Progress®



OpenEdge® Replication:
User Guide

OpenEdge®

Table of Contents

Copyright....9

Preface....11

Purpose....11
Audience....11
Organization....12
Using ABL documentation....12
References to ABL compiler and run-time features....12
References to ABL data types....13
Typographical conventions....13
Examples of syntax descriptions....14
Long syntax descriptions split across lines....16
Complex syntax descriptions with both required and optional elements....16
OpenEdge messages....16
Obtaining more information about OpenEdge messages....17

Chapter 1: Introducing OpenEdge Replication....19

What is data replication....19
How OpenEdge Replication works with two databases....20
How OpenEdge Replication works with a replication set....20
Key features of OpenEdge Replication....21
OpenEdge Replication terminology....21
Primary and secondary databases....22
Source and target databases....22
A Replication Set....25
OpenEdge Replication property files....26
OpenEdge Replication server....26
OpenEdge Replication agent....27
OpenEdge Replication Model....28
Failover....29
Failback....29
Transition....29
Source and target architecture requirements....30
OpenEdge Replication activity during normal processing....30
Normal source database activity....30
Normal target database activity....31
What happens if there is a database failure....32
Choosing a hot standby database....32
Synchronization between server and agent....33

OpenEdge Replication utilities and commands....34

Chapter 2: OpenEdge Replication and After-imaging....35

Understanding OpenEdge Replication after-image requirements....35
Calculating current database after-image volume....36
Using after-image VST information....36
Using the RFUTIL command....37
Using the PROMON command....37
Using before-image VST information....38
The role of after-image extents in the Replication process....38
After-image extent types....39
After-image extent states....39
After-image extent sizing on a source database....41
Calculating fixed after-image extent size....41
After-image interdependency between source and target databases....43
Determining OpenEdge Replication network bandwidth....44
Estimating network bandwidth....44
Anticipating additional network overhead....45
Ensuring success when using after-imaging....46
OpenEdge Replication and after-imaging: a summary....46

Chapter 3: Planning for OpenEdge Replication....47

Preliminary planning tasks....47
Step 1: Reviewing after-image information....48
Step 2: Creating a database backup plan....48
Step 3: Evaluating your production database....48
Step 4: Determining if you will be replicating an encryption-enabled database....48
Understanding the importance of after-imaging in OpenEdge Replication planning....49
Determining acceptable latency....50
Determining acceptable target database downtime....50
Determining appropriate failover behavior....50
Understanding limitations and restrictions....51

Chapter 4: Setting Up OpenEdge Replication....53

Overview....53
Getting started....54
Setting up the OpenEdge Replication databases....55
Setting up the source database with an offline backup....55
Setting up the source database with an online backup....57
Setting up the OpenEdge Replication target database....59
Enabling before-image encryption for a Replication-enabled target database....60
Configuring the OpenEdge Replication property files....62
Sample OpenEdge Replication property files....62

Server properties....64
Control agent properties....67
Agent properties....69
Logging level for Replication....71
Configuring the source database property file....72
Configuring the target database property file....74
Choosing the OpenEdge Replication agent startup mode....74
Using the immediate agent startup mode....75
Using the deferred agent startup mode....75
Setting up the deferred agent startup implementation....76
Starting OpenEdge Replication....77
Message logging during startup....78
Starting the source database....78
Starting the target database....78
OpenEdge Replication startup and initialization process....79
Starting OpenEdge Replication with the DBMAN utility or OpenEdge Explorer....80
Database connection considerations....80
Accessing a running replication-enabled database....81
Accessing a target database....81
Accessing a source database....82
Enabling transparent data encryption for an OpenEdge Replication-enabled source database....82
Enabling encryption for a Replication-enabled source database that is online....83
Stopping OpenEdge Replication....84
Shutting down the source database....84
Shutting down the target database....85
Terminating the OpenEdge Replication server when the source database is running....85
Terminating the OpenEdge Replication agent when the target database is running....86
Disabling OpenEdge Replication on the server....86
Disabling OpenEdge Replication on the agent....87

Chapter 5: Moving From Failure to Recovery with OpenEdge Replication....89

Overview of database failure recovery....90
Automated transition functions....90
Possible types of failure....90
Detecting TCP/IP communications failures....91
Source failure recovery after losing connection....91
Target transition after losing connection....92
Restoring a source database....92
Transition 93
Transition scenarios....93
Intuitive transition processing....94
JTA transactions and transition....94
Configuring transition....94

Planning for transition....96
Transition End Trigger....99
OpenEdge Replication from normal activity through failure and recovery....101
Step 1: Primary replication before a failure....102
Step 2: Primary machine failure....102
Step 3: Entering pretransition....103
Step 4: Transitioning the target database to a source database....105
Step 5: Failover....106
Step 6: Primary machine repair complete....107
Step 7: Initiating primary database transition....107
Step 8: Secondary replication is performed....110
Step 9: The Replication failback process....111
Transition of a Replication Set....120
Recovery from transition failures....122
Transition logging....123
Transition properties....124
Setting transition properties....125
Sample of a startup parameter file used by transition....129
Source database startup parameter file....129
Target database startup parameter file....129
Normal database startup parameter file....130
Transition property reference....130
Transition command actions....130
Transition properties summary....131

Chapter 6: Enabling Replication AI Streaming....137

Replication AI Streaming Overview....137
Configuring Replication AI Streaming....139
Observing Replication AI Streaming in the Replication Monitor....139
Observing Replication AI Streaming in VSTs....140
Understand automatic transition with AI Streaming....141

Chapter 7: Reference....145

DSRUTIL utility....146
DSRUTIL applyextent qualifier....147
DSRUTIL canceldefer server qualifier....150
DSRUTIL connectagent database qualifier....151
DSRUTIL disablesitereplication qualifier....152
DSRUTIL monitor qualifier....152
DSRUTIL recovery qualifier....153
DSRUTIL restart agent qualifier....154
DSRUTIL restart server qualifier....154
DSRUTIL startagent database qualifier....155

DSRUTIL status qualifier....155
When the detail or verbose argument is used....156
When the detail or verbose argument is not used....157
DSRUTIL terminate qualifier....159
DSRUTIL transition qualifier....160
DSRUTIL transition failover qualifier....161
DSRUTIL triggertransition qualifier....161
OpenEdge Replication DSRUTIL MONITOR....162
Startup menu....162
Monitoring latency between the source and target databases....163
Replication server status....164
Replication server remote agents....168
Replication remote agents status....169
Replication agent status....173
Replication inter-agent status....179
Virtual system tables for OpenEdge Replication....181
_Repl-Server VST field descriptions....182
_Repl-Agent VST field descriptions....186
_Repl-AgentActivity VST field descriptions....191
_Repl-AgentControl VST field descriptions 195
_Repl-AgentControlActivity VST field descriptions....198
_Repl-AgentAIStreaming VST field descriptions....201
_DbServiceManager VST field descriptions....202
_DbServiceManagerObjects VST field descriptions....204
_Repl-InterAgentActivity VST field descriptions....205
Modifiable VSTs on Replication Targets....207
Utilities and OpenEdge Replication....210
OpenEdge Replication and database management systems....216
DSRUTIL logginglevel qualifier....216

Chapter 8: OpenEdge Replication Quick Command Summary....219

Configuring the OpenEdge Replication property files....220
Setting up the source and target databases....220
Setting up the source database with deferred agent startup for online backup....221
Set up the target database....222
Enabling before-image encryption for a Replication-enabled target database....223
Configuring OpenEdge Replication with deferred agent startup....224
Configuring OpenEdge Replication for one agent....225
Configuring OpenEdge Replication for two agents....226
Starting OpenEdge Replication....227
Stopping OpenEdge Replication....227
Terminating the OpenEdge Replication server and agent....228
Configuring for automatic transition....228
Configuring for manual transition....229

Using manual transition....229
Re-enabling OpenEdge Replication after transition....229
Restarting the OpenEdge Replication server after target shutdown....230
Monitoring an OpenEdge Replication database....231
Backing up an online target database....231

Index....235

Copyright

© **2019 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.**

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

Corticon, DataDirect (and design), DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Deliver More Than Expected, Icenium, Kendo UI, Kinvey, NativeScript, OpenEdge, Powered by Progress, Progress, Progress Software Developers Network, Rollbase, SequeLink, Sitefinity (and Design), Sitefinity, SpeedScript, Stylus Studio, TeamPulse, Telerik, Telerik (and Design), Test Studio, and WebSpeed are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries. Analytics360, AppServer, BusinessEdge, DataDirect Autonomous REST Connector, DataDirect Spy, SupportLink, DevCraft, Fiddler, JustAssembly, JustDecompile, JustMock, Kinvey Chat, NativeChat, NativeScript Sidekick, OpenAccess, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

February 2019

Last updated with new content: Release 12.0

Preface

For details, see the following topics:

- Purpose
- Audience
- Organization
- Using ABL documentation
- Typographical conventions
- Examples of syntax descriptions
- OpenEdge messages

Purpose

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

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.

Audience

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

Organization

Introducing OpenEdge Replication on page 19

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

OpenEdge Replication and After-imaging on page 35

Provides details about database after-imaging functionality and OpenEdge Replication.

Planning for OpenEdge Replication on page 47

Provides a description of planning considerations to keep in mind when using OpenEdge Replication.

Setting Up OpenEdge Replication on page 53

Provides instructions about how to set up, start, and stop OpenEdge Replication.

Moving From Failure to Recovery with OpenEdge Replication on page 89

Provides details about possible failure conditions; an introduction to the transition process; and details about recovering from a database failure on a source or target machine, including information about the transition and failback processes.

Reference on page 145

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

OpenEdge Replication Quick Command Summary on page 219

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

Using ABL documentation

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 go to https://docs.progress.com.

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 documentation uses the following typographical and syntax conventions:

Convention	Description
Bold	Bold typeface indicates commands or characters the user types, provides emphasis, or the names of user interface elements.
Italic	Italic typeface indicates the title of a document, or signifies new terms.
SMALL, BOLD CAPITAL LETTERS	Small, bold capital letters indicate OpenEdge key functions and generic keyboard keys; for example, **GET** and **CTRL**.
KEY1+KEY2	A plus sign between key names indicates a **simultaneous** key sequence: you press and hold down the first key while pressing the second key. For example, **CTRL+X**.
KEY1 KEY2	A space between key names indicates a **sequential** key sequence: you press and release the first key, then press another key. For example, **ESCAPE H**.
Syntax:	
`Fixed width`	A fixed-width font is used in syntax, code examples, system output, and file names.
`Fixed-width italics`	Fixed-width italics indicate variables in syntax.
`Fixed-width bold`	Fixed-width bold italic indicates variables in syntax with special emphasis.

Convention	Description
UPPERCASE fixed width	ABL keywords in syntax and code examples are almost always shown in upper case. Although shown in uppercase, you can type ABL keywords in either uppercase or lowercase in a procedure or class.
Period (.) or colon (:)	All statements except DO, FOR, FUNCTION, PROCEDURE, and REPEAT end with a period. DO, FOR, FUNCTION, PROCEDURE, and REPEAT statements can end with either a period or a colon.
[ ]	Large brackets indicate the items within them are optional.
[ ]	Small brackets are part of ABL.
{ }	Large braces indicate the items within them are required. They are used to simplify complex syntax diagrams.
{ }	Small braces are part of ABL. For example, a called external procedure must use braces when referencing arguments passed by a calling procedure.
|	A vertical bar indicates a choice.
. . .	Ellipses indicate repetition: you can choose one or more of the preceding items.

Examples of syntax descriptions

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

Syntax

```
ACCUM aggregate expression
```

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

```
FOR EACH Customer NO-LOCK:
  DISPLAY Customer.Name.
END.
```

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

Syntax

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

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

Syntax

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

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

Syntax

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

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

Syntax

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

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

Syntax

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

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

Syntax

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

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

Syntax

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

Long syntax descriptions split across lines

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

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

Syntax

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

Complex syntax descriptions with both required and optional elements

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

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

Syntax

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

OpenEdge messages

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

- **Execution messages** inform you of errors encountered while OpenEdge is running a procedure; for example, if OpenEdge cannot find a record with a specified index field value.

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

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

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

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

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

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

Obtaining more information about OpenEdge messages

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

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

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

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

1. Start the Procedure Editor:

   ```
   OpenEdge-install-dir/bin/pro
   ```

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

1

Introducing OpenEdge Replication

This chapter provides an introduction to data replication in general and OpenEdge® Replication specifically.

For details, see the following topics:

- What is data replication
- OpenEdge Replication terminology
- Source and target architecture requirements
- OpenEdge Replication activity during normal processing
- What happens if there is a database failure
- OpenEdge Replication utilities and commands

What is data replication

In general, data replication has two major real-time functions:

- To distribute copies of information to one or more sites
- To keep data constantly available to customers, even if there is a database failure

Using OpenEdge Replication, you can replicate a local OpenEdge database, known as the *source database*, to up to two remote OpenEdge databases, known as *target databases*, that are running on one or more machines. This duplication allows you to keep OpenEdge databases identical while also providing a hot standby in case a database fails. If a database does fail, a replica becomes active, ensuring that mission-critical data is available 24 hours a day, seven days a week to your users.

Once you install, configure, and start OpenEdge Replication, replication happens automatically, ensuring minimal or no disruption in the event of unplanned downtime or disaster.

How OpenEdge Replication works with two databases

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 (for example, if the primary machine were to shut down unexpectedly), you can move all database update activity from the source database to the target database through the transition process.

Once the primary machine becomes available again, you see another target available, and can fail over 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 overall 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, depending on how you set the transition properties), making the secondary database (formerly a target) a source database.
5. All database activity is moved to the secondary database, which is now functioning as a source database. The secondary database now becomes the production database.
6. Once the machine hosting the primary database is fixed, the replication process continues as follows:
   a. 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.
   b. The new target can receive replicated data from the current source.
   c. At a convenient time, you can perform the failback process to return the databases to the roles originally established, with the primary database as the source and the secondary database as the target. You can do this by transitioning the secondary database using the failover command.

How OpenEdge Replication works with a replication set

Adding a Replication Set, to an OpenEdge Replication configuration, changes the failover activity. The biggest benefit of a Replication Set is the ability to transition all available databases together. The goal of the Replication Set is to minimize the risk of a single point of failure and the need to rebase.

With a Replication Set, if a Replication agent loses communication contact with the Replication server (for example, if the primary machine were to shut down unexpectedly), you can move all database update activity from the source database one of the target databases, and the other target becomes an agent to the new source.

Once the primary machine becomes available again, you can move 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.

An additional benefit of the Replication Set, is continued replication during planned downtime. For example, you can manually transition your Replication Set with the `dsrutil transition failover` command, and perform maintenance on the primary machine. The activities are moved to the secondary machine (formerly a target, but now the source). At this point, the primary machine can be shut down for maintenance, and later brought back online. When back online, the database on the primary machine can be synchronized with the new source. At the same time, replication is still running to the other target, preventing replication from a single point of failure.

The overall 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, depending on how you set the transition properties), making the secondary database (formerly a target) a source database.
5. All database activity is moved to the secondary database, which is now functioning as a source database. The secondary database now becomes the production database.
6. The second target transitions to be an agent of the new secondary database.

Key features of OpenEdge Replication

Data replication as performed by OpenEdge Replication provides the following features:

- Automated, real-time replication of databases for failover or disaster recovery
- A choice of one or two target database configurations for a single source database
- Online or offline backup of source and target databases
- Data integrity between source and target databases
- Failover functionality, which allows you to move database activity from a source database to a target database
- Failback functionality, which allows you to move back database activity from a target database to a source database
- The continuation of source database activity while administration tasks are being performed
- Replication activity reporting

OpenEdge Replication terminology

The following sections provide a brief introduction to OpenEdge Replication terminology and to the application's supporting architecture.

- Primary and secondary databases on page 22
- Source and target databases on page 22
- OpenEdge Replication property files on page 26

- OpenEdge Replication server on page 26
- OpenEdge Replication agent on page 27
- A Replication Set on page 25
- OpenEdge Replication Model on page 28
- Failover on page 29
- Failback on page 29
- Transition on page 29

Primary and secondary databases

There are two distinct database roles within OpenEdge Replication: the *primary database* and the *secondary database*.

The primary database is the database that is updated from your application; you initially enable the primary database as the OpenEdge Replication source database.

The secondary database is the initial replica of the primary database; you initially enable the secondary database as the OpenEdge Replication target database. For each source database, you can configure up to two target databases.

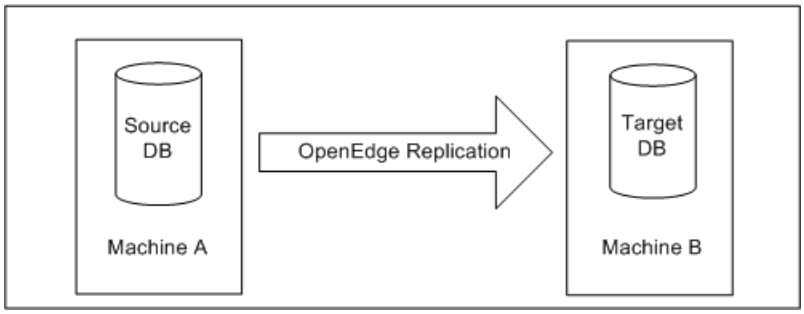
Source and target databases

Within the context of OpenEdge Replication, you set up one primary database and one or two secondary databases. You set up the primary database as the source database, and the secondary database(s) as the target database.

The source database and the target database must be running the same version of Progress OpenEdge and on the same platform.

As shown in the figure below, the OpenEdge Replication process copies the source database onto the target database and keeps those two databases identical. To accomplish this, OpenEdge Replication moves data from the source to the target in blocks of the same format as those written to database after-imaging files.

Figure 1: OpenEdge Replication from one site to another



The figure above also shows the source and target databases as residing on separate machines, which is the recommended configuration. Setting up the source and target on separate machines allows the target database to continue if the machine hosting the source database fails.

The source database

The source database is the database where users do their work and make database updates. The database has both write access and read access.

In addition, the source database:

- Is not considered a source database until you enable it as an OpenEdge Replication source database.
- Is the database from which the OpenEdge Replication server replicates data to the target database.
- Uses after-imaging to capture all database activity performed. 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 transaction log records containing a description of all database changes to the after-image files.

The target database

The target database is an identical copy of the source database. You can set up either one target database or two target databases (for additional redundancy). The source and target databases should reside on separate machines so the target can run if the source machine fails.

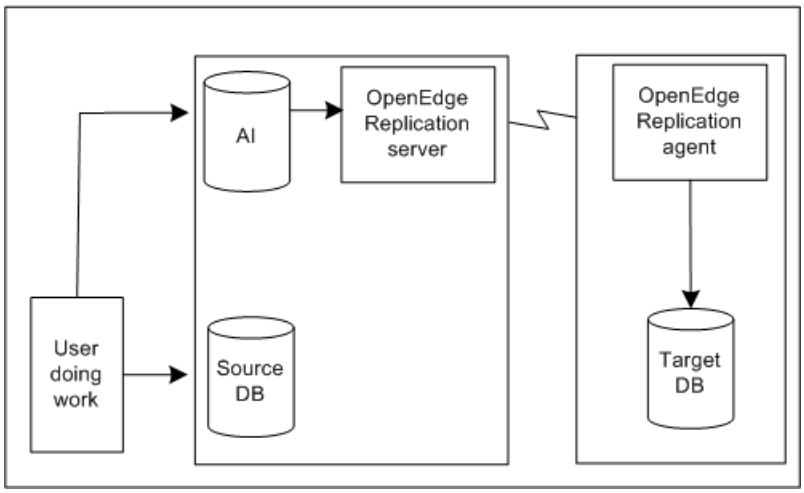
The target database:

- 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 source database.
- Does not allow database updates by anything other than the OpenEdge Replication agent.
- 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 Accessing a running replication-enabled database on page 81. See Utilities and OpenEdge Replication on page 210 for supported non-database write activity.

You can back up an online or an offline target database. For details, see Backing up an online target database on page 231.

The figure below shows the source and target databases in the OpenEdge Replication model.

Figure 2: Source and target databases



Other source and target database characteristics

When you enable OpenEdge Replication, it is important to consider its effects on the database functions and features described in this section.

- Target quiet points on page 24
- Schema lock on page 24

Target quiet points

If there is a quiet point enabled on the target, replication stops until the target quiet point is disabled.

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 a server property (Schema-Lock-Action) to specify the action an agent should take if an exclusive schema lock is not granted. You can set the property such that the agent waits until the exclusive schema lock is granted, or you can choose to have the agent force an exclusive schema lock. In this latter instance, 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.

Schema-Lock-Action and other properties are described in Configuring the OpenEdge Replication property files on page 62.

A Replication Set

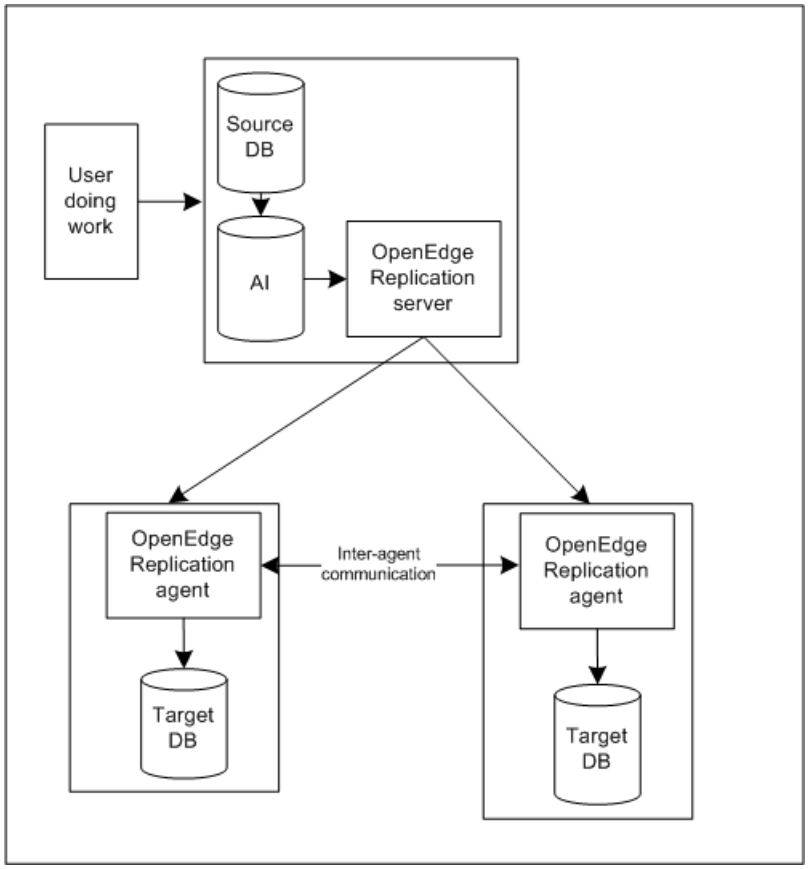
A Replication Set is a two-target Replication configuration where the two targets are aware of each other, and are configured to continue replication in the event of a server failure.

The awareness between the targets allow for one of the targets to transition to the source, and synchronize with the remaining target. You can configure a Replication Set to prioritize which target is responsible for transitioning to the source. For example, a Replication Set can be configured so that the secondary target transitions to a source when both the original source and the primary target are lost. When configuring a Replication Set, all databases must have after-imaging enabled.

Communication between agents is required for Replication Sets. This is known as inter-agent communication. The initial connection between agents occurs during the startup of the agents. After the server has connected to all of the agents, it enters a state for inter-agent connection initialization. When the server is in this state, it sends a message to each agent with the connection information needed to connect to the other agent. When each connected agent receives the message, the agent calls a message dispatch function to handle connecting to the other agent. The listening agents process inter-agent connection requests in the same manner that server-agent connections are processed. After the connected agent has completed the inter-agent network with the other agent, it sends a confirmation message to the server to let the server know that the inter-agent network has been established. Once the server has received confirmation that the inter-agent network has been established from all agents, the server begins startup synchronization. With the new inter-agent communication network, agents can ping each other to determine if the agent is still alive and connected.

The figure below shows the source and target databases, and inter-agent communication in an OpenEdge Replication Set model.

Figure 3: Replication Set



OpenEdge Replication property files

The OpenEdge Replication source database and target database properties are stored in *property files*. The properties whose values you set in the property files control different aspects of your OpenEdge Replication environment.

OpenEdge Replication provides a sample of a source property file (named `source.repl.properties`) and a target property file (named `target.repl.properties`). You can make a copy of each of these sample files and modify them to match your desired OpenEdge Replication settings.

If you want, you can maintain identical property files on both the source and target database directories. In this instance, each file must contain full server and agent details.

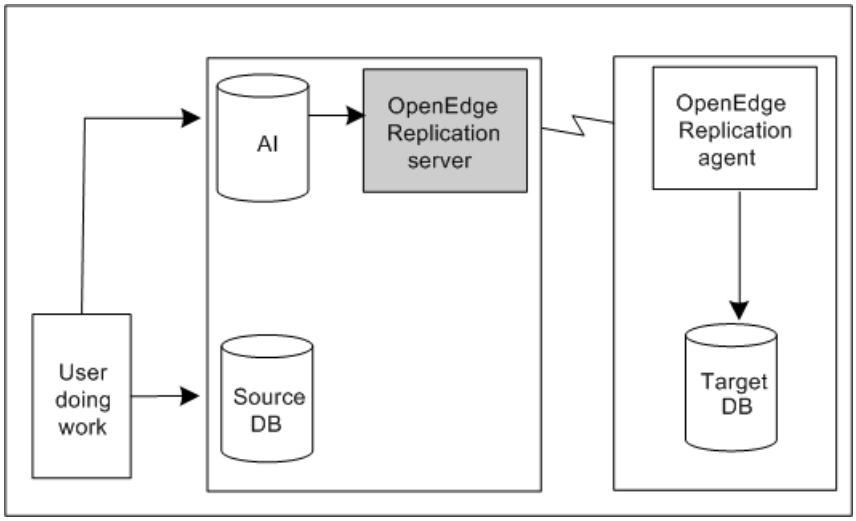
For more information about configuring the property files, see Configuring the OpenEdge Replication property files on page 62.

OpenEdge Replication server

For OpenEdge Replication to succeed in keeping the source and target databases identical, communication must occur between the two databases, allowing propagation of transaction log records from the source 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, if there are two target databases). Also, the OpenEdge Replication server process provides communications for startup, schema locks, and server-side recovery if a failure occurs.

The following figure highlights the OpenEdge Replication server in the OpenEdge Replication model.

Figure 4: OpenEdge Replication server



The OpenEdge Replication server:

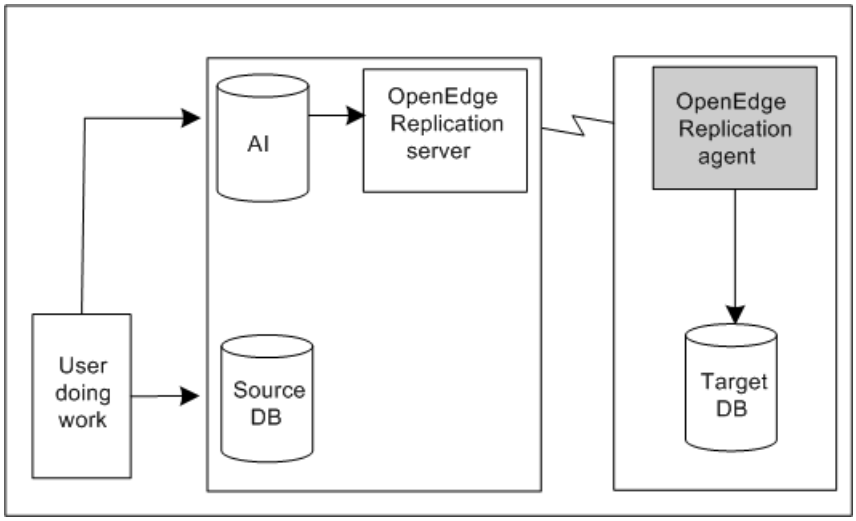
- Must reside on the source database machine.
- Connects to the source database and establishes and maintains communications with the database server.
- Establishes, maintains, and controls communication with the OpenEdge Replication agents.
- Sends source database updates to the target database to keep the databases identical. This is achieved by using the AI transaction log to send AI blocks of information to the target.

OpenEdge Replication agent

For the OpenEdge Replication server to succeed in keeping the source and target databases identical, it requires that an OpenEdge Replication agent be present on the target database 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 details about which actions to follow if the connection to the OpenEdge Replication server is lost.

The following figure highlights the OpenEdge Replication agent process in the OpenEdge Replication model.

Figure 5: OpenEdge Replication agent



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

- Must reside on the target database machine.
- 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, which enforces user read-only functionality while providing the benefits of OpenEdge Replication multi-user access to a database.

  For more information about Enhanced Read-Only mode, see Enhanced Read-Only mode on page 28.

Designating a critical agent

OpenEdge Replication allows a maximum of two agents per server. You can designate one agent as a *critical agent*, which means that transition will start automatically for that agent's target database during failure processing. For example, if you have two target databases, only the database managed by a critical agent will automatically transition during failure processing.

You can define only one critical agent. If you configure a second critical agent, it will be changed to a noncritical agent. All agents by default are noncritical.

You identify an agent as critical in the source database property file. For more information, see Configuring the OpenEdge Replication property files on page 62.

Enhanced Read-Only mode

When the OpenEdge Replication agent is running, the target 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.

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 client read-only functionality, is a database server concept. The 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 on the source database, failover transitions the target database.

- If you have a Replication Set defined, the primary target transitions to the source, and the other target becomes a target of the new source.
- Without a Replication Set, one or both targets transition to a source or a normal (non-Replication-enabled) OpenEdge database, however the targets cannot synchronize with each other after the transition, and a rebase is required to reestablish replication.

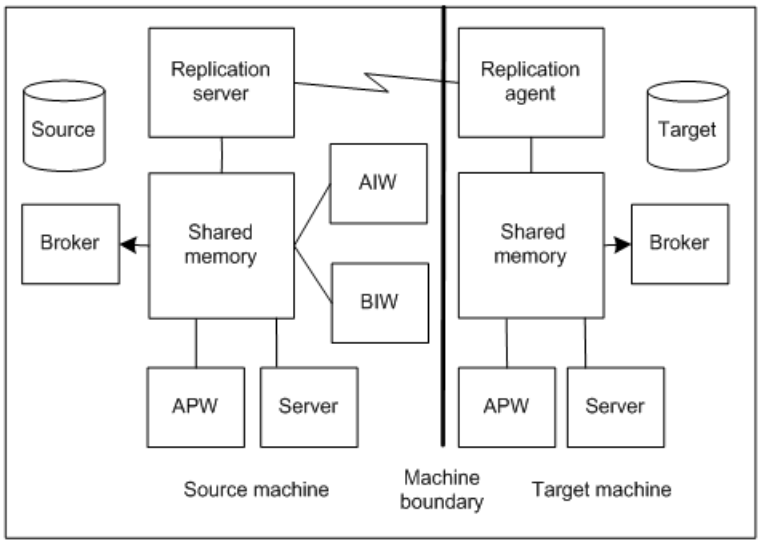
  At this point, all future connections to the target database(s) are full access. All current connections are disconnected; when they reconnect, they are full access. See OpenEdge Replication activity during normal processing on page 30 for more information about the transitioning of a target database to a normal OpenEdge database.

OpenEdge Replication Model

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

The following figure illustrates the OpenEdge Replication model coexisting in a standard database environment with TCPIP communication.

Figure 6: OpenEdge Replication model



Although the source and target databases **can** reside on the same machine, it is beneficial to locate them on separate machines so the target can run if the source machine fails.

The Asynchronous Page Writer (APW) keeps writable buffers at a low count. The APW ensures that a supply of empty buffers is available so the database engine does not have to wait for database buffers to be written to disk.

The After-image Writer (AIW) improves performance by continually writing after-image (AI) buffers to disk soon after OpenEdge fills the buffers. The after-imaging feature lets you recover a database that was damaged when a failure caused the loss of the database or primary recovery (BI) area. When you enable after-imaging, OpenEdge writes notes to the after-image files that contain a description of all changes to the database. You can run only one AIW process per database.

Failover

In the event of a failure of either the primary database or the machine on which it runs, *failover* occurs. During failover, database activity is moved to a secondary database.

Failback

When the primary machine is again available after a failover has occurred, *failback* can occur, allowing all database update activity to then occur on that machine. 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.

You typically run the failback process after the following events occur:

1. A source database has failed.
2. Database processing has moved to the secondary (target) database.
3. The source database been repaired, and you are ready to reinstate the primary database as the source.

Transition

Transition refers to the changing of the database role to perform failover. The database role may change when:

- There is a failure on the primary machine (which is hosting the source database).
- The primary machine is again up and running and you want to move activity from the secondary database back to the primary database.

If a database failure, such as a lost TCP/IP connection between the OpenEdge Replication server and the OpenEdge Replication agent, does occur, failure processing starts. If a failure occurs on the OpenEdge Replication server, the server starts a process known as *failure recovery*. If a failure occurs on the OpenEdge Replication agent, the agent prepares to perform *transition*. Transition takes place if the connection to the server remains lost.

There are two versions of transition: automatic and manual. For more information about automatic and manual transition, see Choosing a hot standby database on page 32.

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 supported platform.
- The database block sizes.
- If large file support is enabled, it must be enabled on both databases.
- Before-Image (BI) block sizes.

In addition, the values on the target database must be greater than or equal to the values on the source for the following database startup parameters:

- Lock Table Entries (`-L`)
- Number of Users (`-n`)
- Maximum JTA Transactions (`-Maxxids`)
- The sum of Number of Users (`-n`) and Maximum Servers (-Mn)

If the values for the target are less than the source, the agent will not start. For details about these parameters, see *OpenEdge Data Management: Database Administration*.

For added protection against data loss, locate the target database on a different machine from the source.

OpenEdge Replication activity during normal processing

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, by default, not allowed. However, you can change this default if you want. (See the description of the `defer-agent-startup` property in Choosing the OpenEdge Replication agent startup mode on page 74 for more information.)

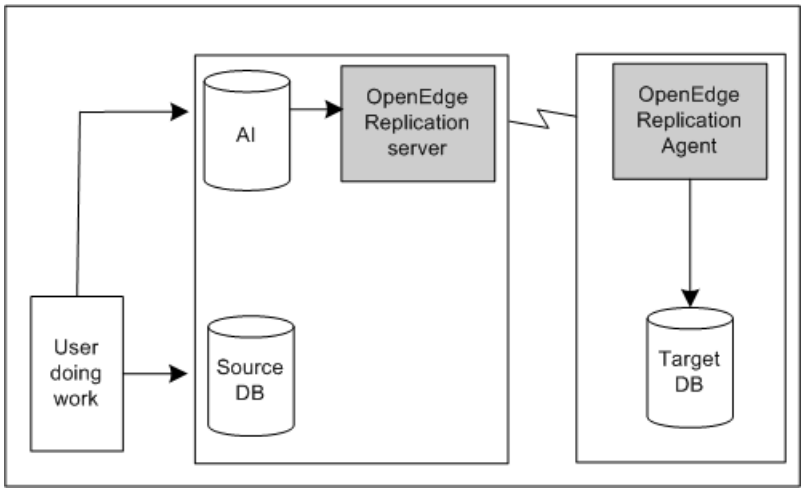
If a process attempts to connect to either database during this phase, the process will block until OpenEdge Replication has completed its work and normal database activity starts. See Accessing a running replication-enabled database on page 81 for further details about what access is permitted and when it is permitted.

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.

The following figure shows AI blocks sent from the server to the agent.

Figure 7: AI blocks sent from server to agent

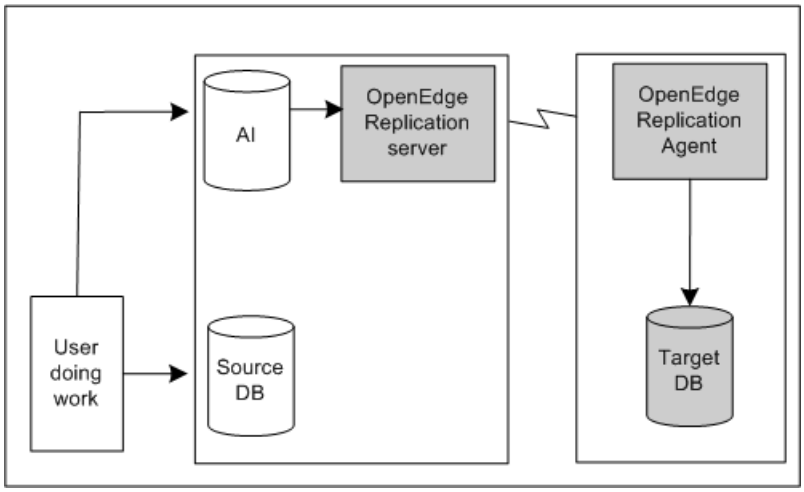


Normal target database activity

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

This continuous roll forward is shown in the following figure.

Figure 8: Continuous roll forward on agent



For the most part, the OpenEdge Replication agent performs no specific action for individual transaction log records other than applying them to the target database, as the figure above 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.

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. You must delete this file whenever you restore the source or target database, or when you re-enable OpenEdge Replication.

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.

What happens if there is a database failure

If a database failure—for example, a lost TCP/IP connection between the OpenEdge Replication server and the OpenEdge Replication agent—does occur, failure processing starts. If a failure occurs on the OpenEdge Replication server, the server starts a process known as *failure recovery*. If a failure occurs on the OpenEdge Replication agent, the agent prepares to perform *transition*. Transition takes place if the connection to the server remains lost.

Choosing a hot standby database

When there is a failure in communication between the Replication server on the primary machine and the Replication agent on the secondary machine, all database activity is failed over to the secondary machine. If failover occurs, the Replication-enabled target database is available as a *hot standby*. A hot standby database is a database that is updated and ready to go immediately.

You can have up to two target databases per source database. Only one of the target databases can be set to transition automatically.

You can identify only one target database as critical because you do not want users to be updating two different databases after automatic transition has occurred. If you have a second target database, that database is not transitioned unless you manually transition it. This provides you with options for your hot standby choice. You can find details about automatic transition and choosing one target database as critical in Configuring the OpenEdge Replication property files on page 62.

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. It is possible that 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 DSRUTIL utility on page 146 and Monitoring latency between the source and target databases on page 163.

The advantage to automatic 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 set for transition 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.

For OpenEdge Replication to continue normal operations after a failure and the re-establishment of a connection, the OpenEdge Replication server and the OpenEdge Replication agent must perform *synchronization*.

The sections that follow describe synchronization and its role in communication failure recovery, including transition.

Synchronization between server and agent

Database integrity is a major concern of all database users and is compounded when replication is in use, since two or more databases must remain identical. OpenEdge Replication achieves database integrity by performing synchronization. Synchronization ensures that the source database and the target database are identical.

Synchronization is performed by the OpenEdge Replication server and the OpenEdge Replication agent during database startup, replication startup, and failure recovery processing.

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, and the database changes must be replicated to any and all target databases. During the startup synchronization process, all database activity that might have occurred during database startup is replicated to the target database.

If the server detects a communication failure with the agent

After a communication failure with the OpenEdge Replication agent, the OpenEdge Replication server performs a failure recovery process. During failure recovery, the server attempts to recommunicate with the agent or agents that have lost connection with it.

If the failure recovery operation is successful and 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, however, and AI extents continue to fill.

If the agent detects a communication failure with the server

When the OpenEdge Replication agent detects a communication failure with the OpenEdge Replication server, the agent performs an action called transition. You can determine how long you want the agent to wait to begin transition by setting a length-of-wait-time value in the source database property file.

If the OpenEdge Replication agent receives a communication attempt from the server during the length of wait time specified in the property file and the connection is successfully 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 from the server within the length of time you set, one of two events occurs:

- If you have configured automatic transition in the property file, the target database is transitioned. If you have a Replication Set defined, one target transitions to a source and continues replicating to the other target.

  If no Replication Set is defined, the target begins transition according to the defined transition properties, and becomes a source or a normal (non-Replication-enabled) OpenEdge database.

- If you have configured manual transition in the property file, the OpenEdge Replication agent continues to wait for a connection until either:

- You perform manual transition, using the DSRUTIL utility.
- You shut down the database.

For more information about the DSRUTIL utility, see DSRUTIL utility on page 146.

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 DSRUTIL utility on page 146.
- The DSRUTIL utility monitor capabilities, which allow you to monitor OpenEdge Replication. See OpenEdge Replication DSRUTIL MONITOR on page 162.
- Virtual System Tables, which allow you to access OpenEdge Replication through ABL or SQL. See Virtual system tables for OpenEdge Replication on page 181.

2

OpenEdge Replication and After-imaging

To use OpenEdge Replication, you must set up after-imaging on the source database. This chapter describes the role of after-imaging in replication.

For details, see the following topics:

- Understanding OpenEdge Replication after-image requirements
- Calculating current database after-image volume
- The role of after-image extents in the Replication process
- Determining OpenEdge Replication network bandwidth
- Ensuring success when using after-imaging
- OpenEdge Replication and after-imaging: a summary

Understanding OpenEdge Replication after-image requirements

To use OpenEdge Replication, you must activate after-imaging on the source database. Additionally, if you define a Replication Set, you must activate after-imaging on your target databases as well. When you enable after-imaging, the database engine writes notes containing a description of all database changes to the after-image logs. You can use this information generated by after-imaging to recover a database that was damaged when a failure caused the loss of either the database itself or its primary recovery (before-image) area.

Before you start using OpenEdge Replication, it is recommended that you calculate how much after-image volume, on average, 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 you need.

Although you cannot calculate exact data volume, since it is dependent on the type of transactions within the application and how much activity is going on against the database, you can arrive at an estimate if you know the average amount of after-image volume on your system.

The details in the following sections provide information related to:

- Calculating current database after-image volume
- The role of the after-image extents in the Replication process
- Using after-imaging with OpenEdge Replication

For more general information about database after-imaging, see *OpenEdge Data Management: Database Administration*.

Calculating current database 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 the average load and the maximum load may be enough to saturate the network, causing additional network problems.

You can calculate after-image volume by using any of the following typical methods:

- After-image virtual system table (VST) information
- The RFUTIL command
- The PROMON command
- 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 relevantinformation are _AiLog-TotWrites and _AiLog-BytesWritn. The _AiLog-TotWrites field's value is an integer representing the number of after-image blocks written to after-image files.

Refer to the 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 you intend to establish as 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: AI Log** appears, as shown:

```
01/23/12        Activity: AI Log
14:29:40        01/23/12 14:26 to 01/23/12 14:29 (3 min 9 sec)

                              Total         Per Min      Per Sec      Per Tx

Total AI writes                   2               1         0.01        1.00
AIW AI writes                     0               0         0.00        0.00
Records written                  68              22         0.36       34.00
Bytes written                  7379            2343        39.04     3689.50
Busy buffer waits                 0               0         0.00        0.00
Buffer not avail                  0               0         0.00        0.00
Partial writes                    1               0         0.01        0.50
Log force waits                   0               0         0.00        0.00
 Enter <return>, A, L, R, S, U, Z, P, T, or X (? for help):
```

Notice the elapsed time (shown here in bold) on the second line of the sample log.

Example

To gather a sample for 60 minutes:

1. 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.
2. Record the number from the **Total AI 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 you gather, the more accurate the average is.
3. 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** (**AI Log**). The after-image block size is the second-to-last entry.
4. Take this number and divide it by 1024 to arrive at the block size (8192 / 1024 = 8).
5. 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 relevant information are _BiLog-TotWrites and _BiLog-BytesWritn. The _BiLog-TotWrites field's value is an integer representing the number of before-image blocks written to before-image files. 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 for the database, this method is the only choice in calculating after-image volume. Note that you must set up after-imaging for any database you intend to use as an OpenEdge Replication source database.

The role of after-image extents in the Replication process

After-image extents contain a series of notes grouped together into after-image blocks. In the event of a source database failure, you can use the after-image extents with the database roll-forward recovery process to restore the source database to its pre-failure condition, without losing completed transactions that occurred since the last database backup.

There are two after-image extent types—fixed-length or variable-length— and there are five after-image extent states—empty, busy, locked, full, or archived.

After-image extent types

The two types of after-image extents are *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 the following table.

Table 1: After-image extent types: benefits and drawbacks

Type	Benefit	Drawback
Fixed-length	**Performance asset** — 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.	**Full extent management** — It is possible for multiple extents to fill within a cycle of extent management. You must account for this possibility in after-image planning and implementation.
Variable-length	**Performance loss** — 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.	**Full disks** — Extents could grow to take up all of the available disk space. **Larger file management** — Management of variable extents can involve working with very large files, since the extents can grow to any size. Take into account the possibility that you may need to back up or move these large files.

After-image extent states

The following table describes the five different after-image extent states that can exist when a database is enabled as an OpenEdge Replication source database.

Table 2: After-image extent states

After-image extent state	Description
EMPTY	The after-image extent is empty and ready to be used by the database.

After-image extent state	Description
BUSY	The after-image extent is currently being written to by the database. Transaction log records continue to accumulate until an extent switch is performed. An extent switch occurs when the BUSY AI extent becomes full, or a database utility executes a switch. For example, RFUTIL AIMAGE NEW explicitly causes an extent switch, or an online backup performs an extent switch as part of its processing. For databases running OpenEdge Replication, a BUSY extent state is changed to LOCKED when the extent switch occurs. For databases without replication, the state is changed to FULL.
LOCKED	The after-image extent is no longer being written to, but has not been replicated in its entirety to the target database(s). No new logs are written to the extent,but it cannot be emptied until all transaction log records have been sent to and applied by the Replication agent. Once all the transaction log records in the extent have been replicated, the state is changed as follows: • If you are using AI File Management in addition to Replication, the state changed to ARCHIVED. • If you are not using AI File Management, the state is changed to FULL.
FULL	The after-image extent is FULL and is ready to be emptied. (All processing by Replication and AI File Management is completed.)
ARCHIVED	The after-image extent has been archived by the AI File Management Utility, but has not been fully replicated to the Replication target database. For details about the utility, see *OpenEdge Data Management: Database Administration*.

As users perform database changes, the actions required to make those changes are recorded as transaction log records, which are organized into AI blocks and then written into the AI file. 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 blocks to the target database.

When a source 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 OpenEdge Replication is restarted, all non-replicated database activity is replicated to the target using the data in the after-image extents. It is essential, therefore, 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 database.

Reusing after-image space is accomplished by emptying FULL-UNLOCKED after-image extents using the following command:

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

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 be enough 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 a database that is enabled as a source database. These additional transaction log records require more after-image space, referred to as *replication overhead*. In order to calculate the replication overhead, the estimated after-image size information (obtained in The role of after-image extents in the Replication process on page 38) must be multiplied by 1.5.

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 3: After-image extent sizing calculations (hourly data) on page 41 and Table 4: After-image extent sizing calculations (weekly data) on page 42 .

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 the following table are performed.

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

Description	Measurement/estimate
Average after-image data generated in one typical work hour	72.5MB, which is the average of the following: • 70MB in hour 1 • 67MB in hour 2 • 79MB in hour 3 • 74MB in hour 4
OpenEdge Replication after-image overhead	50%, or 1.5 multiplier
Number of processing hours during a typical workday	16 hours per day

Description	Measurement/estimate
Number of processing hours during the typical workweek	80 (16 hours per day x 5 days)
Estimate of after-image data generated in one day	1160MB, or 1.13GB (72.5MB x 16 hours per day)
Estimate of after-image data generated in one workweek	5800MB, or 5.66GB (1160MB x 5 days)
Estimate of after-image data generated in one hour with OpenEdge Replication overhead	109MB (72.5MB x 1.5)
Estimate of after-image data generated in one day with OpenEdge Replication overhead	1740MB (1160MB x 1.5)
Estimate of after-image data generated in one workweek with OpenEdge Replication overhead	8700MB (1740MB x 5 days)
Single fixed after-image extent size	**450MB (109MB x 4 hours)**
Total number of after-image extents necessary to handle one week of processing	**19 extents (8700MB / 450MB)**

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 the following table are performed.

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

Description	Measurement/estimate
Total after-image data generated in one typical week	15000MB, or 14.6 GB
OpenEdge Replication after-image overhead	50%, or 1.5 multiplier
Number of processing hours during a typical workday	16 hours per day
Number of processing hours during the typical workweek	112 hours (16 hours per day x 7 days)
After-image data generated in one hour	134MB(15000MB / 112 hrs in week)
After-image data generated in one day	2144MB, or 2.09GB(134MB x 16 hours)
Estimate of after-image data generated in one hour with OpenEdge Replication overhead	201MB (134MB x 1.5)
Estimate of after-image data generated in one day with OpenEdge Replication overhead	3216MB, or 3.14GB(2144GB x 1.5)

Description	Measurement/estimate
Estimate of after-image data generated in one workweek with OpenEdge Replication overhead	22512MB (3,216MB x 7 days)
Single fixed after-image extent size	**804MB (201MB x 4 hours)**
Total number of after-image extents necessary to handle one week of processing	**28 extents (22512MB / 804MB)**

Using variable extents

Estimating the volume of after-image extent size is not necessary when you are 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.

After-image interdependency between source and target databases

Source and target databases are interdependent in two ways. First, the target database requires after-image blocks and transaction log records (notes) generated by the source database in order to remain synchronized with the source database. All source database update operations generate transaction log records 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 remain in a LOCKED state until they are replicated in their entirety. Source database activity occurring without replication actively being performed relies upon the available space in the 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 determining the size of after-image extents for a source database. When a database is enabled as a source database, the database generates additional transaction log records that are written into the recovery log (BI) and the after-image log. You must consider the overhead generated by these additional transaction log records as you develop and implement successful after-image sizing and management schemes for OpenEdge Replication.

The replication-specific transaction log records group logical operations and consist of a begin operation and an end operation.

Determining OpenEdge Replication network bandwidth

There is additional replication overhead that is required to transfer the after-image blocks over the network to the target system. This additional overhead is estimated at ten percent over the amount of the message volume sent between the server and agent 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 means fewer messages are sent, which lowers 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.

The presence of after-image data is the most significant determinant in how much bandwidth is 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:

```
AID x AIRM = AIDR
AIDR x AIRO = AIDRO
AIDRO X (PRD) = AIDROP
```

`AID`

The total size of all the after-image blocks.

`AIRM`

The value 1.5; the multiplier for additional after-image notes due to OpenEdge Replication.

`AIDR`

The number of after-image blocks with OpenEdge Replication.

`AIRO`

The value 1.1; the multiplier for OpenEdge Replication overhead.

`AIDROP`

The number of after-image blocks with OpenEdge Replication and overhead.

(PRD)

The period of time, as follows:

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

Example

- Consider the following example, where:
  - AID is 89MB.
  - Sample length is 60 minutes.
  - PRD is 1.

```
89MB x 1.5 = 133.5MB
133.5MB x 1.1 = 146.85MB
146.85MB x 1 = 146.85MB
```

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

- In the example that follows:
  - AID is 360 MB.
  - Sample length is 24 hours.
  - PRD is /1440.

```
360MB x 1.5 = 540MB (566, 231, 040 bytes)
5540MB X 1.1 = 594MB (622, 854, 144 bytes)
594MB/1440 = 422 KB (432, 537)
```

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 (approximately four percent) of this information 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.

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 about archiving your after-image extents, see *OpenEdge Data Management: Database Administration*.

OpenEdge Replication and after-imaging: a summary

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 also be taken into account when determining the configuration of the after-image extent sizes and archiving.

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

3

Planning for OpenEdge Replication

This chapter describes how to plan for OpenEdge Replication.

Note: For details about configuring replication, see Setting Up OpenEdge Replication on page 53.

For details, see the following topics:

- Preliminary planning tasks
- Understanding the importance of after-imaging in OpenEdge Replication planning

Preliminary planning tasks

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

1. Reviewing after-imaging information.
2. Creating a database backup plan.
3. Evaluating your production database.
4. Determining if you will be replicating an encryption-enabled database.
5. Understanding the importance of after-imaging in OpenEdge Replication planning

Step 1: Reviewing after-image information

To use OpenEdge Replication, you must activate after-imaging on the source database. You can find general details about after-imaging in OpenEdge Replication and After-imaging on page 35

For details about after-imaging specifically related to planning, see Understanding the importance of after-imaging in OpenEdge Replication planning on page 49.

If you are unfamiliar with database after-imaging in general, also *OpenEdge Data Management: Database Administration*.

Step 2: Creating a database backup plan

Prior to using 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*.

Step 3: 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 Step 4: Determining if you will be replicating an encryption-enabled database on page 48 .
- **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 can 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 from 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 database 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 causes interrupted user access to your databases.

Step 4: Determining if you will be replicating an encryption-enabled database

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 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.

The replication process for databases also enabled for encryption

The interaction between replication and encryption on a database that is 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 details about transparent data encryption and OpenEdge Replication, see Enabling transparent data encryption for an OpenEdge Replication-enabled source database on page 82. For more information about transparent data encryption in general, see *OpenEdge Data Management: Database Administration*.

Understanding the importance of after-imaging in OpenEdge Replication planning

As described in OpenEdge Replication and After-imaging on page 35 after-imaging must be in place on the source database when you are using OpenEdge Replication. Your ability to configure after-imaging with OpenEdge Replication requires that you understand and consider your business requirements. This section discusses the issues related to planning that you must consider before running after-imaging with OpenEdge Replication.

- Determining acceptable latency on page 50
- Determining acceptable target database downtime on page 50

- Determining appropriate failover behavior on page 50
- Understanding limitations and restrictions on page 51

Determining acceptable latency

Latency within OpenEdge Replication is defined as the time between an 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 is 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.

For more information about latency, see Reference on page 145.

Determining 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 can be based on a specific time frame, or on the volume of data that needs to be applied to the database to synchronize the source and target databases. Once the out of date threshold is reached and you decide to disable replication, you will have to begin the replication enablement processes to restart the replication process.

Determining appropriate failover behavior

The failover process moves database activity from a source database to a target database in the event of a failure in the source database or on the primary machine. Determining when to initiate failover directly correlates with 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 the 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 shuts 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.

Understanding 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 can 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 about restoring a database, see Restoring a source database on page 92.
- 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 about activities that are allowed and disallowed, see Utilities and OpenEdge Replication on page 210.
- OpenEdge Replication does not support databases enabled with two-phase commit.
- OpenEdge Replication does support databases enabled for JTA transactions. You must enable your source database for JTA transactions. JTA transactions are automatically enabled on target database when once both databases are started and replication activity resumes. For more information on JTA transactions, see *OpenEdge Data Management: Database Administration* and *OpenEdge Data Management: SQL Development*.
- OpenEdge Replication requires at least one ABL broker in order to function. This ABL broker must be started before any secondary broker. Specifically, the ABL broker must be the first broker started, the `-DBService` startup argument must be specified, and the broker must support ABL clients. When secondary brokers are started, you must not specify the `-DBService` startup argument.

4

Setting Up OpenEdge Replication

This chapter describes how to set up OpenEdge Replication.

Note: Before setting up OpenEdge Replication, be sure to review the details in OpenEdge Replication and After-imaging on page 35 and Planning for OpenEdge Replication on page 47.

For details, see the following topics:

- Overview
- Setting up the OpenEdge Replication databases
- Configuring the OpenEdge Replication property files
- Choosing the OpenEdge Replication agent startup mode
- Starting OpenEdge Replication
- Accessing a running replication-enabled database
- Enabling transparent data encryption for an OpenEdge Replication-enabled source database
- Stopping OpenEdge Replication

Overview

Setting up OpenEdge Replication involves the following overall steps:

- Setting up the OpenEdge Replication databases
- Configuring the OpenEdge Replication property files
- Choosing the OpenEdge Replication implementation method
- Starting the source database
- Starting the target database

These procedures are described in this chapter.

Note: The information in the following sections is provided based on the expectation that you are familiar with using OpenEdge databases and their supported commands. If you are not familiar, refer to *OpenEdge Data Management: Database Administration*.

Getting started

Before you can set up OpenEdge Replication, you must consider the following factors:

- Whether you plan to use an online or an offline backup of the source database to create the target database
- The mode you plan to use to start the agent on the target database

Choosing an online or an offline backup of the source database

As you plan for OpenEdge Replication, you identify the database you will use as a source. You then use a backup of that database to create the target database.

You can choose to use either an online backup or an offline backup of the source to create the target database. The steps you follow to set up the source database and enable it for replication are, however, dependent upon the backup method that you choose.

For the steps you follow to set up the source database, see Setting up the OpenEdge Replication databases on page 55 .

Choosing the OpenEdge Replication agent startup mode

You can choose between two agent startup modes for OpenEdge Replication: *immediate agent startup* or *deferred agent startup*.

The immediate agent startup mode requires that the OpenEdge Replication agents start before the length of time you set in the `connect time-out` agent property has expired. The connect time-out specifies how many seconds the OpenEdge Replication agent will wait for connection from the OpenEdge Replication server before the replication agent shuts itself down. 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 mode allows the source database and the OpenEdge Replication server to start without connecting to an agent. This startup mode 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 immediate agent startup. Deferred agent startup allows online backups, which will minimize the downtime of the database. For more information, see Setting up the OpenEdge Replication databases on page 55 and Setting up the source database with an online backup on page 57 .

You can also alternate between startup modes from session to session.

For details about starting the agent in either mode, see Using the immediate agent startup mode on page 75 and Using the deferred agent startup mode on page 75 .

Setting up the OpenEdge Replication databases

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

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 database you intend to use as the source:

Step 1: Back up the source database on page 55

Step 2: Create a structure file on page 55

Step 3: Enable database after-imaging on page 56

Step 4: Enable the source database for OpenEdge Replication on page 56

Step 5: Perform an incremental backup of the source database on page 57

Step 6: Where to go next on page 57

If you are using the OpenEdge Replication deferred agent startup mode and want to create a target database from an online backup of the source database, see Setting up the source database with an online backup on page 57.

Step 1: Back up the source database

You can use either PROBKUP or the OS copy utility to back up the database. However, keep in mind that if the source database has after-image files and you use the OS copy utility, after-imaging will also be enabled on the target database.

You are not required to set up after-imaging on the target database. If after-imaging is enabled, however, the management of the after-image files on the target must be handled separately from the source, requiring the same degree of effort for both databases.

For PROBKUP, use the following command:

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

Step 2: Create a structure file

Once you back up the source database, you must build a structure file (`.st`). Use the following command to build this file:

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

(Later, you will copy this structure file from the source to the target machine.)

Step 3: Enable database after-imaging

To use OpenEdge Replication, you must enable after-imaging on your source database. If AI is already enabled on the database machine, skip this section and go to Step 4: Enable the source database for OpenEdge Replication on page 56.

Enable AI on your source database

Use the steps that follow to enable after-imaging on your source database:

1. Create a new structure file—named, for example, *source-db-name*_ai.st—and edit it to add after-imaging.

   The following sample shows the structure file:

   ```
   # addai.st
   a C:\OpenEdge\WRK\ f 1024
   a C:\OpenEdge\WRK\ f 2048
   ```

   Be sure to give the new structure file a different name (for example, the one shown in the next step) from the one you created in Step 2: Create a structure file on page 55.

2. Apply *source-db-name*_ai.st to the source database.

   The syntax follows:

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

3. Back up the database.

   The syntax follows:

   ```
   probkup source-db-name source-db-backup-name
   ```

4. Begin after-imaging.

   The syntax follows:

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

Step 4: Enable the source database for OpenEdge Replication

OpenEdge Replication requires that you enable the source database for replication **before** you start the database.

To enable the source database, use 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, use 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.

Step 6: Where to go next

After you complete the tasks described in this section, continue with Setting up the OpenEdge Replication target database on page 59.

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. To enable online backups, you must implement the deferred agent startup mode, which is described in Using the deferred agent startup mode on page 75. You also must enable AI on the source database.

Perform the following steps on the source database:

Step 1: Create a structure file on page 57

Step 2: Enable the database for after-imaging on page 58

Step 3: Enable the source database for OpenEdge Replication on page 59

Step 4: Perform an online backup on page 59

Step 5: Configure the OpenEdge Replication server for deferred agent startup on page 59

Step 6: Where to go next on page 59

Step 1: Create a structure file

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

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

(Later you will copy this structure file from the source to the target machine.)

Step 2: Enable the database for after-imaging

To use OpenEdge Replication, you must enable after-imaging on the source database.

If after-imaging is already enabled for the database, skip this section and go to Step 3: Enable the source database for OpenEdge Replication on page 59.

If after-imaging is not already enabled for the database, use the procedure that follows to enable it.

Enabling after-imaging on your source database

Use the following procedure to enable after-imaging on your source database:

1. Create a new structure file—named, for example, *source-db-name*_ai.st—and edit it to add after-imaging.

   The following is sample of the structure file:

   ```
   # addai.st
   a C:\OpenEdge\WRK\ f 1024
   a C:\OpenEdge\WRK\ f 2048
   ```

   Be sure to give the new structure file a different name from the one you created in Step 1: Create a structure file on page 57.

2. Apply *source-db-name*_ai.st to the source database.

   The syntax is shown:

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

3. Back up the database.

   The syntax is shown:

   ```
   probkup source-db-name source-db-backup-name
   ```

4. Begin after-imaging.

   The syntax is 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 for replication before you start the database. To enable the source database, use the following command:

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

Step 4: Perform an online backup

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

```
probkup online source-db-name source-db-backup-name -REPLTargetCreation
```

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 (shown in bold) is set for 240 minutes, or four hours:

```
[server]
     control-agents=agent-name
     database=source-db-name
     transition=manual
     transition-timeout=1200
     defer-agent-startup=240
```

For more information, see Server properties on page 64.

Step 6: Where to go next

After you complete the tasks described in this section, continue with Setting up the OpenEdge Replication target database on page 59.

Setting up the OpenEdge Replication target database

To set up OpenEdge Replication, you must restore the backup you created of the source database to use as your target database.

The information in the following sections is provided based on the expectation that you are familiar with using OpenEdge databases and their supported commands. If you are not familiar, refer to *OpenEdge Data Management: Database Administration*.

Create the OpenEdge Replication target database

Use the following steps to create the OpenEdge Replication target database.

1. Move or copy the source backup file and the incremental backup file from the source database machine to the target database machine.
2. Copy the structure file (*source-db-name*.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 for the target 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 database directory and rename it with the name of the target database (*target-db-name*.ks).

   The source database keystore was created when you enabled transparent data encryption on the source database.

4. Restore the backup copy of the source database to the target machine.

   The syntax follows:

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

5. If the source database was backed up offline, restore the source database's incremental backup on the target machine.

   For example:

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

6. Enable the target database for OpenEdge Replication.

   The syntax follows:

```
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.

Follow these steps to enable before-image encryption on the Replication-enabled target database:

1. Shut down the source database and the target database.

Use the following command for each database:

```
proshut db-name
```

2. Disable replication on each database.

   Use the following command to disable replication on each database:

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

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

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

   When you enable before-image encryption, the before image is automatically truncated.

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

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

5. Back up the database using the following command:

```
probkup source-db-name source-db-backup-name
```

6. Copy the backup volume or volumes to the target computer.
7. Copy the source database structure file (*source-db-name*.st) to the target database directory on the secondary computer and rename it to match the target database name. 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 database directory and rename it with the name of the target database (*target-db-name*.ks).

   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 using the following command:

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

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.
12. Start both databases.

Configuring the OpenEdge Replication property files

In setting up OpenEdge Replication, you must configure the replication property files for both the OpenEdge Replication server and the OpenEdge Replication agent. The properties control how OpenEdge Replication operates within your source database and target database environment.

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.

The sample properties files are located in *`OpenEdge-install-dir`*`\properties`. You store each property file that you modify and save in the same directory as its corresponding database.

Note: Running an OpenEdge backup (using the PROBKUP utility) does not include the property files. Consider using an operating backup system utility to backup these files.

The following sections describe the sample properties files that are shipped with the product for the source and target databases and provide a definition of the property file directives. Review the samples and the property descriptions and then continue with:

- Configuring the source database property file on page 72
- Configuring the target database property file on page 74

Sample OpenEdge Replication property files

Following are samples of a source property file, a target property file, and a combined source and target database file.

Sample OpenEdge Replication source property file

The following is an excerpt from the sample source properties file that your receive with the OpenEdge Replication product:

```
[server]
    control-agents=agent1
    database=source
    transition=manual
    transition-timeout=600

[control-agent.agent1]
   name=agent1
   database=target
   host=localhost
   port=4501
   connect-timeout=120
   replication-method=async
   critical=0

[transition]
   database-role=normal
```

For information about these properties and their values, see the following:

- Server properties on page 64
- Control agent properties on page 67

Sample OpenEdge Replication target properties file

The following is an excerpt from the sample target properties file that is shipped with the OpenEdge Replication product:

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

[transition]
   database-role=normal
```

For more information about these properties and their values, see the more detailed descriptions in Agent properties on page 69 .

Sample OpenEdge Replication combined properties file

You can choose to have one property file that combines both source database and target database property settings. In the case of this combination file, you must place the file in both the source database directory and the target database directory.

The following is a sample combined properties file:

```
[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

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

[transition]
   database-role=reverse
   auto-begin-ai=0
   auto-add-ai-areas=0
   ai-structure-file=!{value-of:server.database}.addai.st
```

Server properties

Server properties are found after the `[server]` directive in the OpenEdge Replication source properties file.

Property names and values are separated by an equal sign (for example, `transition=auto)`.

The following table provides a list of server properties and their values.

Table 5: Server properties

Property name	Value	Description
`control-agents`	*`agent_name[`*`,` `agent_name``]`	Specifies a comma-separated list of OpenEdge Replication agent names. The comma must be followed by a space. The agent name must match the control agent name specified in the `[control-agent]` properties section head. `ALL` is not allowed as an agent name. Each agent must have a unique name.
`database`	*`db_name`*	Specifies the source database name.

Property name	Value	Description
defer-agent-startup	*minutes*	Specifies how long, in minutes, the server attempts to connect to an agent if the first connection attempt is unsuccessful. The default value is 0 (not active). Set *minutes* to 0, or a value between 5 and 10080, inclusive.
increment-polling-delay	*milliseconds*	Specifies the value, in milliseconds, to increase the polling delay. By default, the polling delay increment is 5 ms. For configurations with AI Streaming enabled, the default is 1 ms.[1] Set *milliseconds* to a value between 1 and 1000, inclusive. The value must be less than maximum-polling-delay.
maximum-polling-delay	*milliseconds*	Specifies the maximum value, in milliseconds, for the polling delay. By default, the maximum polling delay is 500 ms. Set *milliseconds* to a value between 500 and 1000, inclusive.
minimum-polling-delay	*milliseconds*	Specifies the minimum value, in milliseconds, for a polling delay. By default, the minimum polling delay is 5 ms. For configurations with AI Streaming enabled, the default is 20 ms.[1] Set *milliseconds* to a value between 1 and 30, inclusive. The polling delay starts at the minimum, specified by minimum-polling-delay, and automatically increases, by the amount specified by increment-polling-delay, during periods of inactivity to the maximum delay, specified by maximum-polling-delay. Once activity occurs during polling, the delay is reset to the minimum-polling-delay value.

[1] The connect message between the agent and server informs the server when AI Streaming is enabled. In response, the server sets the minimum-polling-delay to 20 ms, and the increment-polling-delay to 1 ms. These values override any previous setting, including values set in the properties file. The change to the property settings is written to the server's database log file when the first AI Streaming agent is connected.

Property name	Value	Description
repl-Keep-Alive	*seconds*	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. The minimum value is 90 seconds; there is no maximum value. The default is 300 seconds.
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, 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.

Property name	Value	Description
transition	auto | manual	Specifies how to transition the target database to a normal database. If a critical agent cannot reconnect with the source database within the time specified in the transition-timeout property, the following occurs: • auto —The agent automatically transitions the target database. • manual —The agent listens for the server until it reconnects with the source database or until the DSRUTIL *db-name* -C transition command transitions the target database. **Note:** Values are case-sensitive. Use auto or manual, not Auto or Manual.
transition-timeout	*seconds*	Specifies the number of seconds the agent waits before it performs auto-transition. This property is ignored when transition=manual. The value is incremented by the sum of the connect-timeout for all configured agents.

Control agent properties

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 match exactly 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).

The following table provides a list of control agent properties and their values.

Table 6: Control agent properties

Property name	Value	Description
connect-timeout	*seconds*	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. *seconds* is an integer greater than or equal to 120 (which is the default) and less than or equal to 86,400. Using this property means you do not have to perform 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.
critical	0 | 1	Specifies whether the agent is critical: • 1 is critical. • 0 is the default value of noncritical. A critical agent is an 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. ALL is not allowed as an agent name. Each agent must have a unique name.

Property name	Value	Description
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	Specifies the type of replication, asynchronous (async). **Note:** The value is case-sensitive. Use async , not Async.

Agent properties

Agent properties define the configuration of 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).

The following table provides a list of agent properties and their values.

Table 7: Agent properties

Property name	Value	Description
ai-streaming	0 | 1	Specifies if AI Streaming is enabled. • 0 — AI Streaming is disabled. • 1 — AI Streaming is enabled. This is the default. When AI Streaming is enabled, the replication agent utilizes multi-threading to pull the AI blocks from the source, and store and apply the data on the target. AI Streaming is enabled by default in Version 12.
connect-timeout	*seconds*	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 (which is the default) and less than or equal to 86,400. Using this property means you do not have to perform 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.

Property name	Value	Description
database	*db_name*	Specifies the target database name.
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.
listener-maxport	*port-number*	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.
listener-minport	*port-number*	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*.
maximum-polling-delay	*milliseconds*	Specifies the maximum value, in milliseconds, for a polling delay. By default, the maximum polling delay is 500 ms. Set *milliseconds* to a value between 500 and 1000, inclusive.
minimum-polling-delay	*milliseconds*	Specifies the minimum value, in milliseconds, for a polling delay. By default, the minimum polling delay is 5 ms. Set *milliseconds* to a value between 1 and 30, inclusive. The polling delay starts at the minimum, specified by minimum-polling-delay, and increases by 5 ms increments, during periods of inactivity to the maximum delay, specified by maximum-polling-delay. Once activity occurs during polling, the delay is reset to the minimum-polling-delay value.

Property name	Value	Description
name	*agent_name*	Specifies the OpenEdge Replication agent name. By convention, the *agent-name* should match the name specified in the source database properties file. *ALL* is not allowed as an agent name. Each agent must have a unique name.
rsb-cache-size	*cache-size*	Specifies the size of the Replication Streaming cache in units of an AI block, when AI Streaming is enabled (ai-streaming=1). If AI Streaming is not enabled, this property is ignored. *cache-size* is a non-negative integer that determines the size of the cache. The default is 5000.

Logging level for Replication

OpenEdge Replication allows you to set the logging level for both the server and the agent. The logging level can be set at startup through the properties files, and can be changed while the system is running through a utility. The messages generated are written to either the repl.agent.lg or the repl.server.lg file, based on where in the properties file the logging level is set, or which database the utility is run against. If a logging level is not specified, logging is off.

Logging Levels

The table that follows describes the logging levels.

Logging Level	Description
0 (None)	Logs no entries. This is equivalent to turning logging off.
1 (Errors)	Logs OpenEdge Replication error messages. This is the minimum amount of logging.
2 (Basic)	Logs "basic" information such as the start and end of transactions and connections as well as summary information. Errors logged at Level 1 are also logged. This is the suggested logging level.
3 (Verbose)	Log all messages and details, including the information logged at Level 1 and 2.

Configuring logging level in properties file

You can set your logging level when you configure your replication and agent(s) in the `repl.properties` file. For both the `[agent]` and `[server]` sections of your properties file, add the following:

```
logging-level=n
```

Where *n* is an integer between 0 and 3.

Setting logging level while Replication is running

You can set or change the logging level while Replication is running for both the Replication server and agents. Use the following command to change the level:

```
dsrutil dbname -C logginglevel new-logging-level
```

dbname

Name of the database; specify the source database for the server or a target database for an agent.

new-logging-level

The new logging level to set. It must be an integer between 0 and 3.

The new logging level remains in effect until you change it again online, or until Replication restarts. If the Replication server or agent is restarted, the logging-level value set in the `repl.properties` file is used.

Configuring the source database property file

Before you start OpenEdge Replication on the source database, you must configure the replication server. You configure the server by saving and then editing the sample source properties file provided when you install OpenEdge Replication as part of OpenEdge. Each of the properties is listed and described in Server properties on page 64.

To configure the source database property file:

1. Make a copy of the sample `source.repl.properties` file, which is located by default in *OpenEdge-install-dir*\properties.
2. Name this new file *db-name*`.repl.properties`, where *db-name* is the name of your source database.
3. Place this new file in the same directory as your source database.
4. To change any of the default property values, simply modify them in the property file, as follows:
   - 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 Management or OpenEdge Explorer.
   - f you are in Windows, you can use OpenEdge Management or OpenEdge Explorer to change the properties file. You can also use Notepad.

   For information about configuring the properties file in OpenEdge Management or OpenEdge Explorer, see *OpenEdge Management and OpenEdge Explorer: Configuration*.

Properties file example

The following is an example of a `source.repl.properties` file for an OpenEdge Replication server that is configured with two agents:

```
# OpenEdge Replication properties file for a database that will be used
#  as a source database for OpenEdge Replication. This is a two agent #  configuration.
[server]
     control-agents=agent1, agent2
     database=your source db name
     transition=manual
     transition-timeout=1200

[control-agent.agent1]
     name=agent1
     database=your target db 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 db 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.
- *agent2* in the `control-agents=`*agent2* of the `[server]` section must match the control agent name specified in the `[control-agent.agent2]` section head.
- `ALL` is not allowed as an agent name. Each agent must have a unique name.

When you finish editing the file, save it.

Agent rules 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. This is because the source and target databases may not be synchronized, which will cause a failure.
- Do not rename an OpenEdge Replication agent after OpenEdge Replication has been started once.
- You can add an OpenEdge Replication agent (the maximum is two agents in total) as long as the target databases are created as instructed.

Configuring the target database property file

Before you start OpenEdge Replication on the target database, you must configure the agent. You configure the agent by saving and then editing the sample target properties file provided when you install OpenEdge Replication as part of OpenEdge. Each of the properties is listed and described in Agent properties on page 69.

Use the following steps to configure the target database property file:

1. Make a copy of the `target.repl.properties` file, which is located by default in *`OpenEdge-install-dir`*`\properties`.
2. Name this new file *`db-name`*`.repl.properties`, where *`db-name`* is the name of your target database.
3. Place this new file in the same directory as the target database.
4. To change any of the default property values, edit and save 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 or OpenEdge Explorer to change your properties file. You can also use OpenEdge Management if you have purchased a license.

   For more information about configuring the properties file in OpenEdge Management or OpenEdge Explorer, see *OpenEdge Management and OpenEdge Explorer: Configuration*.

Properties file example

The following is an example of a `target.repl.properties` file for an OpenEdge Replication agent:

```
[agent]
   name=agent1
   database=target-db-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
```

Choosing the OpenEdge Replication agent startup mode

You can choose between two OpenEdge Replication agent startup modes: immediate agent and deferred agent.

Using the immediate agent startup mode

The following is an overview of the tasks you must perform to use the immediate agent startup mode for OpenEdge Replication. Immediate agent startup mode does not allow source or target database activity until both the server and agents complete their startup and initialization phases..

To set up the immediate agent startup mode for OpenEdge Replication:

1. Using the `PROENV` utility, ensure that you have the `$DLC`, `$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 database you intend to use as the source.
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 that 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 Setting up the OpenEdge Replication databases on page 55.
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 Setting up the OpenEdge Replication target database on page 59.
5. To set OpenEdge Replication property values, do the following:
   a) Configure the server properties file.
   b) Configure the agent properties file.

   For more information, see Configuring the OpenEdge Replication property files on page 62.
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.

Using the deferred agent startup mode

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

Note that when you use 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 block until failure recovery is complete.

To use deferred agent startup, you must specify the value of 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 following:

- Configuring the OpenEdge Replication property files on page 62
- Server properties on page 64

The deferred agent startup mode 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 goes into deferred agent startup mode.
2. While in deferred agent startup mode, the OpenEdge Replication server attempts to connect every five minutes until all agent(s) are connected or until the time-out specified as the value for the `defer-agent-startup` property 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 DSRUTIL connectagent database qualifier on page 151.

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

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

Setting up the deferred agent startup implementation

To set up 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: This step 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 Setting up the source database with an online backup on page 57.

   b) Create a structure file (`.st`) of the source database.
   c) Ensure that 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: This step is not necessary if you intend to create an online backup. See Setting up the source database with an online backup on page 57.

4. Configure your OpenEdge Replication server by doing the following:

   a) Set the `defer-agent-startup` property to a valid time-out value in the OpenEdge Replication server properties file.

   b) Configure the OpenEdge Replication server properties file for your other server properties.

   For more information, see Configuring the OpenEdge Replication property files on page 62.

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 Setting up the OpenEdge Replication target database on page 59.

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 the DSRUTIL `connectagent` command to force a connection without waiting. For more information about `connectagent`, see DSRUTIL connectagent database qualifier on page 151.

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

Starting OpenEdge Replication

Once you set up the source and target databases and their corresponding property files, you can start OpenEdge Replication. Simply start your source and target databases with an OpenEdge Replication qualifier. For more information, see the following sections:

- Starting the source database on page 78
- Starting the target database on page 78

The order in which you start the databases does not matter. However:

- It is recommended that if you did not set the `defer-agent-startup` property to a non-zero value in your server properties file, you start the target database first so that the OpenEdge Replication server does not time out.
- If you do start the source database first, start the target database before the `connect-timeout` property in the `[control-agent.`*`name`*`]` section of the server properties file expires.

Alternatively, you can start OpenEdge Replication with the database utility dbman or OpenEdge Explorer (or OpenEdge Management, if you have a license).

Message logging during startup

While OpenEdge Replication is starting up, a log is produced that you can review 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 only in Replication, because neither database logging nor standard output is available.

Starting the source database

Use the following command to start the source database:

```
proserve source-db-name -DBService replserv -S {port | service name}
```

The `-DBService` argument, which is case-sensitive, instructs the broker to start the OpenEdge Replication server. Unless you are using the `defer-agent-startup` method, database activity cannot begin until OpenEdge Replication has completed its startup and initialization.

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. For more information, see Source and target architecture requirements on page 30.

If there is a configuration error, the OpenEdge Replication server will be brought down even if there is no critical agent; and messages will be sent to the database log on both the source side and the target side. (When a critical agent exists for a server, transition starts automatically for that agent's target database during failure processing.)

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 target-db-name -DBService replagent -S {port | service name }
```

The database broker will monitor the *port* or *service name* specified by 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 specify 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.

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 the handshaking 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 does 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 reinserts itself back into the AI Block write process, where the OpenEdge database is unblocked, and normal database and OpenEdge Replication activity continues.

Before synchronization begins, the OpenEdge Replication server attempts to connect to its configured agent(s). If the server cannot connect to its configured agent(s) on the first connection attempt, it goes into deferred agent startup. While in the deferred agent startup state, the OpenEdge Replication server waits five minutes, and then attempts the OpenEdge Replication agent connection again. The OpenEdge Replication server remains in this state until all agent(s) are connected, or until the time-out specified in the defer-agent-startup property is reached.

Startup, initialization, and synchronization are performed once all the agents are connected. 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 block until failure recovery is complete.

Starting OpenEdge Replication with the DBMAN utility or OpenEdge Explorer

Once you enable OpenEdge Replication and set the properties for both the source and the target databases, you can start OpenEdge Replication automatically with either the DBMAN utility or OpenEdge Explorer (or OpenEdge Management if you have a license), 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 actually 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 OpenEdge Management or OpenEdge Explorer, see *OpenEdge Management and OpenEdge Explorer: Configuration*.

Database connection considerations

As OpenEdge Replication is started, all database connections to your Replication source are disallowed until OpenEdge Replication startup and initialization phases are complete, unless you set 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.

The following table shows the results of connecting during startup and initialization based on the type of client connecting.

Table 8: Connection results during startup and initialization

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

Type of client	Connection results
AppServer Remote Client	Ends in an error stating that the broker cannot spawn the server. A message is sent to the database log.
WebSpeed Self Serve Client	Not permitted until database connections are allowed.
WebSpeed Remote Client	Ends in an error stating that the broker cannot spawn the server. A message is sent to the database log.

Accessing a running replication-enabled database

Once OpenEdge Replication is running, database access is controlled differently for the source and target. Access is also dependent on the type of product installed.

The following sections describe the different types of connections allowed.

Accessing a target database

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 OpenEdge Replication Plus is installed, any process can access the target database for read-only use. If updates are attempted, an error occurs.
- If OpenEdge Replication is installed, access to the target database is limited to the system-level tools shown in the table that follows.

Limited logins are allowed immediately after the agent completes its startup. Normal logins that allow the access listed in the table that follows are allowed after synchronization completes.

Table 9: Standard system-level access permitted to target database

Tool	Description
Shut down	PROSHUT command to shut down the database
Monitor	PROMON utility that displays database information
AIW	A background process that writes AI buffers to disk soon after they are filled
BIW	A background process that continually writes filled BI buffers to disk
WDOG	The watchdog that cleans up after improperly terminated processes
OpenEdge Replication agent	Receives information from the OpenEdge Replication server and updates the target database

Tool	Description
OpenEdge Replication Utility DSRUTIL	Allows you to perform specific OpenEdge Replication agent and target database commands
Database agent	Provides information to OpenEdge Management or OpenEdge Explorer about the status of the database

If the OpenEdge Replication agent is transitioning the target database to a normal OpenEdge database (one not enabled for replication), critical system-level commands are allowed to access the database, as shown in the table that follows. Transition refers to the changing of the database role after a database failure and is described in detail in the sections:

- Transition on page 93
- Overview of database failure recovery on page 90

Table 10: System-level access to target database permitted during transition

System-level activity	Description
Shut down	The PROSHUT command shuts down the database.
Monitor	The PROMON utility displays database information.
WDOG	The watchdog cleans up after improperly terminated processes.

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

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

Accessing a source database

All normal database access is allowed on the source database if either OpenEdge Replication or OpenEdge Replication Plus is installed.

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 on the source database.

   Use the following command:

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

3. Update the Encryption Policy on the source database as required.

   Use the following command:

```
proutil source-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 source database for transparent data encryption.) The keystore must have the same name as the target database; if it does not, rename it once it is on the machine where the target resides.

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.

Enabling encryption for a Replication-enabled source database that is online

To enable encryption for a Replication-enabled source database that is online, use the following procedure:

1. Add the Encryption Policy Area to **both** the source database and target databases.
2. Enable encryption.

   Use the following command:

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

3. Update the Encryption Policy as required.

   Use the following command:

```
proutil 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 (*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 until the target database keystore file is available.

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—*db-name*.ks—to the target database directory immediately after encryption is enabled on the source database.

Stopping OpenEdge Replication

To shut down and stop database activity, you use the PROSHUT or PROMON utility first on the source database and then the target database.

Shutting down the source database

This section describes how to perform a shutdown of your source database.

To perform a shutdown, type 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.

You can also perform a forced (emergency) shutdown by typing the following command:

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

Note: You cannot use this command if you started the database with the No Crash Protection (-i) parameter.

The database will immediately shut down, and a warning is placed in the log file. With a forced shutdown, the AI buffers are not flushed.

After a forced shutdown, however, the source and target databases are not considered identical. At this point, OpenEdge Replication stops and you must follow the steps in Starting OpenEdge Replication on page 77 to restart it.

For more information about how to use the PROSHUT and PROMON utilities, see *OpenEdge Data Management: Database Administration.*

Shutting down the target database

You can perform a shutdown of your target database.

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

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

The following actions 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.

You can perform a forced (emergency) shutdown by typing the following command:

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

Note: You cannot use this command if you started the database with the No Crash Protection (`-i`) parameter.

The database will immediately shut down, and a warning is placed in the log file. With a forced shutdown, the AI buffers are not flushed.

After a forced shutdown, however, the source and target database are not considered to be identical. At this point, OpenEdge Replication stops and you must follow the steps in Starting OpenEdge Replication on page 77to restart it.

Terminating the OpenEdge Replication server when the source database is running

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

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

When the OpenEdge Replication server terminates activity, the server responds, based on the value of the `agent-shutdown-action` property, as follows:

- If the `agent-shutdown-action` is `NORMAL`, then the replication agent(s) terminate, and the target database stays up.
- If the `agent-shutdown-action` is `RECOVERY`, then the replication agent(s) remain active, in a stand-by state, waiting for the replication server to reconnect.

The status of the database is not affected by OpenEdge Replication server termination.

Terminating the OpenEdge Replication agent when the target database is running

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

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

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

Disabling OpenEdge Replication on the server

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

To disable OpenEdge Replication after you have terminated the server:

1. Release any pending waits.

   Use the following command:

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

2. Disable OpenEdge Replication.

   Use the following command:

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

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

Disabling OpenEdge Replication on the agent

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

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

5

Moving From Failure to Recovery with OpenEdge Replication

OpenEdge Replication allows you to handle various database failure conditions and recovery solutions.

For details, see the following topics:

- Overview of database failure recovery
- Possible types of failure
- Restoring a source database
- Transition
- OpenEdge Replication from normal activity through failure and recovery
- Transition of a Replication Set
- Recovery from transition failures
- Transition logging
- Transition properties
- Sample of a startup parameter file used by transition
- Transition property reference

Overview of database failure recovery

When you enable OpenEdge Replication, you replicate a database on a primary machine to one or two 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). If communication is not resumed, transition begins.

Once you transition the secondary database to a source database, all database update activity is *failed over* to the secondary database. If you have configured a three database Replication Set, when one of the secondary (target) databases becomes the source, the other target continues its role as a replication target, and it receives updates from the new source. If you have a three database configuration, but no Replication Set, the second target database stops receiving replication updates.

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.

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 automated functions (among others):

- 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

Possible types of failure

When OpenEdge Replication is running, normal database activity occurs. If OpenEdge Replication encounters a failure, failure processing begins.

A lost connection, in which an OpenEdge Replication server loses contact with its agents or an agent loses contact with its server, can occur for a variety of reasons. When a lost connection occurs, the source goes into failure recovery and the target goes into transition.

The reasons why a lost connection occurs are varied. For example, a database may crash, the OpenEdge Replication server or agent may terminate abnormally, or the system hosting the replication server or agent may crash.

There may also be a break in the TCP/IP connection between the OpenEdge Replication server and its agents.

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 `Repl-Keep-Alive` server property to enable pinging between the server and the agent. A ping is sent every thirty seconds. The `Repl-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, a connection failure condition is set and failure recovery begins. The minimum value is 90 seconds; there is no maximum value.

For more information about configuring `Repl-Keep-Alive`, see Server properties on page 64.

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 agents 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 database activity, as if OpenEdge Replication were 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 schema updates are being performed.
3. If the OpenEdge Replication server is able to reconnect to the OpenEdge Replication agent, it again begins processing AI blocks from the database. When it gets within ten AI blocks of the last AI block written, the OpenEdge Replication server temporarily stalls 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.
4. When synchronization is completed, the OpenEdge Replication server reinserts itself back into the AI block write process. The database is unstalled, allowing normal database activity and replication activity to continue.

If the OpenEdge Replication server is unable to reconnect to all agents or to the critical agent in the configured `connect-timeout` period, the OpenEdge Replication server terminates, and source database activity continues. In other words, if there is no critical agent, the server must be able to reconnect to all agents; or it will terminate. If one agent is specified as the critical agent, the server will continue if it can reconnect to it.

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 database. 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 as determined by the `transition-timeout` value in the OpenEdge Replication server (source) 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 pretransition 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 (as configured in the transition properties) to a normal OpenEdge database. A normal OpenEdge database is not replication-enabled; 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.

Caution: If your source and target databases are enabled for JTA transactions and have JTA transactions in a prepared state, you must manually resolve those transactions before transitioning. See *OpenEdge Data Management: Database Administration* for information on JTA and manually resolving transactions.

For more information about transition, see Transition on page 93 .

Restoring a source database

In the event of a catastrophic failure, you may find that you need to restore a source database from backup. Before you can restore a database that is enabled for OpenEdge Replication, you must first disable replication.

Note: Restore a database only when you are sure you must do so. Restoring should be used as a last resort, since you will lose the data for any activity performed after you backed up the database.

To restore a Replication-enabled source database:

1. Disable OpenEdge Replication on the source database.

Use the following command:

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

2. Restore your source database.

   Use the following command:

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

   Be sure you delete the *db-name*`.repl.recovery` file any time you restore.You can find the `.recovery` file in the source database directory.

   If the database you are restoring was previously enabled for OpenEdge Replication, the database is again enabled for OpenEdge Replication after the restore. You cannot restart OpenEdge Replication 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.

Transition

A key component of replication failover and failback is the process known as transition. *Transition* refers to the changing of the database role —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.

Transition scenarios

Transition can occurs in a number of different situations, as follows:

- The role of a **target** database can be transitioned to that of a **normal** (non-Replication-enabled) 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 already 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.
- If you have defined a **Replication Set**, both source and target can transition together. See Transition of a Replication Set on page 120 for more details.
- You can configure transition to restart a 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?

JTA transactions and transition

When your source database is a JTA-enabled OpenEdge database, JTA transactions are replicated from the source database to the target database or databases. An External Transaction Manager governs transactions on a source database, not the target database(s). An External Transaction Manager can have read access to a target database.

You can not transition a database with JTA transactions in a prepared state. You can use PROMON to manually resolve outstanding transactions, however, this manual intervention should only be performed on a replication source database when there is an unrecoverable catastrophic failure to JTA transaction manager. On a target database JTA transactions can only be resolved when Replication Agent is in the "pre-transition" state of OpenEdge Replication.

For more information on JTA transactions, see *OpenEdge Data Management: Database Administration* and *OpenEdge Data Management: SQL Development*.

Configuring transition

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

Choosing either one or two target databases

OpenEdge Replication allows for the configuration of 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 `source.repl.properties` file have `critical` set to `1` and `transition` set to `auto`.

If you set more than one target database to `crtitical=1` and `transition=auto`, OpenEdge Replication recognizes only the first target database specified in the `source.repl.properties` file for automatic transition. The second database must 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 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 waits 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 is 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 the following figure as a guide.

Figure 9: Source properties file with automatic transition

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

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

As the previous figure 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

When manual transition is set up and the OpenEdge Replication agent loses contact with the OpenEdge Replication server, the agent 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, you must configure the following properties as described:

- 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, at some other time, changed to `auto`.

The sample properties file in the following figure can be used as a guide.

Figure 10: Source properties file with manual transition

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

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

As the previous figure shows, the OpenEdge Replication agent, `agent1`, will wait for connection from the OpenEdge Replication server indefinitelyuntil the DBA performs transition.

To perform transition manually, use the following command:

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

Planning for transition

To use transition effectively, you must set **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.

In the Transition properties, modify the `responsibility` property (which is currently used for information purposed only) to identify the type of database. The possible values are `primary` and `secondary`.

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

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

  In the Transition properties, set the `database-role` property to identify the new role of the database once it is transitioned. Possible values are `reverse` and `normal`. If you choose `reverse`, the role of the database is reversed such that a source database becomes a target database and a target database becomes a source database. If you choose `normal`, which is the default value, the role of the database becomes that of a normal database, meaning that it is no longer enabled for replication.

- **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.

  In the Transition properties, modify the after-imaging properties `auto-begin-ai`, `auto-add-ai-areas`, and `ai-structure-file` to set AI behavior, as follows:

  - You can set the `auto-begin-ai` property value to `0` if you do not want AI to begin automatically after a target-to-source transition. To begin AI automatically after a target-to-source transition, set the value to `1`.
  - You can set the `auto-add-ai-areas` property to `0` if you do not want to 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). To add 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), set the value to `1`.
  - You can set 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?**

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

  In the Transition properties, 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, as follows:

  - You can set the `backup-method` property to `mark` to identify the database as backed up; marking the database as backed up does not allow future AI extents to be used when recovering from a disaster.

    To back up the database offline by using the OpenEdge PROBKUP utility, set the property to `full-offline`. 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.

    To back up the database online after the database is restarted, which occurs after the database has been transitioned, set the property to `full-online`.

  - You can set the `backup-arguments` property for the arguments required for the full online and offline backups performed for the database.

    Keep in mind that 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. Begin the arguments with *`device-name`*, and do not use backup validation parameters (such as `-vp` and `-vf`).

- You can set 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`*.

- **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.

  - In the Transition properties, modify the following automatic transition properties to specify whether the database should be automatically restarted after transition:

    - `restart-after-transition`
    - `start-secondary-broker`
    - `source-startup-arguments`, `target-startup-arguments`, or `normal-startup-arguments`
    - `source-secondary-broker-startup-arguments`, `target-secondary-broker-startup-arguments`, or `normal-secondary-broker-startup-arguments`

    You can set the `restart-after-transition` property to `0` if you do not want to restart the database automatically after transition is performed. Set the property to `1` to restart the database automatically after transition is performed. When you specify this value, you must also supply the `*-startup-arguments` properties, or the database startup will fail.

    Set the `start-secondary-broker` property to `1` to start a secondary broker after transition is performed.

  - Set the `*-startup-arguments` property or properties as follows:

    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
    ```

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 Sample of a startup parameter file used by transition on page 129 .

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

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

  In the Transition properties, modify the `recovery-backup-arguments` property, which determines whether transition automatically attempts a recovery. Specify the device name, followed by any other backup arguments.

Once you make each of these decisions, you can set the transition properties accordingly, as described in Setting transition properties on page 125 . (See Transition properties summary on page 131 for a complete list of all transition properties.)

Transition End Trigger

OpenEdge Replication supports a post-transition trigger. The transition end trigger is a user-created script that is configured to run at the end of transition. Executing a transition end trigger gives the database administrator the opportunity to insert additional instructions at a controlled point in the transition, such as directing applications to re-connect to a new source database. The trigger is supported by fail-over transition, when a source and target switch roles, and recovery transition, when a source fails and a target transitions to be the source for the other target.

Configuring transition trigger

The transition end trigger is configurable. You can add the transition end trigger property to the properties file of the source, target(s), or both. Indicate that you want the transition end trigger to execute by adding the following line to the `[transition]` section of your properties file:

```
transition-end-trigger=1
```

If not specified, the `transition-end-trigger` value is 0, indicating that no trigger is to execute. If set to a value other than 0 or 1, or not properly inserted in the `[transition]` section of the properties file, an error is written to the *dbname*`.repl.agent.lg` file.

If the `transition-end-trigger` is enabled on more than one database involved in the transition process, then the trigger may be executed multiple times, once for each participating database where it is enabled. Depending on your configuration, the transition end trigger may involve one database (a source or a target), two databases (one source plus a target), or three databases (one source plus two targets).

The trigger file is not validated prior to transition.

Transition trigger

The transition trigger is a script with very specific requirements. The file must be:

- Named `transProc` on Unix or `transProc.bat` in Windows.
- Located in the target database directory, the directory containing the *dbname*`.db` file
- Set as "executable" by your operating system

Caution: An OpenEdge server calling an external script is a potential security risk. Restricted access to the script is advised.

The trigger file has five input parameters, in the following order:

1. Current database name (the database the trigger is called from after transition)
2. Transition status (success/failure)
3. Database name (new source database name)
4. Host name (host for new source database)
5. Port (port name or number for database broker)

Note: All five input parameters are not always available. If the transition has failed, only the first two parameters (current database and status) are valid. In instances such as an offline transition, the host name and port number parameters of the new source, may not be available.

Once configured to execute, the transition end trigger executes regardless of the success or failure of the replication transition, by spawning a new process at the end of transition. Within the trigger code, you can use the transition status to determine execution paths based on the success or failure of transition.

The trigger file is not validated prior to transition. If the `transProc` file does not exist in the expected directory, or cannot be executed, an error message is written to the *dbname*`.lg` file.

If the trigger script is not found, the following error message is written:

```
(293) ** db_directory/transProc was not found.
```

If the trigger script is not executable, or you do not have permission, the following error message is written:

```
(294) ** You do not have permission to access file db_directory/transProc.
```

If the trigger script successfully executes, the following messages are written:

```
RPLU    6: (-----) Transition operation: Executing transition end trigger.
RPLU    6: (-----) Executing transProc with args[0]: /db_directory/transProc
RPLU    6: (-----) Executing transProc with args[1]: db1
RPLU    6: (-----) Executing transProc with args[2]: 0
RPLU    6: (-----) Executing transProc with args[3]: /db_directory/db2
RPLU    6: (-----) Executing transProc with args[4]: localhost
RPLU    6: (-----) Executing transProc with args[5]: 2625
RPLU    6: (13551) The transition of this database has completed normally.
```

Replication Monitor

You can verify the setting of `transition-end-trigger` property in the Replication Monitor.

For the server, select `S` for the **Replication server status** screen. The lower portion of the screen, with an enabled trigger, is shown below:

```
      AI structure file:                    addai.st
   Backup method:                           Mark
   Transition end trigger:                  1

   Configured agents:                       agent1, agent2
   Repl keep alive:                         300
   Schema lock action:                      Wait
   Agent shutdown action:                   Recovery
```

For an agent, select `A` for the **Replication agent status** screen. The lower portion of the screen, with an enabled trigger, is shown below:

```
Additional transition information:
    Replication set:                    1
    Database role:                      Reverse
    Transition to agents:               agent1,agent2
    Restart after transition:           1
    Automatically begin AI:             1
    Automatically add AI areas:         1
    AI structure file:                  addai.st
    Backup method:                      Full-offline
    Transition end trigger:             1
```

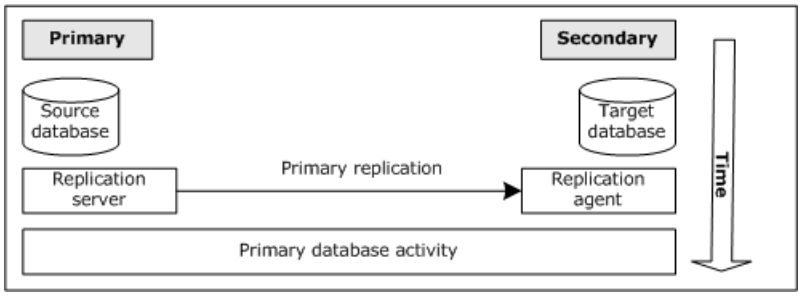
OpenEdge Replication from normal activity through failure and recovery

Refer to the following sections for a description of the replication process from normal activity through a failure condition, pretransition, transition, failover, failback, and recovery.

Step 1: Primary replication before a failure

During typical OpenEdge Replication operation, replication 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 the following figure.

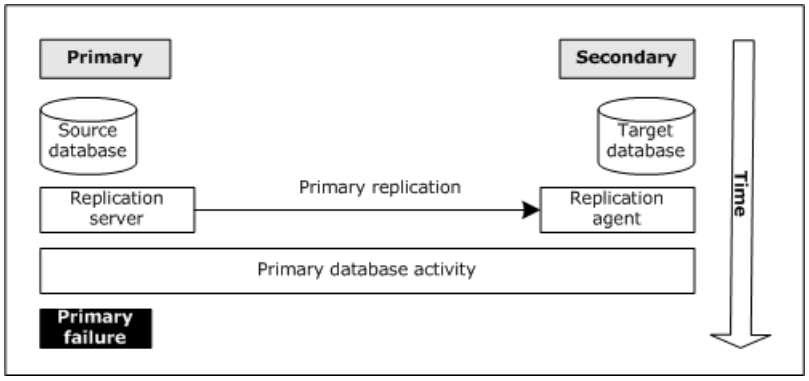
Figure 11: Primary replication before a failure



Step 2: Primary machine failure

Replication activity continues until a failure occurs on the primary machine, as shown in the following figure.

Figure 12: Primary machine failure



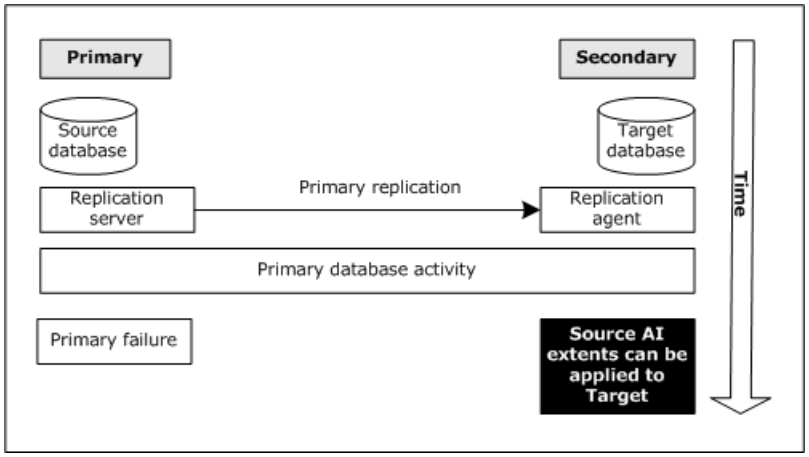
The Replication agent recognizes the failure as a TCP/IP communication error. Typical failures might be due to a network outage between the primary and secondary machines, a primary database crash, or a primary hardware crash.

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.

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 the following figure.

Figure 13: 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.

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 unable to reach the target. For example, a TCP/IP failure could lose packets that were awaiting 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]` section 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.
- If Replication AI streaming, the `ai-streaming` property under the `[agent]` section in the properties file, is enabled, the RSB cache must be empty before applying extents. If it is not empty, attempting to apply an extent, results in the following message:

```
The extent cannot be applied. There are %d unapplied blocks remaining in the RSB
 cache. (19292)
```

Note: While the replication agent main thread is in the applying-extent state, you cannot run the `triggertranistion` command. With AI Streaming replication, you cannot run subsequent `applyextent` commands while the agent is in the applying-extent state. The writer thread must first empty the RSB cache and the main agent thread will move out of the applying-extent state.

To apply AI extents:

1. Display failure recovery information.

   Use the following command:

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

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

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

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

   Determine the file number of the last-applied AI extent. In this example the last-applied extent is `3`. Since the value after `Completely Applied to Target` is `No`, you will use `3` as the *`extent-name`* in the next step.

   If the value after `Completely Applied to Target` was `Yes`, you would use `4` as the *`extent-name`* in the next step, 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 DSRUTIL applyextent qualifier on page 147.

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

```
dsrutil target-db-name -C applyextent qualified-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).

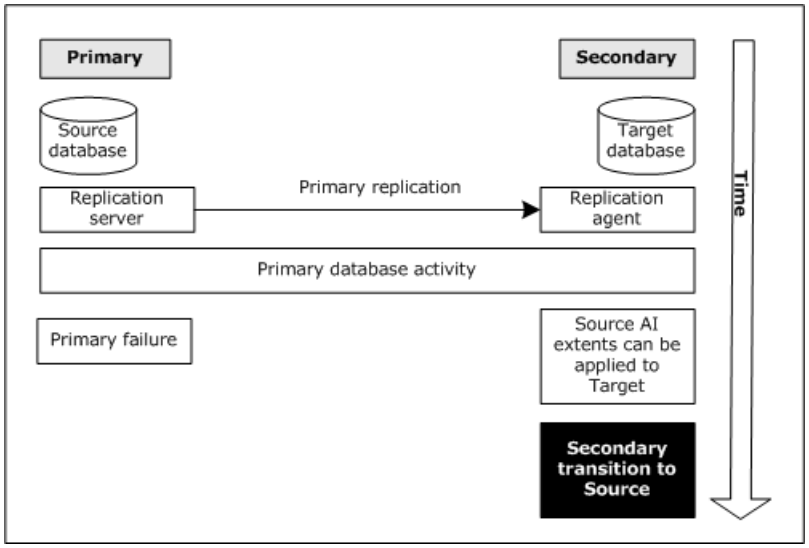
If the source database after-image extents cannot be applied to the target database before transition, the target database will not contain any of the unapplied source database transactions. In this case, unless the transactions can be recreated and re-entered after transition, you **will** lose an indeterminate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

After you apply the AI extent, you can begin a manual transition of the target. For more information, see Setting up manual transition on page 96.

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 the following figure.

Figure 14: Transitioning the target to a source database



To transition the database successfully:

1. Properly configure the transition properties. See Setting transition properties on page 125 for details.
2. Use the following command to perform transition:

```
dsrutil target-db-name -C transition
```

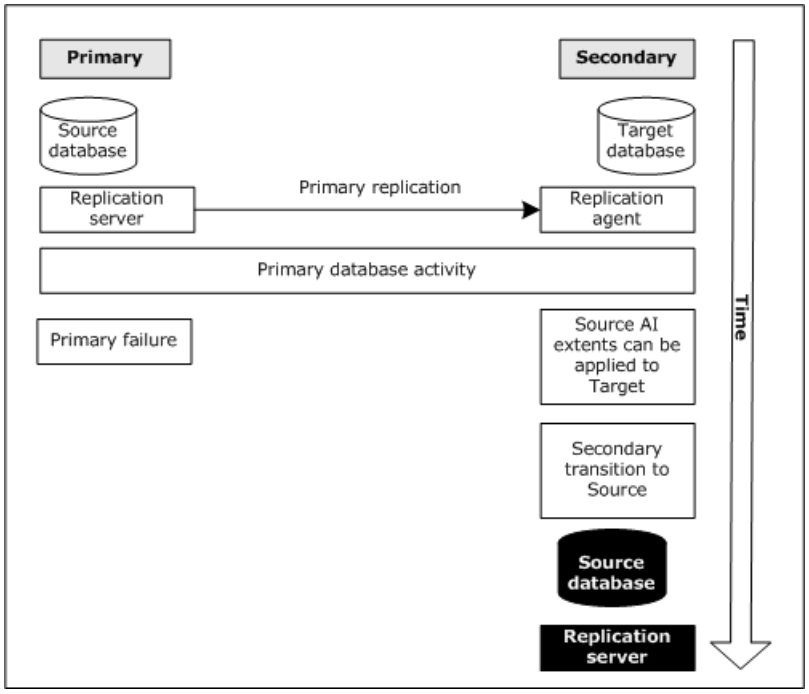
You can also configure transition to occur automatically, so that neither you nor another DBA needs to explicitly execute the command to initiate it.

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 15: Secondary database transitioned to source database



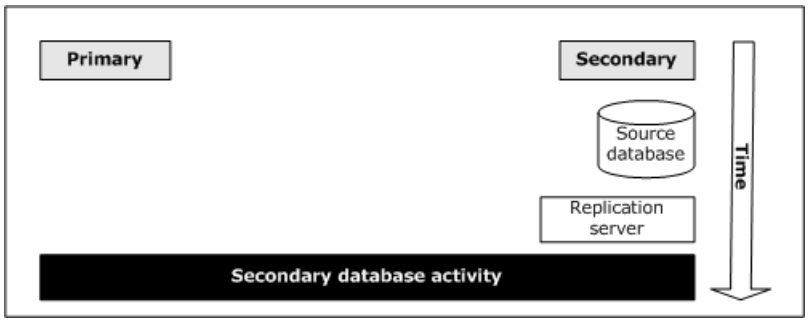
You can configure transition to start the secondary database automatically once the transition process completes, or you can start the database manually. The decision is up to you.

Since the secondary database is now a source database (as shown in the previous figure), 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 Server properties on page 64.

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 the following figure.

Figure 16: 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 and the machine is again up and running as shown in the following figure, you can begin the initial steps involved with failing back production processing to the primary computer.

Figure 17: Primary machine repair complete

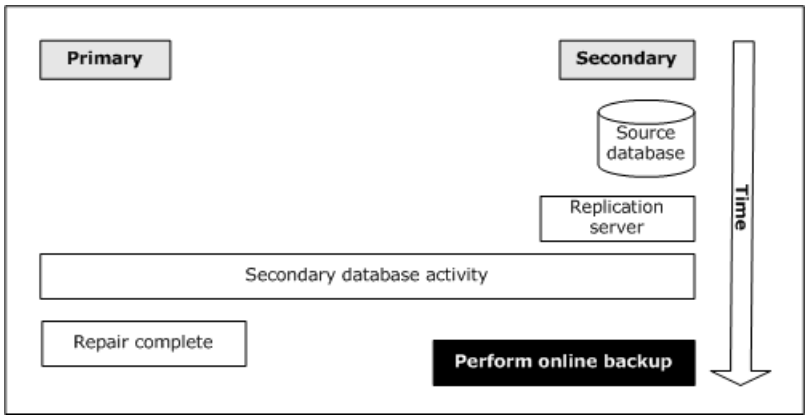


Step 7: Initiating primary database transition

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 knows as *secondary replication*.

As shown in the following figure, setting up secondary replication begins with a backup of the secondary database. Using an online backup limits the amount of downtime required.

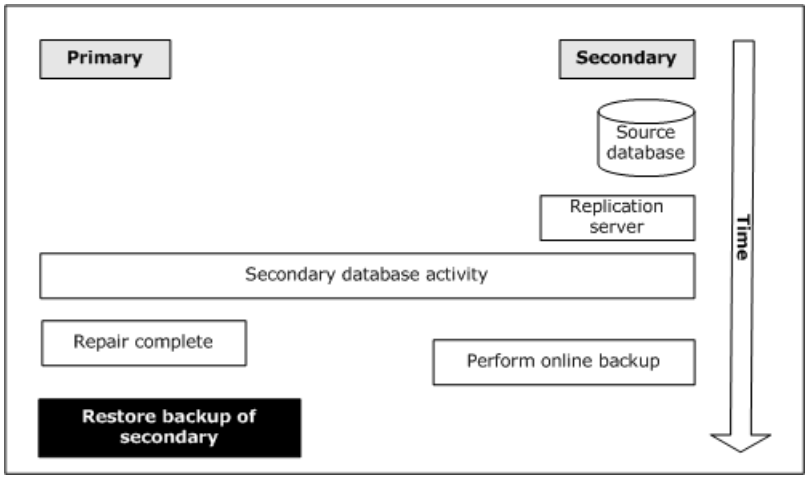
Figure 18: Performing an online backup of secondary database



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 the following figure.

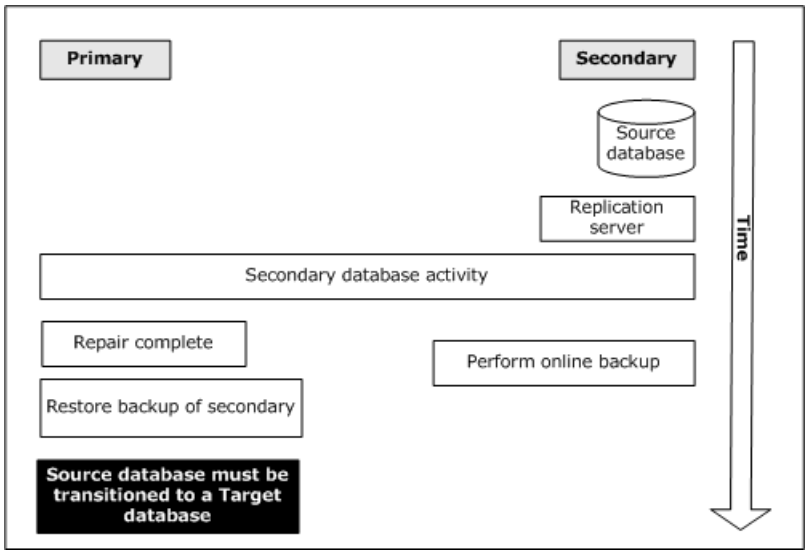
Figure 19: Restoring backup of secondary database



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 Step 2: Create a structure file on page 55.

You must now transition the primary database from a source to a target, as shown in the following figure.

Figure 20: Transitioning source database to target



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

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

```
prorest primary-db-namebackup-file -REPLTransition
```

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

```
prorest primary-db-namebackup-file
```

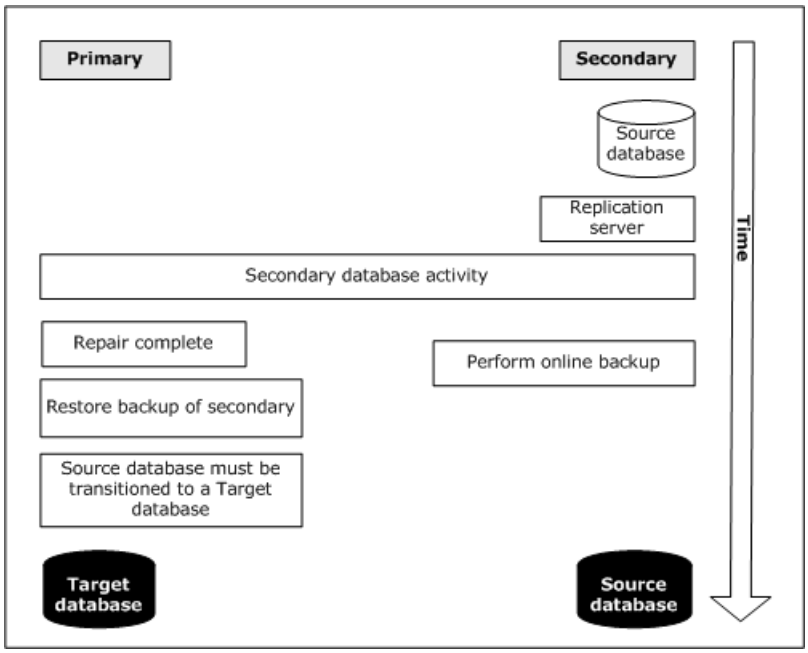
  Then use the following command for transition:

```
proutil primary-db-name -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 the following figure.

Figure 21: Transition completes



You can now perform secondary replication.

Step 8: Secondary replication is performed

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

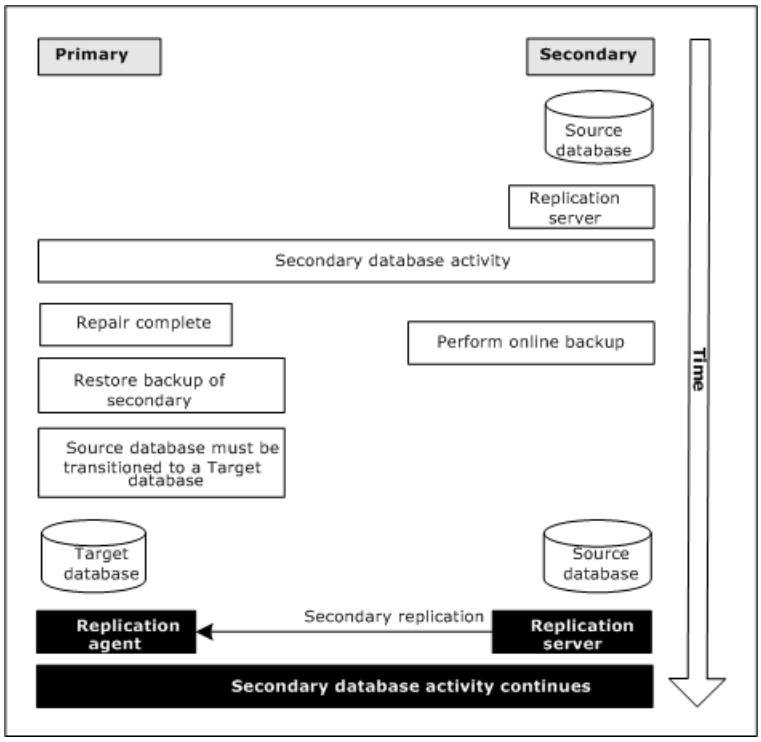
To restart the Replication server if it has terminated, use the following command:

```
dsrutil secondary-db-name -C restart server
```

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

As shown in the following figure, all secondary database activity that has taken place since the online backup was performed will be replicated to the primary database, as it is now the target database.

Figure 22: Secondary replication occurs



Step 9: The Replication failback process

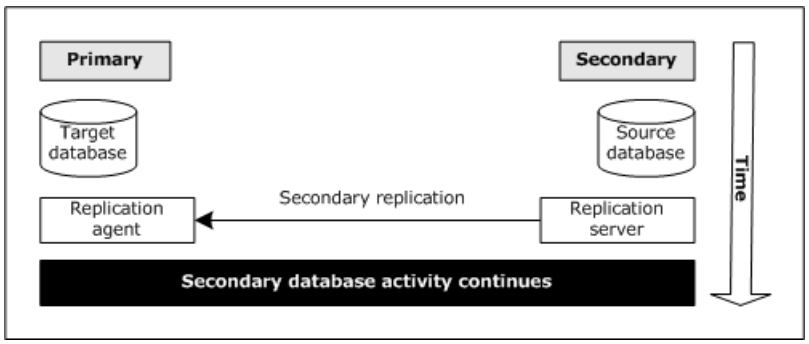
You can perform OpenEdge Replication failback production processing to the primary computer by using either of the following methods:

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

Failback processing using transition failover

When secondary replication is being performed, as shown in the following figure, both databases are up and running and all secondary transactions are being replicated to the primary database.

Figure 23: Secondary replication continues (before transition failover)

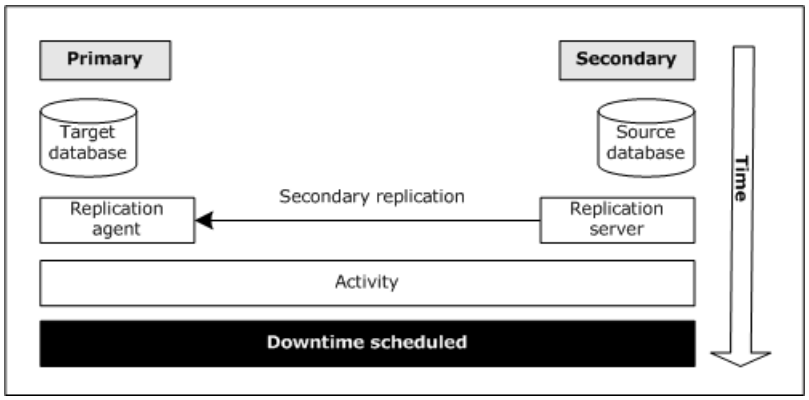


At this point, the secondary is considered the production database and the primary is considered the replica.

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.

The following figure illustrates the scheduling of database downtime.

Figure 24: Scheduling downtime to perform failback



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

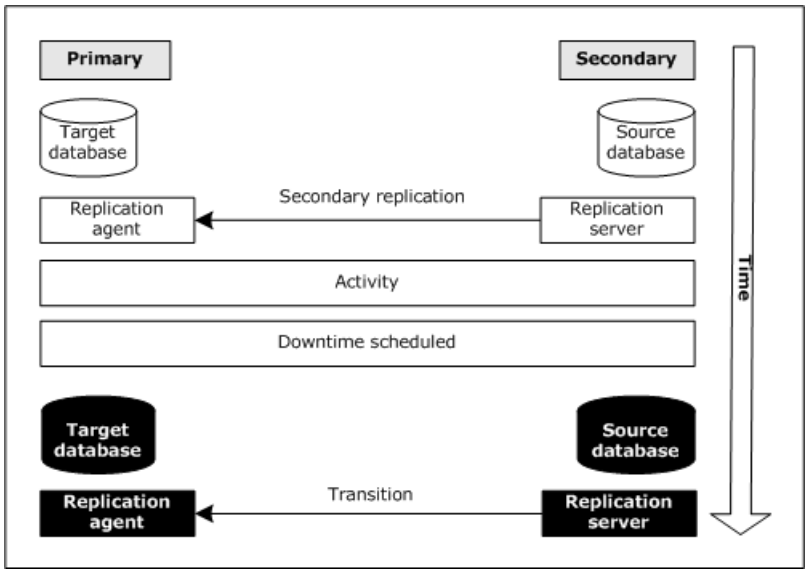
```
dsrutil secondary-db-name -C transition failover
```

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.

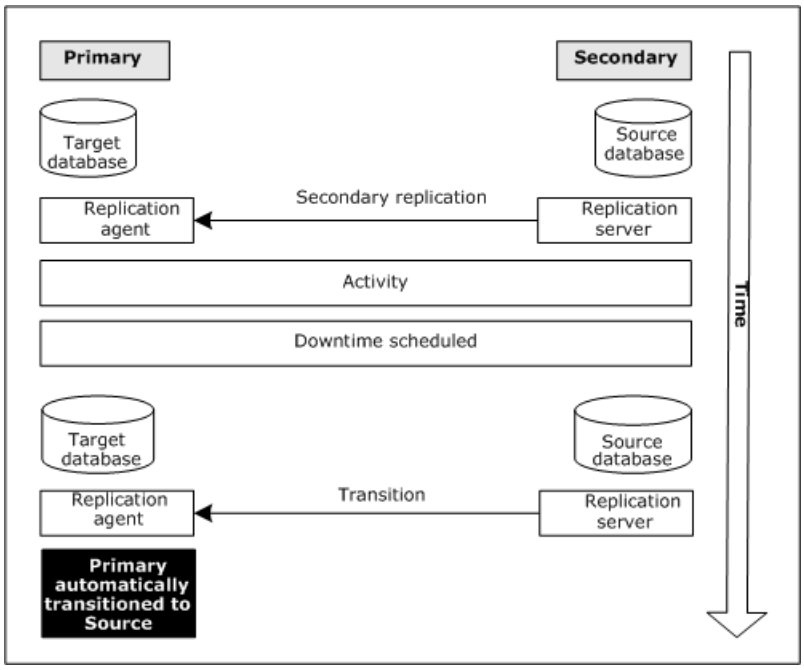
Once the startup synchronization process is complete, the actual transition process can begin, as shown in the following figure.

Figure 25: Initiating transition of the primary database



Once the startup synchronization process completes normally, the transition of the primary database is performed as configured, as shown in the following figure.

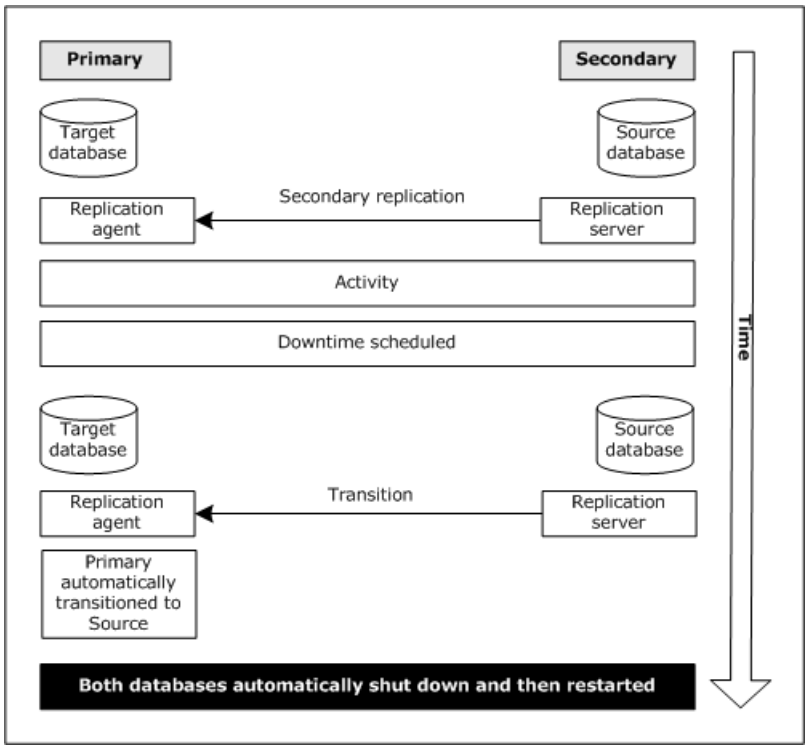
Figure 26: Continuing transition of the primary database



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

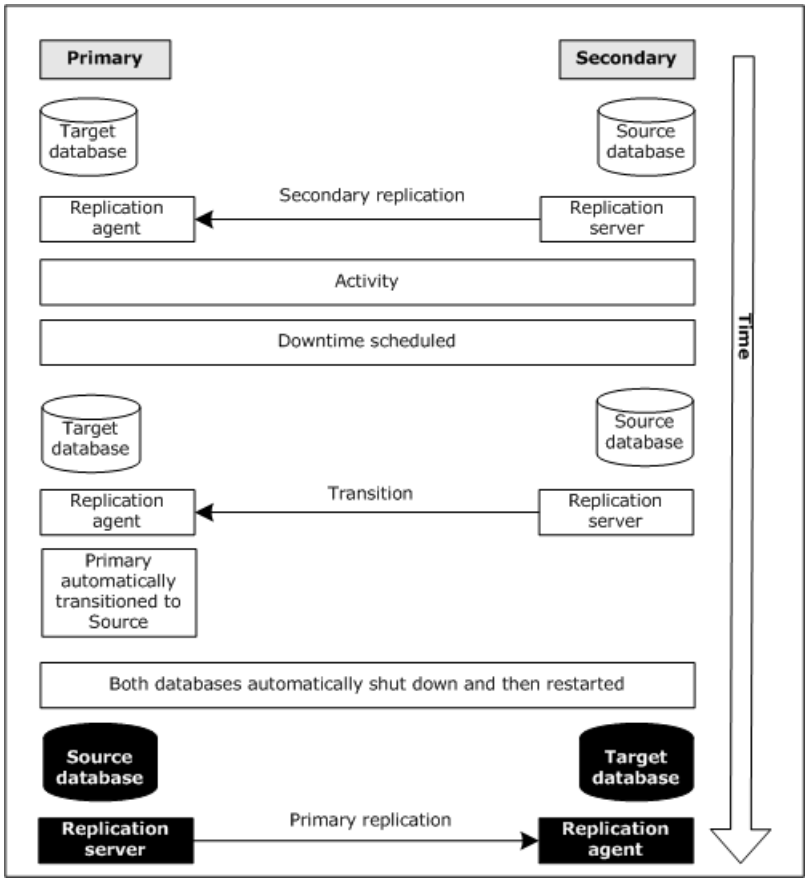
Once transition completes normally for both databases, the databases will be restarted in their new roles, as shown in the following figure.

Figure 27: Databases started in new roles



As shown in the following figure, 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 28: 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:

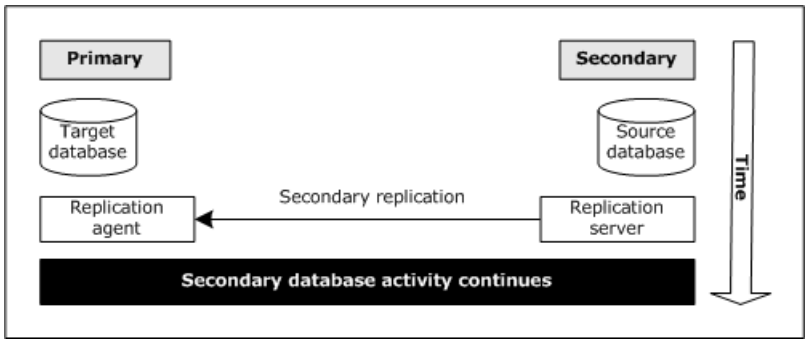
- The transition process is an all-or-nothing event. If transition fails for either database, both databases are restored to their original state, providing that recovery backups were properly configured. The result is that the primary database remains the target and the secondary database remains the 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 the following figure, both databases are up and running and all secondary transactions are being replicated to the primary database.

Figure 29: Secondary replication continues (before controlled transition)

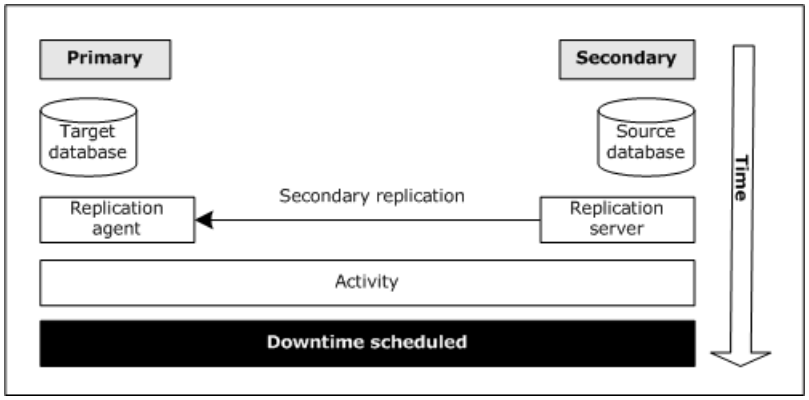


At this point, the secondary is considered the production database and the primary is considered the replica.

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.

The following figure illustrates the scheduling of database downtime.

Figure 30: 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.

Caution: Do not restart the databases after transition, so that you are able to 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.

Initiating failback with controlled transition

Use the following procedure to initiate the failback process using controlled transition:

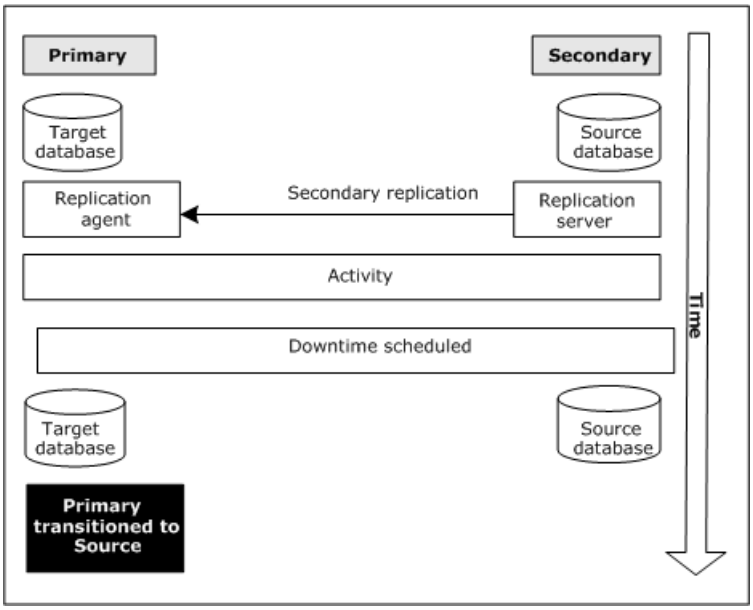
1. Shut down and restart both databases. Doing so ensures that all source activity is flushed and, in turn, replicated to the target database.
2. Verify the synchronization of the databases by doing one of the following:
   - Examining the database log file for a message confirming that the databases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

     ```
     dsrutil source-db-name -C status -detail
     ```

   When a status of 3049 is returned, both databases are synchronized.
3. Shut down the databases again.
4. Issue the following command on the primary machine:

   ```
   dsrutil primary-db-name -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.

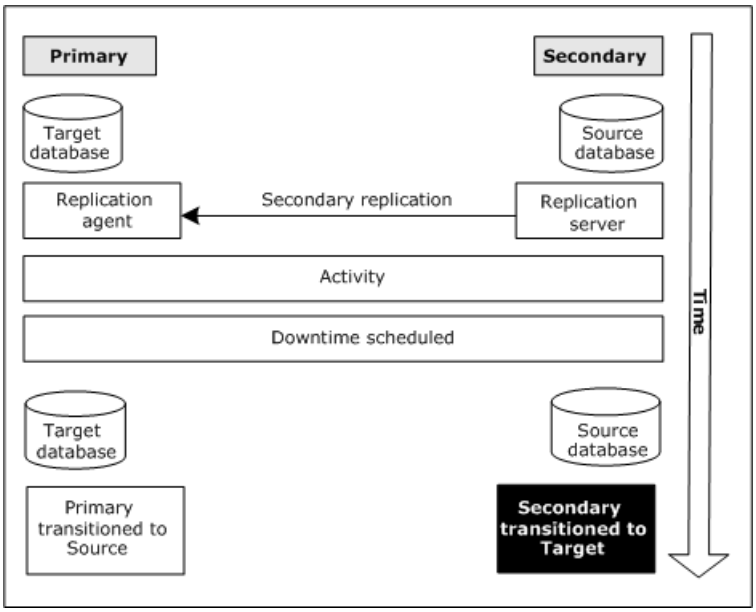
   After the transition of the primary database to source database completes, transition of the secondary database can begin.
5. On the secondary machine, issue the following command to transition the secondary database to a target database:

   ```
   dsrutil secondary-db-name -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:

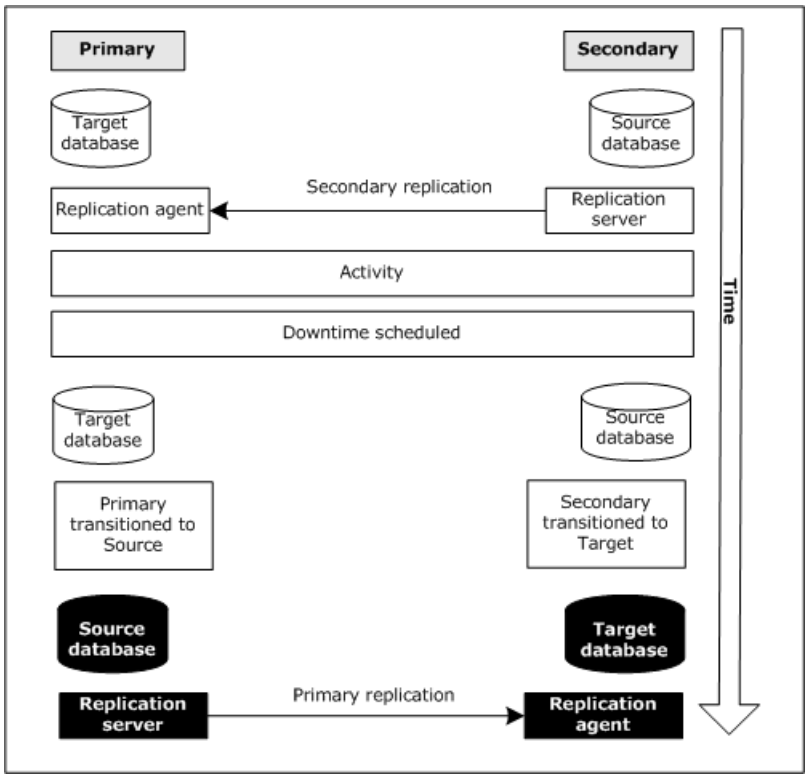


At this point the roles that both databases had during secondary replication have been reversed:

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

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 the following figure.

Figure 31: 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.

Transition of a Replication Set

Transition of a Replication Set occurs in one of two scenarios:

- **Recovery transition** — If the source fails, the two targets transition together. After the transition completes, the primary target is transitioned to a source database, and the other is its target.
- **Failover transition** — If all three replicas are online, the source and two targets to transition together.

In both transition scenarios, replication continues after the transition completes.

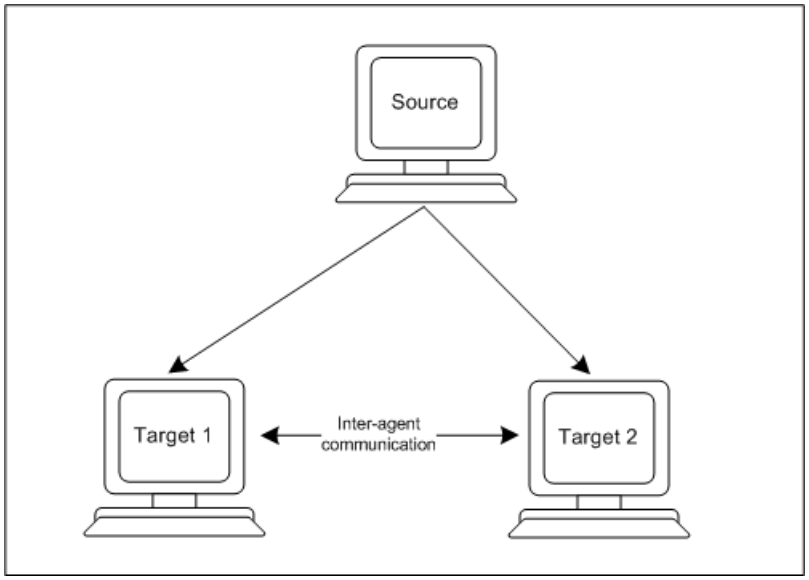
The transition section of the server properties file must be configured for a Replication Set. For example:

```
[transition]
  replication-set=1
  transition-to-agents=agent1,agent2
  database-role=reverse
  restart-after-transition=1
```

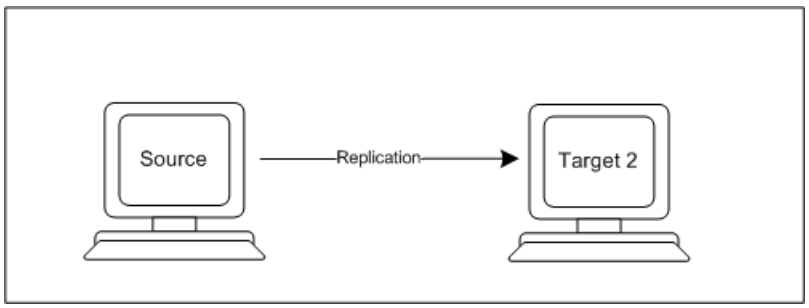
Recovery transition

Recovery transition of a Replication Set occurs when the source database is lost. The primary target transitions to the source, and the secondary target becomes a target for the new source.

During normal Replication Set processing, the three replicas are configured as shown:



If the source becomes unavailable and the Replication Set transitions, the resulting configuration is as shown:



Replication continues, but the Replication Set is suspended. Re-adding the third replica requires a rebase. After the database is rebased and added as a target, the Replication Set is reinstated.

Failover transition

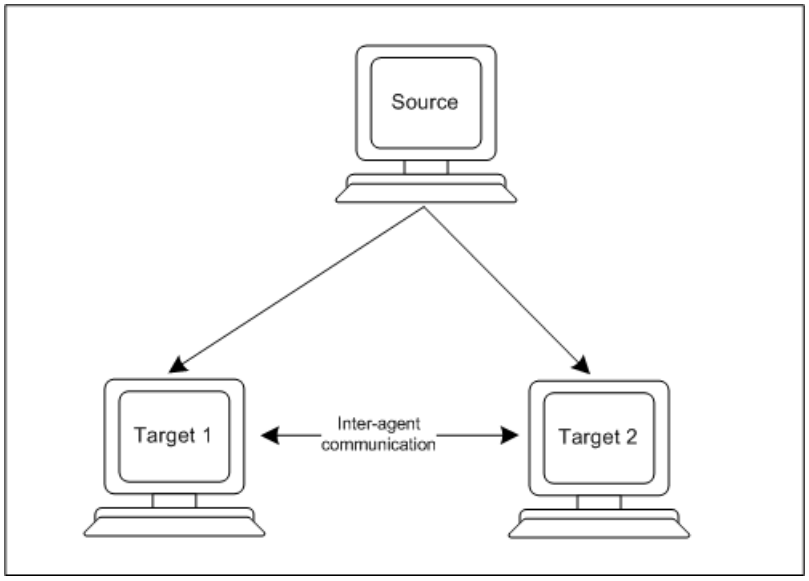
Failover transition of a Replication Set is the planned switch of a target and source replica.

Failover transition is initiated by executing the following command on the source database:

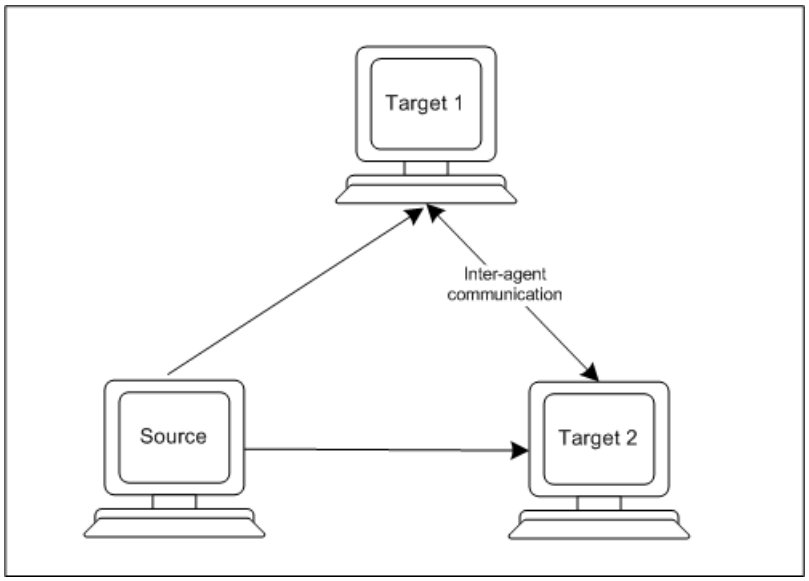
```
dsrutil source-db -C transition failover
```

All three databases of the Replication Set should be online for failover transition to successfully execute.

Before failover transition, the Replication Set is configured as shown:



After executing failover transition on a Replication Set, all three members of the set remain, but the source and the first agent listed in the transition-to-agents property, have switched roles, as shown:



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, as they existed 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` allows 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 *db-name*`.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:
  1. The current database log file, *db-name*`.lg`, is renamed *db-name*`.lg.save`.
  2. The database is deleted.
  3. The database is restored from the recovery backup. Ensure that you have the latest database structure file, *db-name*`.st`, in the database directory.
  4. The saved database log, *db-name*`.lg.save`, is renamed *db-name*`.lg`.
  5. 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 complex process. The transition process does provide status and progress messages; in addition, you can activate detailed transition logging.

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

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

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

This information is sent to `stdout` by the DSRUTIL utility. 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, transition will output information to the database log file *db-name*.lg and, if instructed, to a separate transition log. Transition will perform additional logging to a separate log file named *db-name*.repl.util.lg when the -logging argument is supplied to DSRUTIL as follows:

```
dsrutil db-name -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=source-db-name
    transition=manual
    transition-timeout=600
    agent-shutdown-action=recovery
    repl-keep-alive=90

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

[agent]
   name=agent1
   database=source-db-name
   listener-minport=4387
   listener-maxport=4500

[transition]
   database-role=reverse
   responsibility=primary
   auto-begin-ai=1
   auto-add-ai-areas=1
   transition-to-agents=agent1
   ai-structure-file=source-db-name.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-db-name
   listener-minport=4387
   listener-maxport=4500

[server]
    control-agents=agent1
    database=source-db-name
    transition=manual
    transition-timeout=600
    defer-agent-startup=60
    agent-shutdown-action=recovery
    repl-keep-alive=90

[control-agent.agent1]
   name=agent1
   database=target-db-name
   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=target-db-name.addai.st
   restart-after-transition=1
   source-startup-arguments=-DBService replserv
   target-startup-arguments=-S 6931 -DBService replagent
   backup-method=full-offline
   backup-arguments=target-db-name.sav
   incremental-backup-arguments=ks2.sav.inc
```

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

See Transition properties summary on page 131 for a description of each transition property. For details about setting the properties, see Setting transition properties on page 125 .

Setting transition properties

You can set the transition properties that match the decisions you made in your planning. Once you make the decisions outlined in Step 1: Primary replication before a failure on page 102 , you must set the transition properties that support those decisions, as listed in the table that follows. (See Transition properties summary on page 131 for a complete list of all transition properties.)

Table 11: Setting transition properties

To implement this transition planning decision . . .	Do this . . .
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.

To implement this transition planning decision . . .	Do this . . .
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`*.
Do you want the database to restart automatically after transition?	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` • `start-secondary-broker, source-secondary-broker-startup-arguments, target-secondary-broker-startup-arguments,` or `normal-secondary-broker-startup-arguments` Specify either of these values for the `restart-after-transition` property: • `0` — Do not automatically restart the database after transition is performed.

To implement this transition planning decision . . .	Do this . . .
	• 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 either of these values for the start-secondary-broker property: • 0 — Do not automatically start a secondary broker for the database after transition is performed. • 1 — Automatically start a secondary broker for the database after transition is performed. When you specify this value, you must also supply the *-secondary-startup-arguments properties, or the broker 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 • 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.

To implement this transition planning decision . . .	Do this . . .
	For additional details about the `source-`, `target-`, or `normal-startup-arguments`, see Sample of a startup parameter file used by transition on page 129.
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 of a startup parameter file used by transition

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 (non-Replication-enabled) database. (The file is optional in this case.)

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 Transition properties summary on page 131 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 Transition properties summary on page 131 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 Transition properties summary on page 131 for details about the `normal-startup-arguments` property.

Transition property reference

The sections that follow provide reference information related to OpenEdge Replication transition and transition properties.

Transition command actions

The following table describes the action taken by the `dsrutil` *`db-name`* `-C transition` command when it is executed for databases in varying states.

Table 12: Transition command actions

Database type	Database state	Online or offline	Description of transition performed
Source	Database is newly enabled	Offline	An error occurs because the database is newly enabled.
Source	Database is newly enabled	Online	An error occurs because the database is online and has never been replicated.
Source	Database has been replicated	Offline	The database is transitioned as instructed in the `repl.properties` file.
Source	Database has been replicated	Online	In order to transition a source database in this state, the Replication server must be communicating with the Replication agent to convey a transition and failover request to the Replication agent.
Target	Database has been replicated	Offline	In order to transition a target database in this state, the database cannot be newly transitioned.

Database type	Database state	Online or offline	Description of transition performed
Target	Database has been replicated	Online	In order to transition a target database in this state, the following conditions must be met: • The Replication agent cannot be communicating with the Replication server. • The Replication agent must be in pretransition. • All source AI extents must be applied to the target database by using `dsrutil db-name -C applyExtent`. You must explicitly perform this step, possibly multiple times depending on the number of source AI extents.
Normal	—	Online	An error occurs because the database is not enabled for replication.

Transition properties summary

The following table 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 13: Transition properties

Property name	Type and length	Description
`replication-set`	integer	Defines if source and target replicas transition together. Possible values are as follows: • `0` — The source and targets do not transition together (the default). • `1` — The source and targets transition together. See A Replication Set on page 25 for details.
`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.

Property name	Type and length	Description
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 andthe target-startup-arguments.
start-secondary-broker	integer	Set to **1** to automatically start a secondary broker after transition.
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.pfBecause 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

Property name	Type and length	Description
target-startup-arguments	character[256]	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 *port-number* or *service-name*: -DBService replagent -S { *port-number* | *service-name* }
normal-startup-arguments	character[256]	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
source-secondary-broker-startup-arguments	character[256]	For secondary broker. See the description of source-startup-arguments for details.
target-secondary-broker-startup-arguments	character[256]	For secondary broker. See the description of target-startup-arguments for details.
normal-secondary-broker-startup-arguments	character[256]	For secondary broker. See the description of normal-startup-arguments for details.
auto-begin-ai	integer	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.

Property name	Type and length	Description
`transition-to-agents`	character[256]	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. If `replication-set=1`, the first agent in the `transition-to-agents` list is selected to become the new source. 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` **Note:** If both `dr-transition-to-agents` and `transition-to-agents` are defined, `dr-transition-to-agents` is used in *recovery* transition and `transition-to-agents` is used for *failover* transition. If `dr-transition-to-agents` is not defined,`transition-to-agents` is used for both types of recovery.
`dr-transition-to-agents`	character[256]	**For Recovery Transition only.**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. If `replication-set=1`, the first agent in the `transition-to-agents` list is selected to become the new source. 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` **Note:** If both `dr-transition-to-agents` and `transition-to-agents` are defined, `dr-transition-to-agents` is used in *recovery* transition and `transition-to-agents` is used for *failover* transition. If `dr-transition-to-agents` is not defined,`transition-to-agents` is used for both types of recovery.

Property name	Type and length	Description
`auto-add-ai-areas`	integer	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.
`ai-structure-file`	character[256]	Name of the structure file that contains the list of AI areas to add.
`backup-method`	character[15]	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: `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`** — The database will be backed up offline by using the PROBKUP utility. The backup is performed in two steps: The first backup is a full backup, which is performed before AI is enabled for the database. The second backup is an incremental backup, which is performed after AI is enabled and after the role of the database is changed. • **`full-online`** — An online backup is performed after the database is restarted, which occurs after the database has been transitioned.
`backup-arguments`	character[256]	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 `-vp` and `-vf`). The arguments should begin with *`device-name`*.

Property name	Type and length	Description
`incremental-backup-arguments`	character[256]	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 *`device-name`*.
`recovery-backup- arguments`	character[256]	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 `-vp` and `-vf`). The arguments should begin with *`device-name`*.
`apply-streaming-data`	integer	**For auto-transition only.** If Replication AI Streaming is enabled, and there is data in the RSB cache at the time of transition, indicate if the data should be applied or ignored before transition. The following values are valid: • **1** — Apply all data in the RSB cache before transition. • **0** — Do not apply any data in the RSB cache before transition. **Note:** Replication AI Streaming is enabled by default in OpenEdge Release 12.0

6

Enabling Replication AI Streaming

This chapter describes how to manage and monitor Replication AI Streaming.

For details, see the following topics:

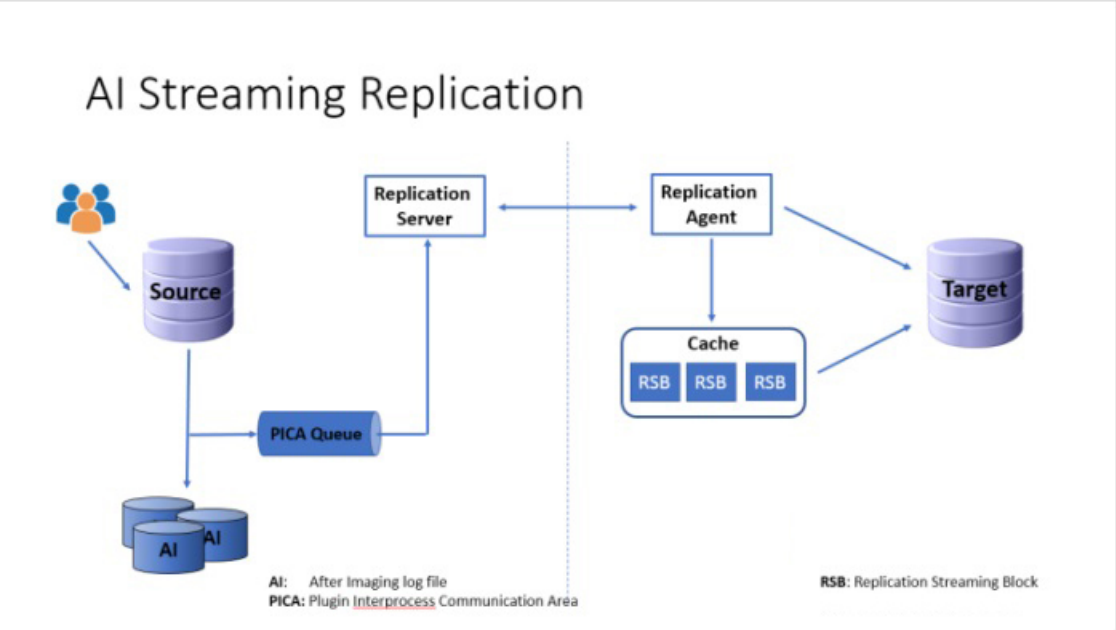
- Replication AI Streaming Overview
- Configuring Replication AI Streaming
- Observing Replication AI Streaming in the Replication Monitor
- Observing Replication AI Streaming in VSTs
- Understand automatic transition with AI Streaming

Replication AI Streaming Overview

Database admins need to ensure their disaster recovery systems have the most up-to-date database changes so that in the event of a failure, recovery can occur quicker, and retain as much data as possible. Replication AI Streaming addresses this need.

Without Replication AI Streaming enabled, AI blocks are sent to the targets when they are generated, and are received by the agent and applied on the target databases. If AI data cannot be processed fast enough on the targets, it is queued on the source database machine. In the event of a source crash, this data can be lost and lead to data consistency issues on the targets.

To minimize the data loss caused by AI data queued on the source, Replication AI Streaming streams the AI blocks to the targets as they are generated. The agent on the target is multi-threaded to improve performance. Replication AI Streaming allows the targets to stay as close as possible to the source database because source AI data is hardened on the targets once it is received. Replication AI Streaming is also designed to reduce the latency as AI data is less likely to be queued on the source side (PICA). The following figure demonstrates how AI Streaming works.



Streaming Replication Architecture

The Replication server sends AI data to the target as described in the *OpenEdge Replication and After-imaging* chapter, but the Replication agent caches the incoming AI blocks in the Replication Stream Block (RSB) Cache. The blocks are then read from the RSB cache and applied to the target. Database admins have the ability to observe the Replication AI Streaming operations in the replication log with logging-level set to 3 on the replication agent.

Replication AI Streaming is enabled by default in OpenEdge Release 12.0. The sections that follow contain descriptions of how to configure your Replication Agent property file and how to observe the Replication AI Streaming operations in the Replication Monitor and through VSTs.

Configuring Replication AI Streaming

Configuring Replication AI Streaming is controlled by properties. Add the following properties to the `[agent]` section of your Replication properties file(s):

- **`ai-streaming`**

  The `ai-streaming` property enables or disables this feature. The property is enabled by default. Set as follows:

  - **Enabled:** `ai-streaming=1`
  - **Not enabled:** `ai-streaming=0`

- **`rsb-cache-size`**

  The `rsb-cache-size` property sizes your RSB cache. Set as follows:

  `rsb-cache-size=`*`n`*

  Where *`n`* is the size of the RSB cache in units of Replication Stream Blocks. The default is 1000.

Observing Replication AI Streaming in the Replication Monitor

You can observe the functioning of Replication AI Streaming in the Replication Monitor. Start the monitor on your target machine, and select **Replication Agent Status**.

Observe the status of Replication AI streaming, and the cache, with the following lines of the output:

```
AI Streaming enabled:                       1
Replication streaming block cache:
    Cache size:                             5000
    Free cache entries:                     999
    Used cache entries:                     1
    Cache high water mark:                  175
    Number of times cache filled:           0
```

Observe the status of any latency between the target and the source, with the following lines of the output:

```
    AI Block Information:
        Source RDBMS Block (Seq / Block):       4 / 6905
        Last Received Block (Seq / Block):      4 / 6884
        Last Processed Block (Seq / Block):     4 / 6884
    Latency Information:
        Repl Server behind Source DB by:        0     second(s)
        Current Source Database Transaction:    2303
        Last Transaction Applied to Target:     2303
        Target Current as of (Target, Source):  Tue May 15 18:15:58 2018, Tue May 15
18:15:58 2018 with delta of 000:00:00
        Number of blocks to send on Source:     0 as of Tue May 15 18:16:40 2018
        Number of blocks to process on Target:  0 as of Tue May 15 18:16:40 2018
        Time delay to process the last block:   0 second(s)
```

For descriptions of each line of the **Replication Agent Status** output, see Replication agent status on page 173.

Observing Replication AI Streaming in VSTs

The _Repl-AgentAIStreaming table within the Replication Virtual System Tables (VSTs) provides insight into the configuration and functioning of Replication AI Streaming.

- _Repl-AgentAIStreaming

  The _Repl-AgentAIStreaming provides information about the RSB cache. It contains the following fields:

  - `_ReplAgtAIStr-AgentID`
  - `_ReplAgtAIStr-AgentName`
  - `_ReplAgtAIStr-TotalCacheEntries`
  - `_ReplAgtAIStr-FreeCacheEntries`
  - `_ReplAgtAIStr-UsedCacheEntries`
  - `_ReplAgtAIStr-CacheHWM`
  - `_ReplAgtAIStr-TimesCacheFilled`

  For detailed descriptions of the fields, including data types, see _Repl-AgentAIStreaming VST field descriptions on page 201.

- _Repl-Agent

  The _Repl-Agent VST features the following field:

  - `_ReplAgt-AIStreaming`

  For detailed descriptions of this field, and the rest of the _ReplAgent VST, see _Repl-Agent VST field descriptions on page 186.

- _Repl-AgentActivity

  The _Repl-AgentActivity VST features the following fields:

- _ReplAgtAct-LastRecvBlockSeq
- _ReplAgtAct-LastRecvBlockBlock
- _ReplAgtAct-ServerBlocksToSend
- _ReplAgtAct-LastBlockCountTime
- _ReplAgtAct-BlockProcDelay

For a detailed description of these fields, and the rest of the _ReplAgentActivity VST, see _Repl-AgentActivity VST field descriptions on page 191.

Understand automatic transition with AI Streaming

The behavior of the data in your RSB cache, when an automatic transition occurs, is controlled by the transition property, `apply-streaming-data`. The `apply-streaming-data` property may be set to either 1 or 0. The setting indicates the following:

- **`apply-streaming-data=1`** — When the transition timeout expires, the replication agent waits for all entries in the RSB cache to be applied to the target database before continuing with auto-transition.
- **`apply-streaming-data=0`** — When the transition timeout expires, the replication agent discards all entries in the RSB cache and continues with auto-transition.

The `apply-streaming-data` property defaults to 0 unless explicity set to 1 in the properties file. Setting the `apply-streaming-data` property to values other than 1 is the equivalent of setting it to 0.

For all possible configurations of replication (one target, two target, replication set)[2], when `apply-streaming-data=1`, all the data in the RSB cache is written to the target database before transition.

Auto-transition for one-target or two-targets (no replication set)

In this configuration, the agents handle data in the RSB cache with the following steps:

1. The agent reaches time to transition.
2. Check the `apply-streaming-data` setting.
   - If 1, check if the RSB cache is empty.
     - **Yes** — initiate auto-transition, writing this message to the log file:

```
Agent's RSB cache is empty.  Ready for auto-transition.   (19317 )
```

     - **No** — reset the time to transition and return to **Step 1**, after writing these messages to the log file:

```
%l<num> blocks exist in RSB cache.  The Agent will transition after all blocks
 in the RSB cache have been applied to the database. (19300)
The transition time has been reset to allow for RSB cache to empty before auto
 transition. (19301)
```

[2] Progress Software recommends configuring two-target replication environments as a replication set to avoid rebasing after transition.

- If 0, check if the RSB cache is empty.
  - **Yes** — initiate auto-transition, writing this message to the log file:

```
Agent's RSB cache is empty.  Ready for auto-transition.   (19317 )
```

  - **No** — initiate auto-transition, writing this message to the log file:

```
Automatic transition is starting, there are up to %l<num> blocks in the RSB
cache that will not be processed. (19299)
```

Auto-transition for a two target replication set

In this configuration, the agents handle data in the RSB cache with the following steps. If at any time during the processing of the RSB cache prior to transition, the agents lose inter-agent communication, the replication set is broken and the agents transition as single agents, using the prior process.

1. The local agent reaches time to transition. (Each agent acts as a local agent, communicating with the other "remote" agent.)
2. Check if the RSB cache has been processed.
   - **Yes** — Proceed to **Step 3**.
   - **No** — Process the RSB cache.

     Check the `apply-streaming-data` setting:
     - **1** — Process RSB cache data before transition.

       Check if the RSB cache is empty.
       - **Yes** — Ping the remote agent that the RSB cache is empty, and the local agent is ready for transition. Write the following message to the log file:

```
Agent's RSB cache is empty.  Ready to transition with replication-set. (19311)
```

       - **No** — Ping the remote agent that the RSB cache is not empty, the local agent is processing the cache, and is not ready for transition. Write the following messages to the log file and repeat this step:

```
%l<num> blocks exist in RSB cache.  The Agent will transition after all blocks
 in the RSB cache have been applied to the database. (19300)
The transition time has been reset to allow for RSB cache to empty before
auto transition. (19301)
```

     - **0** — Do not process RSB cache data before transition.

       Check if the RSB cache is empty.

- **Yes** — Ping the remote agent that RSB cache is empty and the local agent is ready for transition. Write the following message to the log file:

```
Agent's RSB cache is empty.  Ready to transition with replication-set.
(19311)
```

- **No** — Discard the remaining blocks in the RSB cache. Ping the remote agent that cache is empty and the local agent is ready for transition. Write the following message to the log file:

```
Agent is ready to transition with replication-set.  There are up to %l<num>
 blocks in the RSB cache that will not be processed.(19312)
```

Note: When the remote agent receives a ping from the local agent, it sets its time to transition to the current time to begin handling of its own RSB cache. The remote agent writes the following message to its log file:

```
Remote agent has signaled that it processed its RSB data and is ready for
auto-transition. (19316)
```

3. Check communications status with the remote agent.

   Has the remote agent sent a message indicating that it has completed processing its RSB cache and is ready to transition?

   - **No** — Ping the remote agent to confirm it is still connected. Reset the time to transition, and return to **Step 1** after writing the following message to the log file:

```
Waiting for remote agent to process its RSB data and signal that it is ready to
auto-transition.(19314)
```

   - **Yes** — Proceed to **Step 4**.

4. Confirm that both agents are ready to proceed with auto-transition.

   Both agents write the following message to their log file:

```
Both agents in the replication-set have processed their RSB data and are ready for
auto-transition.(19315)
```

   The priority agent initiates transition, and the other agent waits to be notified by the priority agent that transition processing is proceeding.

7

Reference

This chapter provides OpenEdge Replication reference information related to the DSR utility, the DSR utility monitor, virtual system tables, and other utilities.

For details, see the following topics:

- DSRUTIL utility
- DSRUTIL applyextent qualifier
- DSRUTIL canceldefer server qualifier
- DSRUTIL connectagent database qualifier
- DSRUTIL disablesitereplication qualifier
- DSRUTIL monitor qualifier
- DSRUTIL recovery qualifier
- DSRUTIL restart agent qualifier
- DSRUTIL restart server qualifier
- DSRUTIL startagent database qualifier
- DSRUTIL status qualifier
- DSRUTIL terminate qualifier
- DSRUTIL transition qualifier
- DSRUTIL transition failover qualifier

- DSRUTIL triggertransition qualifier
- OpenEdge Replication DSRUTIL MONITOR
- Virtual system tables for OpenEdge Replication
- Utilities and OpenEdge Replication
- OpenEdge Replication and database management systems
- DSRUTIL logginglevel qualifier

DSRUTIL utility

Once you initially set up, enable, and start OpenEdge Replication, you can use the DSRUTIL utility to perform specific OpenEdge Replication server, OpenEdge Replication agent, source database, and target database requests.

Syntax

```
DSRUTIL db-name -C ACTION [ Server | Agent ] [ name | ALL ]
[ -Passphrase ]
```

Parameters

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.

-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 1.

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 listed in the table that follows and described in the following sections.

Table 14: DSRUTIL utility qualifiers

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

DSRUTIL applyextent qualifier

Allows you to apply AI extents, generated by your source database, directly to a target database. This is useful when there is a source failure and there are AI blocks in transit between the server and agents.

Syntax

```
dsrutil db-name -C applyextent extent-name
```

Parameters

db-name

The name of the target database.

extent-name

The fully qualified path name of the extent to apply. See the section below for details on how to identify an extent to apply.

Your Replication configuration must meet the following requirements in order to apply extents:

- 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.
- If you have AI streaming enabled, your RSB cache must be empty. The `applyextent` command will cause the AI blocks contained in the extent to be written to the RSB cache. If it is not empty, attempting to apply an extent, results in the following message:

```
The extent cannot be applied. There are %d unapplied blocks remaining in the RSB
 cache. (19292)
```

Note: While the replication agent main thread is in the applying-extent state, you cannot run the `triggertranistion` command. With AI Streaming replication, you cannot run subsequent `applyextent` commands while the agent is in the applying-extent state. The writer thread must first empty the RSB cache and the main agent thread will move out of the applying-extent state.

Identifying the AI extent to apply

Identifying the AI extent to apply is a multi-step process.

- First, determine which extent to apply by using the following command:

```
dsrutil db-name -C recovery Agent
```

The recovery output provides the output **After Image File Number** and **Completely Applied to Target**, as shown:

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

If **Completely Applied to Target** is **No**, use the **After Image File Number** provided. If **Completely Applied to Target** is **Yes**, increment the **After Image File Number** value by 1.

- After determining the AI Image File number, locate the file, as follows:
  - If the source database is available, use the following command to generate a list of after-image extents for the source database:

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

Executing the command produces the following output:

```
Extent:  1
Status:  Full
  Type:  Variable Length
  Path:  /vobs_repl/solaris/bin/ks1.a1
  Size:  120
  Used:  1
 Start:  Wed Oct 26 08:32:12 2005
 Seqno:  1

Extent:  2
Status:  Full
  Type:  Variable Length
  Path:  /vobs_repl/solaris/bin/ks1.a2
  Size:  4728
  Used:  4534
 Start:  Wed Oct 26 08:32:14 2005
 Seqno:  2

Extent:  3
Status:  Busy
  Type:  Variable Length
  Path:  /vobs_repl/solaris/bin/ks1.a3
  Size:  4728
  Used:  4456
 Start:  Wed Oct 26 08:33:29 2005
 Seqno:  3

Extent:  4
Status:  Empty
  Type:  Variable Length
  Path:  /vobs_repl/solaris/bin/ks1.a4
  Size:  120
  Used:  0
 Start:  N/A
 Seqno:  0
```

Match the Seqno from this output to the **After Image File Number** provided by the DSRUTIL recovery output. Apply all BUSY and FULL extents beginning with this extent.

- If the source database is unavailable, but its after-image extents are available on a SAN or NAS device that the target machine has access to, you can determine the first after-image extent to apply, by examining the AI files.

  1. 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 database you specify in the *db-name* location in the command can be any database provided it has AI enabled and an AI block size that is the same as the original source database. However, this command must be run offline, so the database specified must be offline.

     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)
     ```

  2. Match the **This is 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.

For more information on applying extents, see Manually applying after-image extents on page 103.

DSRUTIL canceldefer server qualifier

Instructs the OpenEdge Replication server to stop attempting to reconnect. This action is applicable only if the defer-agent-startup property was set to a valid non-zero time that has not yet expired.

Syntax

```
dsrutil db-name -C canceldefer server
```

Parameters

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

```
dsrutil db-name -C connectagent { name | ALL }
```

Parameters

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.

name | ALL

Connect one OpenEdge Replication agent (*name*) or all OpenEdge Replication agents (`ALL`).

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

```
dsrutil db-name -C disablesitereplication { Source | Target }
```

Parameters

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

```
dsrutil db-name -C monitor
```

Parameters

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 OpenEdge Replication DSRUTIL MONITOR on page 162 .

DSRUTIL recovery qualifier

Displays the replication recovery information.

Syntax

```
dsrutil db-name -C recovery { server | agent }
```

Parameters

db-name

The name of the database to perform the action on. The name of the database must be the first argument and must be a valid name.

The replication recovery information for the server looks similar to the following:

```
Replication version:              4.0
  Date created:                     Wed Oct 26 08:32:58 2005
  Date last written:                Wed Oct 26 08:33:46 2005

  Replication server information:
    Number of agents:               1
    Number of unused agents:        1
    Last modified:                  Wed Oct 26 08:32:08 2005
    Master block update count:      21

  Remote Agent information:
    Remote Agent 1
      Identification:               1
      Agent name:                   agent1
      Last AI block acknowledged:   area: 14, seq: 3, loc: 4259840, offset: 3189
      Last modified:                Wed Oct 26 08:33:29 2005
      Last AI block ACK time:       Wed Oct 26 08:33:46 2005
      Remote agent host:            localhost
      Remote agent database:        ks2
```

The replication recovery information for the agent looks similar to the following:

```
Replication version:              4.0
  Date created:                     Wed Oct 26 08:32:57 2005
  Date last written:                Wed Oct 26 08:34:36 2005

  Replication local agent information:
    Last Block:                     Complete
    Last block received location:   area: 14, seq: 3, loc: 4554752, offset: 0
    Last block processed location:  area: 0, seq: 0, loc: 0, offset: 0
    Last block ACKed location:      area: 14, seq: 3, loc: 4259840, offset: 3189
    Last block received:            no date
    Last block ACKed:               no date
    ID of the last TX begin:        4345
    ID of the last TX end:          4345
    Time of last TX end:            Wed Oct 26 08:33:46 2005
    Last AI Extent processed
      AIMAGE BEGIN date:            Wed Oct 26 08:32:12 2005
      AIMAGE NEW date:              Wed Oct 26 08:33:29 2005
      After-Image File Number:      3
      File Last Opened:             Wed Oct 26 08:33:29 2005
      Completly Applied to Target:  No
```

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

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

DSRUTIL restart agent qualifier

Restarts the OpenEdge Replication agent. The advantage of using this command is that you do not have to restart your target database.

If the OpenEdge Replication agent is currently running, it is not restarted and an error is logged.

Syntax

```
dsrutil db-name -C restart agent
```

Parameters

db-name

The name of the target 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 the Replication agent terminates, you can restart it with DSRUTIL restart agent qualifier on page 154, or 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

```
dsrutil db-name -C restart server
```

Parameters

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 DSRUTIL connectagent database qualifier on page 151.

DSRUTIL status qualifier

Allows you to query the status of a Replication server or agent.

Syntax

```
dsrutil db-name -C status agentname
     [ -detail | -verbose ]
```

Parameters

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 When the detail or verbose argument is used on page 156.

-verbose

Provides the detail value, as shown in When the detail or verbose argument is used on page 156, and provides a brief text description.

The return codes listed in the following table are valid.

Table 15: Replication status return codes

Code	Description
0	The command completed normally, and the status code was sent to stdout.
2	There was a generic database open error.
3	The database was opened, but it is not enabled for replication.

When the detail or verbose argument is used

When the -detail argument is specified, the numeric value from the Status column below is returned. When the -verbose argument is specified, the numeric value is returned, along with the text in the Description column below. The output is returned via stdout.

See the following table for a list and description of the possible status code values and descriptions.

Table 16: Detail and Verbose return codes

Status	Description
0	Unknown
1001	Initial Connection
1002	Initializing
1003	Target Database in Quiet Point
1032	Initial Connection Failed
1033	Recovery Failed
1034	Invalid Target Database Configuration
1035	Agent Failed

Status	Description
1036	Agent is Ignored
1037	Agent is Stopped
1038	Agent is Terminated
1063	Agent is Ended
2080	Pre Transition
2081	Applying After-image Extent
2082	Transitioning
2083	Listening
2084	Waiting while JTA transactions are resolved
3048	Startup Synchronization
3049	Normal Processing
3050	Recovery Synchronization
3051	Online backup of Target Database
3052	Target Database in a Quiet Point
3053	Target Database in BI Stall
3054	Target Database in AI Stall
3055	Being Transitioned

When the detail or verbose argument is not used

The return code equals zero when the `-detail` argument or `-verbose` is not used, and one of the status codes in the following table is returned via `stdout`.

Table 17: Return code zero status code

If the status code is . . .	It reflects the status for . . .
1*xxx*	The server

If the status code is . . .	It reflects the status for . . .
2*xxx*	The control agent
3*xxx*	The agent

See the following table 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 18: Status code values

Status	Class	Detail status	Value
100	Connecting.	Initial connection.	1001
100	Connecting.	Initializing.	1002
100	Connecting.	Server Initialization.	6001
100	Connecting.	Connecting to Agents.	6002
100	Connecting.	Configuring Agent(s).	6003
101	Processing is taking place.	Startup Synchronization.	3048
101	Processing is taking place.	Normal Processing.	3049
101	Processing is taking place.	Recovery Synchronization.	3050
101	Processing is taking place.	Recovery Processing.	6004
101	Processing is taking place.	Startup Synchronization.	6005
101	Processing is taking place.	Normal Processing.	6021
102	Activity is halted.	Online backup of the Target Database.	3051
102	Activity is halted.	Target Database in Quiet Point.	3052
102	Activity is halted.	Target Database is in a BI stall.	3053
102	Activity is halted.	Target Database is in an AI stall.	3054
103	Pre-transition state.	Pre-transition.	2080
103	Pre-transition state.	Applying After-image Extent.	2081
104	Agent is listening.	Listening.	2083
105	Transition.	Transitioning.	2082
199	Inactive.	Initial Connection Failed.	1032

Status	Class	Detail status	Value
199	Inactive.	Recovery Failed.	1033
199	Inactive.	Invalid Target Database Configuration.	1034
199	Inactive.	Agent Failed.	1035
199	Inactive.	Agent is Ignored.	1036
199	Inactive.	Agent is Stopped.	1037
199	Inactive.	Agent is Terminated.	1038
199	Inactive.	Agent is Ended.	1063
199	Inactive.	Server is ended.	6060
255	Unknown.	Unknown.	0

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

```
dsrutil db-name -C { terminate server | agent }
```

Parameters

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.

Note

When the OpenEdge Replication server terminates activity, the server responds, based on the value of the `agent-shutdown-action` property, as follows:

- If the `agent-shutdown-action` is `NORMAL`, then the replication agent(s) terminate, and the target database stays up.
- If the `agent-shutdown-action` is `RECOVERY`, then the replication agent(s) remain active, in a stand-by state, waiting for the replication server to reconnect.

DSRUTIL transition qualifier

Instructs an OpenEdge Replication agent to transition the target database. The database transitions to the role defined in the properties file (source or normal). Use this command to perform a manual transition of a target database that is in a pre-transition state.

Syntax

```
dsrutil db-name -C transition [ -Passphrase ] [ -logging [ 1 | 2 ] ]
```

Parameters

db-name

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

`-C transition`

Causes the Replication agent to start transition.

`-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.

`-logging`

Turns on transition logging.

`1`

Produces summary logging.

`2`

Produces detailed logging.

DSRUTIL transition failover qualifier

Instructs an online OpenEdge Replication source database to switch roles with an online target database.

Syntax

```
dsrutil source-db-name -C transition failover [ -Passphrase ] [ -logging [ 1 | 2 ] ]
```

Parameters

source-db-name

The name of the database, currently in the role of the source replica to perform the action on.

`-C transition failover`

Causes the Replication source to start the process of switching roles.

`-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.

`-logging`

Turns on transition logging.

`1`

Produces summary logging.

`2`

Produces detailed logging.

Failover transition reverses the roles of a source and target database. For details of the steps involved in failover transition, see Failback processing using transition failover on page 111.

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

```
dsrutil db-name -C triggertransition
```

Parameters

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 the following figure appears.

Figure 32: DSRUTIL Monitor Startup menu screen

```
OpenEdge Replication Monitor     Page 1

     Database: /twoagent/tdba

    S.  Replication server status
    R.  Replication server remote agents
    A.  Replication agent status
    I.  Replication inter-agent status

    M.  Modify display defaults
    Q.  Quit

    Enter your selection:
```

As shown in the previous figure, the first two lines of the Startup Menu are the utility title line and the fully qualified name of the database you specify. The remaining lines are defined in the table that follows.

Table 19: DSRUTIL monitor startup

Field	Description
`Replication server status`	Instructs you to enter the letter **S** to display the Replication server Status display screen
`Replication server remote agents`	Instructs you to enter the letter **R** to display the Replication remote agent Selection menu
`Replication agent status`	Instructs you to enter the letter **A** to display the Replication local agent Status display screen
`Replication inter-agent status`	Instructs you to enter the letter **I** to display the Replication inter-agent status display screen. The inter-agent status is only available when a replication set is enabled.
`Modify display defaults`	Instructs you to enter the letter **M** to display the current utility display settings and prompts the user for any desired changes
`Quit`	Instructs you to enter the letter **Q** to exit the utility
`Enter your selection`	Instructs you to enter a selection for the action the utility is to perform

Monitoring latency between the source and target databases

If there is a database failure of some kind and 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. The target database will be behind because 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 yet been applied to the target database, DSRUTIL MONITOR provides latency information from both the server's perspective and agent's perspective.

From the OpenEdge Replication server perspective

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

- The current database 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 the database

From the 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 about the DSRUTIL MONITOR reports on latency from the OpenEdge Replication server and agent perspectives, see Table 19: DSRUTIL monitor startup on page 163.

Replication server status

If you select **Replication server status** from the Startup Menu of the DSRUTIL Monitor Utility, the screen shown in the following figure appears.

Figure 33: OpenEdge Replication server status screen

```
OpenEdge Replication Monitor                    Page 1

  Database:  C:\OE120WRK\repl1

  Database is enabled as OpenEdge Replication:  Source

  Server is:                                    In Normal Processing
  Number of configured agents:                  2
  Delay Interval (current / min / max):         500 / 5 / 500
  Recovery information:
      State:                                    No recovery being performed
      Agents needing recovery:                  0
      Agents connected:                         0
      Agents in synchronization:                0
  Transition information:
      Type:                                     Manual
      Replication set:                          1
      Database role:                            Reverse
      Transition to agents:                     agent1, agent2
      Recovery transition to agents:            agent1, agent2

      Restart after transition:                 1
      Automatically begin AI:                   ?
      Automatically add AI areas:               1
      AI structure file:                        addai.st
      Backup method:                            Mark
      Transition end trigger:                   0
  Configured agents:                            agent1, agent2
  Repl keep alive:                              300
  Schema lock action:                           Wait
  Agent shutdown action:                        Recovery
```

If you have enabled deferred agent startup, you will see the following additional lines after the `Number of configured agents` line.

```
  Defer Agent Startup :
      Continue connection attempts until:     Mon Feb 26 16:37:46 2018
      Deferred Agent startup will expire in : 9 Hr 58 Min 37 Sec
      Next connection attempt in :            4 Min 41 Sec
      Connection attempts performed :         3
      Agent(s) currently connected :          0
```

As shown in the first figure above, the first two lines of the Replication server status screen are the utility title line and the fully qualified name of the database you specify. The remaining lines are defined in the table that follows.

Table 20: Replication server status

Field	Description
`Database is enabled as OpenEdge Replication`	Indicates whether the database is enabled as a replication source or target.
`Server is`	Describes how the server is processing information: • **Normal processing** — The server is processing information in the normal fashion. • **Performing initialization** — The server is performing initialization. • **Performing startup** • **synchronization** — The server is in the process of synchronizing the target databases with the source database. • **Connection**— The server connection to agent(s). • **Configuring connected agents**— The server is performing handshaking with the agent. • **Performing failure recovery** — The server is attempting failure recovery from a connection failure. • **Connecting to Agent(s)**— The server is in deferred agent startup mode, and waiting to connect to agents. • **Unknown**— The server is in an unknown state.
`Number of configured agents`	Shows the number of agents currently configured to operate with the server.
`Defer Agent Startup`	Shows information related to deferred agent startup.
`Continue connection attempts until`	Shows the time, day, and date when connection attempts will stop.
`Deferred Agent Startup will expire in`	Shows the remaining duration of deferred agent startup.
`Next connection attempt in`	Shows when the next connection attempt starts.
`Connection attempts performed`	Shows how many connection attempts have occurred.
`Agent(s) currently connected`	Shows the number of connected agents.

Field	Description
`Delay Interval (current / min / max)`	Shows the amount of time, in milliseconds, the server will wait between polling the database for information to be replicated. **Current** is the current value, **min** is the minimum value, and **max** is the maximum value. Polling is used to increase the performance of the server, and the wait is used to limit the amount of overhead when no data is available to be replicated.
`Recovery information`	Shows failure-recovery-related details.
`Agents needing recovery`	Shows the number of remote agents requiring failure recovery.
`Agents connected`	Shows the number of remote agents currently connected to the server. (Note that since the database is not going through recovery, the number of **Agents connected** indicated under **Recovery information** is correctly reflected as zero.)
`Agents in synchronization`	Shows the number of remote agents in the process of being brought up to date.
`Transition information`	Shows Replication transition details.
`Type`	Shows the type of transition to be performed: • **Manual** — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database • **Automatic**— Indicates a transition from a target database to a normal OpenEdge database will take place without intervention
`Transition timeout limit`	Shows the maximum amount of time that will elapse before the transition of a target database to a normal OpenEdge database. **Note:** This line does not appear in manual transition, as the timeout limit is shown for automatic transition only.
`Replication set`	Indicates if Replication is configured for Replication sets. Set to 1 if configured, 0 if not.
`Database role`	Indicates the role of the source database in transition. Valid values include: • Normal • Reverse

Field	Description
`Transition to agents`	Displays the names of the agents for transition. If the next line (`Recovery transition to agents`) is set, this field represents the agent list for *failover* transition; if not set, this line represents the agent list for both *failover* and *recovery* transition. If `replication-set=1`, the first agent in this is selected to become the new source.
`Recovery transition to agents`	Displays the names of the agents for recovery transition.
`Restart after transition`	Displays if the source database is configured to be restarted after transition. Set 1 for restart, 0 for no restart.
`Automatically begin AI`	When Replication Set =1, indicates whether to start after-imaging on a target database that is transition to a source. Set 1 to start after-imaging, 0 to not start. If Replication set is not enabled, set to the unknown value, "?"
`Automatically add AI areas`	Indicates if after-image areas should be automatically added to the target database that is transitioning to a source. Set to 1 to add after-image areas, 0 not to add AI.
`AI structure file`	The name of the structure file to use if automatically adding AI areas. If Automatically add AI is not enabled, set to the unknown value, "?"
`Backup method`	The backup method used during transition. Valid entries include: • Full-offline • Full-online • Mark
`Configured agents`	Displays the names of the configured replication target agents
`Repl keep alive`	Displays the time, in seconds, that replication will wait and attempt to re-establish communication before starting transition.

Field	Description
Schema lock action	Action to take if schema lock exists. Valid entries include: • Force • Wait
Agent shutdown action	Action to take when agent shuts down. Valid entries include: • Normal • Recovery • Wait

Replication server remote agents

If you select **Replication server remote agents** from the Startup Menu of the DSRUTIL Monitor Utility (shown in), the screen shown in the following figure appears.

Figure 34: OpenEdge Replication server remote agents screen

```
 OpenEdge Replication Monitor            Page 1

      Database: /twoagent/sdb

      Database is enabled as OpenEdge Replication:  Source

      Remote Agents Configured:
      ID   Name     Host Name     Database
      1.   agent1   localhost     tdba
      2.   agent2   localhost     tdbb

      Q. Quit      Enter your selection:
```

As shown in the previous figure, 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 the table that follows.

Table 21: Replication server remote agents details

Field	Description
Database is enabled as OpenEdge Replication	Indicates whether the database is enabled as a replication source or target.
Remote Agents Configured	Displays a list of the currently configured remote agents. Each remote agent is identified by an **ID**, the agent's **Name**, the **Host Name**, and the target **Database**.

Field	Description
Quit	Instructs you to type the letter Q to exit the **Replication Server Remote Agents** menu.
Enter your selection	Instructs you to type the ID of the remote agent whose information is to be displayed.

Replication remote agents status

If you select the **Replication remote agents status** from the Startup Menu of the DSRUTIL Monitor Utility, the screen shown in the following figure appears.

Figure 35: OpenEdge Replication remote agents status screen

```
OpenEdge Replication Monitor          Page 1

     Database: C:\OE120WRK\repl1
    Agent:
        Name:                                   agent1
        ID:                                     1
        Host name:                              127.0.0.1
        Target database:                        C:\OE120WRK\repl2
        State:                                  Normal Processing
        Critical:                               No
        Method:                                 Asynchronous
    Server/Agent connection time:               Fri Mar 16 10:14:16 2010
    Remote agent is waiting for:                Nothing
    Recovery state:                             No recovery being performed
    Maximum bytes in TCP/IP message:            8512
    Server/Agent connection timeout:            60.000 seconds
    Transition information:
        Type:                                   Manual
    The last block was sent at:                 Fri Mar 16 10:14:18 2018
    Activity information:
        Blocks sent:                            497
        Blocks acknowledged:                    1
        Synchronization points                  8
    AI Block Information:
        Current RDBMS Block (Seq / Block):      3 / 313
        Last Sent Block (Seq / Block):          3 / 313
    Server to agent load check interval:        10 blocks
    Time between server and agent load checks:  7002.666 seconds
    Time taken to respond to load check:        0.093 seconds
    Port:                                       1025
    Last message was received at:               Mon Mar 19 20:28:00 2018
    Connection state:                           Connected
```

As shown in the previous figure, 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 the table that follows.

Table 22: Replication remote agents status details

Field	Description
Agent	Shows a selection of basic remote agent information.

Field	Description
Name	Shows the name of the remote agent.
ID	Shows the remote agent identification number.
Host name	Shows the host on which the target database associated with the remote agent resides.
Target database	Shows the name of the database associated with the remote agent.
State	Shows details related to what the server knows about the remote agent. For example: • **Normal processing** — The server and the agent are performing normal processing. • **Initial connection** — The agent is waiting for initial connection from server. • **Startup synchronization** — The server and the agent are synchronizing. • **Initialization** — The agent is being initialized by the server. • **Initial connection failed** — The server could never connect to an agent. • **Invalid target database configuration** — Something in the target database does not match the source. • **Agent terminated** — The target database shut down or the agent terminated using `Terminate agent`. • **Online backup of Target Database** — An online target database backup is being performed. • **Recovery synchronization** — Recovery synchronization is being performed. • **Recovery failed** — Failure recovery failed. • **Unknown** — The agent is in an unknown state.
Critical	Shows whether the remote agent is a critical agent (**Yes**) or is not (**No**).
Method	Shows the replication method: **Asynchronous**. As of OpenEdge Release 12.0, asynchronous replication is the only supported replication method. Synchronous replication is no longer supported.

Field	Description
Remote agent is waiting for	Shows why the agent might be waiting. The field value might be either one of the following: • **Nothing** — Indicates no waiting is taking place • **Schema lock request** — Indicates the server is waiting for the agent to acquire the schema lock so the database schema can be changed
Recovery state	Shows where the server and the agent are in the failure recovery process. The field value can be one of the following: • **Failed for the agent** — Failure recovery could not be completed for the agent. • **No recovery being performed** — There is currently no failure recovery being done. • **Just entered recovery** — The server has just started failure recovery and will determine which remote agents remain connected and which must be reconnected. • **Server attempting connection** — The server is attempting to connect to those remote agents no longer communicating to the server. • **Initialize synchronization with agents** — The server is initializing the agent failure recovery synchronization. • **Synchronizing agents** — The connected agents are in the process of being brought up to date with the database changes made to the source database. • **Recovery complete** — The failure recovery of all connected agents has been completed.
Maximum bytes in TCP/IP message	Shows the maximum number of bytes used for the TCP/IP communication messages.
Server/Agent connection time	Shows the date and time at which the server and agent connected.
Server/Agent connection timeout	Shows the number of seconds after which the OpenEdge Replication server will stop attempting to connect to the agent.
Transition information	Shows remote agent-related transition details.
Type	Shows the type of transition to be performed: • **Manual** — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database • **Automatic** — Indicates a transition from a target database to a normal OpenEdge database will take place without intervention

Field	Description
`Timeout limit`	Shows the maximum amount of time that will elapse before the transition of a target database to a normal OpenEdge database; shown for automatic transition only.
`The last block was sent at`	Shows the date and time the last block was sent to the agent.
`Activity information`	Shows activity information.
`Blocks sent`	Shows the number of blocks sent to the agent.
`Blocks acknowledged`	Shows the number of blocks acknowledged by the agent.
`Synchronization points`	Shows the number of synchronization points that have occurred.
`AI Block Information`	Provides latency information that shows how far behind OpenEdge Replication is in updating the target database. This is important if the target database is transitioned due to source database failure.
`Current RDBMS Block (Seq / Block)`	Shows the current database AI block. `Seq` is the AI extent sequence number. It can be viewed by using `rfutil LIST`.
`Last Sent Block (Seq / Block)`	Shows the last AI block that was sent to the OpenEdge Replication agent.
`Server to agent load check interval`	Shows the number of blocks the server will send to the agent, at which point the server will wait for an acknowledgment from the agent for the block just sent.
`Time between server and agent load checks`	Shows the average elapsed time between the load checks.
`Time taken to respond to load check`	Shows the average elapsed time the agent took to acknowledge the server for the block just sent.
`Port`	Listening port of the agent
`Last message was received at`	Date-time the last message was received
`Connection state`	State of the connection. Valid values include: • Connected • Disconnected • Listening

Replication agent status

If you select the **Replication agent status** from the Startup Menu of the DSRUTIL Monitor Utility, the screen shown in the following figure appears.

Figure 36: OpenEdge Replication agent status screen

```
OpenEdge Replication Monitor                  Page 1
Database:  /scratch/doc/120/demo/repl2

    Database is enabled as OpenEdge Replication:  Target

    Agent:
        Name:                                   agent1
        ID:                                     1
        Host name:                              127.0.0.1
        State:                                  Normal Processing
        Ready:                                  Yes
        Critical:                               No
        Method:                                 Asynchronous
    Agent is waiting for:                       Nothing
    Maximum bytes in TCP/IP message:            8496
    Server/Agent connection time:               Tue May 15 18:08:41 2018
    Delay Interval (current / min / max):       10 / 5 / 500
    Transition information:
        Type:                                   Manual
    The last block received at:                 Tue May 15 18:16:40 2018
    Activity information:
        Blocks received:                        1857
        Blocks processed:                       1858
        Blocks acknowledged:                    0
        Notes processed:                        184042
        Transactions started:                   0
        Transactions ended:                     0
        Synchronization points:                 29
    AI Block Information:
        Source RDBMS Block (Seq / Block):       4 / 6905
        Last Received Block (Seq / Block):      4 / 6884
        Last Processed Block (Seq / Block):     4 / 6884
    Latency Information:
        Repl Server behind Source DB by:        0     second(s)
        Current Source Database Transaction:    2303
        Last Transaction Applied to Target:     2303
        Target Current as of (Target, Source):  Tue May 15 18:15:58 2018, Tue May 15
18:15:58 2018 with delta of 000:00:00
        Number of blocks to send on Source:     0 as of Tue May 15 18:16:40 2018
        Number of blocks to process on Target:  0 as of Tue May 15 18:16:40 2018
        Time delay to process the last block:   0 second(s)
    Connect timeout:                            1800
    Listener port range:                        1025-65500
    Current listener port:                      1037
    Additional transition information:
        Replication set:                        1
        Database role:                          Reverse
        Transition to agents:                   agent1,agent2
        Recovery transition to agents:
        Restart after transition:               1
        Automatically begin AI:                 ?
        Automatically add AI areas:             1
        AI structure file:                      addai.st
        Backup method:                          Full-offline
        Transition end trigger:                 0
        AI Streaming enabled:                   1
        Apply Streaming Data:                   1
    Replication streaming block cache:
        Cache size:                             1000
```

```
        Free cache entries:                     999
        Used cache entries:                     1
        Cache high water mark:                  175
        Number of times cache filled:           0
```

As shown in the previous figure, the first two lines of the **Replication agent status** screen are the utility title line and the fully qualified name of the database specified by the user. The remaining lines of the Replication Agent Status screen are defined in the table that follows.

Table 23: Replication Agent Status details

Field	Description
`Database is enabled as OpenEdge Replication`	Indicates whether the database is enabled as a replication source or a target.
`Agent`	Lists basic remote agent information.
`Name`	Lists the name of the remote agent.
`ID`	Lists the remote agent identification number.
`Host name`	Lists the host on which the target database associated with the remote agent resides.

Field	Description
State	Shows agent processing information: • **Normal processing** — The server and the agent are performing normal processing. • **Initial connection** — The agent is waiting for initial connection from server. • **Startup synchronization** — The server and the agent are synchronizing. • **Initialization** — The agent is being initialized by the server. • **Initial connection failed** — The server could never connect to an agent. • **Invalid target database configuration** — Something in the target database does not match the source. • **Agent terminated** — The target database shut down or the agent terminated using `Terminate agent`. • **Online backup of Target Database** — An online target database backup is being performed. • **Recovery synchronization** — Recovery synchronization is being performed. • **Recovery failed** — Failure recovery was unsuccessful. • **Unknown** — The agent is in an unknown state.
Critical	Shows whether the remote agent is a critical agent (**Yes**) or is not (**No**).
Method	Lists the replication method: **asynchronous**.
Agent is waiting for	Shows why the agent might be waiting. The field value might be one of the following: • **Nothing** — Indicates no waiting is taking place • **Schema lock request** — Indicates the server is waiting for the agent to acquire the schema lock so the database schema can be changed • **Encryption objects** — For an encrypted database, the agent's wait status, or what the agent is waiting for
Maximum bytes in TCP/IP message	Shows the maximum number of bytes used for the TCP/IP communication messages.
Server/Agent connection time	Shows the date and time at which the server and agent connected.

Field	Description
Delay Interval (current / min / max)	Shows the amount of time, in milliseconds, the agent will wait between polling the TCP/IP connection for information to be replicated. **Current** is the current value, **min** is the minimum value, and **max** is the maximum value. Polling is used to increase the performance of the server, and the wait is used to limit the amount of overhead when no data is available to be replicated.
Transition information	Shows a group of remote agent-related transition information.
Type	Shows the type of transition to be performed: • **Manual** — Indicates intervention is required to complete the transition of a target database to a normal OpenEdge database • **Automatic** — Indicates a transition from a target database to a normal OpenEdge database will take place without intervention
The last block received at	Shows the date and time the last block was sent to the agent.
Activity information	Shows a selection of activity information.
Blocks received	Shows the number of blocks received from the server.
Blocks processed	Shows the number of blocks processed.
Blocks acknowledged	Shows the number of blocks acknowledged.
Notes processed	Shows the number of AI transaction notes processed by the agent.
Transactions started	Shows the number of transactions started on the agent.
Transactions ended	Shows the number of transactions completed on the agent.
Synchronization points	Shows the number of synchronization points that have occurred.
AI Block Information	Identifies the current AI block that is being written by the source database and provides the last AI block that was sent by Replication.
Source RDBMS Block (Seq / Block)	Provides the current source database block that the OpenEdge Replication agent is processing.
Last Received Block (Seq / Block)	Provides the AI location, defined by AI sequence number and AI block number, of the last block received by the replication agent.
Last Processed Block (Seq / Block)	Provides the AI location, defined by AI sequence number and AI block number, of the last processed block by the replication agent.
Latency Information	Provides latency information that shows how far behind OpenEdge Replication is in updating the target database. This is important if the target database is transitioned due to source database failure.

Field	Description
`Repl Server behind Source DB by`	Provides the number of seconds the OpenEdge Replication server is behind the source database.
`Current Source Database Transaction`	Provides the current source database transaction that the OpenEdge Replication agent is processing.
`Last Transaction Applied to Target`	Provides the last source database transaction that was applied to the target database.
`Target Current As Of`	Provides the date of the last transaction applied to the target database by the OpenEdge Replication agent based on source database time.
`Number of blocks to send on Source`	Provides the number of AI blocks to send on the source database.
`Number of blocks to process on Target`	Provides the number of AI blocks to process on the target database.
`Time delay to process the last block`	Provides the time it takes for an AI block to be processed by the agent after it has been received.
`Connect timeout`	Number of seconds during which agent attempts to reconnect to server.
`Listener port range`	Range of ports for agent
`Current listener port`	Current port in use
`Replication set`	Set to 1 if Replication set is enabled, 0 otherwise.
`Database role`	Role of agent database in transition
`Transition to agents`	Displays the names of the agents for transition. If the next line (`Recovery transition to agents`) is set, this field represents the agent list for *failover* transition; if not set, this line represents the agent list for both *failover* and *recovery* transition. If `replication-set=1`, the first agent in this is selected to become the new source.
`Recovery transition to agents`	Displays the names of the agents for *recovery* transition.
`Restart after transition`	Indicates if restart should occur after transition. Set to 1 for yes, 0 for no.
`Automatically begin AI`	For replication set only, indicates if after-imaging should automatically be started on transition. Set to 1 to begin after-imaging, 0 to not. If replication set is not enabled, set to the unknown value, "?"

Field	Description
Automatically add AI areas	Indicates if after-image areas should be automatically added to the target database that is transitioning to a source. Set to 1 to add after-image areas, 0 not to add AI.
AI structure file	The name of the structure file to use if automatically adding AI areas. If Automatically add AI is not enabled, set to the unknown value, "?"
Backup method	The backup method for database during transition. Valid entries include: • Full-offline • Full-online • Mark
Transition end trigger	Indicates if a transition end trigger should run at the end of transition. Set to 1 to indicate the trigger property is enabled, and set to 0 to indicate no transition end trigger.
AI Streaming enabled	Indicates if Replication AI Streaming is enabled. 1 indicates enabled, 0 indicates not enabled.
Apply Streaming Data	If Replication AI Streaming is enabled, indicates how to treat data in the RSB cache during auto transition. 1 indicates that data in the RSB cache is applied before transition; 0 indicates that data in the RSB cache is not applied before auto transition.
Replication streaming block cache	Shows a group of related information about the cache for Replication AI Streaming.
Cache size	Displays the cache size in units of Replication Streamed Blocks (RSB).
Free cache entries	Displays the number of free cache entries in the RSB cache.
Used cache entries	Displays the number of used cache entries in the RSB cache.
Cache high water mark	Displays the maximum number of cache entries that have been in use at one point in the life cycle of the cache.
Number of times cache filled	Displays the number of times the RSB cache has filled.

Replication inter-agent status

If you select the **Replication inter-agent status** from the Startup Menu of the DSRUTIL Monitor Utility for a replication target, the screen shown in the following figure appears.

Figure 37: OpenEdge Replication remote agents status screen

```
OpenEdge Replication Monitor               Page 1

   Database: /twoagent/repl2

   Database is enabled as OpenEdge Replication:  Target

   Local Agent:                                  agent1
   ID:                                           1
   Host name:                                    127.0.0.1
   Port:                                         1025
   Server latency at last message:               0     second(s)
   Synchronization points:                       8

   Remote Agent:                                 agent2
   ID:                                           2
   Host name:                                    localhost
   Port:                                         1026
   Database name:                                /twoagent/repl3
   Connection state:                             Connected

   Last message sent at:                         Thu Feb  2 17:05:36 2017
   Last message received at:                     Thu Feb  2 17:05:36 2017
   Last ping sent at:                            Thu Feb  2 17:05:36 2017
   Last ping received at:                        Thu Feb  2 17:05:36 2017
   Messages sent:                                9026
   Messages received:                            9027
   Synchronization points:                       3
   Last acknowledged Source AI sequence:         6
   Last acknowledged Transaction:                351
```

As shown in the previous figure, the first two lines of the **Replication inter-agent status** screen are the utility title line and the fully qualified name of the database specified by the user.

The remaining lines of the **Replication inter-agent status** screen are defined in the table that follows.

Table 24: Replication inter-agent status details

Field	Description
`Local Agent`	Displays the name of the local agent.
`ID`	Displays the ID of the local agent.
`Host name`	Displays the host on which local the target database resides.
`Port`	Displays the port on which local agent is listening.
`Server latency at last message`	Displays, in seconds, the latency between the source and the local target.

Field	Description
Synchronization points	Displays the number of synchronization point messages received from the remote agent.
Remote Agent	Displays the name of the remote agent.
ID	Displays the ID of the remote agent.
Host name	Displays the host on which the remote target database resides.
Port	Displays the port on which remote agent is listening.
Database name	Displays the database name for the remote agent.
Connection state	Shows details related to what is known about the remote agent. For example: • **Normal processing** — The server and the agent are performing normal processing. • **Initial connection** — The agent is waiting for initial connection from server. • **Startup synchronization** — The server and the agent are synchronizing. • **Initialization** — The agent is being initialized by the server. • **Initial connection failed** — The server could never connect to an agent. • **Invalid target database configuration** — Something in the target database does not match the source. • **Agent terminated** — The target database shut down or the agent terminated using Terminate agent. • **Online backup of Target Database** — An online target database backup is being performed. • **Recovery synchronization** — Recovery synchronization is being performed. • **Recovery failed** — Failure recovery failed. • **Unknown** — The agent is in an unknown state.
Last message sent at:	Displays the date and time that the last inter-agent message was sent.
Last message received at:	Displays the date and time that an inter-agent message was last received from the remote agent.
Last ping sent at:	Displays the date and time that the last inter-agent ping was sent.

Field	Description
Last message ping at:	Displays the date and time that an inter-agent ping was most recently received.
Messages sent	Shows the number of inter-agent messages sent to the remote agent.
Messages received	Shows the number inter-agent messages received by the local agent.
Synchronization points	Shows the number of synchronization points that have occurred.
Last acknowledged Source AI sequence	The sequence number of the last processed source AI sequence acknowledged from the remote agent.
Last acknowledged transaction	The transaction ID of the last processed transaction acknowledged from the remote agent.

If a remote agent is not connected, the remote data is not displayed and is replaced with the following message:

```
 No connected remote agents.
```

Virtual system tables for OpenEdge Replication

Virtual system tables are available with OpenEdge Replication, as shown in the table that follows.

Table 25: OpenEdge Replication virtual system tables

Virtual system table	Description
OpenEdge Replication Server (_Repl-Server)	Provides detailed information about the OpenEdge Replication server.
OpenEdge Replication Agent (_Repl-Agent)	Provides detailed information about an OpenEdge Replication agent.
OpenEdge Replication Agent Activity (_Repl-AgentActivity)	Provides dynamic information about active OpenEdge Replication agents.
OpenEdge Replication Control Agent (_Repl-AgentControl)	Provides detailed information about the OpenEdge Replication agents this OpenEdge Replication server is controlling.
OpenEdge Replication Agent Control Activity (_Repl-AgentControlActivity)	Provides dynamic information about the active OpenEdge Replication agents this OpenEdge Replication server is controlling.
OpenEdge Replication Agent AI Streaming (_Repl-AgentAIStreaming)	Provides detailed information about the OpenEdge Replication AI streaming cache.

Virtual system table	Description
OpenEdge Replication Service Manager(_DbServiceManager)	Provides detailed information about the OpenEdge Replication plugin manager communication area.
OpenEdge Replication Service Manager Objects (_DbServiceManagerObjects)	Provides detailed information about registered objects of an OpenEdge Database. Possible service objects include: • OpenEdge Replication Server • OpenEdge Replication Agent • OpenEdge RDBMS • OpenEdge Management DB Agent
OpenEdge Replication Inter-agent Activity(_Repl-InterAgentActivity)	Provides detailed information about inter-agent communication.

See the sections that follow for descriptions of the fields in the VSTs listed above.

For more information on using virtual system tables, see *OpenEdge Data Management: Database Administration*.

_Repl-Server VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication virtual system table, _Repl-Server.

Table 26: Virtual system table _Repl-Server field description

Field name	Field number	Data type	Description
_ReplSrv-id	2	INT64	Server Number Format ->>>>>>>>>>>>>>>>>>9
_ReplSrv-AgentCount	3	INTEGER	Agents Count Number of agents that the OpenEdge Replication server is controlling Format >>>>>>>>>9
_ReplSrv-BlocksSent	4	INTEGER	After-image(AI) Blocks Sent Number of AI Blocks the OpenEdge Replication server has sent to connected OpenEdge Replication agents Format >>>>>>>>>9

Field name	Field number	Data type	Description
_ReplSrv-StartTime	5	CHARACTER	Server Start Time Date and time when the OpenEdge Replication server was started Format X(24)
_ReplSrv-LastBlockSentAt	6	CHARACTER	Last Block Sent At Date and time when the OpenEdge Replication server was started Format X(24)
ReplSrv-Database	7	CHARACTER	Database file name Format X(24)
ReplSrv-DatabaseRole	8	CHARACTER	Database role. Valid values are: • Replication source • Replication target Format X(8)
ReplSrv-ServerState	9	CHARACTER	Server state. Valid values are: • Performing Initialization • Connecting to Agents • Configuring Connected Agents • Performing Failure Recovery • Performing Startup Synchronization • In Normal Processing • Unknown Format X(36)
ReplSrv-DeferAgentConnUntil	10	CHARACTER	Date and time that deferred agent connection attempts will cease Format X(24)
ReplSrv-DeferAgentExpire	11	CHARACTER	Time remaining until deferred agent startup expires Format X(24)
ReplSrv-NextConnectIn	12	CHARACTER	Time until next connection attempt Format X(24)

Field name	Field number	Data type	Description
ReplSrv-ConnectAttempts	13	INT64	Number of connection attempts performed Format ->>>>>>>>>>>>>>>>>>>9
ReplSrv-AgentsConnected	14	INTEGER	Number of agents currently connected Format >>>>>>>>>9
ReplSrv-CurDelayInterval	15	INTEGER	Current message delay interval (in seconds) Format >>>>>>>>>9
ReplSrv-MinDelayInterval	16	INTEGER	Minimum message delay interval (in seconds) Format >>>>>>>>>9
ReplSrv-MaxDelayInterval	17	INTEGER	Maximum message delay interval (in seconds) Format >>>>>>>>>9
ReplSrv-RecoveryState	18	CHARACTER	Server recovery state. Valid values are: • No Recovery • Just Entered Recovery • Server Attempting Connection • Initialize Synchronization with Agents • Synchronizing Agents • Recovery Complete Format X(42)
ReplSrv-RecAgentsToRecover	19	INTEGER	Number of agents needing recovery Format >>>>>>>>>9
ReplSrv-RecAgentsConnected	20	INTEGER	Number of agents connected and not needing recovery Format >>>>>>>>>9
ReplSrv-AgentsInSync	21	INTEGER	Number of agents in synchronization Format >>>>>>>>>9

Field name	Field number	Data type	Description
ReplSrv-TransitionType	22	CHARACTER	Transition type. Valid values are: • Manual • Automatic Format X(32)
ReplSrv-TransitionTimeout	23	INTEGER	For automatic transition, the maximum amount of time (in seconds) that the target database will wait until it transitions to a normal database. Format >>>>>>>>>9
_ReplSrv-ReplicationSet	24	INTEGER	Replication set enabled Format >>>9
_ReplSrv-TransDatabaseRole	25	CHARACTER	Role the database takes after transition. Valid values are: • Normal • Reverse Format X(32)
_ReplSrv-TransitionToAgents	26	CHARACTER	Transition to agents list Format X(100)
_ReplSrv-RestartAfterTrans	27	BOOLEAN	Restart after transition Format yes/no
_ReplSrv-AutoBeginAI	28	BOOLEAN	Automatically begin after-imaging during transition Format yes/no
_ReplSrv-AutoAddAI	29	BOOLEAN	Automatically add after-imaging files during transition Format yes/no
_ReplSrv-AIFile	30	CHARACTER	AI structure filename for automatically adding after-imaging Format X(256)
_ReplSrv-BackupMethod	31	CHARACTER	Backup method during transition Format X(32)

Field name	Field number	Data type	Description
_ReplSrv-ConfiguredAgents	32	CHARACTER	List of configured agents Format X(68)
_ReplSrv-ReplKeepAlive	33	INT64	Replication keep alive property (in seconds) Format ->>>>9
_ReplSrv-SchemaLockAction	34	CHARACTER	Method server uses to perform schema locking. Valid values include: • Force • Wait Format X(32)
_ReplSrv-AgentShutdownAction	35	CHARACTER	Action replication agent takes if connection to server is lost. Valid values include: • Normal • Recovery • Wait Format X(32)
_ReplSrv-ReservedChar	36	CHARACTER	Reserved
_ReplSrv-ReservedInt	37	INTEGER	Reserved
_ReplSrv-TransEndTrigger	38	INTEGER	Indicates if a transition end trigger is to execute on transition. Valid values include: • **0** — Execute a transition end trigger • **1** — Do not execute a transition end trigger Format >>>9
_ReplSrv-DRTransitionToAgents	39	CHARACTER	Recovery transition to agents list. Format X(100)

_Repl-Agent VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication VST, _Repl-Agent.

Table 27: Virtual system table _Repl-Agent field descriptions

Field name	Field number	Data type	Description
_ReplAgt-AgentID	2	INT64	Agent number The ID of the OpenEdge Replication agent Format ->>>>>>>>>>9
_ReplAgt-AgentName	3	CHARACTER	Agent Name Name of the OpenEdge Replication agent as configured in the repl.properties file Format X(32)
_ReplAgt-ConnectTime	4	CHARACTER	Agent Connect Time Date and time when the OpenEdge Replication server successfully connected to the OpenEdge Replication agent Format X(24)
_ReplAgt-DBName	5	CHARACTER	Database Name Fully qualified database name that the OpenEdge Replication agent is connected to Format X(128)
_ReplAgt-ServerHost	6	CHARACTER	Server Host Name Name of the host where the OpenEdge Replication server is running Format X(128)
_ReplAgt-Port	7	INTEGER	The agent is connected to this port number The port number the OpenEdge Replication server uses to connect to this OpenEdge Replication agent Format ->>>>>>>>>>9
_ReplAgt-BlocksReceived	8	INT64	Number of blocks received The number of after-image (AI) blocks this OpenEdge Replication server has received Format ->>>>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgt-BlocksProcessed	9	INT64	Number of blocks processed. The number of AI blocks this OpenEdge Replication agent has processed Format ->>>>>>>>>>9
_ReplAgt-BlocksACK	10	INTEGER	Number of block acknowledgements sent by this agent The number of block acknowledgements this OpenEdge Replication agent has sent to the OpenEdge Replication server Format ->>>>>>>>>>9
_ReplAgt-NotesProcessed	11	INTEGER	Number of Notes Processed The number of AI notes this OpenEdge Replication agent has processed Format ->>>>>>>>>>9
_ReplAgt-LastTRID	12	INTEGER	Transaction ID of Last Transaction began The Transaction ID of the last Transaction Begin encountered Format ->>>>>>>>>>9
_ReplAgt-Method	13	CHARACTER	Replication Method The type of replication that is being performed with this OpenEdge Replication agent, **A** — Asynchronous Format X(1)
_ReplAgt-Status	14	INTEGER	A value from the **Value** column in Table 18: Status code values on page 158, as reported by the following command: `DSRUTIL -C status -detail` Format >>>>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgt-CommStatus	15	INTEGER	Agent's Communication Status The OpenEdge Replication agent's TCP communication status: • **1** — Connected • **2** — Disconnected Format >>>>>>>>>9
ReplAgt-DatabaseRole	16	CHARACTER	Database role. Valid values are: • Replication source • Replication target Format X(8)
ReplAgt-Critical	17	INTEGER	Critical agent flag. Valid values are: • **1** — Critical • **0** — Not Critical Format >>>>>>>>>9
ReplAgt-MaxMsgSize	18	INT64	Maximum number of bytes in a TCP message Format ->>>>>>>>>>>>>>>>>>9
ReplAgt-MinDelayInterval	19	INTEGER	Minimum message delay interval (in seconds) Format >>>>>>>>>9
ReplAgt-MaxDelayInterval	20	INTEGER	Maximum message delay interval (in seconds) Format >>>>>>>>>9
ReplAgt-TransitionType	21	CHARACTER	Transition type. Valid values are: • Manual • Automatic Format X(24)
ReplAgt-TimeoutLimit	22	INTEGER	Time (in seconds) agent will try to reconnect before entering transition. Format >>>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgt-ConnectTimeout	23	INT64	Agent's connect timeout. Agent will try to connect to the server this many seconds. Format ->>>>>>>>>>>>>>>>>>>9
_ReplAgt-MinPort	24	INT64	Minimum listener port number Format ->>>>>>>>>>>>>>>>>>>9
_ReplAgt-MaxPort	25	INT64	Maximum listener port number Format ->>>>>>>>>>>>>>>>>>>9
_ReplAgt-ReplicationSet	26	INTEGER	Replication set enabled Format >>>9
_ReplAgt-TransDatabaseRole	27	CHARACTER	Role database takes after transition. Valid values include: • Normal • Reverse Format X(32)
_ReplAgt-TransitionToAgents	28	CHARACTER	Transition to agents Format X(100)
_ReplAgt-RestartAfterTrans	29	BOOLEAN	Restart database after transition Format yes/no
_ReplAgt-AutoBeginAI	30	BOOLEAN	Automatically begin after-imaging during transition Format yes/no
_ReplAgt-AutoAddAI	31	BOOLEAN	Automatically add after-image files during transition Format yes/no
_ReplAgt-AIFile	32	CHARACTER	AI structure filename for adding after-imaging during transition Format X(256)

Field name	Field number	Data type	Description
_ReplAgt-BackupMethod	33	CHARACTER	Backup method during transition. Valid values include: • Full-offline • Full-online • Mark Format X(32)
_ReplAgt-ReservedChar	34	CHARACTER	Reserved
_ReplAgt-ReservedInt	35	INTEGER	Reserved
_ReplAgt-TransEndTrigger	36	INTEGER	Indicates if a transition end trigger is to execute on transition. Valid values include: • **0** — Execute a transition end trigger • **1** — Do not execute a transition end trigger Format >>>9
_ReplAgt-AIStreaming	37	BOOLEAN	Indicates if AI streaming is enabled. Valid values include: • **yes** — Enabled • **no** — Disabled Format yes/no
_ReplAgt-DRTransitionToAgents	38	CHARACTER	Recovery transition to agents list. Format X(100)
_ReplAgt-ApplyStreamingData	39	BOOLEAN	Indicates if AI Streaming data residing in the RSB cache should be applied to the target prior to auto-transition. • **yes** — Apply data • **no** — Do not apply data Format yes/no

_Repl-AgentActivity VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication VST, _ReplAgentActivity.

Table 28: Virtual system table _ReplAgentActivity field description

Field name	Field number	Data type	Description
_ReplAgtAct-AgentID	2	INT64	Agent number Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-AgentName	3	CHARACTER	Agent name Format X(32)
_ReplAgtAct-ConnectTime	4	CHARACTER	Date and time of replication agent/server connect Format X(24)
_ReplAgtActBlocksReceived	5	INT64	The number of after-image (AI) blocks this agent has received Format >>>>>>>>9
_ReplAgtAct-BlocksProcessed	6	INT64	The number of AI blocks this agent has processed Format >>>>>>>>9
_ReplAgtAct-BlocksACK	7	INTEGER	The number of block acknowledgments this agent has sent to the server Format >>>>>>>>9
_ReplAgtAct-NotesProcessed	8	INTEGER	The number of AI notes this agent has processed Format >>>>>>>>9
_ReplAgtAct-LastTRID	9	INTEGER	The TRID of the last transaction on the target Format >>>>>>>>9
_ReplAgtAct-Status	10	INTEGER	Current status of this replication agent Format >>>>>>>>9
_ReplAgtAct-CommStatus	11	INTEGER	Current TCP communication status of this agent. Valid values are: • **1** — Connected • **2** — Disconnected Format >>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgtAct-State	12	CHARACTER	Agent state. Valid values are: • Terminating • In Normal Processing • In Synchronization Processing • In Forced Transition • In Pre-Transition • Ready For Manual Transition Format X(48)
_ReplAgtAct-Ready	13	INTEGER	Agent ready status. Valid values are: • **0** — no • **1** — yes Format >>>>>>>>9
_ReplAgtAct-WaitingFor	14	CHARACTER	What this agent is waiting for. Valid values include: • Nothing • Schema Lock Request • Encryption Objects • Schema Object state change Format X(48)
_ReplAgtAct-CurDelayInterval	15	INTEGER	Current message delay interval (in seconds) Format >>>>>>>>9
_ReplAgtAct-TimeToTransition	16	CHARACTER	Time remaining before transition Format X(24)
_ReplAgtAct-LastBlockReceived	17	CHARACTER	Last block received on/at Format X(24)
_ReplAgtAct-TransStarted	18	INT64	Number of transactions started Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-TransEnded	19	INT64	Number of transactions ended Format ->>>>>>>>>>>>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgtAct-SyncPoints	20	INT64	Number of synchronization points Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-SrcRDBMSSeq	21	INT64	The AI sequence number of the most recent AI block on the source database. -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-SrcRDBMSBlock	22	INT64	The AI block number of the most recent AI block on the source database. -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-LastRecvBlockSeq	23	INT64	The AI sequence number of the last received AI block on the target database. -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-LastRecvBlockBlock	24	INT64	The AI block number of the last received AI block on the target database. -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-LastBlockSeq	25	INT64	Last processed AI block sequence number -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-LastBlockBlock	26	INT64	Last processed AI block -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtAct-ServerLag	27	INTEGER	Seconds the replication server is behind the source database -1 if not available Format >>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgtAct-SourceTRID	28	INTEGER	Last TRID from the source database -1 if not available Format >>>>>>>>9
_ReplAgtAct-LastSyncTimeTgt	29	CHARACTER	Time that the target was last current Format X(24)
_ReplAgtAct-LastSyncTimeSrc	30	CHARACTER	Time that the source was last current Format X(24)
_ReplAgtAct-LastSyncDelta	31	CHARACTER	Delta in Source/Target Sync Format X(24)
_ReplAgtAct-ServerBlocksToSend	32	INTEGER	Number of AI Blocks in the Replication server's PICA queue at time of the last AIBLOCK message Format >>>>>>>>9
_ReplAgtAct-LastBlockCountTime	33	CHARACTER	Date/time the Replication server last sent the PICA queue information to the replication agent Format X(24)
_ReplAgtAct-BlockProcDelay	34	INTEGER	The elapsed time between when the last timestamped block was received by the Replication agent, and when it was applied to the target database Format >>>>>>>>9
_ReplAgtAct-ReservedChar	35	CHARACTER	Reserved for future use
_ReplAgtAct-ReservedInt	36	INTEGER	Reserved for future use

_Repl-AgentControl VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication VST, _Repl-AgentControl.

Table 29: Virtual system table _Repl-AgentControl field description

Field name	Field number	Data type	Description
_ReplAgtCtl-AgentID	2	INT64	Agent number The ID of the OpenEdge Replication agent Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtCtl-AgentName	3	CHARACTER	Agent Name Name of the OpenEdge Replication agent as configured in the repl.properties file Format X(32)
_ReplAgtCtl-ConnectTime	4	CHARACTER	Agent Start Time Date and time when the OpenEdge Replication server successfully connected to the OpenEdge Replication agent Format X(24)
_ReplAgtCtl-RemoteDBName	5	CHARACTER	Remote Database Name Fully qualified database name to which the OpenEdge Replication agent is connected Format X(128)
_ReplAgtCtl-RemoteHost	6	CHARACTER	Remote Host Name Name of the host where the OpenEdge Replication agent is running Format X(128)
_ReplAgtCtl-Port	7	INTEGER	Connected to Agent on Port Number The port number the OpenEdge Replication server uses to connect to this OpenEdge Replication agent Format >>>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgtCtl-BlockSent	8	INTEGER	Blocks Sent to this Agent The number of after-image (AI) blocks the OpenEdge Replication server has sent to this OpenEdge Replication agent Format >>>>>>>>>9
_ReplAgtClt-BlocksACK	9	INTEGER	Number of block acknowledgments received for this agent The number of block acknowledgements this OpenEdge Replication agent has received from its remote counterpart Format >>>>>>>>>9
_ReplAgtCtl-LastBlockSentAt	10	CHARACTER	Last Block Sent At The data and time the last block was sent to this OpenEdge Replication agent Format X(24)
_ReplAgtCtl-Method	11	CHARACTER	Replication Method The type of replication being performed with this OpenEdge Replication agent, **A** — Asynchronous (mode & RP_MODE_ASYNC) Format X(1)
_ReplAgtCtl-status	12	INTEGER	Current status of the agent. A value from the **Value** column in Table 18: Status code values on page 158, as reported by the following command: DSRUTIL -C status -detail Format >>>>>>>>>9
_ReplAgtCtl-CommStatus	13	INTEGER	Agent's Communication Status The OpenEdge Replication agent's TCP communication status: • **1** — Connected • **2** — Disconnected Format >>>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgtCtl-Critical	14	INTEGER	Critical agent flag. Valid values are: • **1** — Critical • **0** — Not Critical Format >>>>>>>>>>9
_ReplAgtCtl-MaxMsgSize	15	INT64	Maximum number of bytes in a TCP message Format >>>>>>>>>>9
_ReplAgtCtl-ConnectionTimeout	16	DECIMAL	Timeout interval, in seconds Format >>>>>>>>>>9.999
_ReplAgtCtl-TransitionType	17	CHARACTER	Transition type. Valid values are: • Manual • Automatic Format X(32)
_ReplAgtCtl-TimeoutLimit	18	INTEGER	Time (in seconds) agent will try to reconnect before entering transition. Format >>>>>>>>>>9

_Repl-AgentControlActivity VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication VST, _Repl-AgentControlActivity.

Table 30: Virtual system table _Repl-AgentControlActivity field description

Field name	Field number	Data type	Description
_ReplAgtCtlAct-AgentID	2	INT64	Agent number Format ->>>>>>>>>>>>>>>>>>>9
_ReplAgtCtlAct-AgentName	3	CHARACTER	Agent name Format X(32)
_ReplAgtCtlAct-ConnectTime	4	CHARACTER	Date and time of replication agent/server connect Format X(24)

Field name	Field number	Data type	Description
_ReplAgtCtlAct-BlocksSent	5	INTEGER	The number of after-image (AI) blocks the server has sent Format >>>>>>>>>9
_ReplAgtCtlAct-BlocksACK	6	INTEGER	The number of AI blocks the agent has processed Format >>>>>>>>>9
_ReplAgtCtlAct-LastBlockSentAt	7	CHARACTER	Time that the last block was received Format X(24)
_ReplAgtCtlAct-Status	8	INTEGER	Current status of the agent. Valid values are: • 1 = Normal • 2 = In Recovery Mode Format >>>>>>>>>9
_ReplAgtCtlAct-CommStatus	9	INTEGER	Current TCP communication status of the agent. Valid values are: • 1 = Connected • 2 = Disconnected Format >>>>>>>>>9
_ReplAgtCtlAct-State	10	CHARACTER	Current state of the agent Format X(48)
_ReplAgtCtlAct-WaitingFor	11	CHARACTER	What this agent is waiting for. Valid values are: • Nothing • Schema Lock Request Format X(24)

Field name	Field number	Data type	Description
_ReplAgtCtlAct-RecoveryState	12	CHARACTER	Recovery state of this agent. Valid values are: • Failed for the Agent • No Recovery Being Performed • Just Entered Recovery • Server Attempting to Connect to Agent • Initialize Synchronization • Synchronizing • Recovery Complete Format X(48)
_ReplAgtCtlAct-SyncPoints	13	INT64	Number of synchronization points Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtCtlAct-CurRDBMSSeq	14	INT64	Current AI Block Sequence -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtCtlAct-CurRDBMSBlock	15	INT64	Current AI Block -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtCtlAct-LastBlockSeq	16	INT64	Last sent AI Block Sequence -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtCtlAct-LastBlockBlock	17	INT64	Last sent AI Block -1 if not available Format ->>>>>>>>>>>>>>>>>>9
_ReplAgtCtlAct-LoadInterval	18	INTEGER	Server to agent load check interval (in blocks) Format >>>>>>>>>9
_ReplAgtCtlAct-LoadChkInt	19	DECIMAL	Time between server and agent load checks (in seconds) Format >>>>>>>>>9.999

Field name	Field number	Data type	Description
_ReplAgtCtlAct-LoadChkResp	20	DECIMAL	Time taken to respond to load checks (in seconds) Format >>>>>>>>>9.999
_ReplAgtCtlAct-LastMsgRecv	21	CHARACTER	Time the last message was received from the remote agent Format X(24)

_Repl-AgentAIStreaming VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication VST, _Repl-AgentAIStreaming.

Table 31: Virtual system table _Repl-AgentAIStreaming field descriptions

Field name	Field number	Data type	Description
_ReplAgtAIStr-AgentID	2	INT64	Agent number The ID of the OpenEdge Replication agent Format ->>>>>>>>>>9
_ReplAgtAIStr-AgentName	3	CHARACTER	Agent Name Name of the OpenEdge Replication agent as configured by the name property in the [agent] section of the repl.properties file Format X(32)
_ReplAgtAIStr-TotalCacheEntries	4	INT64	Total number of entries in the Replication Streaming Block cache in units of Replication Stream Blocks (RSB) The maximum size of the cache is specified by the rsb-cache-size property in the [agent] section of the repl.properties file Format ->>>>>>>>>>9

Field name	Field number	Data type	Description
_ReplAgtAIStr-FreeCacheEntries	5	INT64	Free cache entries in the Replication Streaming Block cache in units of RSBs The number of RSB cache entries currently not in use Format ->>>>>>>>>>9
_ReplAgtAIStr-UsedCacheEntries	6	INT64	Used cache entries in the Replication Streaming Block cache in units of RSBs The number of RSB cache entries currently in use Format ->>>>>>>>>>9
_ReplAgtAIStr-CacheHWM	7	INT64	AI Stream Cache high water mark The maximum number of RSB cache entries that have been in use at one point in the life cycle of the cache Format ->>>>>>>>>>9
_ReplAgtAIStr-TimesCacheFilled	8	INT64	Number of times the Replication Streaming Block cache has filled Format ->>>>>>>>>>9

_DbServiceManager VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication VST, _DbServiceManager.

Table 32: Virtual system table _DbServiceManager field description

Field name	Field number	Data type	Description
_DbSvcMgr-Id	2	INT64	Id is ROWID of the _DBServiceManager record Format ->>>>>>>>>>>>>>>>>>9
_DbSvcMgr-CommunicationAreaSize	3	DECIMAL	Size of the pica queue in kilobytes Format >>>>>>>>9.999
_DbSvcMgr-TotalMsgEntries	4	INT64	Total number of messages available in the pica queue Format ->>>>>>>>>>>>>>>>>>9

Field name	Field number	Data type	Description
_DbSvcMgr-FreeMsgEntries	5	INT64	Number messages on the free list in the pica queue Format ->>>>>>>>>>>>>>>>>>9
_DbSvcMgr-UsedMsgEntries	6	INT64	Total number of messages on all queues Format ->>>>>>>>>>>>>>>>>>9
_DbSvcMgr-MessageHWM	7	INT64	High water mark of used messages Format ->>>>>>>>>>>>>>>>>>9
_DbSvcMgr-PicaFilled	8	INT64	Number of times the pica queue filled Format ->>>>>>>>>>>>>>>>>>9
_DbSvcMgr-LatchHolder	9	INTEGER	User number of last pica latch holder Format ->>>>>>>>>>9
_DbSvcMgr-AccessCount	10	INT64	Number of times locks were taken out Format ->>>>>>>>>>>>>>>>>>9
_DbSvcMgr-AccessCollisions	11	INT64	Number of times collisions occur when locks are held Format ->>>>>>>>>>>>>>>>>>9
_DbSvcMgr-LockPID	12	INTEGER	PID of the pica queue lock holder (if the pica queue locked) 0 if no lock Format >>>>>>>>9
_DbSvcMgr-LockUserType	13	CHARACTER	User type of the pica queue lock holder (if the pica queue locked) "Not available" if the queue is not locked Format X(32)
_DbSvcMgr-LockUserNumber	14	INT64	User number of the pica queue lock holder (if pica queue locked) 0 if no lock Format ->>>>>>>>>>>>>>>>>>9

_DbServiceManagerObjects VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication VST, `_DbServiceManagerObjects`.

Table 33: Virtual system table _DbServiceManagerObjects field description

Field name	Field number	Data type	Description
`_DbSvcMgrObj-Id`	2	INT64	Id is ROWID of the _DBSvcMgrObj record Format ->>>>>>>>>>>>>>>>>>9
`_DbSvcMgrObj-Name`	3	CHARACTER	Service object name Format X(32)
`_DbSvcMgrObj-Ready`	4	CHARACTER	Accept message status. Valid values: • Ready • Not Ready Format X(16)
`_DbSvcMgrObj-Status`	5	CHARACTER	Service object run status. Valid values: • Active • Inactive • Pre-resigstered • Registered • Queue Ready • Running • Busy • Busy Not Processing • DB Change Wait • Unknown Format X(32)
`_DbSvcMgrObj-Msgs`	6	INT64	Number of messages on the service object queue Format ->>>>>>>>>>>>>>>>>>9

Field name	Field number	Data type	Description
_DbSvcMgrObj-LockPID	7	INTEGER	PID of the lock holder (if the service object is locked) Format >>>>>>>>9
_DbSvcMgrObj-LockUserType	8	CHARACTER	User type of the lock holder (if the service object is locked) Format X(32)
_DbSvcMgrObj-LockUserNumber	9	INT64	User number of the lock holder (if the service object is locked) Format ->>>>>>>>>>>>>>>>>>9

_Repl-InterAgentActivity VST field descriptions

The table below contains the field descriptions of the OpenEdge Replication VST, _Repl-AgentControlActivity.

Table 34: Virtual system table _Repl-InterAgentActivity field description

Field name	Field number	Data type	Description
_ReplIntAgtAct-AgentID	2	INT64	Agent number Format ->>>>>>>>>>>>>>>>>>9
_ReplIntAgtAct-AgentName	3	CHARACTER	Agent name Format X(32)
_ReplIntAgtAct-Host	4	CHARACTER	Name of host where the remote agent is running Format X(128)
_ReplIntAgtAct-Port	5	INTEGER	Port the remote agent is listening on Format >>>>>>>>>>>>>>>>>>9
_ReplIntAgtAct-DBName	6	CHARACTER	Database name of the target database Format X(128)

Field name	Field number	Data type	Description
_ReplIntAgtAct-ConnectionState	7	CHARACTER	Connection state. Valid values include: • Connected • Disconnected • Listening Format X(16)
_ReplIntAgtAct-LastMsgSent	8	CHARACTER	Time last message sent to remote agent Format X(24)
_ReplIntAgtAct-LastMsgRecv	9	CHARACTER	Time last message received from remote agent Format X(24)
_ReplIntAgtAct-LastPingSent	10	CHARACTER	Time last ping message sent Format X(24)
_ReplIntAgtAct-LastPingRecv	11	CHARACTER	Time last ping message received Format X(24)
_ReplIntAgtAct-SentMsgs	12	INT64	Total messages sent to remote agent Format ->>>>>>>>>>>>>>>>>>9
_ReplIntAgtAct-RecvMsgs	13	INT64	Total messages received from remote agent Format ->>>>>>>>>>>>>>>>>>9
_ReplIntAgtAct-SyncPoints	14	INT64	Synchronization point notes received from remote agent Format ->>>>>>>>>>>>>>>>>>9
_ReplIntAgtAct-RemoteLastBlkSeq	15	INTEGER	Last acknowledged source AI sequence from remote agent Format ->>>>>>>>>>9
_ReplIntAgtAct-RemoteSourceTRID	16	INT64	Last acknowledged source transaction ID from remote agent Format ->>>>>>>>>>>>>>>>>>9

Modifiable VSTs on Replication Targets

To ease Virtual System Table (VST) updating on replication targets, the following modifiable VST tables and fields are allowed on replication targets.

Database connection (_Connect)

Field name	Data type	Description
_Connect-Misc	CHARACTER	User updatable field.

User connection (_MyConnection)

Table 35:

Field name	Data type	Description
_MyConn-UserMisc	CHARACTER	User updatable field.

Database features (_Database-Feature)

Field Name	Data Type	Description
_DBFeature_Enabled	CHARACTER	"1" indicates the feature is enabled; "0" indicates the feature is not enabled

Note: The only feature that may be enabled on a replication target is Large Database Entry Keys. It may only be enabled.

Database startup parameters file status (_DbParams)

Field name	Data type	Description
_DbParams-Value	CHARACTER	The current value specified for the parameter

Note: The following list contains the Database Startup Parameters that may have their value fields modified on a replication target. Other Startup Parameters are fixed and may not be modified.

- Base Index (-baseindex)
- Base LOB (-baselob)
- Base Table (-basetable)
- Base User Index (-baseuserindex)
- Base User LOB (-baseuserlob)
- Base User Table (-baseusertable)
- BI Writer Scan Delay (-bwdelay)
- Diagnostic Directory (-diagDir)
- Diagnostic Events Value (-diagEvent)
- Diagnostic Events Level (-diagEvtLevel)
- Diagnostic Field Separator (-diagFS)
- Diagnostic Data Format (-diagFormat)
- Diagnostic Pause Length (-diagPause)
- Diagnostic Prefix Value (-diagPrefix)
- Log File Archive Directory (-lgArchiveDir)
- Log File Archive Enable (-lgArchiveEnable)
- Log File Truncate Frequency (-lgTruncateFrequency)
- Log File Truncate Size (-lgTruncateSize)
- Log File Truncate Time (-lgTruncateTime)
- Limit Log File Payload (-limitLgPayload)
- LRU Force Skips (-lruskips)
- LRU Alternate Buffer Pool Force Skips (-lru2skips)
- Nap Time (-nap)
- Nap Maximum (-napmax)
- Login Governor (-nGovernor)
- Number of Checkpoint Statistics (-numcheckpointstats)
- Omit Log Messages (-omitLgMsgs)
- PICA Queue Nap Time (-picanap)
- Prefetch Delay (-prefetchDelay)
- Prefetch Factor (-prefetchFactor)
- Prefetch Num Recs (-prefetchNumRecs)
- Prefetch Priority (-prefetchPriority)
- Page Writer Queue Delay (-pwqdelay)

- Page Writer Queue Minimum (`-pwqmin`)
- Page Writer Scan Delay (`-pwsdelay`)
- Page Writer Scan (`-pwscan`)
- Page Writer Maximum Buffers (`-pwwmax`)
- Record Free Chain Search Depth Factor (`-recspacesearchdepth`)
- Spin Lock Retries (`-spin`)
- Server-side JOIN (`-ssj`)
- Database Client Notification (`-usernotifytime`)
- Database Consistency Check (`-DbCheck`)
- Lock Governor (`-LGovernor`)
- Memory Consistency Check (`-MemCheck`)
- Message Wait (`-Nmsgwait`)
- Autonomous Update Statistics (`-SQLAutoStats`)
- Transaction End (TXE) Lock Retry Limit (`-TXERetryLimit`)
- Transaction End (TXE) Commit Lock Skip Limit (`-TXESkipLimit`)

Utilities and OpenEdge Replication

Many of the OpenEdge database utilities are used to change a database. The table that follows lists the utilities and indicates whether they are allowed to operate on a source or target database.

Table 36: Utility support for OpenEdge Replication

Utility	Qualifier	Source database	Target database
DBTOOL	—	Yes	No
PROBKUP	Online	Yes	Yes
	Incremental	Yes	Yes
	Normal	Yes	Yes
PROCOPY	—	Yes	Yes
PRODB	—	Yes	Yes
PRODEL	—	Yes	Yes
PROLOG	—	Yes	Yes
PROMON	—	Yes	Yes
PROREST	—	No	No
PROSTRCT	Add	Yes	Yes
	Addonline	Yes	Yes
	Builddb	Yes	Yes
	Cluster Clear	Yes	Yes
	Create	NA	NA
	List	Yes	Yes
	Remove	Yes	No
	Repair	Yes	Yes
	Statistics	Yes	Yes
	Unlock	No	No
PROUTIL	2phase begin	No	No
	2phase commit	No	No
	2phase end	No	No
	2phase modify	No	No

Utility	Qualifier	Source database	Target database
	2phase recover	No	No
	auditarchive	Yes	No
	auditload	Yes	No
	auditreconfig	Yes	No
	bigrow	Yes	No
	busy	Yes	Yes
	bulkload	No	No
	chanalys	Yes	Yes
	codepage-compiler	Yes	Yes
	conv1112	NA	NA
	convchar	Yes	No
	convfile	No	No
	dbanalys online	Yes	Yes
	dbanalys offline	Yes	Yes
	datamove	Yes	No
	dbauthkey	No	No
	dbipcs	Yes	Yes
	dbrestrict	Yes	No
	describe	Yes	Yes
	deallocate	Yes	No
	disableauditing	No	No
	disableB2	Yes	No
	disableencryption	Yes	No
	disableJTA	Yes	No
	disablekeyevents	Yes	No
	disablemultitenancy	Yes	No

Utility	Qualifier	Source database	Target database
	disablereadonlypartitions	Yes	No
	disabletablepartitioning	Yes	No
	disptosscreatelimits	Yes	Yes
	dump	Yes	Yes
	dumpspecified online	Yes	Yes
	dumpspecified offline	Yes	Yes
	enableauditing	No	No
	enableB2	Yes	No
	enableencryption	Yes	No
	enableJTA	Yes	No
	enablekeyevents	Yes	No
	enablelargekeys	Yes	No
	enablemultitenancy	Yes	No (enabled automatically on an online source or as part of a restored backup)
	enablereadonlypartitions	Yes	No
	enablesitereplication	Yes	Yes
	enablestoredproc	No	No
	enabletablepartitioning	Yes	No
	epolicy	Yes	Yes
	disablesitereplication		Yes
	holder	Yes	Yes
	idxactivate	Yes	No
	idxanalys online	Yes	Yes
	idxanalys offline	Yes	Yes
	idxbuild	No	No
	idxcheck	Yes	Yes

Utility	Qualifier	Source database	Target database
	idxcompact online	Yes	No
	idxcompact offline	No	No
	idxdeactivate	Yes	No
	idxfix online	Yes	No
	idxfix offline	No	No
	idxmove online	Yes	No
	idxmove offline	No	No
	increaseto	Yes	No
	iostats online	Yes	Yes
	iostats offline	Yes	Yes
	load	No	No
	mtidxbuild	Yes	No
	mvsch	No	No
	partitionmanage alter	Yes	No
	partitionmanage merge	Yes	No
	partitionmanage rename	Yes	No
	partitionmanage split	Yes	No
	partitionmanage truncate	Yes	No
	partitionmanage view	Yes	Yes
	rcodekey	No	No
	set[area|lob|table]createlimit	Yes	No
	set[area|lob|table]tosslimit	Yes	No
	tabanalys online	Yes	Yes
	tabanalys offline	Yes	Yes
	tablemove online	Yes	No
	tablemove offline	No	No

Utility	Qualifier	Source database	Target database
	tpidxbuild	Yes	No
	truncate area	No	No
	truncate bi	Yes	No
	updateschema	No	No
	updatevst	No	No
	viewB2	Yes	No
	wordbreak-compiler	NA	No
	word-rules	No	No
RFUTIL	aimage begin	Yes	Yes
	aimage end	No	Yes
	aimage extent empty	Yes	Yes
	aimage extent full	Yes	Yes
	aimage extent list	Yes	Yes
	aimage extract		
	aimage new	Yes	Yes
	aimage aioff	No	Yes
	aimage query	Yes	Yes
	aimage scan	Yes	Yes
	aimage truncate	No	No
	aiarchiver disable	Yes	No
	aiarchiver enable	Yes	No
	aiarchiver end	Yes	No
	aiarchiver setdir	Yes	No
	aiarchiver setinterval	Yes	No
	aiarchive extent	Yes	No
	aiverify full	Yes	No

Utility	Qualifier	Source database	Target database
	aiverify partial	Yes	No
	mark backup	Yes	Yes
	roll forward	No	No
	roll forward oplock	No	No
	roll forward retry	No	No
	roll opunlock	No	No
	sequence	No	No
SQLDUMP	—	Yes	Yes
SQLLOAD	—	Yes	No
SQLSCHEMA	—	Yes	Yes

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 OpenEdge Management 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 Management or OpenEdge Explorer, 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*.

DSRUTIL logginglevel qualifier

Allows you to set or change the logging level OpenEdge Replication is running for both the Replication server and agents.

Syntax

```
dsrutil db-name -C logginglevel new-logging-level
```

Parameters

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.

new-logging-level

The new logging level to set. It must be an integer between 0 and 3 as described in the following table.

Logging Level	Description
0 (None)	Logs no entries. This is equivalent to turning logging off.
1 (Errors)	Logs OpenEdge Replication error messages. This is the minimum amount of logging.
2 (Basic)	Logs "basic" information such as the start and end of transactions and connections as well as summary information. Errors logged at Level 1 are also logged. This is the suggested logging level.
3 (Verbose)	Log all messages and details, including the information logged at Level 1 and 2.

The new logging level remains in effect until you change it again online, or until Replication restarts. If the Replication server or agent is restarted, the logging-level value set in the `repl.properties` file is used.

8

OpenEdge Replication Quick Command Summary

This chapter provides command summary information.

For details, see the following topics:

- Configuring the OpenEdge Replication property files
- Setting up the source and target databases
- Enabling before-image encryption for a Replication-enabled target database
- 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
- Backing up an online target database

Configuring the OpenEdge Replication property 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 Configuring the OpenEdge Replication property files on page 62.

Setting up the source and target databases

This section provides the steps for setting 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 { file 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-backup-name
```

   d) Begin AI, as shown:

```
rfutil 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}
```

Setting up the source database with deferred agent startup for online backup

Use the following steps 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-backup-name
```

   d) Begin AI, as shown:

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

3. Enable Replication, as shown:

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

4. Back up the database, as shown:

```
probkup online source-db-name source-db-backup-name -REPLTargetCreation
```

5. 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
```

6. Restart the source database, as shown (note that `-DBService` is case-sensitive):

```
PROSERVE -db source-db-name -DBService replserv
-S { port | service name }
```

Set up the target database

Use the following procedure to set up the target database:

1. Move or copy 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 database directory on the target machine, and rename it with the name of the target database (*target-db-name*.ks).
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-namedevice-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 Overview on page 53.

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.

Use the steps that follows 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:

```
DSRUTIL db-name -C DisableSiteReplication source-db-name
```

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

```
DSRUTIL db-name -C DisableSiteReplication target-db-name
```

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-backup-name
```

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:

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

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

```
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.

Use the following procedure to configure OpenEdge Replication with deferred agent startup:

1. Copy the source properties file from `OpenEdge-install-dir/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:

```
[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 Server properties on page 64

Configuring OpenEdge Replication for one agent

You configure OpenEdge Replication for one agent by copying and editing the sample properties file.

Use the following procedure to configure OpenEdge Replication for one agent:

1. Copy the source properties file from *OpenEdge-install-dir*`/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.

Use the following procedure to configure OpenEdge Replication for two agents:

1. Copy the source properties file from *OpenEdge-install-dir*/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, agemt2-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 Starting the source database on page 78 and Starting the target database on page 78.

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 Shutting down the source database on page 84 and Shutting down the target database on page 85.

Terminating the OpenEdge Replication server and agent

You can use the DSRUTIL utility to terminate an OpenEdge Replication server and agent.

Use the following procedure 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 Terminating the OpenEdge Replication server when the source database is running on page 85 and Terminating the OpenEdge Replication agent when the target database is running on page 86.

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 Setting up the OpenEdge Replication target database on page 59.

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 Setting up the OpenEdge Replication target database on page 59.

Using manual transition

Use the DSRUTIL utility anytime after OpenEdge Replication is running to transition a target database to a normal database.

To manually transition a target database, enter the following command:

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

Re-enabling OpenEdge Replication after transition

Once the target database has transitioned to be the new source database, you can re-enable OpenEdge Replication.

To re-enable OpenEdge Replication after transition:

1. In the target database directory, enter the following commands:

```
proshut target-db-name -by
dbutil probkup target-db-name target-db-backup-name
```

2. In the source database directory, enter the following commands:

```
proutil source-db-name -C disablesitereplication source
prorest source-db-name target-db-backup-name
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
```

Restarting the OpenEdge Replication server after target shutdown

Restart the OpenEdge Replication server with the `dsrutil restart 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 -C 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
```

Backing up an online 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.

Note: The Replication agent must be in a "normal processing" state for the online backup to execute. Verify that your agent is in a "normal processing state using one of the following:

- Query the `_ReplAgent` VST. See Virtual system tables for OpenEdge Replication on page 181.
- Query the agent status with DSRUTIL STATUS. See DSRUTIL status qualifier on page 155.
- Check the agent status with DSRUTIL MONITOR. See DSRUTIL monitor qualifier on page 152.

When an online backup of the target database is underway, activity continues on the source database as long as 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, the Replication server will indicate to the database 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 the next step.
   - 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 the previous step), 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.

Index

A

Administration *53*
After-Imaging
  enabling *56*
  extent blocks *31*
  extent sizing on source database *41*
  extent states *39*
  extent types *39*
  extents *38*
  fixed-length extents *39*
  switching extents *31*
  variable-length extents *39*
Agent
  configuration *74*, *164*
  connection timeout *69*, *169*
  critical *67*
  host *67*
  latency report *163*
  listener-maxport *69*
  listener-minport *69*
  polling *64*, *69*
  port *67*
  properties *69*
  rules *73*
  single agent configuration *225*
  status *162*, *173*
  terminating *86*, *146*, *159*, *228*
  two agent configuration *226*
ApplyExtent command action *146–147*
Architecture *21*
Asynchronous
  operation *30*
Automatic transition *95*, *164*, *228*

C

Checkpoint *31*
Commands *34*
Configuring
  agent *74*, *164*, *225–226*
  errors *78*
  OpenEdge Replication *62*
connect-timeout property *67*, *69*
Connection *164*
Control agent properties *64*, *67*
Creating the target database *59*
Critical agent *27*
critical property *67*

D

Data replication *19*
Database
  access *81*
  connection considerations *80*
  enabling for transparent data encryption and replication *49*
  Enhanced Read-only mode
    defined *28*
  schema locks *24*
database property *64*, *67*, *69*
database.repl.recovery *31*
database.repl.recovery file *92*
-DBService argument *78*
_DbSvcMgr-TotalMsgEntries *202*
_DbSvcMgr-FreeMsgEntries *202*
_DbSvcMgr-LockUserType *202*
_DbSvcMgr-UsedMsgEntries *202*
_DbSvcMgr-MessageHWM *202*
_DbSvcMgr-LatchHolder *202*
_DbSvcMgr-AccessCount *202*
_DbSvcMgr-Id *202*
_DbSvcMgr-LockPID *202*
_DbSvcMgr-AccessCollisions *202*
_DbSvcMgr-PicaFilled *202*
_DbSvcMgr-LockUserNumber *202*
_DbSvcMgr-CommunicationAreaSize *202*
_DbSvcMgrObj-LockPID *204*
_DbSvcMgrObj-LockUserNumber *204*
_DbSvcMgrObj-Status *204*
_DbSvcMgrObj-Msgs *204*
_DbSvcMgrObj-Ready *204*
_DbSvcMgrObj-LockUserType *204*
_DbSvcMgrObj-Id *204*
_DbSvcMgrObj-Name *204*
Default implementation *54*
defer-agent-startup property *59*, *64*, *75*, *79*
Deferred agent startup *59*, *75*, *224*
disablesitereplication argument *92*, *146*, *152*
DSRUTIL utility
  agent option *146*
  and passphrase *146*
  and passphrase in transition *160–161*
  and transparent data encryption *146*
  MONITOR
    latency details *163*
  server option *146*
  startup menu *162*

E

Enhanced Read-only mode (ERO)
  defined *28*
Estimating network bandwidth *44*

F

Failover
  defined *50*
Failure recovery
  performing *164*
Fixed-length after-image extents *39*
Functional definitions *21*

H

Hot standby
  defined *32*

I

Implementation
  default *75*
  deferred agent startup *75*
Incremental backup *57*
Initialization process *79*
Inter-agent status *179*
ipver property *67*, *69*

J

JTA transactions *94*

L

Latency
  details provided by DSRUTIL MONITOR *163*
Limitations and restrictions *51*
listener-maxport property *69*
listener-minport property *69*
logginglevel argument *216*

M

Manual transition *96*, *164*, *229*
maximum-polling-delay property *64*, *69*
Message logging *78*
minimum-polling-delay property *64*, *69*

N

name property *67*, *69*
Normal
  database activity *30*
Normal *(continued)*
  processing *164*
  target activity *31*

O

Offline backup *55*
Online backup of source database *57*, *59*, *220*
Online backup of target database *231*
OpenEdge Replication
  administration *53*
  agent *27*
  agent properties *69*
  automatic transition *228*
  commands *34*
  configuring *62*
  control agent properties *67*
  critical agent *27*
  database monitoring *231*
  defined *19*
  disabling *92*
  failover *50*
  feature *21*
  functional definitions *21*
  implementation *74*
  limitations and restrictions *51*
  manual transition *229*
  model *29*
  monitoring *231*
  network bandwidth estimation *44*
  on encrypted databases *48*, *82*, *146*
  overhead *41*, *44*
  planning *47*
  re-enabling after transition *229*
  server *26*
  starting *80*, *227*
  startup *77*
  stopping *84*, *227*
  utilities *34*, *210*
  with transparent data encryption *48*

P

Passphrase (transparent data encryption) *49*, *82*, *146*, *160–161*
Performing initialization status *164*
port property *67*
Pre-transition state *92*
Primary database
  defined *22*
PROBKUP *210*, *231*
PROCOPY *210*
PRODB *210*
PRODEL *210*
PROLOG *210*
PROMON *210*

Properties
agent 69
control agent 67
server 64
transition 124
Property files
sample source 63
sample target 63
source 62
target 63
PROREST 92, 210
PROSERVE 78
PROSHUT 84–85
PROSTRCT 210
PROUTIL 92, 210

R

_Repl-Server VST 181
Re-enabling OpenEdge Replication after transition 229
Recovery
lost connection 91
RECOVERY command action 153
Remote agent status 168–169
Repl-Keep-Alive property 64
repl.server.startup.lg 78
_ReplAgt-CommStatus 186
_ReplAgt-MaxDelayInterval 186
_ReplAgt-AIFile 186
_ReplAgt-AutoBeginAI 186
_ReplAgt-Status 186
_ReplAgt-BlocksReceived 186
_ReplAgt-ConnectTimeout 186
_ReplAgt-ServerHost 186
_ReplAgt-TimeoutLimit 186
_ReplAgt-MinDelayInterval 186
_ReplAgt-MinPort 186
_ReplAgt-Port 186
_ReplAgt-BlocksACK 186
_ReplAgt-BlocksProcessed 186
_ReplAgt-DBName 186
_ReplAgt-LastTRID 186
_ReplAgt-BackupMethod 186
_ReplAgt-MaxMsgSize 186
_ReplAgt-ConnectTime 186
_ReplAgt-DatabaseRole 186
_ReplAgt-TransitionType 186
_ReplAgt-AgentName 186
_ReplAgt-TransitionToAgents 186
_ReplAgt-Method 186
_ReplAgt-AgentID 186
_ReplAgt-RestartAfterTrans 186
_ReplAgt-NotesProcessed 186
_ReplAgt-MaxPort 186
_ReplAgt-ReplicationSet 186
_ReplAgt-AutoAddAI 186
_ReplAgt-TransDatabaseRole 186
_ReplAgt-Critical 186
_ ReplAgtAct-BlocksACK 191
_ ReplAgtAct-LastBlockReceived 191
_ ReplAgtAct-BlocksProcessed 191
_ ReplAgtAct-LastProcBlock 191
_ ReplAgtAct-Status 191
_ ReplAgtAct-CommStatus 191
_ ReplAgtAct-TimeToTransition 191
_ ReplAgtAct-SrcRDBMSSeq 191
_ ReplAgtAct-CurDelayInterval 191
_ ReplAgtAct-LastSyncTimeTgt 191
_ ReplAgtAct-LastProcSeq 191
_ ReplAgtAct-AgentName 191
_ ReplAgtAct-TransEnded 191
_ ReplAgtAct-ServerLag 191
_ ReplAgtAct-State 191
_ ReplAgtAct-AgentID 191
_ ReplAgtAct-SyncPoints 191
_ ReplAgtAct-NotesProcessed 191
_ ReplAgtAct-SourceTRID 191
_ ReplAgtAct-ConnectTime 191
_ ReplAgtAct-TransStarted 191
_ ReplAgtAct-BlocksReceived 191
_ ReplAgtAct-LastTRID 191
_ ReplAgtAct-SrcRDBMSBlock 191
_ ReplAgtAct-WaitingFor 191
_ ReplAgtAct-LastSyncTimeSrc 191
_ReplAgtAIStr-AgentID 201
_ReplAgtAIStr-AgentName 201
_ReplAgtAIStr-CacheHWM 201
_ReplAgtAIStr-FreeCacheEntries 201
_ReplAgtAIStr-TimesCacheFilled 201
_ReplAgtAIStr-TotalCacheEntries 201
_ReplAgtAIStr-UsedCacheEntries 201
_ ReplAgtCtl-ConnectTime 195
_ ReplAgtCtl-Status 195
_ ReplAgtCtl-Port 195
_ ReplAgtCtl-BlocksSent 195
_ ReplAgtCtl-LastBlockSentAt 195
_ ReplAgtCtl-AgentID 195
_ ReplAgtCtl-CommStatus 195
_ ReplAgtCtl-RemoteDBName 195
_ ReplAgtCtl-Method 195
_ ReplAgtCtl-RemoteHost 195
_ ReplAgtCtl-TransitionType 195
_ ReplAgtCtl-AgentName 195
_ ReplAgtCtl-TimeoutLimit 195
_ ReplAgtCtl-ConnectionTimeout 195
_ ReplAgtCtl-BlocksACK 195
_ ReplAgtCtl-MaxMsgSize 195
_ ReplAgtCtl-Critical 195
_ ReplAgtCtlAct-AgentID 198
_ ReplAgtCtlAct-LastBlockSentAt 198
_ ReplAgtCtlAct-WaitingFor 198
_ ReplAgtCtlAct-AgentName 198

_ ReplAgtCtlAct-SyncPoints *198*
_ ReplAgtCtlAct-LoadInterval *198*
_ ReplAgtCtlAct-CurRDBMSBlock *198*
_ ReplAgtCtlAct-BlocksSent *198*
_ ReplAgtCtlAct-CurRDBMSSeq *198*
_ ReplAgtCtlAct-LoadChkResp *198*
_ReplAgtCtlAct-LastMsgRecv *198*
_ ReplAgtCtlAct-ConnectTime *198*
_ ReplAgtCtlAct-State *198*
_ ReplAgtCtlAct-LoadChkInt *198*
_ ReplAgtCtlAct-Status *198*
_ ReplAgtCtlAct-CommStatus *198*
_ ReplAgtCtlAct-BlocksACK *198*
_ ReplAgtCtlAct-RecoveryState *198*
_ ReplAgtCtlAct-LastBlockSeq *198*
_ ReplAgtCtlAct-LastBlockBlock *198*
Replication agent, See Agent
Replication overhead *41*
Replication server, See Server
Replication Set transition *120*
replication-method property *67*
_ReplIntAgtAct-LatPingRecv *205*
_ReplIntAgtAct-Host *205*
_ReplIntAgtAct-SentMsgs *205*
_ ReplIntAgtAct-AgentID *205*
_ ReplIntAgtAct-AgentName *205*
_ReplIntAgtAct-DBName *205*
_ReplIntAgtAct-SyncPoints *205*
_ReplIntAgtAct-LastMsgRecv *205*
_ReplIntAgtAct-LastPingSent *205*
_ReplIntAgtAct-Port *205*
_ReplIntAgtAct-RemoteLastBlkSeq *205*
_ReplIntAgtAct-RemoteSourceTRID *205*
_ReplIntAgtAct-RecvMsgs *205*
_ReplIntAgtAct-ConnectionState *205*
_ReplIntAgtAct-LastMsgSent *205*
_ ReplSrv-DatabaseRole *182*
_ReplSrv-BackupMethod *182*
_ ReplSrv-StartTime *182*
_ ReplSrv-DeferAgentConnUntil *182*
_ReplSrv-AutoBeginAI *182*
_ ReplSrv-RecAgentsToRecover *182*
_ ReplSrv-AgentsConnected *182*
_ ReplSrv-RecAgentsConnected *182*
_ReplSrv-TransDatabaseRole *182*
_ReplSrv-ConnectAttempts *182*
_ReplSrv-RestartAfterTrans *182*
_ReplSrv-SchemaLockAction *182*
_ReplSrv-ReplicationSet *182*
_ReplSrv-AIFile *182*
_ ReplSrv-CurDelayInterval *182*
_ ReplSrv-BlocksSent *182*
_ ReplSrv-ServerState *182*
_ ReplSrv-MaxDelayInterval *182*
_ ReplSrv-RecoveryState *182*
_ReplSrv-ReplKeepAlive *182*
_ ReplSrv-AgentsInSync *182*
_ ReplSrv-TransitionTimeout *182*
_ ReplSrv-MinDelayInterval *182*
_ReplSrv-AgentShutdownAction *182*
_ReplSrv-TransitionToAgents *182*
_ ReplSrv-AgentCount *182*
_ReplSrv-AutoAddAI *182*
_ ReplSrv-TransitionType *182*
_ReplSrv-DeferAgentExpire *182*
_ ReplSrv-Database *182*
_ ReplSrv-LastBlockSentAt *182*
_ ReplSrv-Id *182*
_ReplSrv-ConfiguredAgents *182*
_ReplSrv-NextConnectIn *182*
RESTART command action
  agent *154*
  server *154*
Restoring
  databases *92*
  source database *92*
RFUTIL *210*
Roll forward *32*

S

Schema lock
  request *169*
Schema-Lock-Action property *64*
Secondary database
  defined *22*
Server
  inter- agent status *179*
  latency report *163*
  properties *64*
  recovery *153*
  remote agents details *168*
  remote agents status *169*
  restarting after target shutdown *230*
  status *162*, *164*
  terminating *85*, *228*
Setup
  source *55*, *57*, *220*
  target *59*, *220*
Shutdown
  source *84*
  target *85*
Single agent configuration *225*
Source
  access *82*
  combined properties file *63*
  database
    defined *22*
  enabling *56*, *59*
  incremental backup *57*
  machine *47*
  normal activity *30*

Source *(continued)*
offline backup *55*
online backup *57*, *59*, *220*
properties file *62*
PROSERVE startup *78*
requirements *30*
restoring *92*
setup *55*, *57*, *220*
shutdown *84*
startup *78*
structure files *55*, *57*
SQLDUMP *210*
SQLLOAD *210*
SQLSCHEMA *210*
startAgent command action *146*, *155*
Startup
message logging *78*
process *79*
source *78*
target *78*
Structure files *55*, *57*
Supported utilities *210*
Sync point
defined *31*
Synchronization
defined *33*
processing *164*

T

Target
access *81*
combined properties file *63*
create *59*
database
defined *22*
machine *47*
normal activity *31*
online backup *231*
properties file *63*
PROSERVE startup *78*
quiet points *24*
requirements *30*
restoring *92*
setup *59*, *220*
shutdown *85*
startup *78*
target.repl.properties *74*
Terminate command action *85*, *159*
TERMINATE command action *86*
Terminating
agent *86*, *228*
server *85*, *228*
Transaction end *31*
Transition
automatic *95*
configuration *94*
lost connection *92*
manual *96*
re-enabling OpenEdge Replication after *229*
timeout limit *164*
Transition command action *160–161*
transition property *64*
transition-timeout property *64*
Transparent data encryption
and DSRUTIL *146*
and replication requirements *48*
enabling for a replication-enabled source database *82*
support in replication *48*
TriggerTransition command action *146*, *161*

U

Unknown state *164*
Unsupported utilities *210*
Utilities *34*, *210*

V

Variable-length after-image extents *39*
Virtual System Table *207*
Virtual system tables *181*
VST *207*
VST field descriptions of:
_DbServiceManagerObjects *204*
_DbServiceManager *202*
_Repl-Agent *186*
_Repl-AgentAIStreaming *201*
_Repl-Server *182*
_Repl-AgentActivity *191*
_Repl-AgentControl *195*
_Repl-AgentControlActivity *198*
_Repl-IntAgentActivity *205*