a source, you must also specify the following argument, which is case-sensitive, as an indication to the broker to start the Replication server:

`-DBService replserv`
Do you want the database to restart automatically after transition?

(continued)

- If the database is transitioned to a target database, you must specify the `target-startup-arguments` property. The arguments are appended to the PROSERVE command and are used when the database is started. In most cases, the only argument specified here should be `-pf` followed by a parameter file name, as shown here:

  -pf db-name.target.pf

  Because the database is a target, you must also specify the following arguments (note that `-DBService` is case-sensitive) as an indication to the broker to start the Replication agent and listen on the TCP/IP port as specified:

  -DBService replagent -S { port-number | service-name }

- If the database role is reversed, you must specify the `source-startup-arguments` property and the `target-startup-arguments` property.

For additional details about the source-, target-, or normal-startup-arguments, see the “Sample startup parameter file” section on page 4–38.

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

- Modify the `recovery-backup-arguments` property, which determines whether transition automatically attempts a recovery.

  Specify the device name, followed by any other backup arguments.
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
```

Note that `-DBService` is case-sensitive. If you do not enter the case correctly, an error appears.

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
```

Note that `-DBService` is case-sensitive. If you do not enter the case correctly, an error appears.

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.
The sections that follow 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>
<thead>
<tr>
<th>Database type</th>
<th>Database state</th>
<th>Online or offline</th>
<th>Description of transition performed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Database is newly enabled</td>
<td>Offline</td>
<td>An error occurs because the database is newly enabled.</td>
</tr>
<tr>
<td>Source</td>
<td>Database is newly enabled</td>
<td>Online</td>
<td>An error occurs because the database is online and has never been replicated.</td>
</tr>
<tr>
<td>Source</td>
<td>Database has been replicated</td>
<td>Offline</td>
<td>The database is transitioned as instructed in the <code>rep1.properties</code> file.</td>
</tr>
<tr>
<td>Source</td>
<td>Database has been replicated</td>
<td>Online</td>
<td>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.</td>
</tr>
<tr>
<td>Target</td>
<td>Database has been replicated</td>
<td>Offline</td>
<td>In order to transition a target database in this state, the database cannot be newly transitioned.</td>
</tr>
<tr>
<td>Target</td>
<td>Database has been replicated</td>
<td>Online</td>
<td>In order to transition a target database in this state, the following conditions must be met:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• The Replication agent cannot be communicating with the Replication server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• The Replication agent must be in pretransition.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• All source AI extents must be applied to the target database by using <code>dsrutil db-name -C applyExtent</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>You must explicitly perform this step, possibly multiple times depending on the number of source AI extents.</td>
</tr>
<tr>
<td>Normal</td>
<td>–</td>
<td>Online Offline</td>
<td>An error occurs because the database is not enabled for replication.</td>
</tr>
</tbody>
</table>
## 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

<table>
<thead>
<tr>
<th>Property name</th>
<th>Type and length</th>
<th>Description</th>
</tr>
</thead>
</table>
| database-role             | character[15]   | The new role of the database once it is transitioned. The possible values for this property are as follows:  
  - **reverse** — The role of the database is reversed: A source database becomes a target database, and a target database becomes a source database.  
  - **normal** — 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. |
| responsibility            | character[15]   | This property must contain one of the following values:  
  - **primary** — This database is the primary database.  
  - **secondary** — This database is the secondary database.  
  This property is currently informational. |
| restart-after-transition  | integer         | The database can be automatically restarted after transition is performed.  
  Valid values for the property are 0 and 1. When the property is set to 1, the following *-startup-arguments properties must be supplied, or the database startup will fail:  
  - If the database role is normal, you must specify the `normal-startup-arguments`.  
  - If the database role is reversed, you must specify the `source-startup-arguments` and the `target-startup-arguments`. |
| source-startup-arguments  | character[256]  | 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 PROSERVE command (used to start the database).  
  In most cases, the only argument specified here should be `-pf` followed by a parameter file name. For example:  
  
  `-pf db-name.source pf`  
  Because the database is a source, you must also specify the following argument, which is case-sensitive, as an indication to the broker to start the replication server:  
  
  `-DBService replserv` |
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. In most cases, the only argument specified here should be `-pf` followed by a parameter file name. For example:

```
-pf db-name.target.pf
```

Because the database is a target, you must also specify the following arguments (note that `-DBService` is case-sensitive), as an indication to the broker to start the replication agent and to listen on the TCP/IP port specified with `-S`:

```
-DBService replagent -S { port-number | service-name }
```

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. In most cases, the only argument specified here should be `-pf` followed by a parameter file name. For example:

```
-pf db-name.normal.pf
```

Specify 1 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 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. 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:

```
agent1,agent2
```

This property instructs the transition process to automatically add AI areas to the database if:

- It is transitioned to a source database.
- There are currently no AI areas for the database.

If the value specified for the property is 0, AI areas are not added to the database. If the value specified is 1, AI areas are automatically added to the database by using the structure file specified in the `ai-structure-file` property.

Name of the structure file that contains the list of AI areas to add.
backup-method

Specifies the backup method performed before AI is enabled. The following values are valid:

- **mark** — Mark the database backed up by using the following command:
  
  \[ rftutil \ db-name \ -C \ mark \ backup \]

  Marking the database as backed up does not allow future AI extents to be used when recovering from a disaster.

- **full-offline** — The database will be backed up offline by using the \texttt{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.

- **full-online** — An online backup is performed after the database is restarted, which occurs after the database has been transitioned.

backup-arguments

The arguments required for the full online and offline backups that are performed for the database.

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.

Do not use backup validation parameters (such as \texttt{-vp} and \texttt{-vf}).

The arguments should begin with \texttt{device-name}.

incremental-backup-arguments

The arguments required for the offline incremental backup performed after AI is enabled and the database’s role is reversed.

The arguments should begin with \texttt{device-name}.

recovery-backup-arguments

The arguments required for the full online and offline backups that are performed for the database.

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.

Do not use backup validation parameters (such as \texttt{-vp} and \texttt{-vf}).

The arguments should begin with \texttt{device-name}.
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 Progress Explorer, dbman, or OpenEdge Explorer
• 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.repl.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:

```plaintext
[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 `<OpenEdge-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:

```plaintext
[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:

```plaintext
[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

<table>
<thead>
<tr>
<th>Property name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>control-agents</td>
<td>agent_name[, agent_name]</td>
<td>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.</td>
</tr>
<tr>
<td>database</td>
<td>db_name</td>
<td>Specifies the source database name.</td>
</tr>
<tr>
<td>defer-agent-startup</td>
<td>minutes</td>
<td>Specifies how long, in minutes, the server attempts to connect to an agent if the first connection attempt is unsuccessful. minutes is a value greater than or equal to 0 and less than or equal to 10080.</td>
</tr>
<tr>
<td>maximum-polling-delay</td>
<td>milliseconds</td>
<td>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. milliseconds is a value greater than 500 and less than 1000.</td>
</tr>
<tr>
<td>minimum-polling-delay</td>
<td>milliseconds</td>
<td>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. milliseconds is a value greater than or equal to 1 and less than or equal to 10.</td>
</tr>
</tbody>
</table>
Specifications for server properties include:

<table>
<thead>
<tr>
<th>Property name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>repl-Keep-Alive</td>
<td>seconds</td>
<td>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. By default, this property is enabled and has a value of 300 seconds. The minimum value is 60 seconds; there is no maximum value.</td>
</tr>
</tbody>
</table>
| schema-Lock-Action             | wait | force | Specifies the action an agent takes if an exclusive schema lock is not granted. Possible actions are:
- **wait** — The agent waits until the exclusive schema lock is granted. The server blocks until the exclusive schema lock is granted.
- **force** — 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.
**Note:** Values are case-sensitive. Use `wait` or `force`, but not `Wait` or `Force`.
| agent-shutdown-action          | recovery | normal | Specifies the action an agent takes during a shutdown when the replication server ends. Possible actions are:
- **recovery** — The agent will remain active but in a standby state waiting for the replication server to reconnect.
- **normal** — The agent will terminate; the target database will stay up. |

Table 5–1: Server properties (2 of 3)
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

<table>
<thead>
<tr>
<th>Property name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connect-timeout</td>
<td><em>seconds</em></td>
<td>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. Specifies how many seconds the Replication agent waits for connection from the OpenEdge Replication server before the Replication agent shuts itself down. <em>seconds</em> is an integer greater than or equal to 120 and less than or equal to 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 has elapsed, the OpenEdge Replication agent will terminate and allow some limited system-level target database connections.</td>
</tr>
</tbody>
</table>
| critical        | 0 | 1 | Specifies whether the agent is critical:  
- 1 is critical.  
- 0 is the default value of noncritical.  
A critical agent is an asynchronous agent for the target database that can become the source database if the source database becomes unavailable. |
| database        | *db-name* | Specifies the target database name. |
| host            | *IP-address* | *hostname* | Specifies to the server which host the agent will start on. |
| ipver           | *ipv4* | *ipv6* | 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            | *agent-name* | Specifies the OpenEdge Replication agent name. By convention, the *agent-name* should match the name specified in the target database properties file. This directive is used only by Progress Explorer. ALL is not allowed as an agent name. Each agent must have a unique name. |
| port            | *service-name* | *port-num* | 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* | Specifies the type of replication, either asynchronous (*async*) or synchronous (*sync*).  
**Note:** Values are case-sensitive. Use *async* or *sync*, but not *Async* or *Sync*. |
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

<table>
<thead>
<tr>
<th>Property name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connect-timeout</td>
<td>seconds</td>
<td>Specifies how many seconds the OpenEdge Replication agent will wait for connection from the OpenEdge Replication server before the replication agent shuts itself down. seconds is an integer greater than or equal to 120 and less than or equal to 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.</td>
</tr>
<tr>
<td>database</td>
<td>db_name</td>
<td>Specifies the source database name.</td>
</tr>
<tr>
<td>host</td>
<td>IP-address</td>
<td>Specifies to the server which host the agent will start on.</td>
</tr>
<tr>
<td></td>
<td>hostname</td>
<td></td>
</tr>
<tr>
<td>ipver</td>
<td>ipv4</td>
<td>ipv6</td>
</tr>
<tr>
<td>listener-maxport</td>
<td>port-number</td>
<td>Specifies the maximum TCP port number for the agent. port-number must be greater than the value of listener-minport+1 and less than the maximum allowable port number on the system. For UNIX the maximum port number is 65534. For Windows the maximum port number is 32765.</td>
</tr>
<tr>
<td>listener-minport</td>
<td>port-number</td>
<td>Specifies the minimum TCP port number. The agent selects a port in a range between the values specified by listener-minport and listener-maxport. port-number must be greater than 1024 and less than the value of listener-maxport.</td>
</tr>
<tr>
<td>Property name</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>maximum-polling-delay</td>
<td>milliseconds</td>
<td>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. milliseconds is a value greater than 500 and less than 1000.</td>
</tr>
<tr>
<td>minimum-polling-delay</td>
<td>milliseconds</td>
<td>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. milliseconds is a value greater than or equal to 1 and less than or equal to 10.</td>
</tr>
<tr>
<td>name</td>
<td>agent_name</td>
<td>Specifies the OpenEdge Replication agent name. By convention, the agent-name should match the name specified in the source database properties file. This directive is used only by Progress Explorer. ALL is not allowed as an agent name. Each agent must have a unique name.</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>DSRUTIL db-name -C ACTION [Server</td>
</tr>
<tr>
<td>Windows</td>
<td>DSRUTIL db-name -C ACTION [Server</td>
</tr>
</tbody>
</table>

$db$-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.

-Passphrase

For encryption-enabled databases only, specifies to prompt for a passphrase to authenticate the key store.

Caution: If your database is enabled for transparent data encryption and configured for manual start, you must specify -Passphrase every time the database is opened.

For more information on encryption and key store authentication, see OpenEdge Data Management: Database Administration.
The command actions and modifiers you can use with the DSRUTIL utility are described in the following sections.

<table>
<thead>
<tr>
<th>Qualifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyextent</td>
<td>Allows you to apply AI extents generated by the source directly to the target database.</td>
</tr>
<tr>
<td>canceldefer</td>
<td>Instructs the OpenEdge Replication server to stop attempting to reconnect.</td>
</tr>
<tr>
<td>connectagent</td>
<td>Instructs the OpenEdge Replication server to start one or both of its configured agents.</td>
</tr>
<tr>
<td>disablesitereplication</td>
<td>Allows you to disable OpenEdge Replication while the source database is online.</td>
</tr>
<tr>
<td>monitor</td>
<td>Displays a PROMON-type series of screens that show the current state of replication.</td>
</tr>
<tr>
<td>recovery</td>
<td>Displays the replication recovery information.</td>
</tr>
<tr>
<td>relwaits</td>
<td>Frees up any pending waits that might be outstanding so that database activity can continue.</td>
</tr>
<tr>
<td>restart</td>
<td>Restarts the OpenEdge Replication server.</td>
</tr>
<tr>
<td>startagent</td>
<td>See connectagent.</td>
</tr>
<tr>
<td>terminate</td>
<td>Terminates the currently running OpenEdge Replication server or agent.</td>
</tr>
<tr>
<td>transition</td>
<td>Instructs an OpenEdge Replication agent to transition a replication-enabled database.</td>
</tr>
<tr>
<td>triggertransition</td>
<td>Forces the target database to go into a pre-transition state.</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td><code>dsrutil db-name -C applyextent extent-name</code></td>
</tr>
<tr>
<td>Windows</td>
<td><code>dsrutil db-name -C recovery Agent</code></td>
</tr>
<tr>
<td></td>
<td><code>dsrutil db-name -C applyextent extent-name</code></td>
</tr>
</tbody>
</table>

`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:

```
rutil source-db-name -C aimage list
```

Executing this command produces the following output:

<table>
<thead>
<tr>
<th>Extent:</th>
<th>Status:</th>
<th>Type:</th>
<th>Path:</th>
<th>Size:</th>
<th>Used:</th>
<th>Start:</th>
<th>Seqno:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Full</td>
<td>Variable Length</td>
<td>/vobs_repl/solaris/bin/ks1.a1</td>
<td>120</td>
<td>1</td>
<td>Wed Oct 26 08:32:12 2005</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Full</td>
<td>Variable Length</td>
<td>/vobs_repl/solaris/bin/ks1.a2</td>
<td>4728</td>
<td>4534</td>
<td>Wed Oct 26 08:32:14 2005</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Busy</td>
<td>Variable Length</td>
<td>/vobs_repl/solaris/bin/ks1.a3</td>
<td>4728</td>
<td>4456</td>
<td>Wed Oct 26 08:33:29 2005</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Empty</td>
<td>Variable Length</td>
<td>/vobs_repl/solaris/bin/ks1.a4</td>
<td>120</td>
<td>0</td>
<td>N/A</td>
<td>0</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX Windows</td>
<td><code>dsrutil db-name -C canceldefer server</code></td>
</tr>
</tbody>
</table>

`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 connects 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

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX Windows</td>
<td>dsrutil db-name -C connectagent database { name</td>
</tr>
</tbody>
</table>

$db$-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

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>dsrutil db-name -C disablesitereplication { Source</td>
</tr>
<tr>
<td>Windows</td>
<td></td>
</tr>
</tbody>
</table>

$db-name$

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

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>dsrutil db-name -C monitor</td>
</tr>
<tr>
<td>Windows</td>
<td></td>
</tr>
</tbody>
</table>

$db$-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.

For additional information, see the “OpenEdge Replication DSRUTIL MONITOR” section on page 5–31.
DSRUTIL recovery qualifier

Displays the replication recovery information.

Syntax

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX Windows</td>
<td>dsrutil db-name -C recovery { server</td>
</tr>
</tbody>
</table>

**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 08:32:08 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:

```
<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replication version</td>
<td>4.0</td>
</tr>
<tr>
<td>Date created</td>
<td>Wed Oct 26 08:32:57 2005</td>
</tr>
<tr>
<td>Date last written</td>
<td>Wed Oct 26 08:34:36 2005</td>
</tr>
<tr>
<td>Replication local agent information:</td>
<td></td>
</tr>
<tr>
<td>Last Block</td>
<td>Complete</td>
</tr>
<tr>
<td>Last block received location</td>
<td>area: 14, seq: 3, loc: 4554752, offset: 0</td>
</tr>
<tr>
<td>Last block processed location</td>
<td>area: 0, seq: 0, loc: 0, offset: 0</td>
</tr>
<tr>
<td>Last block ACKed location</td>
<td>area: 14, seq: 3, loc: 4259840, offset: 3189</td>
</tr>
<tr>
<td>Last block received</td>
<td>no date</td>
</tr>
<tr>
<td>Last block ACKed</td>
<td>no date</td>
</tr>
<tr>
<td>ID of the last TX begin</td>
<td>4345</td>
</tr>
<tr>
<td>ID of the last TX end</td>
<td>4345</td>
</tr>
<tr>
<td>Time of last TX end</td>
<td>Wed Oct 26 08:33:46 2005</td>
</tr>
<tr>
<td>Last AI Extent processed</td>
<td></td>
</tr>
<tr>
<td>AIMAGE BEGIN date</td>
<td>Wed Oct 26 08:32:12 2005</td>
</tr>
<tr>
<td>AIMAGE NEW date</td>
<td>Wed Oct 26 08:33:29 2005</td>
</tr>
<tr>
<td>After-Image File Number</td>
<td>3</td>
</tr>
<tr>
<td>File Last Opened</td>
<td>Wed Oct 26 08:33:29 2005</td>
</tr>
<tr>
<td>Completely Applied to Target</td>
<td>No</td>
</tr>
</tbody>
</table>
```

Note that the **After-Image File Number** is the same number that is returned as **Seqno** by the following command:

```
rutil source-db-name -C ailmage list
```
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

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>dsrutil db-name -C relwaits</td>
</tr>
<tr>
<td>Windows</td>
<td>dsrutil db-name -C relwaits</td>
</tr>
</tbody>
</table>

$db-name$

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.

**Note:** Do not enable a quiet point on the source database before you restart the Replication server.

### Syntax

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>dsrutil db-name -C restart server</td>
</tr>
<tr>
<td>Windows</td>
<td></td>
</tr>
</tbody>
</table>

*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**

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td><code>dsrutil db-name -C status agentname -detail</code></td>
</tr>
<tr>
<td>Windows</td>
<td></td>
</tr>
</tbody>
</table>

`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**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The command completed normally, and the status code was sent to stdout.</td>
</tr>
<tr>
<td>2</td>
<td>There was a generic database open error.</td>
</tr>
<tr>
<td>3</td>
<td>The database was opened, but it is not enabled for replication.</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>If the status code is . . .</th>
<th>It reflects the status for . . .</th>
</tr>
</thead>
<tbody>
<tr>
<td>1xxx</td>
<td>The server</td>
</tr>
<tr>
<td>2xxx</td>
<td>The control agent</td>
</tr>
<tr>
<td>3xxx</td>
<td>The agent</td>
</tr>
</tbody>
</table>

See Table 5–7 for a list and description of the possible status code values. A query on the source produces a resulting value that is 1.xxx or 6.xxx. A query on the target produces a value that is 2.xxx or 3.xxx.

Table 5–7: Status code values

<table>
<thead>
<tr>
<th>Status</th>
<th>Class</th>
<th>Detail status</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Connecting.</td>
<td>Initial connection.</td>
<td>1001</td>
</tr>
<tr>
<td>100</td>
<td>Connecting.</td>
<td>Initializing.</td>
<td>1002</td>
</tr>
<tr>
<td>100</td>
<td>Connecting.</td>
<td>Server Initialization.</td>
<td>6001</td>
</tr>
<tr>
<td>100</td>
<td>Connecting.</td>
<td>Connecting to Agents.</td>
<td>6002</td>
</tr>
<tr>
<td>100</td>
<td>Connecting.</td>
<td>Configuring Agent(s).</td>
<td>6003</td>
</tr>
<tr>
<td>101</td>
<td>Processing is taking place.</td>
<td>Startup Synchronization.</td>
<td>3048</td>
</tr>
<tr>
<td>101</td>
<td>Processing is taking place.</td>
<td>Normal Processing.</td>
<td>3049</td>
</tr>
<tr>
<td>101</td>
<td>Processing is taking place.</td>
<td>Recovery Synchronization.</td>
<td>3050</td>
</tr>
<tr>
<td>101</td>
<td>Processing is taking place.</td>
<td>Recovery Processing.</td>
<td>6004</td>
</tr>
<tr>
<td>101</td>
<td>Processing is taking place.</td>
<td>Startup Synchronization.</td>
<td>6005</td>
</tr>
<tr>
<td>101</td>
<td>Processing is taking place.</td>
<td>Normal Processing.</td>
<td>6021</td>
</tr>
<tr>
<td>102</td>
<td>Activity is halted.</td>
<td>Online backup of the Target Database.</td>
<td>3051</td>
</tr>
<tr>
<td>102</td>
<td>Activity is halted.</td>
<td>Target Database in Quiet Point.</td>
<td>3052</td>
</tr>
<tr>
<td>102</td>
<td>Activity is halted.</td>
<td>Target Database is in a BI stall.</td>
<td>3053</td>
</tr>
<tr>
<td>102</td>
<td>Activity is halted.</td>
<td>Target Database is in an AI stall.</td>
<td>3054</td>
</tr>
</tbody>
</table>
Table 5-7: Status code values  

<table>
<thead>
<tr>
<th>Status</th>
<th>Class</th>
<th>Detail status</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>103</td>
<td>Pre-transition state.</td>
<td>Pre-transition.</td>
<td>2080</td>
</tr>
<tr>
<td>103</td>
<td>Pre-transition state.</td>
<td>Applying After-image Extent.</td>
<td>2081</td>
</tr>
<tr>
<td>104</td>
<td>Agent is listening.</td>
<td>Listening.</td>
<td>2083</td>
</tr>
<tr>
<td>105</td>
<td>Transition.</td>
<td>Transitioning.</td>
<td>2082</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Initial Connection Failed.</td>
<td>1032</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Recovery Failed.</td>
<td>1033</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Invalid Target Database Configuration.</td>
<td>1034</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Agent Failed.</td>
<td>1035</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Agent is Ignored.</td>
<td>1036</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Agent is Stopped.</td>
<td>1037</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Agent is Terminated.</td>
<td>1038</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Agent is Ended.</td>
<td>1063</td>
</tr>
<tr>
<td>199</td>
<td>Inactive.</td>
<td>Server is ended.</td>
<td>6060</td>
</tr>
<tr>
<td>255</td>
<td>Unknown.</td>
<td>Unknown.</td>
<td>0</td>
</tr>
</tbody>
</table>

For example, for a source database whose Replication server is connecting to its configured agents, the status returned would be 1001.
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 down the database.

Syntax

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>dsrutil db-name -C { terminate server</td>
</tr>
<tr>
<td>Windows</td>
<td></td>
</tr>
</tbody>
</table>

$db-name$

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 any of the following situations:

- To force a transition when the server is in contact with the agent
- To perform a manual transition if the server and agent have lost connection
- To initiate transition of a Replication-enabled database that is also enabled for encryption and configured for manual startup

Syntax

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>`dsrutil db-name -C transition [failover] [-Passphrase] [-logging [1</td>
</tr>
<tr>
<td>Windows</td>
<td></td>
</tr>
</tbody>
</table>

`db-name`

The name of the source database to perform the action on.

Passphrase

The transition of a database from one role to another requires that many different operations be performed, which could include adding after-image extents, backing up the database, or starting up or shutting down the database. The passphrase is required when these operations are performed for a manual startup database that you have enabled with transparent data encryption.

`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 `rprepl 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

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>dsrutil db-name -C triggertransition</td>
</tr>
<tr>
<td>Windows</td>
<td></td>
</tr>
</tbody>
</table>

$db-name$

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 monitor allows you to monitor OpenEdge Replication and provides the display options described in the following sections.

The command for the monitor is as shown:

dsrutil db-name -C monitor

Startup menu

When you use the DSRUTIL MONITOR command, the screen shown in Figure 5–1 appears.

```
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

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replication server status</td>
<td>Instructs the user to enter the letter S to display the Replication server Status display screen</td>
</tr>
<tr>
<td>Replication server remote agents</td>
<td>Instructs the user to enter the letter R to display the Replication remote agent Selection menu</td>
</tr>
<tr>
<td>Replication agent status</td>
<td>Instructs the user to enter the letter A to display the Replication local agent Status display screen</td>
</tr>
<tr>
<td>Modify display defaults</td>
<td>Instructs the user to enter the letter M to display the current utility display settings and prompts the user for any desired changes</td>
</tr>
<tr>
<td>Quit</td>
<td>Instructs the user to enter the letter Q to exit the utility</td>
</tr>
<tr>
<td>Enter your selection</td>
<td>Instructs the user to enter a selection for the action the utility is to perform</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>OpenEdge Replication Monitor</th>
<th>Page 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database: /vobs_repl/solaris/bin/ks1</td>
<td></td>
</tr>
<tr>
<td>Database is enabled as OpenEdge Replication: Source</td>
<td></td>
</tr>
<tr>
<td>Server is: Connecting to Agent(s)</td>
<td></td>
</tr>
<tr>
<td>Number of configured agents: 1</td>
<td></td>
</tr>
<tr>
<td>Defer Agent Startup:</td>
<td></td>
</tr>
<tr>
<td>Continue connection attempts until: Tue Nov 27 01:18:27 2007</td>
<td></td>
</tr>
<tr>
<td>Deferred Agent startup will expire in: 9 Hr 58 Min 37 Sec</td>
<td></td>
</tr>
<tr>
<td>Next connection attempt in: 4 Min 41 Sec</td>
<td></td>
</tr>
<tr>
<td>Connection attempts performed: 1</td>
<td></td>
</tr>
<tr>
<td>Agent(s) currently connected: 0</td>
<td></td>
</tr>
<tr>
<td>Delay Interval (current / min / max): 5 / 5 / 500</td>
<td></td>
</tr>
<tr>
<td>Recovery information:</td>
<td></td>
</tr>
<tr>
<td>State: No recovery being performed</td>
<td></td>
</tr>
<tr>
<td>Agents needing recovery: 0</td>
<td></td>
</tr>
<tr>
<td>Agents connected: 0</td>
<td></td>
</tr>
<tr>
<td>Agents in synchronization: 0</td>
<td></td>
</tr>
<tr>
<td>Transition information:</td>
<td></td>
</tr>
<tr>
<td>Type: Manual</td>
<td></td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database is enabled as OpenEdge Replication</td>
<td>Indicates whether the database is enabled as a replication source or target.</td>
</tr>
<tr>
<td>Server is</td>
<td>Describes how the server is processing information:</td>
</tr>
<tr>
<td></td>
<td>• Normal processing — The server is processing information in the normal fashion.</td>
</tr>
<tr>
<td></td>
<td>• Performing initialization — The server is performing initialization.</td>
</tr>
<tr>
<td></td>
<td>• Performing startup synchronization — The server is in the process of synchronizing the target databases with the source database.</td>
</tr>
<tr>
<td></td>
<td>• Connection — The server connection to agent(s).</td>
</tr>
<tr>
<td></td>
<td>• Configuring connected agents — The server is performing handshaking with the agent.</td>
</tr>
<tr>
<td></td>
<td>• Performing failure recovery — The server is attempting failure recovery from a connection failure.</td>
</tr>
<tr>
<td></td>
<td>• Unknown — The server is in an unknown state.</td>
</tr>
<tr>
<td>Number of configured agents</td>
<td>Shows the number of agents currently configured to operate with the server.</td>
</tr>
<tr>
<td>Defer Agent Startup</td>
<td>Shows information related to deferred agent startup.</td>
</tr>
<tr>
<td>Continue connection attempts until</td>
<td>Shows the time, day, and date when connection attempts will stop.</td>
</tr>
<tr>
<td>Deferred Agent Startup will expire in</td>
<td>Shows the remaining duration of deferred agent startup.</td>
</tr>
<tr>
<td>Next connection attempt in</td>
<td>Shows when the next connection attempt starts.</td>
</tr>
</tbody>
</table>
Table 5–9: Replication server status

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection attempts performed</td>
<td>Shows how many connection attempts have occurred.</td>
</tr>
<tr>
<td>Agent(s) currently connected</td>
<td>Shows the number of connected agents.</td>
</tr>
<tr>
<td>Delay Interval (current / min / max)</td>
<td>Shows the amount of time, in milliseconds, the server will wait between polling the database for information to be replicated. <strong>Current</strong> is the current value, <strong>min</strong> is the minimum value, and <strong>max</strong> 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.</td>
</tr>
<tr>
<td>Recovery information</td>
<td>Shows failure-recovery-related details.</td>
</tr>
<tr>
<td>Agents needing recovery</td>
<td>Shows the number of remote agents requiring failure recovery.</td>
</tr>
<tr>
<td>Agents connected</td>
<td>Shows the number of remote agents currently connected to the server.</td>
</tr>
<tr>
<td>Agents in synchronization</td>
<td>Shows the number of remote agents in the process of being brought up to date.</td>
</tr>
<tr>
<td>Transition information</td>
<td>Shows Replication transition details.</td>
</tr>
<tr>
<td>Type</td>
<td>Shows the type of transition to be performed:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Manual</strong> — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database</td>
</tr>
<tr>
<td></td>
<td>• <strong>Automatic</strong> — Indicates a transition from a target database to a normal OpenEdge database will take place without intervention</td>
</tr>
<tr>
<td>Transition timeout limit</td>
<td>Shows the maximum amount of time that will elapse before the transition of a target database to a normal OpenEdge database. <strong>Note:</strong> This line does not appear in <strong>Figure 5–2</strong>, as the limit is shown for automatic transition only.</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Host Name</th>
<th>Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>agent1</td>
<td>localhost</td>
<td>tdba</td>
</tr>
<tr>
<td>2</td>
<td>agent2</td>
<td>localhost</td>
<td>tdbb</td>
</tr>
</tbody>
</table>

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**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database is enabled as OpenEdge Replication</td>
<td>Indicates whether the database is enabled as a replication source or target.</td>
</tr>
<tr>
<td>Remote Agents Configured</td>
<td>Displays a list of the currently configured remote agents. Each remote agent is identified by an <strong>ID</strong>, the agent’s <strong>Name</strong>, the <strong>Host Name</strong>, and the target <strong>Database</strong>.</td>
</tr>
<tr>
<td>Quit</td>
<td>Instructs the user to type the letter <strong>Q</strong> to exit the <strong>Replication Server Remote Agents</strong> menu.</td>
</tr>
<tr>
<td>Enter your selection</td>
<td>Instructs the user to type the <strong>ID</strong> of the remote agent whose information is to be displayed.</td>
</tr>
</tbody>
</table>
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.

![Figure 5–4: OpenEdge Replication Remote Agents Status screen](image)

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

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent</td>
<td>Shows a selection of basic remote agent information.</td>
</tr>
<tr>
<td>Name</td>
<td>Shows the name of the remote agent.</td>
</tr>
<tr>
<td>ID</td>
<td>Shows the remote agent identification number.</td>
</tr>
<tr>
<td>Host name</td>
<td>Shows the host on which the target database associated with the remote agent resides.</td>
</tr>
<tr>
<td>Target database</td>
<td>Shows the name of the database associated with the remote agent.</td>
</tr>
<tr>
<td>State</td>
<td>Shows information about what the server knows about the remote agent. For example:</td>
</tr>
<tr>
<td></td>
<td>• Normal processing — The server and the agent are performing normal processing.</td>
</tr>
<tr>
<td></td>
<td>• Initial connection — The agent is waiting for initial connection from server.</td>
</tr>
<tr>
<td></td>
<td>• Startup synchronization — The server and the agent are synchronizing.</td>
</tr>
<tr>
<td></td>
<td>• Initialization — The agent is being initialized by the server.</td>
</tr>
<tr>
<td></td>
<td>• Initial connection failed — The server could never connect to an agent.</td>
</tr>
<tr>
<td></td>
<td>• Invalid target database configuration — Something in the target database does not match the source.</td>
</tr>
<tr>
<td></td>
<td>• Agent terminated — The target database shut down or the agent terminated using Terminate agent.</td>
</tr>
<tr>
<td></td>
<td>• Online backup of Target Database — An online target database backup is being performed.</td>
</tr>
<tr>
<td></td>
<td>• Recovery synchronization — Recovery synchronization is being performed.</td>
</tr>
<tr>
<td></td>
<td>• Recovery failed — Failure recovery failed.</td>
</tr>
<tr>
<td></td>
<td>• Unknown — The agent is in an unknown state.</td>
</tr>
<tr>
<td>Critical</td>
<td>Shows whether the remote agent is a critical agent (Yes) or not (No).</td>
</tr>
<tr>
<td>Method</td>
<td>Shows the replication method: asynchronous or synchronous.</td>
</tr>
</tbody>
</table>
Table 5–11: Replication Remote Agents Status details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote agent is waiting for</td>
<td>Shows why the agent might be waiting. The field value might be one of the following:&lt;br&gt;• Nothing — Indicates no waiting is taking place&lt;br&gt;• Schema lock request — Indicates the server is waiting for the agent to acquire the schema lock so the database schema can be changed</td>
</tr>
<tr>
<td>Recovery state</td>
<td>Shows where the server and the agent are in the failure recovery process. The field value can be one of the following:&lt;br&gt;• Failed for the agent — Failure recovery could not be completed for the agent.&lt;br&gt;• No recovery being performed — There is currently no failure recovery being done.&lt;br&gt;• Just entered recovery — The server has just started failure recovery and will determine which remote agents remain connected and which must be reconnected.&lt;br&gt;• Server attempting connection — The server is attempting to connect to those remote agents no longer communicating to the server.&lt;br&gt;• Initialize synchronization with agents — The server is initializing the agent failure recovery synchronization.&lt;br&gt;• Synchronizing agents — The connected agents are in the process of being brought up to date with the database changes made to the source database.&lt;br&gt;• Recovery complete — The failure recovery of all connected agents has been completed.</td>
</tr>
<tr>
<td>Maximum bytes in TCP/IP message</td>
<td>Shows the maximum number of bytes used for the TCP/IP communication messages.</td>
</tr>
<tr>
<td>Server/Agent connection time</td>
<td>Shows the date and time at which the server and agent connected.</td>
</tr>
<tr>
<td>Server/Agent connection timeout</td>
<td>Shows the number of seconds after which the OpenEdge Replication server will stop attempting to connect to the agent.</td>
</tr>
<tr>
<td>Transition information</td>
<td>Shows remote agent-related transition details.</td>
</tr>
<tr>
<td>Type</td>
<td>Shows the type of transition to be performed:&lt;br&gt;• Manual — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database&lt;br&gt;• Automatic — Indicates a transition from a target database to a normal OpenEdge database will take place without intervention</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Timeout limit</td>
<td>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.</td>
</tr>
<tr>
<td>The last block was sent at</td>
<td>Shows the date and time the last block was sent to the agent.</td>
</tr>
<tr>
<td>Activity information</td>
<td>Shows activity information.</td>
</tr>
<tr>
<td>Blocks sent</td>
<td>Shows the number of blocks sent to the agent.</td>
</tr>
<tr>
<td>Blocks acknowledged</td>
<td>Shows the number of blocks acknowledged by the agent.</td>
</tr>
<tr>
<td>Synchronization points</td>
<td>Shows the number of synchronization points that have occurred.</td>
</tr>
<tr>
<td>AI Block Information</td>
<td>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.</td>
</tr>
<tr>
<td>Current RDBMS Block (Seq / Block)</td>
<td>Shows the current RDBMS AI block. Seq is the AI extent sequence number. It can be viewed by using rfutil LIST.</td>
</tr>
<tr>
<td>Last Sent Block (Seq / Block)</td>
<td>Shows the last AI block that was sent to the OpenEdge Replication agent.</td>
</tr>
<tr>
<td>Server to agent load check interval</td>
<td>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.</td>
</tr>
<tr>
<td>Time between server and agent load checks</td>
<td>Shows the average elapsed time between the load checks.</td>
</tr>
<tr>
<td>Time taken to respond to load check</td>
<td>Shows the average elapsed time the agent took to acknowledge the server for the block just sent.</td>
</tr>
</tbody>
</table>
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.

![Figure 5–5: OpenEdge Replication agent status screen](image)

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.

When you use the Monitor utility for the agent on an encrypted target database, the utility displays the agent’s wait status, as shown in the line in bold in Figure 5–5.
The remaining lines of the Replication Agent Status screen are defined in Table 5–12.

### Table 5–12: Replication Agent Status details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database is enabled as OpenEdge Replication</td>
<td>Indicates whether the database is enabled as a replication source or a target.</td>
</tr>
<tr>
<td>Agent</td>
<td>Lists basic remote agent information.</td>
</tr>
<tr>
<td>Name</td>
<td>Lists the name of the remote agent.</td>
</tr>
<tr>
<td>ID</td>
<td>Lists the remote agent identification number.</td>
</tr>
<tr>
<td>Host name</td>
<td>Lists the host on which the target database associated with the remote agent resides.</td>
</tr>
<tr>
<td>State</td>
<td>Shows agent processing information:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Normal processing</strong> — The server and the agent are performing normal processing.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Initial connection</strong> — The agent is waiting for initial connection from server.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Startup synchronization</strong> — The server and the agent are synchronizing.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Initialization</strong> — The agent is being initialized by the server.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Initial connection failed</strong> — The server could never connect to an agent.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Invalid target database configuration</strong> — Something in the target database does not match the source.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Agent terminated</strong> — The target database shut down or the agent terminated using <code>Terminate agent</code>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Online backup of Target Database</strong> — An online target database backup is being performed.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Recovery synchronization</strong> — Recovery synchronization is being performed.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Recovery failed</strong> — Failure recovery was unsuccessful.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Unknown</strong> — The agent is in an unknown state.</td>
</tr>
<tr>
<td>Critical</td>
<td>Shows whether the remote agent is a critical agent (<strong>Yes</strong>) or not (<strong>No</strong>).</td>
</tr>
<tr>
<td>Method</td>
<td>Lists the replication method: <strong>asynchronous</strong> or <strong>synchronous</strong>.</td>
</tr>
</tbody>
</table>
Table 5–12: Replication Agent Status details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent is waiting for</td>
<td>Shows why the agent might be waiting. The field value might be one of the following:</td>
</tr>
<tr>
<td></td>
<td>• Nothing — Indicates no waiting is taking place</td>
</tr>
<tr>
<td></td>
<td>• Schema lock request — Indicates the server is waiting for the agent to acquire the schema lock so the database schema can be changed</td>
</tr>
<tr>
<td></td>
<td>• Encryption objects — For an encrypted database, the agent’s wait status, or what the agent is waiting for</td>
</tr>
<tr>
<td>Maximum bytes in TCP/IP message</td>
<td>Shows the maximum number of bytes used for the TCP/IP communication messages.</td>
</tr>
<tr>
<td>Server/Agent connection time</td>
<td>Shows the date and time at which the server and agent connected.</td>
</tr>
<tr>
<td>Delay Interval (current / min / max)</td>
<td>Shows the amount of time, in milliseconds, the agent will wait between polling the TCP/IP connection for information to be replicated. Current is the current value, min is the minimum value, and max is the maximum value.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>Transition information</td>
<td>Shows a group of remote agent-related transition information.</td>
</tr>
<tr>
<td>Type</td>
<td>Shows the type of transition to be performed:</td>
</tr>
<tr>
<td></td>
<td>• Manual — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database</td>
</tr>
<tr>
<td></td>
<td>• Automatic — Indicates a transition from a target database to a normal OpenEdge database will take place without intervention</td>
</tr>
<tr>
<td>The last block received at</td>
<td>Shows the date and time the last block was sent to the agent.</td>
</tr>
<tr>
<td>Activity information</td>
<td>Shows a selection of activity information.</td>
</tr>
<tr>
<td>Blocks received</td>
<td>Shows the number of blocks received from the server.</td>
</tr>
<tr>
<td>Blocks processed</td>
<td>Shows the number of blocks processed.</td>
</tr>
<tr>
<td>Blocks acknowledged</td>
<td>Shows the number of blocks acknowledged.</td>
</tr>
<tr>
<td>Notes processed</td>
<td>Shows the number of AI transaction notes processed by the agent.</td>
</tr>
<tr>
<td>Transactions started</td>
<td>Shows the number of transactions started on the agent.</td>
</tr>
<tr>
<td>Transactions ended</td>
<td>Shows the number of transactions completed on the agent.</td>
</tr>
</tbody>
</table>
Table 5–12: Replication Agent Status details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronization points</td>
<td>Shows the number of synchronization points that have occurred.</td>
</tr>
<tr>
<td>AI Block Information</td>
<td>Identifies the current AI block that is being written by the source database and provides the last AI block that was sent by Replication.</td>
</tr>
<tr>
<td>Source RDBMS Block (Seq / Block)</td>
<td>Provides the current source database block that the OpenEdge Replication agent is processing.</td>
</tr>
<tr>
<td>Last Processed Block (Seq / Block)</td>
<td>Provides the last source database block that the OpenEdge Replication agent processed.</td>
</tr>
<tr>
<td>Latency Information</td>
<td>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.</td>
</tr>
<tr>
<td>Repl Server behind Source DB by</td>
<td>Provides the number of seconds the OpenEdge Replication server is behind the source database.</td>
</tr>
<tr>
<td>Current Source Database Transaction</td>
<td>Provides the current source database transaction that the OpenEdge Replication agent is processing.</td>
</tr>
<tr>
<td>Last Transaction Applied to Target</td>
<td>Provides the last source database transaction that was applied to the target database.</td>
</tr>
<tr>
<td>Target Current As Of</td>
<td>Provides the date of the last transaction applied to the target database by the OpenEdge Replication agent based on source database time.</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Virtual system table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenEdge Replication Server _Rep1-Server</td>
<td>Provides detailed OpenEdge Replication server information</td>
</tr>
<tr>
<td>OpenEdge Replication Control Agent _Rep1-AgentControl</td>
<td>Provides detailed information about the OpenEdge Replication agents this OpenEdge Replication server is controlling</td>
</tr>
<tr>
<td>OpenEdge Replication Agent _Rep1-Agent</td>
<td>Provides detailed OpenEdge Replication agent information</td>
</tr>
</tbody>
</table>

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)

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field number</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ReplSrv-AgentCount</td>
<td>2</td>
<td>INTFLD</td>
<td>Agents Count&lt;br&gt;Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9&lt;br&gt;Number of agents that the OpenEdge Replication server is controlling</td>
</tr>
<tr>
<td>_ReplSrv-BlocksSent</td>
<td>3</td>
<td>INTFLD</td>
<td>AI Blocks Sent&lt;br&gt;Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9&lt;br&gt;Number of AI Blocks the OpenEdge Replication server has sent to connected OpenEdge Replication agents</td>
</tr>
</tbody>
</table>
Table 5–15 shows the field descriptions for _Repl-AgentControl.

**Table 5–15: Virtual system table _Repl-AgentControl field description (1 of 3)**

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field number</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ReplAgtCt1-AgentID</td>
<td>2</td>
<td>INTFLD</td>
<td>Agent ID</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The ID of the OpenEdge Replication agent</td>
</tr>
<tr>
<td>_ReplAgtCt1-AgentName</td>
<td>3</td>
<td>CHRFLD</td>
<td>Agent Name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format X(32)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Name of the OpenEdge Replication agent as configured in the repl.properties file</td>
</tr>
<tr>
<td>_ReplAgtCt1-ConnectTime</td>
<td>4</td>
<td>CHRFLD</td>
<td>Agent Start Time</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format X(24)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Date and time when the OpenEdge Replication server successfully connected to the OpenEdge Replication agent</td>
</tr>
<tr>
<td>_ReplAgtCt1-RemoteDBName</td>
<td>5</td>
<td>CHRFLD</td>
<td>Remote Database Name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format X(256)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Fully qualified database name to which the OpenEdge Replication agent is connected</td>
</tr>
<tr>
<td>Field name</td>
<td>Field number</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>_ReplAgtCtl-RemoteHost</td>
<td>6</td>
<td>CHRFLD</td>
<td>Remote Host Name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format X(128)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Name of the host where the OpenEdge Replication agent is running</td>
</tr>
<tr>
<td>_ReplAgtCtl-Port</td>
<td>7</td>
<td>INTFLD</td>
<td>Connected to Agent on Port Number</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The port number the OpenEdge Replication server uses to connect to this OpenEdge Replication agent</td>
</tr>
<tr>
<td>_ReplAgtCtl-BlockSent</td>
<td>8</td>
<td>INDFLD</td>
<td>Blocks Sent to this Agent</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The number of AI blocks the OpenEdge Replication server has sent to this OpenEdge Replication agent</td>
</tr>
<tr>
<td>_ReplAgtCtl-BlocksACK</td>
<td>9</td>
<td>INTFLD</td>
<td>Number of block acknowledgments received for this agent</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The number of block acknowledgements this OpenEdge Replication agent has received from its remote counterpart</td>
</tr>
<tr>
<td>_ReplAgtCtl-LastBlockSentAt</td>
<td>10</td>
<td>CHRFLD</td>
<td>Last Block Sent At</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format X(24)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The data and time the last block was sent to this OpenEdge Replication agent</td>
</tr>
<tr>
<td>_ReplAgtCtl-Method</td>
<td>11</td>
<td>CHRFLD</td>
<td>Replication Method</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format X(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The type of replication that is being performed with this OpenEdge Replication agent:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• A — Asynchronous (mode &amp; RP_MODE_ASYNC)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• S — Synchronous (mode &amp; RP_MODE_SYNC)</td>
</tr>
<tr>
<td>Field name</td>
<td>Field number</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>_ReplAgtCtl-Status</td>
<td>12</td>
<td>INTFLD</td>
<td>A value from the <strong>Value</strong> column in Table 5–7, as reported by the following command: DSRUTIL -C status -detail</td>
</tr>
<tr>
<td>_ReplAgtCtl-CommStatus</td>
<td>13</td>
<td>INTFLD</td>
<td>Agent’s Communication Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The OpenEdge Replication agent’s TCP communication status:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>•  1 — Connected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>•  2 — Disconnected</td>
</tr>
</tbody>
</table>
Table 5–16 shows the field descriptions for _Repl-Agent.

Table 5–16: Virtual system table _Repl-Agent field descriptions

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field number</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ReplAgt-AgentID</td>
<td>2</td>
<td>INTFLD</td>
<td>Agent ID&lt;br&gt;Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9&lt;br&gt;The ID of the OpenEdge Replication agent</td>
</tr>
<tr>
<td>_ReplAgt-AgentName</td>
<td>3</td>
<td>CHRFLD</td>
<td>Agent Name&lt;br&gt;Format X(32)&lt;br&gt;Name of the OpenEdge Replication agent as configured in the repl.properties file</td>
</tr>
<tr>
<td>_ReplAgt-StartTime</td>
<td>4</td>
<td>CHRFLD</td>
<td>Agent Start Time&lt;br&gt;Format X(24)&lt;br&gt;Date and time when the OpenEdge Replication agent was started</td>
</tr>
<tr>
<td>_ReplAgt-ConnectTime</td>
<td>5</td>
<td>CHRFLD</td>
<td>Agent Connect Time&lt;br&gt;Format X(24)&lt;br&gt;Date and time when the OpenEdge Replication server successfully connected to the OpenEdge Replication agent</td>
</tr>
<tr>
<td>_ReplAgt-DBName</td>
<td>6</td>
<td>CHRFLD</td>
<td>Database Name&lt;br&gt;Format X(128)&lt;br&gt;Fully qualified database name that the OpenEdge Replication agent is connected to</td>
</tr>
<tr>
<td>_ReplAgt-ServerHost</td>
<td>7</td>
<td>CHRFLD</td>
<td>Server Host Name&lt;br&gt;Format X(128)&lt;br&gt;Name of the host where the OpenEdge Replication server is running</td>
</tr>
<tr>
<td>_ReplAgt-Port</td>
<td>8</td>
<td>INTFLD</td>
<td>The agent is connected to this port number&lt;br&gt;Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9&lt;br&gt;The port number the OpenEdge Replication server uses to connect to this OpenEdge Replication agent</td>
</tr>
</tbody>
</table>
Table 5–16:  Virtual system table _Repl-Agent field descriptions

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field number</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ReplAgt-BlocksReceived</td>
<td>9</td>
<td>INTFLD</td>
<td>Number of blocks received</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The number of AI blocks this OpenEdge Replication server has received</td>
</tr>
<tr>
<td>_ReplAgt-BlocksProcessed</td>
<td>10</td>
<td>INTFLD</td>
<td>Number of blocks processed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The number of AI blocks this OpenEdge Replication agent has processed</td>
</tr>
<tr>
<td>_ReplAgt-BlocksACK</td>
<td>11</td>
<td>INTFLD</td>
<td>Number of block acknowledgements sent by this agent</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The number of block acknowledgements this OpenEdge Replication agent has sent to the OpenEdge Replication server</td>
</tr>
<tr>
<td>_ReplAgt-NotesProcessed</td>
<td>12</td>
<td>INTFLD</td>
<td>Number of Notes Processed</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The number of AI notes this OpenEdge Replication agent has processed</td>
</tr>
<tr>
<td>_ReplAgt-LastTRID</td>
<td>13</td>
<td>INTFLD</td>
<td>Transaction ID of Last Transaction began</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The Transaction ID of the last Transaction Begin encountered</td>
</tr>
<tr>
<td>_ReplAgt-LastEND</td>
<td>14</td>
<td>INDFLD</td>
<td>Transaction ID of last Transaction End</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format -&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The Transaction ID of the last Transaction End encountered</td>
</tr>
<tr>
<td>_ReplAgt-Method</td>
<td>15</td>
<td>CHRFLD</td>
<td>Replication Method</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Format X(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The type of replication that is being performed with this OpenEdge Replication agent:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- A — Asynchronous</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- S — Synchronous</td>
</tr>
</tbody>
</table>
### Table 5–16: Virtual system table _Repl-Agent field descriptions (3 of 3)

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field number</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ReplAgt-Status</td>
<td>16</td>
<td>INTFLD</td>
<td>A value from the <strong>Value</strong> column in Table 5–7, as reported by the following command: DSRUTIL -C status -detail</td>
</tr>
</tbody>
</table>
| _ReplAgt-CommStatus | 17          | INTFLD    | Agent’s Communication Status  
The OpenEdge Replication agent’s TCP communication status:  
- 1 — Connected  
- 2 — Disconnected |
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

<table>
<thead>
<tr>
<th>Utility</th>
<th>Source database</th>
<th>Target database</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROBKUP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Online</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Incremental</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PROCOPY</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PRODB</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PRODEL</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PROLOG</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PROMON</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PROREST</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PROSTRCT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Builddb¹</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>List</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Remove</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Repair</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Statistics</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Unlock</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Table 5–17: Utility support for OpenEdge Replication

<table>
<thead>
<tr>
<th>Utility</th>
<th>Source database</th>
<th>Target database</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROUTIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2phase begin</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>2phase commit</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>2phase end</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>2phase modify</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>2phase recover</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>bigrow</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>busy</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>bulkload</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>chanalys</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>cmpdb</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>codepage-compiler</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>conv89</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>convchar</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>convfile</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>dbanalys</td>
<td></td>
<td></td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>offline</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>dbauthkey</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>dbipcs</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>dump</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>dumpspecified</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>offline</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>enablelargfiles</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>enablesitereplication</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>disablesitereplication</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>holder</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Utility</td>
<td>Source database</td>
<td>Target database</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>PROUTIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>idxanalys</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>offline</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>idxblockreport</td>
<td></td>
<td></td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>offline</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>idxbuild</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>idxcheck</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>idxcompact</td>
<td></td>
<td></td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>offline</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>idxfix</td>
<td></td>
<td></td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>offline</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>idxmove</td>
<td></td>
<td></td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>offline</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>iostats</td>
<td></td>
<td></td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>offline</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>load</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>mvsch</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>rcodekey</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>tabanalys</td>
<td></td>
<td></td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>offline</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>tablemove</td>
<td></td>
<td></td>
</tr>
<tr>
<td>online</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>offline</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>truncate area</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>truncate bi</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>updatevst</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>wordbreak-compiler</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>word-rules</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Table 5–17: Utility support for OpenEdge Replication

<table>
<thead>
<tr>
<th>Utility</th>
<th>Source database</th>
<th>Target database</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFUTIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aimage begin</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>aimage end</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>aimage extent empty</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>aimage extent full</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>aimage extent list</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>aimage new</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>aimage scan</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>aimage truncate</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>mark backup</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>roll forward</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>roll forward retry</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SQLDUMP</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SQLLOAD</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>SQLSCHEMA</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

1. Note that if you run prostruct buildDb on either a source or target database, the database is no longer replication-enabled. You must re-enable the database for replication.
Once you enable OpenEdge Replication and set the properties for both the source and the target databases, you can start OpenEdge Replication automatically with Progress Explorer, the dbman utility, or OpenEdge Explorer, instead of the PROSERVE command. This is possible because the database configurations that make OpenEdge Replication possible are saved in the conmgr.properties file.

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.

For details about starting Replication with either Progress Explorer or OpenEdge Explorer, see the Progress Explorer online help or OpenEdge Management and OpenEdge Explorer: Configuration.
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 Progress Explorer or OpenEdge Explorer.

- 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. Using OpenEdge Explorer or OpenEdge Management, you can configure properties for replication-enabled databases.

For more information, see OpenEdge Management: Database Management, OpenEdge Management: Reporting, and the section about configuring database properties in OpenEdge Management and OpenEdge Explorer: Configuration.
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 the 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 3–15.
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. Verify that after-imaging is enabled. If it 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:

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

4. Enable the source database for Replication, as shown:

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

5. Perform an incremental backup, as shown:

   `probkup source-db-name incremental { source-db-incrementalbackup-name | device-name }`
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. Verify that after-imaging is enabled. 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
      ```

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. Back up the database, as shown:
   
   ```
   probkup source-db-name source-db-name.bak
   ```

6. Restart the source database, as shown (note that `-DBService` is case-sensitive):
   
   ```
   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. If the source database is encrypted, copy the source database keystore (`source-db-name.ks`) to the target machine, rename it with the name of the target database (`target-db-name.ks`), and place it in the target database directory.

4. 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
```

5. Enable Replication, as shown:

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

6. Start the server for the target, as shown (note that `-DBService` is case-sensitive):

```
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.
Enabling before-image encryption for a Replication-enabled target database

To enable before-image encryption on the target database, you must recreate the target database from the source database.

To enable before-image encryption on the Replication-enabled target database:

1. Shut down the source database and the target database, using the following command for each database:

   ```
   proshut db-name
   ```

2. Use the following command to disable replication on the source database:

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

3. Use the following command to disable replication on the target database:

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

4. If before-image encryption is not already enabled on the source database, enable it using the following command:

   ```
   proutil source-db-name -C enableencryption -biencryption enable
   ```

   The before image area is truncated.

5. Enable the database as a Replication source database using the following command:

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

6. Back up the database, as shown:

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

7. Copy the backup volume or volumes to the target computer.

8. Copy the source database structure file to the target database directory on the secondary computer. Make any modifications necessary to the file to match the configuration of the target database.

9. Copy the source database key store (source-db-name.ks) to the target machine and place it into the target database directory.
10. Restore the target database from the source backup volume or volumes:

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

11. Enable the database as a target database using the following command:

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

12. Configure the Replication property files for both the source and target databases, as follows:
   
a. Locate the sample source database property file (source.repl.properties) and the sample target database property file (target.repl.properties), found in `OpenEdge-install-dir\properties`.

b. Copy each file to the same directory as its corresponding database, and rename it to match the name of the database.

c. Modify the copied property files, if necessary.

13. Start both databases.
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.repl.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:

```plaintext
[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.repl.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.repl.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.repl.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.repl.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
   ```

**Note:** In a two-agent scenario, one Replication agent may connect to the Replication server and then terminate while the server is connecting to the second Replication agent. In this situation, the Replication server fails and does not attempt to reconnect to the agents. You must restart both target databases and then restart the Replication server, which will try again to connect to both agents.
Starting OpenEdge Replication

After you set up the databases and configure 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 (note that -DBService is case-sensitive):

   ```
   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 3–18 and the “Starting the target database” section on page 3–19.
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 3–25 and the “Shutting down the target database” section on page 3–26.
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 3–26 and the “Terminating the OpenEdge Replication agent” section on page 3–27.
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 3–12.
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 3–12.
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:

```
dsrtutil 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 3–43.
**Restarting the 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 (note that -DBService is case-sensitive):

   ```
   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 3–28
  - enabling 3–8
  - extent blocks 3–31
  - extent sizing on source database 2–11
  - extent states 2–5
  - extent types 2–5
  - extents 2–4
  - fixed-length extents 2–4
  - full extents 3–28
  - switching extents 3–32
  - variable-length extents 2–4
- Agent 1–7
  - configuration 3–17, 5–33
  - connection timeout 5–9, 5–38
  - critical 5–8
  - host 5–8
  - latency report 3–29
  - listener-maxport 5–9
  - listener-minport 5–9
  - polling 5–5, 5–10
  - port 5–8
  - properties 5–9
  - rules 3–16
  - single agent configuration 6–9
  - status 5–31, 5–40
  - terminating 3–27, 5–12, 5–28, 6–13
  - two agent configuration 6–10
- ApplyExtent command action 5–12, 5–13

## Architecture 1–3

## Asynchronous
- configuration 1–10
- operation 1–11

## Automatic transition 3–38, 5–34, 6–14

## B

- Before-Image truncate 1–15

## Benefits of OpenEdge Replication 1–2

## C

- Checkpoint 3–32
- Commands 3–30
- Configuring
  - agent 3–17, 5–33, 6–9, 6–10
  - errors 3–18
    - OpenEdge Replication 3–15
- Connection 5–33
  - connect-timeout property 5–8, 5–9
- Control agent properties 5–5, 5–7
- Creating the target database 3–12
- Critical agent 1–9
  - critical property 5–8
Index

D
Data replication 1–2
Database
  access 3–23
  availability 1–13
  Before Image truncate 1–15
  connection considerations 3–22
  crash 3–35
  enabling for transparent data encryption and replication 2–3
  Enhanced Read-only mode
    defined 1–14
    integrity 1–13
    schema locks 1–14
database property 5–5, 5–8, 5–9
database.repl.recovery 3–32
database.repl.recovery file 3–20
db.repl.properties 5–3
-DBService argument 3–18
Default implementation 3–3
deferrer-agent-startup property 3–4, 3–11, 3–21, 5–5
Deferred agent startup 3–4, 3–6, 3–11, 6–8
disableitereplication argument 3–20, 5–12, 5–18
DSRUTIL utility 3–30, 3–39, 5–11
  agent option 5–11
  and passphrase 5–11
  and passphrase in transition 5–29
  and transparent data encryption 5–11
  MONITOR 5–31
    latency details 3–29
    server option 5–11
    startup menu 5–31
E
Emptying AI extents 3–28
Endian ordering
  defined 1–6
Enhanced Read-only mode (ERO)
  defined 1–14
Estimating network bandwidth 2–15

F
Failover
  defined 2–19
Failure
  conditions 3–35
  processing 1–12
  recovery 1–12, 3–34
Failure recovery
  performing 5–33
Fixed-length after-image extents 2–4
Full AI extents 3–28
Functional definitions 1–3

H
Hot standby
  defined 3–34

I
Implementation 3–3
  default 3–3
  deferred agent startup 3–4
Incremental backup 3–9
Initialization process 3–21
ipver property 5–8, 5–9

L
Latency 3–29, 3–34, 5–40
  details provided by DSRUTIL MONITOR 3–29
Limitations and restrictions 2–19
listener-maxport property 5–9
listener-minport property 5–9
Logical operation 2–9
Lost connection 3–35

M
Manual transition 3–39, 5–34, 6–15
maximum-polling-delay property 5–5, 5–10
Message logging 3–19
minimum-polling-delay property 5–5, 5–10

N
name property 5–8, 5–10
Normal
database activity 3–31
processing 5–33
target activity 3–31

O
Offline backup 3–7
Online backup of source database 3–9, 3–11, 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 6–14
benefits 1–2
commands 3–30
configuring 3–15
control agent properties 5–7
critical agent 1–9
database monitoring 6–19
defined 1–2
disabling 3–20
failover 2–19
feature 1–2
functional definitions 1–3
implementation 3–3
limitations and restrictions 2–19
manual transition 6–15
model 1–9
monitoring 6–19
network bandwidth estimation 2–15
on encrypted databases 1–2, 1–13, 2–3, 3–46, 5–11
overhead 2–11, 2–15
planning 2–2
property files 5–3
re-enabling after transition 6–16
server 1–6
server properties 5–5
starting 5–55, 6–11
startup 3–18
stopping 3–25, 6–12
utilities 3–30, 5–51
with transparent data encryption 2–3

P
Passphrase (transparent data encryption)
1–13, 2–3, 3–46, 5–11, 5–29
Performing initialization status 5–33
port property 5–8
Pre-transition state 3–37
Primary database
defined 1–3
PROBKUP 1–16, 5–51
PROCOPY 5–51
PRODB 5–51
PRODEL 5–51
PROLOG 5–51
PROMON 5–51
Properties 5–5
agent 5–9
control agent 5–7
server 5–5
transition 4–32
Property files 5–3
sample combined 5–4
sample source 5–3
sample target 5–4
source 5–3
target 5–4
PROREST 3–20, 5–51
PROSERVE 3–19
PROSHUT 3–25, 3–26
PROSTRCT 5–51
PROUTIL 3–20, 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 5–46
_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 3–36
RECOVERY command action 5–20
Re-enabling OpenEdge Replication after transition 6–17
RELWAITS command action 5–22
Remote agent status 5–35, 5–36
repl.server.startup.lg 3–19
Replication agent. See Agent
Replication overhead 2–11
Replication server. See Server
replication-method property 5–8
Repl-Keep-Alive property 5–6
RESTART command action 5–12, 5–23
Restoring databases 3–20
source database 3–20
RFUTIL 5–54
Roll forward 3–32
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 3–29
properties 5–5
recovery 5–20
remote agents details 5–35
remote agents status 5–36
restarting after target shutdown 6–18
status 5–31, 5–32, 5–33
terminating 3–26, 6–13
Setup
source 3–7, 3–9, 6–3
target 3–12, 6–3, 6–5
Shutdown
source 3–25
target 3–26
Single agent configuration 6–9
Source
access 3–24
combined properties file 5–4
database defined 1–3
enabling 3–9, 3–10
incremental backup 3–9
machine 2–2
normal activity 3–31
offline backup 3–7
online backup 3–9, 3–11, 6–4
properties file 5–3
PROSERVE startup 3–18
requirements 1–6
restoring 3–20
setup 3–7, 3–9, 6–3
shutdown 3–25
startup 3–18
structure files 3–8, 3–10
INDEX

SQLDUMP 5–54
SQLLOAD 5–54
SQLSCHEMA 5–54

startAgent command action 5–12, 5–24

Startup 3–18
message logging 3–19
process 3–21
source 3–18
target 3–19

Structure files 3–8, 3–10

Supported utilities 5–51

Sync point
defined 3–32

Synchronization
defined 1–12
processing 5–33

Synchronous
collection 1–10
operation 1–11

T

Target
access 3–23
combined properties file 5–4
create 3–12
database
defined 1–3
machine 2–2
normal activity 3–31
online backup 1–15
properties file 5–4
PROSERVE startup 3–19
quiet points 1–15
requirements 1–6
restoring 3–20
setup 3–12, 6–3, 6–5
shutdown 3–26
startup 3–19
transition 3–38
target.repl.properties 3–17
TCP/IP communications 1–10

TERMINATE command action 3–26, 3–27, 5–28
Terminating
agent 3–27, 6–13
server 3–26, 6–13

Transaction end 3–32

Transition 4–4
automatic 3–38
configuration 3–38
defined 3–38
lost connection 3–37
manual 3–39
processing 3–40
re-enabling OpenEdge Replication after
6–17
timeout limit 5–34

Transition command action 5–29
transition property 5–7
transition-timeout property 5–7

Transparent data encryption
and DSRUTIL 5–11
and replication requirements 2–3
enabling for a replication-enabled source
database 3–46
support in replication 1–2, 1–13, 2–3

TriggerTransition command action 5–12,
5–30

U

Unknown state 5–33

Unsupported utilities 5–51

Utilities 3–30, 5–51

V

Variable-length after-image extents 2–4

Virtual system tables 5–44

VST field descriptions 5–44