Oracle® Procedural Gateway for APPC

Installation and Configuration Guide

Release 9.2.0.1.0 for UNIX

May 2002
Part No. A96648-01
# Contents

Send Us Your Comments ........................................................................................................................ xiii

Preface ........................................................................................................................................................ xv

Intended Audience ............................................................................................................................... xv
Related Documents .......................................................................................................................... xv
Conventions .......................................................................................................................................... xvi
Documentation Accessibility ........................................................................................................... xviii

1 Release Information

Product Set ............................................................................................................................................... 1-2
Changes and Enhancements Release 9.2.0.1.0 .................................................................................. 1-2
Changes ............................................................................................................................................... 1-2
Oracle Call Interface .......................................................................................................................... 1-2
APPC Log File .................................................................................................................................. 1-2
Enhancements .................................................................................................................................... 1-2
Password Encryption Utility for Gateway Initialization File .................................................... 1-3
AIX-Only: Password Encryption Utility for Resource Recovery Manager ................................ 1-3
Bugs Fixed in Version 9i ...................................................................................................................... 1-3
Known Restrictions ............................................................................................................................ 1-5

2 Introduction to Oracle Procedural Gateway for APPC

Overview .............................................................................................................................................. 2-2
Features .............................................................................................................................................. 2-2
Terms ................................................................................................................................................... 2-4
## Architecture

- Implementation ................................................................................................................. 2-9
- Starting the Gateway ........................................................................................................ 2-9
- Communication .................................................................................................................. 2-9
- Remote Procedural Call Functions ................................................................................... 2-10
  - Remote Transaction Initiation ....................................................................................... 2-10
  - Data Exchange .............................................................................................................. 2-10
  - Remote Transaction Termination .................................................................................. 2-11

## Remote Transaction Termination

### Transaction Types

### 3 System Requirements

#### Hardware Requirements

- Processor ................................................................................................................................. 3-2
- Memory ........................................................................................................................................ 3-2
- Network Attachment ............................................................................................................... 3-2
- CD-ROM Drive ......................................................................................................................... 3-3
- HP-UX Only .............................................................................................................................. 3-3
- Disk Space ................................................................................................................................. 3-3

#### Software Requirements

- Software Requirements for AIX-Based Systems ................................................................. 3-3

#### Documentation Requirements

### 4 Installing and Configuring the Gateway

#### Before You Begin

- Planning to Upgrade the Gateway ....................................................................................... 4-2
- Upgrade Considerations ......................................................................................................... 4-3
- Restoration ............................................................................................................................... 4-3

#### Performing Pre-upgrade Procedures

#### Performing Pre-Installation Procedures

#### Installing the Gateway Software

#### Installation Steps

- Step 1: Login as DBA and Create Login User ID ................................................................. 4-6
- Step 2: Create the Product Installation Directory ............................................................... 4-6
- Step 3: Set the ORACLE_HOME Environment Variable ....................................................... 4-7
- Step 4: Set the DISPLAY Environment Variable ................................................................. 4-7
Step 5: Mount the CD-ROM .......................................................... 4-8
  Mounting the CD-ROM for AIX-Based Systems .................. 4-8
  Mounting the CD-ROM for HP-UX ........................................ 4-8
  Mounting the CD-ROM for Solaris ........................................ 4-9
Step 6: Start the Oracle Universal Installer ............................. 4-9
  For AIX-Based Systems ......................................................... 4-9
  For HP-UX ........................................................................... 4-10
  For Solaris ........................................................................... 4-10
Step 7: Step through the Oracle Universal Installer ................. 4-10
  Oracle Universal Installer on AIX-Based Systems and HP-UX 4-11
  Oracle Universal Installer on Solaris ....................................... 4-12
Configuring Oracle Procedural Gateway for APPC .................. 4-13
  Preconfiguration Steps .......................................................... 4-13
    Oracle Server Coexistent with Gateway Installation/Upgrade 4-13
    Stand-alone Gateway Installation/Upgrade ......................... 4-14
  Configuring the Oracle Integrating Server for First-Time Installations 4-15
  Upgrading the Oracle Integrating Server from Previous Releases 4-17
  First-Time Installation and Configuration Steps .................... 4-19
  Optional Configuration Steps to Permit Multiple Users .......... 4-19
  Configuration ................................................................. 4-23
    Configuring the SNA Communication Packages .................. 4-23
    Configuring the Gateway ............................................... 4-23
  Configuring Your Network .................................................. 4-24
  Configuring Commit-Confirm .............................................. 4-26
  AIX-Based Systems Only: Two-Phase Commit .................... 4-26
  Configuring the OLTP .......................................................... 4-27
    Configuring CICS/ESA .................................................... 4-27
    Configuring IDMS-DC MVS .............................................. 4-28
    Configuring IMS/TM ...................................................... 4-30
    Configuring APPC/MVS ............................................... 4-31
Verifying the Installation ......................................................... 4-32
  Verifying the Gateway Installation ...................................... 4-32
  Verifying the OLTP Configuration ....................................... 4-33
    CICS Verification .......................................................... 4-33
    IDMS-DC Verification .................................................... 4-34
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS/TM Verification</td>
<td>4-35</td>
</tr>
<tr>
<td>APPC/MVS Verification</td>
<td>4-36</td>
</tr>
<tr>
<td>Performing Postinstallation Procedures</td>
<td>4-37</td>
</tr>
<tr>
<td>Installing Sample Applications</td>
<td>4-37</td>
</tr>
<tr>
<td>Accessing the Gateway from Other Oracle Servers</td>
<td>4-38</td>
</tr>
<tr>
<td>Accessing Other OLTPs from the Oracle Integrating Server</td>
<td>4-39</td>
</tr>
<tr>
<td>Deinstalling Your Oracle Procedural Gateway for APPC</td>
<td>4-39</td>
</tr>
</tbody>
</table>

5 Security Requirements

| Overview of Security Requirements                                     | 5-2  |
| Authenticated Application Logons                                       | 5-3  |
| Defining and Controlling Database Links                               | 5-4  |
| Link Accessibility                                                    | 5-4  |
| Links and CONNECT Clauses                                             | 5-4  |
| Passwords in the Gateway Initialization File                          | 5-5  |
| Using the pg4pwd Utility                                              | 5-5  |
| AIX-Based Systems Only: Using the pg4rrmpwd Utility                   | 5-6  |

6 Configuring the SNA Communication Package on AIX-Based Systems

| Using SNA Security Validation                                         | 6-2  |
| Specifying SNA Conversation Security                                  | 6-2  |
| SNA Security Option SECURITY=None                                     | 6-2  |
| SNA Security Option SECURITY=PROGRAM                                  | 6-2  |
| SNA Security Option SECURITY=SAME                                     | 6-3  |
| Processing Inbound Connections                                        | 6-4  |
| Independent Versus Dependent LUs                                     | 6-4  |
| Creating SNA Profiles for the Gateway                                | 6-5  |
| Profile Types                                                         | 6-5  |
| SNA Server Profiles                                                   | 6-5  |
| SNA Node Profile                                                      | 6-6  |
| Link Station Profile                                                  | 6-6  |
| Mode Profile                                                          | 6-6  |
| Local LU Profile                                                      | 6-7  |
| Partner LU Profile                                                    | 6-8  |
| Partner LU Location Profile                                           | 6-8  |
7 Configuring the SNA Communication Package on HP-UX

Using SNA Security Validation ................................................................. 7-2
Specifying SNA Conversation Security .................................................. 7-2
   SNA Security Option SECURITY=NONE .............................................. 7-2
   SNA Security Option SECURITY=PROGRAM ...................................... 7-2
   SNA Security Option SECURITY=SAME ............................................ 7-3
Processing Inbound Connections ............................................................. 7-4
   Steps for Configuring the Communications Interfaces ....................... 7-4
The SNAplus2 Configuration Tool .......................................................... 7-4
Creating SNAplus2 Profiles for the Gateway ......................................... 7-4
Independent Versus Dependent LUs ...................................................... 7-5
   Sym Dest Name .................................................................................. 7-6
Creating SNA definitions for the Gateway ............................................. 7-6
Sample SNAplus2 Definitions ............................................................... 7-6
Configuring SNAplus2 .......................................................................... 7-7
Testing the Connection ......................................................................... 7-10

8 Configuring the SNA Communication Package on Solaris

Using SNA Security Validation ................................................................. 8-2
Specifying SNA Conversation Security .................................................. 8-2
   SNA Security Option SECURITY=NONE .............................................. 8-2
   SNA Security Option SECURITY=PROGRAM ...................................... 8-2
   SNA Security Option SECURITY=SAME ............................................ 8-3
Processing Inbound Connections ............................................................. 8-4
Configuring SunLink SNA Peer-to-Peer Version 9.0 ............................... 8-4
   Overview of the Configuration Process .............................................. 8-4
   Independent Versus Dependent LUs .................................................. 8-5
   Setting up a SunLink P2P LU6.2 Server Name .................................... 8-5
   Setting up a Configuration File .......................................................... 8-6
      CP Directive .................................................................................. 8-7
   TRLINE or SDLCLINE Directives....................................................... 8-7
DLC Directive .................................................................................................................. 8-7
LU Directive .................................................................................................................. 8-7
PTNR_LU Directive ...................................................................................................... 8-8
MODE Directive .......................................................................................................... 8-8
Side Information Files .................................................................................................. 8-8
Starting the SunLink SNA PU2.1 Server ................................................................. 8-10
Test the Connection .................................................................................................. 8-10
Configuring SNAP-IX Version 6 .............................................................................. 8-11
Before You Begin .................................................................................................... 8-11
SNAP-IX Configuration Tool .................................................................................. 8-11
Creating SNAP-IX Profiles for the Gateway ......................................................... 8-11
Independent Versus Dependent LUs ..................................................................... 8-11
Creating SNA Definitions for the Gateway ............................................................. 8-12
Sample SNAP-IX Definitions ................................................................................ 8-12
Configuring SNAP-IX .............................................................................................. 8-13
Invoking xsnaadmin ................................................................................................ 8-13
Configuring the SNA node ...................................................................................... 8-13
Adding a Port ............................................................................................................. 8-13
Create a Link Station ................................................................................................ 8-14
Create Local LUs ....................................................................................................... 8-14
Create Partner LUs ................................................................................................... 8-14
Create Mode and CPI-C Profiles .......................................................................... 8-15
Testing the Connection .......................................................................................... 8-15

9 AIX-Based Systems Only: Implementing Two-Phase Commit

Introduction .................................................................................................................. 9-2
Supported OLTPs (Online Transaction Processors) ................................................. 9-2
Two-Phase Commit Architecture .............................................................................. 9-2
Required Components .............................................................................................. 9-3
Roles of the Components of Two-Phase Commit .................................................. 9-4
Interactions ................................................................................................................ 9-5
Two-Phase Commit Flow ........................................................................................ 9-6
Gateway Server and RRM LU6.2 Log Tables ......................................................... 9-7
PreConfiguration Tasks ......................................................................................... 9-10
Choose a Dedicated Local LU ............................................................................... 9-10
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Up RRM Execution Group</td>
<td>9-11</td>
</tr>
<tr>
<td>Configuring Two-Phase Commit</td>
<td>9-11</td>
</tr>
<tr>
<td>Configuring the Oracle Logging Server</td>
<td>9-12</td>
</tr>
<tr>
<td>Configuring the Gateway Initialization Parameters</td>
<td>9-12</td>
</tr>
<tr>
<td>Configuring the RRM Initialization Parameters</td>
<td>9-14</td>
</tr>
<tr>
<td>Configuring the SNA Server</td>
<td>9-15</td>
</tr>
<tr>
<td>Configuring CICS/ESA</td>
<td>9-17</td>
</tr>
<tr>
<td>Post-Configuration Tasks</td>
<td>9-18</td>
</tr>
<tr>
<td>Set Up RRM Startup and Shutdown Procedures</td>
<td>9-18</td>
</tr>
<tr>
<td>Customizing the rrmstart script</td>
<td>9-18</td>
</tr>
<tr>
<td>RRM Recovery</td>
<td>9-20</td>
</tr>
<tr>
<td>RRM issued message PGA-21215</td>
<td>9-21</td>
</tr>
<tr>
<td>RRM issued message PGA-21218</td>
<td>9-21</td>
</tr>
<tr>
<td>Two-Phase Commit With CICS/ESA</td>
<td>9-23</td>
</tr>
<tr>
<td>Application Design Requirements</td>
<td>9-24</td>
</tr>
<tr>
<td>Manual Recovery of In-Doubt Transactions</td>
<td>9-26</td>
</tr>
<tr>
<td>Logging Two-Phase Commit Activity from CICS</td>
<td>9-29</td>
</tr>
<tr>
<td>PGALUWID Subroutine</td>
<td>9-31</td>
</tr>
<tr>
<td>Periodic Log Cleanup</td>
<td>9-33</td>
</tr>
<tr>
<td>10 Migration and Coexistence with Existing Gateways</td>
<td></td>
</tr>
<tr>
<td>Migrating An Existing Gateway Instance to New Release</td>
<td>10-2</td>
</tr>
<tr>
<td>Step 1: Install the new Release</td>
<td>10-2</td>
</tr>
<tr>
<td>Step 2: Transferring <code>initsid.ora</code> Gateway Initialization File Parameters</td>
<td>10-2</td>
</tr>
<tr>
<td>Backout Considerations When Migrating to New Releases</td>
<td>10-2</td>
</tr>
<tr>
<td>Parameter Changes for Version 4 to Release 9i of the Gateways</td>
<td>10-2</td>
</tr>
<tr>
<td>Parameter Changes for Version 8 or Earlier to Oracle9.2.0 Gateway</td>
<td>10-5</td>
</tr>
<tr>
<td>Migrating from Oracle9.0.1 to Oracle9.2.0</td>
<td>10-6</td>
</tr>
<tr>
<td>Oracle Net Considerations</td>
<td>10-6</td>
</tr>
<tr>
<td>A Gateway Initialization Parameters</td>
<td></td>
</tr>
<tr>
<td>Gateway Initialization Parameter File</td>
<td>A-2</td>
</tr>
<tr>
<td>PGA Parameters</td>
<td>A-2</td>
</tr>
<tr>
<td>PGA Parameters on AIX</td>
<td>A-2</td>
</tr>
<tr>
<td>PGA Parameters on HP-UX and Solaris</td>
<td>A-6</td>
</tr>
</tbody>
</table>
Known Problems in Previous Releases

- Release 8.0.6.1.0
- Release 4.0.1.1.0

Index
Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us at the following E-mail address:

   infoibm_us@oracle.com

If you would like a reply, please include your name, address and telephone number.

If you have problems with the software, please contact your local Oracle Support Services.
Preface

Intended Audience

Read this guide if you are responsible for tasks such as:

- determining hardware and software requirements
- installing, configuring, or administering Oracle Procedural Gateway for APPC
- developing applications that access remote host databases through the gateway
- determining security requirements
- determining and resolving problems

Before using this guide to administer the gateway, you should understand the fundamentals of the operating system for your platform and procedural gateways.

Related Documents

The Oracle Procedural Gateway for APPC Installation and Configuration Guide for UNIX is included as part of your product shipment. Also included is:

- Oracle Procedural Gateway for APPC Messages Guide
- Oracle Procedural Gateway for APPC User’s Guide for UNIX

You might also need Oracle server and Oracle Net documentation. The following is a useful list of the Oracle publications that are referenced in this book:

- Oracle9i Server Installation Guide
- Oracle9i Server Administrator’s Guide
- Oracle9i Server Application Developer’s Guide
Other related publications are listed in "Documentation Requirements" on page 3-6. Refer to the Oracle Technical Publications Catalog and Price Guide for a complete list of documentation provided for Oracle products.

## Conventions

Examples of input and output for the gateway and Oracle environment are shown in a special font:

```
$ mkdir /ORACLE/your_name
```

All output is shown as it appears. For input, refer to the list of conventions and their meanings in the following table:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>example text</td>
<td>Words or phrases, such as <code>mkdir</code> and <code>ORACLE</code>, must be entered exactly as spelled and in the letter case shown. In this example, <code>mkdir</code> must be entered in lowercase letters and <code>ORACLE</code> in uppercase letters.</td>
</tr>
<tr>
<td>italic text</td>
<td>Italicized uppercase or lowercase, such as <code>your_name</code>, indicates that you must substitute a word or phrase, such as the actual directory name.</td>
</tr>
<tr>
<td><strong>BOLD text</strong> or <strong>bold italic TEXT</strong></td>
<td>Bold words or phrases refer to a file or directory structure, such as a directory, path, or file ID.</td>
</tr>
<tr>
<td>...</td>
<td>Ellipses indicate that the preceding item can be repeated. You can enter an arbitrary number of similar items.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Curly braces indicate that one of the enclosed arguments is required. Do not enter the braces themselves.</td>
</tr>
<tr>
<td></td>
<td>Vertical lines separate choices.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Square brackets enclose optional clauses from which you can choose one or none. Do not enter the brackets themselves.</td>
</tr>
</tbody>
</table>
Other punctuation, such as commas, quotation marks or the pipe symbol (|) must be entered as shown unless otherwise specified. Directory names, file IDs and so on appear in the required letter case in examples. The same convention is used when these names appear in text, and the names are highlighted in **bold**. The use of *italics* indicates that those portions of a file ID that appear in *italics* can vary.

Gateway commands, file IDs, reserved words, and keywords appear in uppercase in examples and text. UNIX commands, environment variables, and keywords appear in the required letter case in examples and text. Reserved words and keywords must always be entered as shown; they have reserved meanings within the Oracle system.

**SQL*Plus Prompts**
The SQL*Plus prompt, `SQL>`, appears in SQL statement and SQL*Plus command examples. Enter your response at the prompt. Do not enter the text of the prompt, `SQL>`, in your response.

**UNIX Prompts**
The UNIX prompt, `$`, appears in UNIX command examples. Enter your response at the prompt. Do not enter the text of the prompt, `$`, in your response. A dollar sign, `$`, is also part of some UNIX directory names and should not be mistaken for a prompt character.

**PGAU Prompts**
The PGAU prompt, `PGAU>`, appears in PGAU command examples. Enter your response at the prompt. Do not enter the text of the prompt, `PGAU>`, in your response.

**Directory Names**
Throughout this document, there are references to the directories in which product-related files reside. `$ORACLE_HOME` is used to represent the Oracle home directory. This is the default location for Oracle products. If you have installed into a location other than `$ORACLE_HOME`, replace all references to `$ORACLE_HOME` with the drive and path specification you have used.
Storage Measurements

Storage measurements use the following abbreviations:

- K, for kilobyte which equals 1024 bytes
- M, for megabyte which equals 1 048 576 bytes
- G, for gigabyte which equals 1 073 741 824 bytes

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at:

http://www.oracle.com/accessibility/

Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.
This chapter provides information about this release of Oracle Procedural Gateway for APPC. It contains the following sections:

- **Product Set** on page 1-2
- **Changes and Enhancements Release 9.2.0.1.0** on page 1-2
- **Known Restrictions** on page 1-5
Product Set

The following table lists the production components included on the distribution CD for Oracle Procedural Gateway for APPC:

<table>
<thead>
<tr>
<th>Component</th>
<th>Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Procedural Gateway for APPC</td>
<td>9.2.0.1.0</td>
</tr>
<tr>
<td>Procedural Gateway Administration Utility</td>
<td>9.2.0.1.0</td>
</tr>
<tr>
<td>AIX only: Resource Recovery Manager</td>
<td>9.2.0.1.0</td>
</tr>
<tr>
<td>Oracle Net</td>
<td>9.2.0.1.0</td>
</tr>
</tbody>
</table>

Changes and Enhancements Release 9.2.0.1.0

The following sections detail the changes and enhancements in this release of Oracle Procedural Gateway for APPC.

Changes

The following sections provide information on changes.

Oracle Call Interface

Because of changes to the Oracle Call Interface (OCI), Oracle Procedural Gateway for APPC internal codes for calling COMMIT/CONFIRM have changed. This does not affect user functions.

APPC Log File

The Oracle Procedural Gateway for APPC log file is improved, and provides more information about transactions. For a sample of the new log file, refer to Chapter 6 of the Oracle Procedural Gateway for APPC User’s Guide for UNIX.

Enhancements

Components of the Oracle Procedural Gateway for APPC are based on Oracle9i Server release 9.2.0.1.0 (the Oracle Integrating Server).
Password Encryption Utility for Gateway Initialization File
This release of the gateway includes a utility to support encryption of plain-text passwords in the gateway initialization file. For more information, refer to Chapter 5.

AIX-Only: Password Encryption Utility for Resource Recovery Manager
This release of the gateway includes another utility to support encryption of plain text passwords, which is called LOG_PASS in the initialization file for local LU. For more information, refer to “AIX-Based Systems Only: Using the pg4rrmpwd Utility” on page 5-6.

Bugs Fixed in Version 9i
The following table lists the bugs that have been fixed in version 9i of the gateway, along with their descriptions.

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1276298</td>
<td>When using an invalid value for receive parameter for the pgaxfer procedure, the user receives an ORA 28511 error message.</td>
</tr>
<tr>
<td>1302253</td>
<td>Oracle Procedural Gateway for APPC procedure statements executed on the Oracle integrating servers would hang in SQL*Plus, without giving any messages.</td>
</tr>
<tr>
<td>1336805</td>
<td>When the PGA_SECURITY_TYPE parameter was set to PROGRAM and the PGA_CAPABILITY parameter was set to COMMITCONFIRM, the user received an authentication error but could not see who was complaining about it.</td>
</tr>
<tr>
<td>1404454</td>
<td>The pg4appc log file should be improved and have the received buffer in the log file.</td>
</tr>
<tr>
<td>1411694</td>
<td>The user receives message ORA-28527 when the PGA_CAPABILITY is set to READ_ONLY.</td>
</tr>
<tr>
<td>1472800</td>
<td>Multi-row queries failed on the following error messages: ORA-01401 and ORA-06512.</td>
</tr>
<tr>
<td>1519088</td>
<td>User received sporadic abends when inserting CICS records to VSAM file.</td>
</tr>
<tr>
<td>1677939</td>
<td>Oracle Procedural Gateway for APPC would partially transfer low values to VSAM files.</td>
</tr>
</tbody>
</table>
### Table 1–1 Bugs Fixed in version 9i

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1722467</td>
<td>When the PGA_SECURITY_TYPE parameter was set to PROGRAM and the user specified the user ID and password through database link explicit CONNECT information, the query failed with message PGA-20910 RC=6.</td>
</tr>
<tr>
<td>1724988</td>
<td>When the programmer used RPC PGAINIT_SEC, no matter what syncelevel was being used, the following message was received: “invalid SYNCHLEVEL, 152, specified; valid range is 0:1.”</td>
</tr>
<tr>
<td>2092204</td>
<td>When using PGATCTL procedure with specifying the function code as &quot;D&quot;, the user receives the message PGA-20932 with an invalid function code.</td>
</tr>
<tr>
<td>2128785</td>
<td>Oracle Server Heterogeneous Services place unwanted trace files into /tmp directory.</td>
</tr>
</tbody>
</table>
Known Restrictions

The following restrictions are known to exist in this release of the product.

**AIX-Based Systems Only: Two-Phase Commit Provides No Automatic Recovery With CICS/ESA**
When an update transaction with CICS/ESA at LU6.2 Sync level 2 fails, CICS always rolls back or commits the transaction (based on the INDOUBT parameter of the CICS transaction definition), making automatic recovery by the Oracle integrating server impossible.

**Multibyte Character Sets are Not Supported for Numeric Data and Clauses**
As of version 3.4, the Oracle Procedural Gateway for APPC supports multibyte character set data for IBM VS COBOL II PIC G datatypes. However, the non numeric character data (such as $, (,), +, -,) that is allowed in DISPLAY datatypes and PIC 9 edit masks must still be specified in EBCDIC. The non numeric character data is not subject to MBCS translation.

**CICS Transactions Do Not Allow PF Key Emulation**
When performing a CICS transaction using the Oracle Procedural Gateway for APPC, you cannot emulate CICS PF keys.

**APPC PIP Data is Not Supported**
You cannot define and transmit APPC PIP data in this release of the Oracle Procedural Gateway for APPC.

**Floating Point Datatype Conversion is Not Supported**
Conversion of the floating point datatype is not supported by the Oracle Procedural Gateway for APPC.

**Transaction Programs are Responsible for all Data Compression and Decompression**
The Oracle Procedural Gateway for APPC does not provide exits for compression and decompression facilities. All data exchanged between the Oracle Procedural Gateway for APPC and the transaction must be in uncompressed format.

**PGAU USAGE(NULL) on DEFINE/REDEFINE DATA Not Implemented**
The USAGE(NULL) keyword on the DEFINE or REDEFINE DATA PGAU statements is not yet implemented, even though it is documented.
Known Restrictions

**PGAU Does Not Process COBOL COPY REPLACE Correctly**

When COBOL input to the PGAU DEFINE DATA statement contains a COPY REPLACE clause, only the first replacement is made correctly.

Oracle Procedural Gateway for APPC enables users to initiate transaction program execution on remote online transaction processors (OLTPs). The Oracle Procedural Gateway for APPC provides Oracle applications with seamless access to IBM mainframe data and services through Remote Procedural Call (RPC) processing.

This chapter discusses the architecture, uses, and features of the Oracle Procedural Gateway for APPC. It contains the following sections:

- **Overview** on page 2-2
- **Features** on page 2-2
- **Terms** on page 2-4
- **Architecture** on page 2-7
- **Transaction Types** on page 2-11
Overview

Oracle Procedural Gateway for APPC extends the Remote Procedural Call (RPC) facilities available with the Oracle server. The gateway enables any client application to use PL/SQL to request execution of a remote transaction program (RTP) residing on a host. The gateway provides RPC processing to systems using the SNA APPC (Advanced Program-to-Program Communication) protocol. This architecture allows efficient access to data and transactions available on the IBM mainframe.

The use of a generic and standard protocol, APPC, allows the gateway to access a multitude of systems. The gateway can communicate with virtually any APPC-enabled system, including IBM Corporation’s CICS on any platform, IBM Corporation’s IMS and APPC/MVS, and Computer Associate’s IDMS. These transaction monitors provide access to a broad range of systems, allowing the gateway to access many datastores, including VSAM, DB2 (static SQL), IMS, IDMS, ADABAS and others.

The gateway requires no Oracle software on the remote system. Thus, the gateway uses existing transactions with little or no programming effort on the remote system.

Features

Oracle Procedural Gateway for APPC provides the following benefits:

- Fast interface
  The gateway is optimized so that remote execution of a program is achieved with minimum network traffic. The interface to the gateway is an optimized PL/SQL stored procedure specification (called the “TIP” or “transaction interface package”) precompiled in the Oracle Integrating Server. Because there are no additional software layers on the remote system, overhead occurs only when your program executes.

- Location transparency
  Client applications need not be operating system-specific. For example, your application can call a program in a CICS region on MVS. If you move the program to a CICS region on OS/2, then you need not change the application.

- Application transparency
  Users calling applications that execute a remote transaction program are unaware that a request is sent to a host.
- **Flexible interface**
  You can use the gateway to interface with existing procedural logic or to integrate new procedural logic into an Oracle Integrating Server environment.

- **Oracle server integration**
  The integration of the Oracle server with the gateway enables the gateway to benefit from existing and future Oracle features. For example, the gateway can be called from an Oracle stored procedure or database trigger.

- **Wide selection of tools**
  The gateway supports any tool or application that supports PL/SQL.

- **PL/SQL code generator**
  Oracle Procedural Gateway for APPC provides a powerful development environment, including:
  - a dictionary to store information relevant to the remote transaction
  - a tool to generate the PL/SQL Transaction Interface Package, or TIP
  - a report utility to view the information stored in the gateway dictionary
  - a complete set of tracing and debugging facilities
  - a wide set of samples to demonstrate the use of the product against datastores such as DB2, IMS, IDMS, CICS, and ADABAS

- **Site autonomy and security**
  Oracle Procedural Gateway for APPC provides site autonomy, allowing you to do such things as authenticate users. It also provides role-based security compatible with any security package running on your mainframe computer.

- **Automatic conversion**
  Through the TIP, the following conversions are performed:
  - ASCII to and from EBCDIC
  - remote transaction program datatypes to and from PL/SQL datatypes
  - national language support for many languages
Terms

The following terms are used throughout this guide:

Gateway Initialization File
This file is known as `initsid.ora` and it contains parameters that govern the operation of the gateway. Refer to Appendix A, "Gateway Initialization Parameters" for more information.

Gateway Remote Procedure
Oracle Procedural Gateway for APPC provides prebuilt remote procedures. In general, the following three remote procedures are used:

- PGAINIT, which initializes transactions
- PGAXFER, which transfers data
- PGATERM, which terminates transactions

Refer to "Remote Procedural Call Functions" in Chapter 1 and to Appendix C, "Gateway RPC Interface" in the Oracle Procedural Gateway for APPC User’s Guide for more information about gateway remote procedures.

Oracle Integrating Server
This is any Oracle server instance that communicates with Oracle Procedural Gateway for APPC for purposes of performing remote procedural calls to execute remote transaction programs (RTP). The Oracle Integrating Server can be on the same system as the gateway or on a different system. If it is on a different system, then Oracle Net is required on both systems. Refer to Figure 2–2, "Oracle Procedural Gateway for APPC Architecture" for a view of the gateway architecture.

OLTP (Online Transaction Processor)
OLTP is any of a number of online transaction processors available from other vendors, including CICS, IMS/TM, and IDMS-DC.

PGA (Procedural Gateway Administration)
P GA is a general reference within this guide to all or most components comprising Oracle Procedural Gateway for APPC. This term is used when references to a specific product or component are too narrow.
Terms

PGAU (Procedural Gateway Administration Utility)
PGAU is the tool that is used to define and generate PL/SQL transaction interface packages (TIPs). Refer to Chapter 2, "Procedural Gateway Administration Utility" in the Oracle Procedural Gateway for APPC User’s Guide for more information about PGAU.

PG DD (Data Dictionary)
This component of Oracle Procedural Gateway for APPC is a repository of remote host transaction definitions and data definitions. PGAU accesses definitions in the PG DD when generating TIPs. The PG DD has datatype dependencies because it supports the PGAU and is not intended to be directly accessed by the customer. Refer to Appendix A, "Procedural Gateway for APPC Data Dictionary" in the Oracle Procedural Gateway for APPC User’s Guide for a list of PG DD tables.

PGDL (Procedural Gateway Definition Language)
PGDL is the collection of statements used to define transactions and data to the PGAU.

PL/SQL Stored Procedure Specification (PL/SQL package)
This is a precompiled PL/SQL procedure that is stored in Oracle Integrating Server.

RPC (Remote Procedural Call)
RPC is a programming call that executes program logic on one system in response to a request from another system. See "Gateway Remote Procedure" on page 2-4 for more information, and refer to Appendix C, "Gateway RPC Interface" in the Oracle Procedural Gateway for APPC User’s Guide as well.

RTP (Remote Transaction Program)
A remote transaction program is a customer-written transaction, running under the control of an OLTP, which the user invokes remotely using a PL/SQL procedure. To execute a remote transaction program through the gateway, you must use RPC to execute a PL/SQL program to call the gateway functions.

TIP (Transaction Interface Package)
A TIP is an Oracle PL/SQL package that exists between your application and the remote transaction program. The transaction interface package, or TIP, is a set of PL/SQL stored procedures that invoke the remote transaction program through the
gateway. TIPs perform the conversion and reformatting of remote host data using PL/SQL and UTL_RAW/UTL_PG functions.

**UTL_RAW PL/SQL Package (the UTL_RAW Functions)**

This component of Oracle Procedural Gateway for APPC is a series of data conversion functions for PL/SQL RAW variables and remote host data. The types of conversions performed depend on the language of the remote host data. Refer to "UTL_RAW Functions" in Appendix D of the *Oracle Procedural Gateway for APPC User’s Guide* for more information.

**UTL_PG PL/SQL Package (the UTL_PG Functions)**

This component of Oracle Procedural Gateway for APPC is a series of COBOL numeric data conversion functions. Refer to "NUMBER_TO_RAW and RAW_TO_NUMBER Argument Values" in Appendix D of the *Oracle Procedural Gateway for APPC User’s Guide* for supported numeric datatype conversions.

Figure 2–1, "Relationship of Gateway and Oracle Integrating Server on UNIX Host" illustrates where the terminology discussed in the preceding sections applies within the gateway’s architecture.
The architecture of Oracle Procedural Gateway for APPC consists of three components:

1. **Oracle Integrating Server**

   This server should include the distributed option. The Oracle Integrating Server is usually installed on the same system as Oracle Procedural Gateway for APPC.

   If you install the Oracle Integrating Server on a system other than the system on which the gateway is installed, then you must install Oracle Net with the Oracle Integrating Server and with the gateway. The Oracle Integrating Server must be
capable of connecting to the gateway through any supported Oracle Net protocol.

Refer to Chapter 1, "Release Information" for a list of Oracle Net protocols currently supported by the gateway and tools.

The Oracle Integrating Server can also be used for non-gateway applications.

2. The gateway

Oracle Procedural Gateway for APPC must be installed on a workstation that can run the required version of the operating system.

3. An OLTP (online transaction processor)

The OLTP must be on a system accessible to the system using the SNA APPC protocol. Multiple Oracle Integrating Servers can access the same gateway. A single system gateway installation can be configured to access more than one OLTP.

Figure 2–2 illustrates the architecture of Oracle Procedural Gateway for APPC as described above.

Figure 2–2  Oracle Procedural Gateway for APPC Architecture
Implementation

The gateway has some of the same components as an Oracle database instance on UNIX. The gateway has the following components:

- a home directory, similar to the one associated with an Oracle instance’s ORACLE_HOME environment variable
- a system identifier, identified as *sid* or ORACLE_SID
- an initialization parameter file, similar to the Oracle Integrating Server’s *initsid.ora* file.

The gateway does not have:

- control, redo log, or database files
- the full set of subdirectories and ancillary files associated with an installed Oracle server

Starting the Gateway

The gateway is not started in the same way as an Oracle database instance. It has no background processes and does not need a management utility such as Oracle Enterprise Manager. Each Oracle Integrating Server user session that accesses a particular gateway creates an independent process on UNIX that runs the gateway server and executes SNA functions to communicate with an OLTP.

Communication

All of the communication between the Oracle Integrating Server, the gateway, and the target system are handled through a transaction interface package (TIP). The TIP is a standard PL/SQL package that provides the following functions:

- declares the PL/SQL variables that can be exchanged with a remote transaction program
- calls the gateway packages that handle the APPC communication for starting the conversation, exchanging data, and terminating the conversation
- handles all datatype conversions between PL/SQL datatypes and the target program datatypes

The Procedural Gateway Administration Utility (PGAU), provided with the gateway, automatically generates the TIP specification.
The gateway is identified to the Oracle Integrating Server using a database link. The database link is the same construct used to identify other Oracle server databases. The functions in the gateway are referenced in PL/SQL as:

\textit{function\_name@dblink\_name}

\section*{Remote Procedural Call Functions}

Oracle Procedural Gateway for APPC provides a set of functions that are invoked by the client through remote procedural call (RPC). These functions direct the gateway to initiate, transfer data with, and terminate remote transaction programs running under an OLTP on another system.

The following table lists the remote procedural call functions and the correlating commands that are invoked in the gateway and remote system.

<table>
<thead>
<tr>
<th>Application</th>
<th>Oracle TIP</th>
<th>Gateway</th>
<th>Remote System</th>
</tr>
</thead>
<tbody>
<tr>
<td>call tip_init</td>
<td>tip_init</td>
<td>PGAINIT</td>
<td>Initiate program</td>
</tr>
<tr>
<td></td>
<td>call pgainit@gateway</td>
<td></td>
<td></td>
</tr>
<tr>
<td>call tip_main</td>
<td>tip_main</td>
<td>PGAXFER</td>
<td>Exchange data</td>
</tr>
<tr>
<td></td>
<td>call pgaxfer@gateway</td>
<td></td>
<td></td>
</tr>
<tr>
<td>call tip_term</td>
<td>tip_term</td>
<td>PGATERM</td>
<td>Terminate program</td>
</tr>
<tr>
<td></td>
<td>call pgaterm@gateway</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\subsection*{Remote Transaction Initiation}

The TIP initiates a connection to the target system using one of the gateway functions, PGAINIT. PGAINIT provides, as input, the required SNA parameters to start a conversation with the target transaction program. These parameters are sent across the SNA network, which returns a conversation identifier to PGAINIT. Any future calls to the target program use the conversation identifier as an INPUT parameter.

\subsection*{Data Exchange}

After the conversation is established, a procedural gateway function called PGAXFER can exchange data in the form of input and output variables. PGAXFER sends and receives buffers to and from the target transaction program. The gateway sees a buffer as only a RAW stream of bytes. The TIP that is residing in the Oracle Integrating Server is responsible for converting the application’s PL/SQL datatypes to RAW before sending the buffer to the gateway. It is also responsible for...
converting RAW to the PL/SQL datatypes before returning the results to the application.

**Remote Transaction Termination**

When communication with the remote program is complete, the gateway function PGATERM terminates the conversation between the gateway and the target system. PGATERM uses the conversation identifier as an INPUT parameter to request conversation termination.

**Transaction Types**

Oracle Procedural Gateway for APPC supports three types of transactions that read data from and write data to remote systems:

- **one-shot**
  
  In a one-shot transaction, the application executes initialization, exchanges data and terminates the connection, all in a single call.

- **persistent**
  
  In a persistent transaction, multiple calls to exchange data with the remote transaction can be executed before terminating the conversation.

- **multi-conversational**
  
  In a multi-conversation transaction, the procedural gateway server can be used to exchange multiple records in one call to the remote transaction program.

Refer to “Remote Host Transaction Types” in Chapter 4, “Client Application Development” of the *Oracle Procedural Gateway for APPC User’s Guide* for more information about transaction types.

The following list demonstrates the power of the Oracle Procedural Gateway for APPC:

- You can initiate a CICS transaction on the mainframe to retrieve data from a VSAM file for a PC application.
- You can modify and monitor the operation of a remote process control computer.
- You can initiate an IMS/TM transaction that executes static SQL in DB2.
- You can initiate a CICS transaction that returns a large number of records in a single call.
This chapter describes the system requirements of Oracle Procedural Gateway for APPC. It contains the following sections:

- **Hardware Requirements** on page 3-2
- **Software Requirements** on page 3-3
- **Documentation Requirements** on page 3-6
Hardware Requirements

The hardware requirements for this release of the gateway are described in the following sections.

Processor

Table 3–1 lists the processor requirements for Oracle Procedural Gateway for APPC for your platform.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Processor</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX-Based Systems</td>
<td>IBM RS/6000</td>
</tr>
<tr>
<td>HP-UX</td>
<td>HP 9000 Series HP-UX that can run the required version of HP-UX</td>
</tr>
<tr>
<td>Solaris</td>
<td>A Sun SPARC workstation that can run the required version of Solaris</td>
</tr>
</tbody>
</table>

Memory

For most installations, Oracle Corporation recommends 256 of real memory to support Oracle Procedural Gateway for APPC.

The following factors affect the memory requirements of the gateway server process:

- number of concurrent APPC connections open by each user
- number of data items being transferred between the gateway and the remote transaction program
- additional factors such as configured network buffer size
- the Oracle Net protocol adapters that were included during the gateway installation

Network Attachment

Oracle Procedural Gateway for APPC requires any network attachment supported by the SNA Communication Package for your platform.
CD-ROM Drive

A CD-ROM drive is required for installation of the software.

**HP-UX Only**
Oracle uses ISO 9660 format CD-ROM disks with RockRidge extensions.

Disk Space

*Table 3–2 lists the disk space requirements for the Oracle Procedural Gateway for APPC for your platform.*

<table>
<thead>
<tr>
<th>Platform</th>
<th>Disk Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX-Based Systems</td>
<td>1500 MB</td>
</tr>
<tr>
<td>HP-UX</td>
<td>800 MB</td>
</tr>
<tr>
<td>Solaris</td>
<td>500 MB</td>
</tr>
</tbody>
</table>

Software Requirements

The system software configuration described in these requirements is supported by Oracle Corporation provided that the underlying system software products are supported by their respective vendors. Verify the latest support status with your system software vendors.

**Software Requirements for AIX-Based Systems**

Depending on your platform, the software requirements are as follows:

- *Table 3–3 lists the software requirements for AIX-Based Systems*
- *Table 3–4 lists the software requirements for HP-UX*
- *Table 3–5 lists the software requirements for Solaris*

*Table 3–6 lists the software requirements which are common across the UNIX platforms.*

*Table 3–3  Software Requirements for AIX-Based Systems*

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>AIX version 4.3.3 or AIX 5L</td>
</tr>
</tbody>
</table>
The following table lists the software requirements specific to HP-UX.

**Table 3–4  Software Requirements for HP-UX**

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>HP-UX version 11.0 is required on a 64-bit system.</td>
</tr>
<tr>
<td>Communications</td>
<td>HP-UX SNAPPlus2 Link Release 11.x and HP-UX SNAPPlus2 API Release 11.x or higher are required.</td>
</tr>
</tbody>
</table>

The following table lists the software requirements specific to Solaris.

**Table 3–5  Software Requirements for Solaris**

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>Sun Solaris version 2.6 or higher.</td>
</tr>
<tr>
<td>Communication</td>
<td>Sunlink SNA Peer-to-Peer Communication Server version 9 SNAP-IX version 6</td>
</tr>
</tbody>
</table>

The following table lists the software requirements which are common across the UNIX platforms.

**Table 3–6  Software Requirements Common Across UNIX Platforms**

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Integrating Server</td>
<td>The latest patchset for Oracle9i release 1 (9.0.1), Oracle9i release 2 (9.2.0), and Oracle8i release 8.1.7. If necessary, refer to your vendor for the latest patchset information.</td>
</tr>
</tbody>
</table>
OLTP

The OLTP must support mapped APPC conversations. If the OLTP transaction programs to be executed through the gateway perform database updates, then the APPC verbs CONFIRM, CONFIRMED, and SEND_ERR must be supported by the OLTP. These verbs implement APPC sync level 1.

All resources controlled by an OLTP that can be updated by transaction programs invoked through the gateway must be defined as recoverable resources to the OLTP and host system if COMMIT/ROLLBACK capability is required for those resources. For example, a VSAM file updated by a CICS transaction must be defined to CICS as a recoverable file for COMMIT/ROLLBACK to control the updates.

**CICS/MVS and CICS/ESA**

Release 4.3 or higher is required

**CICS/VSE**

Release 2.3 or higher is required.

**IDMS-DC**

Release 12 or higher is required.

**IMS/TM**

Release 5.1 or later is required.

**APPC/MVS**

OS/390 V2R7 is required.

**Important:** For a list of known restrictions, be sure to read the “Known Restrictions” section on page 1-5 before proceeding with installation of the gateway.

**Oracle networking products**

Oracle Net Client and Oracle Adapter must be installed on the system where the Oracle9i Server is installed. In addition, Oracle Net and the Oracle Adapter must be installed on the system where the gateway is installed.

---

**Table 3–6 Software Requirements Common Across UNIX Platforms**

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLTP</td>
<td>The OLTP must support mapped APPC conversations. If the OLTP transaction programs to be executed through the gateway perform database updates, then the APPC verbs CONFIRM, CONFIRMED, and SEND_ERR must be supported by the OLTP. These verbs implement APPC sync level 1. All resources controlled by an OLTP that can be updated by transaction programs invoked through the gateway must be defined as recoverable resources to the OLTP and host system if COMMIT/ROLLBACK capability is required for those resources. For example, a VSAM file updated by a CICS transaction must be defined to CICS as a recoverable file for COMMIT/ROLLBACK to control the updates.</td>
</tr>
<tr>
<td>CICS/MVS and CICS/ESA</td>
<td>Release 4.3 or higher is required</td>
</tr>
<tr>
<td>CICS/VSE</td>
<td>Release 2.3 or higher is required.</td>
</tr>
<tr>
<td>IDMS-DC</td>
<td>Release 12 or higher is required.</td>
</tr>
<tr>
<td>IMS/TM</td>
<td>Release 5.1 or later is required.</td>
</tr>
<tr>
<td>APPC/MVS</td>
<td>OS/390 V2R7 is required.</td>
</tr>
<tr>
<td>Oracle networking products</td>
<td>Oracle Net Client and Oracle Adapter must be installed on the system where the Oracle9i Server is installed. In addition, Oracle Net and the Oracle Adapter must be installed on the system where the gateway is installed.</td>
</tr>
</tbody>
</table>
Documentation Requirements

Oracle recommends that you read the following documentation on products other than the gateway:

- PL/SQL User’s Guide and Reference
- Oracle9i Installation Guide for UNIX
- Oracle9i Administrator’s Reference
- Oracle9i Server Concepts
- Oracle9i Server Messages
- Oracle Net Administrator’s Guide
- Programmer’s Guide to the Oracle Call Interface

For Operating System and SNA communications package references, refer to the appropriate vendor documentation.
The following topics in this chapter describe how to install and configure Oracle Procedural Gateway for APPC:

- Before You Begin on page 4-2
- Planning to Upgrade the Gateway on page 4-2
- Performing Pre-upgrade Procedures on page 4-3
- Performing Pre-Installation Procedures on page 4-4
- Installing the Gateway Software on page 4-5
- Installation Steps on page 4-6
- Configuring Oracle Procedural Gateway for APPC on page 4-13
- Verifying the Installation on page 4-32
- Performing Postinstallation Procedures on page 4-37
- Deinstalling Your Oracle Procedural Gateway for APPC on page 4-39
Before You Begin

Configuring an online transaction processor to allow access by the gateway requires actions on the OLTP and on certain components of the host operating system. Although no Oracle software is installed on the host system, access to, and some knowledge of the host system and the OLTP are required. Although this chapter includes some information about host system and OLTP installation steps, you must ensure that you have the applicable OLTP and host system documentation available.

Some of the configuration actions on the OLTP might require you to restart the OLTP. In preparation for this, have your host system programmer or DBA review the instructions for your OLTP to allow for any necessary preparations.

To install and configure the gateway with a single Oracle Integrating Server and a single OLTP, perform the procedures described in this chapter. The final section, "Performing Postinstallation Procedures", contains information about expanding the configuration to multiple integrating servers and multiple OLTPs.

Planning to Upgrade the Gateway

This section is only for customers upgrading from a previous release of Oracle Procedural Gateway for APPC. If you are installing for the first time, begin with "Performing Pre-Installation Procedures" on page 4-4.

Refer to Appendix C, "Summary of Changes in Previous Versions" for information on changes or corrected problems for earlier releases of the gateway.
Upgrade Considerations

Upgrade considerations are as follows:

- PGAU control files from any earlier release are upward compatible and you do not need to change them.
- After upgrade, the PG DD contains all of its earlier entries without modification. New PGAU control information has been added along with some columns to support new features, but no customer entries are altered by the upgrade.
- All TIPs from Oracle Procedural Gateway for APPC Release 4.0.1 or earlier must be recompiled, due to changes in the following:
  - PL/SQL compatibility
  - gateway server RPC interface
  - UTL_PG interface

Caution: An upgraded PG Data Dictionary (PG DD) cannot be accessed by an earlier release of PGAU.

Restoration

If you want to restore a previous release of PGA, then you must restore the following components to their previous versions:

- PGAU
- PG DD
- Gateway server

Performing Pre-upgrade Procedures

Perform the following steps to prepare for upgrading Oracle Procedural Gateway for APPC to current versions:

1. Make backups of altered PGA shipped files.
2. Remove or rename any old gateway directories.
Performing Pre-Installation Procedures

Before you install Oracle Procedural Gateway for APPC, perform the following pre-installation procedures:

■ Ensure that your system meets all of the hardware and software requirements of Oracle Procedural Gateway for APPC.
  – A CD-ROM drive is required.
    Oracle Procedural Gateway for APPC is currently shipped on CD-ROM media. The hardware required for the medium on which the gateway was ordered must be installed on your system.
    – Refer to Chapter 3, "System Requirements" for a list of the other hardware and software requirements.

■ Ensure that your security requirements are met.
  Refer to Chapter 3, "System Requirements" for more information about the security requirements for connections and data access on your OLTP.

■ Decide on an SID (system identifier) for your gateway. This SID is used in the Configuration section on page 4-23.
  The SID must be unique and must not be used by any other gateway or Oracle Integrating Server on the system.

■ Ensure that your system can communicate with the OLTP using the SNA Communication Package for your platform.
  For more information about setting up and configuring SNA Communication Packages to run Oracle Procedural Gateway for APPC, refer to the chapter on SNA Communication Packages for your platform:
  – For AIX-Based Systems, refer to Chapter 6, "Configuring the SNA Communication Package on AIX-Based Systems".
  – For HP-UX, refer to Chapter 7, "Configuring the SNA Communication Package on HP-UX".
  – For Solaris, refer to Chapter 8, "Configuring the SNA Communication Package on Solaris".

If you need general information about installing Oracle products and using the Oracle Universal Installer, then refer to the Oracle9i Installation Guide.
You can install Oracle Procedural Gateway for APPC in either of the following ways:

1. On the same system as the existing Oracle Integrating Server.
   All tasks for this type of installation or upgrade are discussed in this section.

2. Stand-alone without a local Oracle Integrating Server.

---

**Note:** In a stand-alone installation, PGAU executes on the same system as the gateway, not on the system that the Oracle Integrating Server is on. This might be a consideration in determining where to store and access PGAU input control files and output TIP's that are generated.

---

**Attention:** When the gateway is installed as a stand-alone or is accessed remotely, an Oracle Net protocol supported by your Oracle Integrating Server must be installed with the gateway to allow communication between the Oracle Integrating Server and the gateway. All of the supporting components for the selected Oracle Net protocol must be running on the gateway system. For example, if you select an Oracle Net protocol adapter, then the TNS listener must be configured and started on the gateway system. For complete information on installing and configuring Oracle Net, refer to the appropriate Oracle Net documentation.

---

**Installing the Gateway Software**

For general information about installing Oracle products and how to use the Oracle Universal Installer, refer to the [Oracle9i Installation Guide](#).

Oracle Procedural Gateway for APPC must be installed in its own separate Oracle home directory. Do not install the gateway in the same Oracle home directory as the Oracle integrating server. This is required to isolate the gateway from Oracle Integrating Server upgrades that might cause incompatibilities if the gateway executables were relinked with later versions of the Oracle server libraries.

The gateway product set includes some of the same components as the Oracle Integrating Server, and the components of Oracle Procedural Gateway for APPC must be Release 9.2.0.1.0 to ensure that the gateway functions correctly.
Installation Steps

You can install the Oracle Integrating Server and Oracle Procedural Gateway for APPC on the same system or on separate systems.

If you install them on the same system, then you must install the Oracle Integrating Server before installing the gateway. Start Oracle Procedural Gateway for APPC upgrades only after the Oracle Integrating Server release 9.2.0.1.0 (or higher) is operational.

Step 1: Login as DBA and Create Login User ID
Log in as the Oracle database administrator (DBA) user. If you are not currently a DBA user, then contact your system administrator to create a DBA login user ID. Refer to the Oracle9i Installation Guide for login information.

Step 2: Create the Product Installation Directory
When you create a new directory, Oracle Corporation recommends that you use the version number as part of the path name. Doing so allows different versions of the same Oracle product to be installed under one Oracle directory tree. The product installation directory is also known as the $ORACLE_HOME for the gateway, and it has also been called the pg4appc directory. The gateway directory (product installation directory) must be separate from the Oracle Integrating Server.

Note: If the Oracle Integrating Server and the gateway reside on separate systems, then be sure that a product directory named oracle exists.

For example, to create the product directory, enter:

$ mkdir /oracle

Then create the product installation directory, /oracle/pga/9.2.0.

For example, enter:

$ mkdir /oracle/pga
$ mkdir /oracle/pga/9.2.0
$ chown oracle:dba /oracle/pga/9.2.0
$ chmod 755 /oracle/pga/9.2.0
Step 3: Set the ORACLE_HOME Environment Variable

The ORACLE_HOME environment variable must point to the directory that you created in Step 2. The command to set the environment variable depends on the shell that you are using.

For example, if you are a Bourne or Korn shell user, then enter:

$ ORACLE_BASE=/oracle/pga; export ORACLE_BASE
$ ORACLE_HOME=/oracle/pga/9.2.0; export ORACLE_HOME

If you are a C shell user, then enter:

$ setenv ORACLE_BASE /oracle/pga
$ setenv ORACLE_HOME /oracle/pga/9.2.0

---

**Note:** Make sure that the TNS_ADMIN environment variable points to the path where your tnsnames.ora and listener.ora files are. These files are usually in the $ORACLE_HOME/network/admin directory.

If you are a Bourne or Korn shell user, then enter:

$ TNS_ADMIN=$ORACLE_HOME/network/admin
$ export TNS_ADMIN

If you are a C shell user, then enter:

$ setenv TNS_ADMIN $ORACLE_HOME/network/admin

---

Step 4: Set the DISPLAY Environment Variable

Set the DISPLAY environment variable to the X-server display where the Oracle Universal Installer is displayed.

For example, if you are using a Bourne or Korn shell, then enter:

$ DISPLAY=machine:0; export DISPLAY

If you are a C shell user, then enter:

$ setenv DISPLAY machine:0

Refer to the Oracle9i Installation Guide for more information.
Step 5: Mount the CD-ROM

The following sections describe how to mount the CD-ROM for each of the platforms.

Mounting the CD-ROM for AIX-Based Systems
Insert the CD in your CD-ROM drive and enter:

```shell
$ su root
# mkdir /cdrom
# mount -r -v cdrfs /dev/cd0 /cdrom
# exit
```

Mounting the CD-ROM for HP-UX
To mount the CD-ROM, perform the following:

1. Use a system editor to add the following line to the `/etc/pfs_fstab` file.

   ```
   device_file  mount_point  filesystem_type  translation_method
   
   For example:
   
   /dev/dsk/c5t2d0 /SD_CDROM pfs-rrrip xlat=unix 10
   
   The first entry is the CD-ROM device file; the second is the mount point. The third entry indicates that the CD-ROM to be mounted is in ISO9660 format with RockRidge extension.
   
   2. Log in as the root user:
   
   ```shell
   $ su root
   
   3. Enter the following command:
   
   ```shell
   # nohup/usr/sbin/pfs_mountd &
   
   4. Enter the following file command:
   
   ```shell
   # nohup/usr/sbin/pfzd &
   
   5. Insert the CD into the tray and enter the following command to mount the CD-ROM:
   
   ```shell
   # /usr/sbin/pfs_mount /SD_CDROM
   ```
6. Exit the root account.
   # exit

Change directories to /SD_CDROM where you can see a lowercase listing of the directories and files on the CD-ROM. The mounted CD-ROM should appear as another read-only file system.

Mounting the CD-ROM for Solaris
Insert the CD in your CD-ROM drive and enter:
$ cd /cdrom

Step 6: Start the Oracle Universal Installer
The following sections describe how to start the Oracle Universal Installer for each of the platforms.

The Oracle Universal Installer is provided on the distribution CD-ROM with the gateway. For general information about installing Oracle products and how to use the Oracle Universal Installer, refer to the Oracle9i Installation Guide.

For AIX-Based Systems
Start the Oracle Universal Installer, as follows:

1. Start the Oracle Universal Installer with the following command:
   $ cd cdrom_mount_point_directory
   $ ./runInstaller

   A window appears with the question "Has the rootpre.sh script been run by root?" Before you can proceed with the installation, this script must have been run on the install system of the gateway.

2. If you have already run this script (either during a previous installation of this version of the gateway, or during an installation of the Oracle9i Server), then type "Y" and begin stepping through the Oracle Universal Installer. Otherwise, follow step 3 below.

3. If the rootpre.sh script has not yet been run, type "N" and complete the following steps:
   a. Log on as the root user.
b. Change directory to the install directory on the CD-ROM.
c. Run the rootpre.sh script.
d. Exit from the root user.
e. Restart the Oracle Universal Installer.

For HP-UX
Start the Oracle Universal Installer, as follows:

```bash
$ cd cdrom_mount_point_directory
$ ./runInstaller
```

For Solaris
Start the Oracle Universal Installer, as follows:

```bash
$ cd cdrom_mount_point_directory
$ ./runInstaller
```

**Step 7: Step through the Oracle Universal Installer**

---

**Caution:** Oracle Universal Installer automatically installs the Oracle-supplied version of the Java Runtime Environment (JRE). This version is required to run Oracle Universal Installer and several Oracle assistants. Do not modify the JRE except by using a patch provided by Oracle Support Services. The Installer also installs JDK 1.3.1 on Solaris and Windows NT. On AIX-Based Systems and HP-UX the Installer prompts for the downloaded.installed location of JDK 1.3.1.

---

Oracle Universal Installer is a menu-driven utility that guides you through installing the gateway by prompting you with action items. The action items and the sequence in which they appear depend on your platform.

The following sections describe how to go through the installation for each of the platforms.
Oracle Universal Installer on AIX-Based Systems and HP-UX

Use the following table as a guide to step through the Oracle Universal Installer, performing the actions described in the Response column.

<table>
<thead>
<tr>
<th>Prompt</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Oracle Universal Installer: Welcome</td>
<td>Click &quot;Next&quot;.</td>
</tr>
<tr>
<td>2. Oracle Universal Installer: File Locations</td>
<td>Specify the source and destination directories and click &quot;Next&quot;.</td>
</tr>
<tr>
<td>3. Oracle Universal Installer: Available Products</td>
<td>Select &quot;Oracle 9i Database 9.2.0.1.0&quot; and click &quot;Next&quot;.</td>
</tr>
<tr>
<td>4. Oracle Universal Installer: Installation Types</td>
<td>Select &quot;Custom and click Next&quot;.</td>
</tr>
<tr>
<td></td>
<td>b. Select &quot;Oracle Transparent Gateways 9.2.0.1.0&quot;, open up this row.</td>
</tr>
<tr>
<td></td>
<td>c. Select &quot;Oracle Procedural Gateway for APPC 9.2.0.1.0&quot;.</td>
</tr>
<tr>
<td></td>
<td>d. Click &quot;Next&quot;.</td>
</tr>
<tr>
<td>6. Oracle Universal Installer: Choose JDK Home Directory</td>
<td>Enter the home directory for JDK and click &quot;Next&quot;.</td>
</tr>
<tr>
<td>7. Setup Privileges</td>
<td>You must run the root.sh configuration script from the SORACLE_HOME directory at this point. Leave the installation window open, run the script as the root user from another window, then come back to the installation screen and click OK to continue.</td>
</tr>
<tr>
<td>8. Oracle Net Configuration Assistance: Welcome</td>
<td>Click &quot;Cancel&quot;.</td>
</tr>
<tr>
<td>9. Oracle Net Configuration Assistance:</td>
<td>Click &quot;Yes&quot;.</td>
</tr>
<tr>
<td>10. Oracle Universal Installer: Configuration Tools</td>
<td>Click &quot;Exit&quot;.</td>
</tr>
<tr>
<td>11. Oracle Universal Installer: End of Installation</td>
<td>Click &quot;Exit&quot;.</td>
</tr>
<tr>
<td>12. Exit</td>
<td>Click &quot;Yes&quot;.</td>
</tr>
</tbody>
</table>
Oracle Universal Installer on Solaris

Use the following table as a guide to step through the Oracle Universal Installer, performing the actions described in the Response column.

<table>
<thead>
<tr>
<th>Prompt</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Oracle Universal Installer: Welcome</td>
<td>Click &quot;Next&quot;.</td>
</tr>
<tr>
<td>2. Oracle Universal Installer: File Locations</td>
<td>Specify the source and destination directories and click &quot;Next&quot;.</td>
</tr>
<tr>
<td>3. Oracle Universal Installer: Available Products</td>
<td>Select &quot;Oracle 9i Database 9.2.0.1.0&quot; and click &quot;Next&quot;.</td>
</tr>
<tr>
<td>4. Oracle Universal Installer: Installation Types</td>
<td>Select &quot;Custom&quot; and click &quot;Next&quot;.</td>
</tr>
<tr>
<td></td>
<td>b. Select &quot;Oracle Transparent Gateways 9.2.0.1.0&quot;, open up this row.</td>
</tr>
<tr>
<td></td>
<td>c. Select &quot;Oracle Procedural Gateway for APPC 9.2.0.1.0&quot;.</td>
</tr>
<tr>
<td></td>
<td>d. Click &quot;Next&quot;.</td>
</tr>
<tr>
<td>6. Oracle Universal Installer: SNA Network Software</td>
<td>Specify your SNA package and click &quot;Next&quot;.</td>
</tr>
<tr>
<td>7. Setup Privileges</td>
<td>You must run the root.sh configuration script from the root directory at this point. Leave the installation window open, run the script as the root user from another window, then come back to the installation screen and click &quot;OK&quot; to continue.</td>
</tr>
<tr>
<td>8. Oracle Net Configuration Assistance: Welcome</td>
<td>Click &quot;Cancel&quot;.</td>
</tr>
<tr>
<td>9. Oracle Net Configuration Assistance:</td>
<td>Click &quot;Yes&quot;.</td>
</tr>
<tr>
<td>10. Oracle Universal Installer: Configuration Tools</td>
<td>Click &quot;Exit&quot;.</td>
</tr>
<tr>
<td>11. Oracle Universal Installer: End of Installation</td>
<td>Click &quot;Exit&quot;.</td>
</tr>
<tr>
<td>12. Exit</td>
<td>Click &quot;Yes&quot;.</td>
</tr>
</tbody>
</table>

Oracle Procedural Gateway for APPC is now installed.
When the Oracle Universal Installer confirms that the installation is complete, verify that the installation procedure was successful. To do this, read the contents of the installation log file, which is located in the `$ORACLE_HOME/install` directory. The default file name is `make.log`.

---

**Attention:** Print the contents of the `$ORACLE_HOME/pg4appc/doc/README.doc` file and read it all. It contains important information about the installation. After reading the `README.doc` file, continue with the following section.

---

**Configuring Oracle Procedural Gateway for APPC**

Configuring Oracle Procedural Gateway for APPC involves working with the following components:

- the Oracle Integrating Server
- your UNIX system
- your network
- the OLTP

**Preconfiguration Steps**

Follow these steps to prepare for the configuration of your Oracle Integrating Server.

**Oracle Server Coexistent with Gateway Installation/Upgrade**

If you are installing or upgrading the gateway on the system where your Oracle Integrating Server is already installed, then your `ORACLE_HOME` environment variable must be changed to point to the Oracle Integrating Server `$ORACLE_HOME` directory.

For example, to set the `ORACLE_HOME` environment variable to the Oracle integrating server `/oracle/product/9.2.0` directory, run the following:

- If you are running the Bourne or Korn shell, enter:
  ```
  $ ORACLE_HOME=/oracle/product/9.2.0;export ORACLE_HOME
  ```

- If you are running the C shell, enter:
  ```
  $ setenv ORACLE_HOME /oracle/product/9.2.0
  ```
You must now define soft links to allow access to the gateway administrative files from the Oracle Integrating Server's $ORACLE_HOME directory and to allow access to PGAU from your existing path.

**Note:** If you are upgrading the gateway, then you must rename or remove previously-defined links or directories from your Oracle Integrating Server $ORACLE_HOME directory. To rename them, enter the following commands:

```
$ cd $ORACLE_HOME
$ mv pg4appc pg4appc_backup
$ cd bin
$ mv pgau pgau_backup
```

To set up the soft links to the current gateway administrative files and PGAU, enter commands similar to the following:

```
$ cd $ORACLE_HOME
$ ln -s /oracle/pga/9.2.0/pg4appc pg4appc
$ ln -s /oracle/pga/9.2.0/bin/pgau pgau
```

Now proceed to "Configuring the Oracle Integrating Server for First-Time Installations" on page 4-15 if this is a first-time install, or with "Upgrading the Oracle Integrating Server from Previous Releases" on page 4-17 if this is an upgrade.

**Stand-alone Gateway Installation/Upgrade**

If you are installing or upgrading the gateway stand-alone (on a system that has no Oracle Integrating Server installed), then you must transfer some of the gateway administrative files to the system where your Oracle Integrating Server is installed. The files are in the gateway $ORACLE_HOME/pg4appc/admin directory. All files in this directory that have the suffix .sql, .pkh, and .pkb should be copied into a similarly-named directory in the Oracle Integrating Server's $ORACLE_HOME directory.

Your Oracle Integrating Server DBA can create the directory with the following commands:

```
$ cd $ORACLE_HOME
$ mkdir pg4appc
$ mkdir pg4appc/admin
```
Use whatever file transfer mechanism is available on your system to copy all of the
.sql, .pkh, and .pkb files from the gateway
$ORACLE_HOME/pg4appc/admin directory to the Oracle Integrating Server
$ORACLE_HOME/pg4appc/admin directory.

Proceed with "Configuring the Oracle Integrating Server for First-Time Installations" on page 4-15 if this is a first-time install, or with "Upgrading the Oracle Integrating Server from Previous Releases" on page 4-17 if this is an upgrade.

Configuring the Oracle Integrating Server for First-Time Installations

Follow these steps to configure your Oracle Integrating Server if you are installing Oracle Procedural Gateway for APPC for the first time:

1. Ensure that the UTL_RAW PL/SQL package has been installed on your Oracle Integrating Server. All PGAU-generated TIP specifications use UTL_RAW, which provides routines for manipulating raw data.

   a. Use SQL*Plus to connect to the Oracle Integrating Server as user SYS.

   b. Enter the following command:

      DESCRIBE UTL_RAW;

      The DESCRIBE command produces output on your screen. If you browse through the output, you should see some functions, including a compare function. If you do not see this output, then continue the UTL_RAW installation by performing steps c and d below.

      If the DESCRIBE command indicates success, then your Oracle Integrating Server has UTL_RAW installed and you can proceed to Step 2.

   c. From SQL*Plus, run the utlraw.sql and prvtrawb.plb scripts in the Oracle Integrating Server $ORACLE_HOME/rdbms/admin directory, in the following order:

      SQL> @$ORACLE_HOME/RDBMS/admin/UTLRAW.SQL
      SQL> @$ORACLE_HOME/RDBMS/admin/PRVTRAWB.SQL

2. Ensure that the DBMS_OUTPUT standard PL/SQL package is enabled on your Oracle Integrating Server. The sample programs and installation verification programs on the distribution media use this standard package.

   a. If necessary, use SQL*Plus to connect to the Oracle Integrating Server as user SYS.
b. Enter the following command:

```
DESCRIBE DBMS_OUTPUT;
```

The describe statement produces output on your screen. If you browse through that output, you should see some functions, including a put_line function.

If you do not see this output, then you must create the DBMS_OUTPUT package. Refer to the Oracle9i Application Developer's Guide for more information about creating the DBMS_OUTPUT package. After successful installation of the DBMS_OUTPUT package, issue the describe statement.

If the describe statement indicates success, then your Oracle Integrating Server has DBMS_OUTPUT created, and you can proceed to Step 3.

3. Install the UTL_PG PL/SQL package. All PGAU-generated TIP specifications use UTL_PG, which provides routines for performing numeric conversions to and from raw data.

   a. If necessary, use SQL*Plus to connect to the Oracle Integrating Server as user SYS.

   b. From SQL*Plus, run the `utlpg.sql` and `prvtpgb.plb` scripts in the Oracle Integrating Server `$ORACLE_HOME/rdbms/admin` directory, in the following order:

```
SQL> @$ORACLE_HOME/RDBMS/ADMIN/UTLPG.SQL
SQL> @$ORACLE_HOME/RDBMS/ADMIN/PRVTPGB.PLB
```

4. Install the Heterogeneous Services (HS) catalogs.

   a. If necessary, use SQL*Plus to connect to the Oracle Integrating Server as user SYS.

   b. Enter the following command:

```
DESCRIBE HS_FDS_CLASS;
```

The describe statement produces output on your screen. If the describe statement indicates success, then heterogeneous services catalogs have been created on your Oracle Integrating Server and you can proceed to Step 5.

If the describe statement does not indicate success, then you must create Heterogeneous Services catalogs and you must follow step c below:

   c. If it is necessary to create the Heterogeneous Services catalog, enter the following command:
5. Create a public database link to access Oracle Procedural Gateway for APPC:
   Use SQL*Plus to connect to the Oracle Integrating Server as user SYSTEM. You can use the following SQL*Plus sample whether the Oracle Integrating Server and the gateway are on the same system or on different systems. In the following sample, pgasrv is the tns_name_entry assigned to the gateway in the tnsnames.ora file.

   SQL> CREATE PUBLIC DATABASE LINK PGA USING 'PGASRV';

6. Create the gateway administrator user PGAADMIN and install the PG DD.
   a. Use SQL*Plus to connect to the Oracle Integrating Server as user SYSTEM.
   b. From SQL*Plus, run the pgacr8au.sql script in the
      $ORACLE_HOME/pg4appc/admin directory. This script creates the
      PGAADMIN user ID.

      The initial password defined for PGAADMIN is PGAADMIN. Use the
      ALTER USER command to change the password. For further information,
      refer to the Oracle9i Database SQL Reference.
   c. Use SQL*Plus to connect to the Oracle Integrating Server as user
      PGAADMIN.
   d. From SQL*Plus, run the pgddcr8.sql script in the
      $ORACLE_HOME/pg4appc/admin directory. This script installs the
      PG DD.
   e. From SQL*Plus, connect to the Oracle Integrating Server as user SYS.
   f. Grant execution privileges on DBMS_PIPE to PGAADMIN:

      SQL> GRANT EXECUTE ON DBMS_PIPE TO PGAADMIN;

Proceed with Step 9. Do not perform Steps 7 and 8.

Upgrading the Oracle Integrating Server from Previous Releases

7. Upgrade Oracle Procedural Gateway for APPC to current version levels.
   a. Use SQL*Plus to connect to the Oracle Integrating Server as user SYS.
   b. Install the UTL_RAW package body. From SQL*Plus, run the prvtrawb.plb
      script from the $ORACLE_HOME/rdbms/admin directory. This script
      upgrades the UTL_RAW package body.
c. Install the UTL_PG package body. From SQL*Plus, run the `prvtpgb.plb` script from the `$ORACLE_HOME/rdbms/admin` directory. This script upgrades the UTL_PG package body.

If the `prvtrawb.plb` and `prvtpgb.plb` scripts complete successfully, then continue with Step 8. If they fail because specifications do not exist or were invalidated, then consider reinstalling the package specifications in Step 7d.

d. If possible, avoid reinstalling the package specifications. You might have to reinstall the package specifications, however, if the UTL_RAW or UTL_PG package has been invalidated or deinstalled. Oracle Corporation recommends that you avoid reinstalling the package specifications because any dependent objects (such as existing user TIPs) are invalidated and subsequently need to be recompiled also. The Oracle database recompiles automatically as the TIPs are referenced. Because client applications are dependent on TIPs, the client applications subsequently need recompilation also. The impact of this is a one-time performance delay while recompilation of the TIPs and dependent client applications proceeds.

Important: Before proceeding with this step, make sure that you are in the `$ORACLE_HOME/pg4appc/admin` directory.

TIPs were split into separate specification and body files in release 3.3 to avoid these cascaded recompilations in later releases.

If you need to reinstall the package specifications, run the following scripts:

- `$ORACLE_HOME/rdbms/admin/utlrw.sql` to upgrade the UTL_RAW package specification
- `$ORACLE_HOME/rdbms/admin/utlpq.sql` to upgrade the UTL_PG package specification.

After the scripts have run, repeat Steps b and c.

8. Upgrade the PG DD as follows before executing the new PGAU:

a. If necessary, use SQL*Plus to connect to the Oracle Integrating Server as user PGAADMIN.

b. From SQL*Plus, run the `pgddupgr.sql` script in the `$ORACLE_HOME/pg4appc/admin` directory. This script upgrades the PG DD.
First-Time Installation and Configuration Steps

Perform these steps only if you are installing the gateway for the first time. If you are upgrading, then proceed to "Optional Configuration Steps to Permit Multiple Users" on page 4-19.

9. Install the TIP trace access PL/SQL routines. These routines require that the DBMS_PIPES standard PL/SQL package is installed and that PGAADMIN has execute privileges on it. For more information on DBMS_PIPES, refer to the Oracle9i Application Developer’s Guide.
   a. If necessary, use SQL*Plus to connect to the Oracle Integrating Server as user PGAADMIN.
   b. From SQL*Plus, run the pgatiptr.sql script in the $ORACLE_HOME/pg4appc/admin directory. This script creates PL/SQL routines that can be called to read and purge trace information created by PGAU-generated TIP specifications. It also creates public synonyms for these routines. The script prompts you for the necessary user IDs and passwords.

10. Install the GPGLOCAL package. This package is required for compilation and execution of all PGAU-generated TIP specifications. TIP developers should be granted execute privileges on GPGLOCAL (see optional step 12).
   a. Use SQL*Plus to connect to the Oracle Integrating Server as user PGAADMIN.
   b. From SQL*Plus, run the gpglocal.pkh script in the $ORACLE_HOME/pg4appc/admin directory. This script compiles the GPGLOCAL package specification.
   c. From SQL*Plus, run the gpglocal.pkb script in the $ORACLE_HOME/pg4appc/admin directory. This script compiles the GPGLOCAL package body.

Optional Configuration Steps to Permit Multiple Users

The following configuration steps are optional. Perform these steps if you want to allow users other than PGAADMIN to perform PG DD operations using PGAU.

11. Create public synonyms for the PG DD to allow other users to access the tables:
   a. Use SQL*Plus to connect to the Oracle Integrating Server as user SYSTEM.
b. From SQL*Plus, run the `pgddcr8s.sql` script in the `$ORACLE_HOME/pg4appc/admin` directory. This script creates public synonyms for the PG DD.

12. Create roles for accessing the PG DD, performing definitions of transactions, and generating TIP specifications. The PGAADMIN user can grant these roles to other users as necessary.

a. Use SQL*Plus to connect to the Oracle Integrating Server as user PGAADMIN.

b. From SQL*Plus, run the `pgddcr8r.sql` script in the `$ORACLE_HOME/pg4appc/admin` directory. This script creates two roles, PGDDDEF and PGDDGEN. The PGDDDEF role provides SELECT, INSERT, UPDATE, and DELETE privileges against some of the PG DD tables, and select privileges against others, and allows execution of the PGAU DEFINE, GENERATE, REDEFINE, REPORT, and UNDEFINE statements. The PGDDGEN role provides select privileges against the PG DD tables, and allows execution of the PGAU GENERATE and REPORT statements only.

13. Grant access to PGA required packages.

TIP developers require access to the following PL/SQL packages, which are shipped with the Oracle Integrating Server:

- DBMS_PIPE in the `$ORACLE_HOME/rdbms/admin` directory
- UTL_RAW in the `$ORACLE_HOME/rdbms/admin` directory
- UTL_PG in the `$ORACLE_HOME/rdbms/admin` directory

Explicit grants to execute these packages must be made to TIP developers.

These grants can be private, as in the following example:

```
$ sqlplus SYS/pw@database_specification_string
SQL> GRANT EXECUTE ON UTL_RAW TO tip_developer;
SQL> GRANT EXECUTE ON UTL_PG TO tip_developer;
SQL> GRANT EXECUTE ON DBMS_PIPE TO tip_developer;
SQL> CONNECT PGAADMIN/pw@database_specification_string
SQL> GRANT EXECUTE ON PGAADMIN.PURGE_TRACE TO tip_developer;
SQL> GRANT EXECUTE ON PGAADMIN.READ_TRACE TO tip_developer;
SQL> GRANT EXECUTE ON PGAADMIN.GPGLOCAL TO tip_developer;
SQL> exit
```

Alternatively, these grants can be public, as in the following example:
$ sqlplus SYS/pw@database_specification_string
SQL> GRANT EXECUTE ON UTL_RAW TO PUBLIC;
SQL> GRANT EXECUTE ON UTL_PG TO PUBLIC;
SQL> GRANT EXECUTE ON DBMS_PIPE to PUBLIC;
SQL> CONNECT PGAADMIN/pw@database_specification_string
SQL> GRANT EXECUTE ON PGAADMIN.PURGE_TRACE TO PUBLIC;
SQL> GRANT EXECUTE ON PGAADMIN.READ_TRACE TO PUBLIC;
SQL> GRANT EXECUTE ON PGAADMIN.GPGLOCAL TO PUBLIC;
SQL> EXIT

You can use either private or public grants. Both are sufficient for using PGA. Public grants are easier and can be performed now. If you use private grants, then they must be issued each time a new TIP developer user ID is created.

SQL scripts for performing these grants are provided in the
$ORACLE_HOME/pg4appc/admin directory. The pgddapub.sql script performs these grants for public access to the packages. The pgddadev.sql script performs the grants for private access to the packages by a single TIP developer. If you are going to use private grants, then you must run the pgddadev.sql script once for each TIP developer’s user ID:

a. Use SQL*Plus to connect to the Oracle Integrating Server as user PGAADMIN.

b. From SQL*Plus, run the appropriate script (pgddapub.sql or pgddadev.sql) from the SORACLE_HOME/pg4appc/admin directory. The script performs the necessary grants as described earlier. You are prompted for the required user IDs, passwords, and database specification strings. If you are using private grants, then repeat this step for each user ID requiring access to the packages.

14. If you are upgrading from a previous release of the gateway, and if you want to upgrade your existing TIPs with new function and maintenance, then regenerate existing TIP specifications using the PGAU GENERATE statement.
**Note:** The Procedural Gateway Administrative Utility (PGAU) has been enhanced to automatically upgrade existing PG DD entries with a new attribute when a PGAU GENERATE command is executed. To support this enhancement, add a new privilege to the PGDDGEN role. To do this - as the PGAADMIN user, use SQL*Plus to connect to the Oracle Integrating Server where the PG DD is stored. Then issue the following SQL command:

```sql
SQL> GRANT INSERT ON PGA_DATA_VALUES TO PGDDGEN;
```

**a.** Invoke PGAU in the directory path where the PGAU control files are generated and where TIPs are stored:

```
$ pgau
PGAU> CONNECT PGAADMIN/pgadmin@database_specification_string
PGAU> GENERATE tranname
PGAU> EXIT
```

For more information about the GENERATE command, refer to the PGAU GENERATE command section in Chapter 2, of the *Oracle Procedural Gateway for APPC User’s Guide*.

Note that it is not necessary to define the PG DD entries again.

**b.** Invoke SQL*Plus in the same directory path where the newly-generated TIP specifications are stored.

```
$ sqlplus tip_owner/pw@database_specification_string
SQL> @tipname.pkh
SQL> @tipname.pkb
SQL> exit
```

PGAU GENERATE produces the TIP in two output files: a specification and a body. You must compile both, first the specification and then the body.

For more information about the GENERATE command, refer to the PGAU GENERATE command section in Chapter 2, of the *Oracle Procedural Gateway for APPC User’s Guide*.
Configuration

To configure your system for Oracle Procedural Gateway for APPC, perform the following steps.

Configuring the SNA Communication Packages

Configure the SNA Communication Packages profiles for APPC connections.

Configure the profiles to define LU6.2 conversations with the OLTP. Refer to the chapter on SNA Communication Packages for your platform:

- For AIX-Based Systems, refer to Chapter 6, "Configuring the SNA Communication Package on AIX-Based Systems".
- For HP-UX, refer to Chapter 7, "Configuring the SNA Communication Package on HP-UX".
- For Solaris, refer to Chapter 8, "Configuring the SNA Communication Package on Solaris".

Configuring the Gateway

To configure the gateway, perform the following:

1. Tailor the Oracle Procedural Gateway for APPC parameters.

Parameters specific to Oracle Procedural Gateway for APPC are supplied in the gateway parameter file, `initsid.ora`, which is in the `$ORACLE_HOME/pg4appc/admin` directory. A sample gateway parameter file, `initPGA.ora` is provided in this subdirectory.

   **Note:** In the `initsid.ora` file, substitute your `pg4appc` SID name for "sid" in this file name.

The parameters fall into two categories:

- Gateway initialization parameters.

These parameters control the general operation of the gateway in the Oracle environment.
Configuring Oracle Procedural Gateway for APPC

Important: Before performing the following step, refer to Appendix A, "Gateway Initialization Parameters" for information about tailoring gateway initialization and PGA parameters. Pay special attention to the information on using the PGA_CAPABILITY parameter.

- PGA parameters.

PGA parameters control the APPC interface portion of the gateway. Use the SET gateway initialization parameter to specify PGA parameters. Oracle Corporation recommends that you group all SET commands for PGA parameters at the end of the init<sid>.ora file.

Note: Misspelled parameters are ignored.

2. Use the sample pg4hoa1.sh file to specify the name of the gateway executable (pg4asrv). The sample file resides at /oracle/pga/9.2.0/bin. Refer to the "Sample pg4hoa1.sh (Boot Parameter) File" on page A-15 for a sample of the file.

Note: In the sample pg4hoa1.sh file, you must modify the directory of the executable to your specific directory. Incorrect directories are not detected by the gateway, and you may receive error message ORA-28509.

Configuring Your Network

The gateway must be defined to the TNS listener, and a service name must be defined for accessing the gateway. To do this, perform the following steps:

1. Add an entry for the gateway to the listener.ora file:

   (SID_DESC=
    (SID_NAME=PGA)
    (ORACLE_HOME=/oracle/pga/9.2.0)
    (PROGRAM=pg4hoa1.sh)
   )

2. Add a service name for the gateway to the tnsnames.ora file on the system where your Oracle Integrating Server is located. The service name is specified in the USING parameter of the database link defined for accessing the gateway.
from the Oracle Integrating Server. For example, if you are using the IPC protocol adapter and your gateway sid is PGA, add the following entry to tnsnames.ora:

```sql
pgaipc=
   (DESCRIPTION =
      (ADDRESS = (PROTOCOL = ipc) (KEY=key))
      (CONNECT_DATA = (SID=PGA))
      (HS=))
}
```

In this example, key is the IPC key defined in the listener.ora file for the IPC protocol. You can use the IPC protocol only if the Oracle Integrating Server and the gateway are on the same system.

If you are using the TCP/IP protocol adapter, and if your gateway sid is PGA, then add the following entry to tnsnames.ora:

```sql
pgatcp=
   (DESCRIPTION =
      (ADDRESS = (PROTOCOL= TCP)(Host= gateway)(Port= port))
      (CONNECT_DATA = (SID=PGA))
      (HS=))
}
```

In this example, port is the TCP port defined in the listener.ora file for the TCP protocol, and gateway is the TCP/IP host name of the system where the gateway is located.
Configuring Oracle Procedural Gateway for APPC

**Note:** If you are installing a stand-alone gateway, then you must define Oracle Integrating Server to PGAU by adding a service name to `tnsnames.ora` on the system where your gateway resides. For example:

```plaintext
ora_server =
  (DESCRIPTION=
    (ADDRESS =
      (PROTOCOL= TCP)
      (PORT= port)
      (HOST= ora_srv)
    )
    (CONNECT_DATA= (SID= ora_server))
  )
```

In this example
- `port` is the TCP port defined in the Oracle Integrating Server `listener.ora` for the TCP protocol;
- `key` is the HOST key;
- `ora_srv` is the TCP/IP host name of the system where the Oracle Integrating Server resides;
- `ora_server` is the SID of the Oracle Integrating Server.

Refer to *Oracle Net Administrator’s Guide* for more information about configuring the network.

**Configuring Commit-Confirm**

If you plan to implement commit-confirm, then refer to "Implementing Commit-Confirm" in the *Oracle Procedural Gateway for APPC User’s Guide* for detailed information.

**AIX-Based Systems Only: Two-Phase Commit**

For information on two-phase commit on AIX-Based Systems, refer to Chapter 9, "AIX-Based Systems Only: Implementing Two-Phase Commit".
Configuring the OLTP

The steps for configuring your OLTP to communicate with Oracle Procedural Gateway for APPC vary, depending on which OLTP you are using and on which platform the OLTP is running. CICS/ESA, IMS/TM, APPC/MVS, and IDMS-DC MVS are the currently supported OLTPs.

**Note:** You do not need to perform the configuration steps for an OLTP if this is not a first-time installation for that OLTP.

### Configuring CICS/ESA

If your OLTP is CICS/ESA, then perform the following steps to configure CICS and MVS for communication with the gateway:

1. Configure MVS VTAM for the SNA communication packages APPC connection to your system. At least one independent LU must be available to gateway.

2. Check the VTAM logmode table used by CICS. (The table name is specified in the `MODETAB` parameter in the VTAM APPL definition for CICS.) Ensure that an entry exists for APPC sessions with parallel session and sync-level support. The `orapl62.asm` file in the `$ORACLE_HOME/pg4appc/sna` directory contains a sample mode entry, including comments that indicate the required values in the mode entry.

3. Using your file transfer facility, transfer the following files from the `$ORACLE_HOME/pg4appc/demo/CICS` directory to the MVS system on which you run CICS:
   - `dfhcsdup.jcl` - JCL to run the CICS DFHCSDUP utility
   - `pgaflip.asm` - assembler source for the CICS FLIP transaction
   - `pgaflip.jcl` - JCL to assemble and linkedit the CICS FLIP transaction

4. Using the comments in the `dfhcsdup.jcl` file, tailor the JCL and input statements to match your system setup, and submit it for batch execution. Performing this step updates your CICS system definitions.

**Note:** AIX-Based Systems Only:

If you plan to use two-phase commit, modify the definition of the PGASESS session profile to specify `MODENAME(ORASYN62)` if you use the distributed sample mode entry.
5. Using the instructions in the `pgaflip.jcl` file comments, tailor the JCL to match your system setup, and submit it for batch execution. Performing this step assembles and linkedit the `pgaflip.asm` file into a load module library accessible to your CICS system through the DFHRPL DD statement in the CICS startup procedure.

6. Log on to your CICS system and enter the following transaction:

   `CEDA INSTALL GROUP(ORAPGA)`

   This transaction installs the CICS connection and session definitions for APPC communication with the gateway on UNIX. It also installs definitions for the sample CICS programs and transactions provided with the gateway.

Your CICS configuration is now complete.

**Configuring IDMS-DC MVS**

If your OLTP is IDMS-DC MVS, perform the following steps to configure IDMS-DC and MVS for communication with the gateway:

1. Ensure that your IDMS-DC system is at release 12 or higher.

2. Configure MVS VTAM for the communication interface APPC connection to UNIX. At least one independent LU must be available for use by the gateway.

3. If your IDMS-DC system does not have APPC support, then set up a separate MVS VTAM APPL definition for use by the IDMS-DC LU6.2 interface. IDMS-DC cannot use the same VTAM APPL for both VTAM terminal and VTAM APPC support.

   For more information, refer to vendor documentation.

4. Check the VTAM logmode table used by IDMS-DC. (The table name is specified in the MODETAB parameter in the VTAM APPL definition for IDMS-DC.) Ensure that an entry exists for APPC sessions with parallel session and sync level support. The `orapl62.asm` file in the `$ORACLE_HOME/pg4appc/sna` directory contains a sample mode entry, including comments that indicate the required values in the mode entry.

5. Modify the Mode Profile to be used for your IDMS-DC connections.

   To bypass a bug in IDMS-DC, set the "Auto ACTIVATIONS limit" field to the same value as the "maximum number of SESSIONS" field. Without this setting, IDMS-DC rejects the first data packet received from UNIX on each conversation, making it impossible for the gateway to communicate with an IDMS-DC transaction.
6. Using **CA-IDMS System Generation** as a guide, perform the following IDMS tasks:

   a. Enable the IDMS-DC multiple session service manager, RHDCCNOS.

   b. Define a VTAM line for APPC use, if one is not already defined.

   c. Define PTERMs and LTERMs for communications with the UNIX independent LU, as defined to VTAM on MVS.

      Set up two PTERM/LTERM definitions to use the SNASVCMG mode entry for communications between the SNA Service Managers on MVS and UNIX. Set up one of these definitions as a contention winner and set up the other one as a contention loser. Set up additional PTERM/LTERM definitions to use the mode entry defined in the SNA communication packages Profile DEFINE/MODE entry for use by the gateway sessions. One PTERM/LTERM definition is required for each concurrent session with the gateway. Set up these definitions with contention off.

      In the `$ORACLE_HOME/pg4appc/demo/IDMS` directory, the `appcdef.doc` file provides sample IDMS-DC definitions for a VTAM line with PTERMs and LTERMs.

   d. Define the IVP and sample programs and transactions to IDMS-DC.

      In the `$ORACLE_HOME/pg4appc/demo/IDMS` directory, the `trandef.doc` file provides sample IDMS-DC definitions for the IVP and sample programs and transactions. These definitions should not be modified.
7. Using your file transfer facility, transfer the following files from the 
   SORACLE_HOME/pg4appc/demo/IDMS directory to the MVS system on 
   which you run IDMS-DC:
   - pgaflip.asm - the assembler source for the IDMS-DC FLIP program
   - pgaflip.jcl - JCL to assemble and linkedit the IDMS-DC FLIP program

8. Using the comments in the pgaflip.jcl file, tailor the JCL to match your system 
   setup and submit it for batch execution. Performing this step assembles and 
   linkedit the pgaflip.asm file into a load module library accessible to your 
   IDMS-DC system through the CDMSLIB DD statement in the IDMS-DC startup 
   procedure.

The IDMS-DC configuration is now complete.

Configuring IMS/TM

If your OLTP is IMS/TM, then perform the following steps to configure IMS/TM 
and MVS for communication with the gateway:

1. Configure your IMS system for the APPC.

2. Configure MVS VTAM for the SNA APPC connection to UNIX. At least one 
   independent LU must be available for use by the gateway, unless you are using 
   the IMS LU6.1 Adapter for LU6.2 applications. In this case, you must have one 
   dependent LU defined for each concurrent session. For example, if you want to 
   support 10 concurrent sessions, then you must have 10 dependent LUs defined.

3. Check the VTAM logmode table used by IMS/TM. The table name is specified 
   by the MODETAB parameter in the VTAM APPL definition.

   For APPC/IMS, ensure that an entry exists for APPC sessions with sync-level 
   support and parallel session support. The oralu62.asm and oraplu62.asm files 
   in the SORACLE_HOME/pg4appc/sna directory contain sample mode entries 
   for single session and parallel session support, respectively. The samples 
   include comments that indicate the required values in the mode entries.
4. Using your file transfer facility, transfer the following files from the
SORACLE_HOME/pg4appc/demo/IMS directory to the MVS system on which
you run IMS/TM:
   - pgaflip.asm is assembler source for IMS FLIP transaction;
   - pgaflip.jcl is JCL to assemble and linkedit IMS FLIP transaction;
   - imsgen.asm is IMS stage 1 gen definitions for the IMS FLIP transaction.
5. Add the statements in the imsgen.asm file to your IMS stage 1 gen and run
your IMS stage 1 and stage 2 gens. Use the online change utility to enable the
new transaction definition.
6. Using the comments in the pgaflip.jcl file, tailor the JCL to match your system
setup and submit it for batch execution. This assembles and linkedit the
pgaflip.asm file into a load module library that is accessible to your IMS/TM
system and creates a PSB and an ACB for the FLIP transaction.
7. Perform the tasks necessary on your system to make the new transaction
available to IMS/TM. Depending on your system setup, you might have to
restart IMS.

The IMS/TM configuration is now complete.

Configuring APPC/MVS
If your OLTP is APPC/MVS, then perform the following steps to configure
APPC/MVS for communication with the gateway:
1. Configure MVS VTAM for the SNA APPC connection to UNIX. At least one
independent LU must be available for use by the gateway.
2. Check the VTAM logmode table used by APPC/MVS. (The table name is
specified by the MODETAB parameter in the VTAM APPL definition for
APPC/MVS.) Ensure that an entry exists for APPC sessions with sync level and
parallel session support. The oraplu62.asm file in the
SORACLE_HOME/pg4appc/sna directory contains a sample mode entry,
including comments that indicate the required values in the mode entry.
3. Allocate a partitioned dataset (PDS) on your MVS system where the sample
files are placed. The PDS should be allocated with RECFM=FB, LRECL=80, and
a BLKSIZE appropriate for the device type on which it resides. Approximately
two tracks of 3390 disk space are required with one directory block. Oracle
Corporation suggests naming this partitioned dataset (PDS)
ORAPGA.APPCMVS.SAMPLIB.
4. Using your file transfer facility, transfer the following files from the
   SORACLE_HOME/pg4appc/demo/MVS directory to the MVS PDS you
   allocated in the previous step, using the following specified member names:
   - **pgaflip.jcl** is JCL to add an APPC/MVS TP profile and to define the
     execution environment for the transaction. Store this file in your MVS PDS
     as member PGAFLIPJ.
   - **pgaflip.rex** is the REXX source for the APPC/MVS PGAFLIP transaction.
     Store this file in your MVS PDS as member PGAFLIP.

5. Using the comments in the **pgaflip.jcl** file, tailor the JCL to match your system
   setup and submit it for batch execution. Performing this step defines the
   APPC/MVS TP profile for the PGAFLIP transaction and stores it in the
   APPC/MVS profile dataset. Ensure that you change the dataset name in the
   JCL to match the name of the MVS PDS allocated in Step 3.

The APPC/MVS configuration is now complete.

### Verifying the Installation

To verify the gateway installation and the OLTP configuration, perform the
following procedures after installing Oracle Procedural Gateway for APPC.

### Verifying the Gateway Installation

**Note:** If your database link name is not "PGA," modify
demonstration .sql files to give them the particular database link
name that you created in Step 5 of “Configuring the Oracle
Integrating Server for First-Time Installations” on page 4-15. You
must modify the following .sql files:
- **pgavsn.sql**
- **pgaecho.sql**
- **pgacics.sql**
- **pgaidms.sql**
- **pgaims.sql**
- **pgamvs.sql**
To verify the gateway software installation using the database link PGA previously created, perform the following steps:

1. Using SQL*Plus, connect to your Oracle Integrating Server from the client system as user PGAADMIN.

2. Run `$ORACLE_HOME/pg4appc/demo/pgavsn.sql`.
   The server version number banner appears at your terminal.

3. Run `$ORACLE_HOME/pg4appc/demo/pgaecho.sql`.
   The following message appears:
   ```
   ==> Congratulations, your installation was successful. <==
   ```

**Verifying the OLTP Configuration**

The procedure for verifying your OLTP configuration varies, depending on which OLTP you are using and depending on which platform the OLTP is running. CICS/ESA, IMS/TM, APPC/MVS, and IDMS-DC MVS are the currently supported OLTPs.

**CICS Verification**

If your OLTP is CICS/ESA, perform the following steps to verify the CICS configuration:

1. To verify that the FLIP transaction is installed correctly, log on to your CICS system and enter the following transaction, replacing `flip` with the transaction ID you chose for FLIP when you configured your CICS system for the gateway.

   ```
   flip THIS MESSAGE
   ```

   The following output appears at your terminal:

   ```
   EGASSEM SIHT pilf
   ```

2. Log on to UNIX and change directory to `$ORACLE_HOME/pg4appc/demo/CICS` directory.

3. Using your file transfer facility, transfer the `pgacics.sql` file to the client system from which you access the Oracle Integrating Server. If the clients are local to the Oracle Integrating Server, then this step is not necessary.
4. Modify the `pgacics.sql` file. Customize the following three items used for accessing the gateway and the CICS system, as described in the comments at the beginning of the file:
   - the CICS transaction ID
   - the side profile name
   - the logmode entry name

5. Ensure that the SNA communication package on your system has been started.

6. Log on to your CICS system and enter this transaction, where `name` is the name of the CONNECTION definition installed by the DFHCSDUP job you ran in the CICS configuration steps:
   ```
   CEMT SET CONNECTION (name) ACQUIRED
   ```
   This transaction activates the CICS connection to UNIX.

7. Using SQL*Plus, connect to your Oracle Integrating Server from the client system.

8. Run `pgacics.sql`.
   The following message appears:
   ```
   ==> Congratulations, your gateway is communicating with CICS ===
   ```
   Your CICS installation verification is complete.

### IDMS-DC Verification

If your OLTP is IDMS-DC MVS, perform the following steps to verify the IDMS-DC configuration:

1. Log on to UNIX and change directory to $ORACLE_HOME/pg4appc/demo/IDMS.

2. Using your file transfer facility, transfer the `pgaidms.sql` file to the client system from which you access the gateway. If the clients are local to the Oracle Integrating Server, then this step is not necessary.

3. Modify the `pgaidms.sql` file. Customize the following three items used for accessing the gateway and the IDMS-DC system, as described in the comments at the beginning of the file:
   - the IDMS-DC transaction ID
Verifying the Installation

Installing and Configuring the Gateway 4-35

4. Check that the SNA communication package on your system has been started.

5. Using the IDMS-DC DCMT transaction, display the LU6.2 line to check that
sessions have been started with your system. Enter the following command,
where linename is the name of the LINE defined for LU6.2 communications
with your system:

   DCMT DIS LINE linename

   Each defined L-TERM/P-TERM should show a status of INSRV. Any other
   status indicates a problem with the IDMS-DC APPC interface

6. Using SQL*Plus, connect to your Oracle Integrating Server.

7. Run pgaidms.sql.

   The following message appears:

   ==> Congratulations, your gateway is communicating with IDMS-DC<==

Your IDMS-DC installation verification is now complete.

IMS/TM Verification

If your OLTP is IMS/TM, perform the following steps to verify the IMS/TM
configuration:

1. To verify that the FLIP transaction is installed correctly, log on to your IMS/TM
system and enter the following transaction (replacing flip with the transaction
ID you chose for FLIP when you configured your IMS/TM system for the
gateway):

   flip THIS MESSAGE

   The following output appears on your terminal:

   EGASSEM SIHT

2. Log on to UNIX and change directory to
   $ORACLE_HOME/pg4appc/demo/IMS.

3. Using your file transfer facility, transfer the pgaims.sql file to the client system
   from which you access the gateway. If the clients are local to the Oracle
   Integrating Server, then this step is not necessary.
4. Modify the **pgaims.sql** file. Customize the following three items used for accessing the gateway and the IMS/TM system, as described in the comments at the beginning of the file:
   - the IMS/TM transaction ID
   - the side profile name
   - the logmode entry name

5. Ensure that the SNA communication package on your system has been started.

6. Using SQL*Plus, connect to your Oracle Integrating Server from the client system.

7. Run *pgaims.sql*.

   The following message appears:

   ```
   => Congratulations, your gateway is communicating with IMS/TM <==
   ```

Your IMS/TM installation verification is now complete.

**APPC/MVS Verification**

If your OLTP is APPC/MVS, perform the following steps to verify the APPC/MVS configuration:

1. Verify that your APPC/MVS subsystem is active.

2. Log on to the UNIX system where the gateway is installed and change directory to `$ORACLE_HOME/pg4appc/demo/MVS`.

3. Using your file transfer facility, transfer the **pgamvs.sql** file to the client system from which you access the gateway. If the clients are local to the Oracle Integrating Server, then you do not need to do this.

4. Modify the **pgamvs.sql** file. Customize the following three items used for accessing the gateway and the APPC/MVS system, as described in the comments at the beginning of the file:
   - the APPC/MVS transaction ID
   - the side profile name
   - the logmode entry name

5. Ensure that the SNA communication package on your system has been started.
6. Using SQL*Plus, connect to your Oracle Integrating Server from the client system.

7. Run `pgamvs.sql`.
   The following message appears:
   
   => Congratulations, your gateway is communicating with APPC/MVS <=

Your APPC/MVS installation verification is now complete.

**Performing Postinstallation Procedures**

The following are optional steps that you can perform as necessary. Installation of the sample applications for your OLTP is recommended to help you to fully understand how the gateway works and how it interfaces with your OLTP.

**Installing Sample Applications**

Your Oracle Procedural Gateway for APPC package contains sample PL/SQL procedures and OLTP transaction programs that demonstrate the capabilities of Oracle Procedural Gateway for APPC. Samples are provided for the following:

**APPC/MVS**
- list MVS dataset information

**CICS/ESA**
- ADABAS inquiry
- DB2 inquiry
- DB2 multi-row inquiry
- DB2 update
- VSAM inquiry
- VSAM update
- DLI inquiry
- FEPI DB2 inquiry
- FEPI VSAM inquiry
- IDMS-DC MVS
Performing Postinstallation Procedures

- IDMS/R inquiry
- IMS/TM
- IMS inquiry using IVTNO and IVTNV sample transactions
- IMS PARTS inquiry (CPI-C)
- IMS PARTS update (CPI-C)

Additional samples are added to the distribution media in later releases of the product. Wherever possible, the sample applications use the sample databases provided with the database products.

For this release, full documentation on installing and using the sample applications is available in the README.doc files in the following directories:

- $ORACLE_HOME/pg4appc/demo
- $ORACLE_HOME/pg4appc/CICS
- $ORACLE_HOME/pg4appc/IMS
- $ORACLE_HOME/pg4appc/MVS
- $ORACLE_HOME/pg4appc/IDMS

Accessing the Gateway from Other Oracle Servers

To access Oracle Procedural Gateway for APPC from other Oracle servers, perform the following steps:

1. Set up the Oracle Integrating Server with local or Oracle Net access to Oracle Procedural Gateway for APPC.

2. Create a database link from the Oracle Integrating Server to the gateway, as described in "Configuring the Oracle Integrating Server for First-Time Installations" on page 4-15.

3. Ensure that the UTL_RAW, UTL_PG, and DBMS_OUTPUT PL/SQL packages are installed on the Oracle Integrating Server.
Accessing Other OLTPs from the Oracle Integrating Server

To access other OLTPs from the Oracle Integrating Server, perform the following steps:

1. Configure the SNA communication package for the OLTP. If the same physical connection is used, only side information and partner LU information must be new. For more information:
   - For AIX-Based Systems: refer to Chapter 6, "Configuring the SNA Communication Package on AIX-Based Systems".
   - For HP-UX, refer to Chapter 7, "Configuring the SNA Communication Package on HP-UX".
   - For Solaris, refer to Chapter 8, "Configuring the SNA Communication Package on Solaris".

2. Configure the additional OLTP, as described in "Configuring the OLTP" on page 4-27.

Deinstalling Your Oracle Procedural Gateway for APPC

If you decide to deinstall Oracle Procedural Gateway for APPC, perform the following steps:

1. Drop the PG DD (packages and procedures):
   a. Use SQL*Plus to connect to the Oracle Integrating Server as user PGAADMIN.
   b. From SQL*Plus, run the pgdddel.sql script in the $ORACLE_HOME/pg4appc/admin directory.

2. Log in as DBA. Refer to "Step 1: Login as DBA and Create Login User ID" on page 4-6 for more information.

3. Set the ORACLE_HOME environment variable. Refer to "Step 3: Set the ORACLE_HOME Environment Variable" on page 4-7 for more information.

4. Verify the DISPLAY and ORACLE_HOME environment variables. Refer to "Step 4: Set the DISPLAY Environment Variable" on page 4-7 for more information.

5. Mount the CD-ROM. Refer to "Step 5: Mount the CD-ROM" on page 4-8 for more information.

6. Start the Oracle Universal Installer by running the following command:
Deinstalling Your Oracle Procedural Gateway for APPC

$ ./runInstaller

7. Step through the Oracle Universal Installer. Use the prompts listed in the following table as a guide for deinstallation, following the instructions in the Response column.

**Table 4–3  Steps to DeInstall the Gateway Using Oracle Universal Installer**

<table>
<thead>
<tr>
<th>Prompt</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Oracle Universal Installer</td>
<td>Click &quot;Deinstall Products ...&quot;.</td>
</tr>
<tr>
<td>2. Inventory</td>
<td>Check &quot;Oracle Procedural Gateway for APPC 9.2.0.1.0&quot; and click &quot;Remove&quot;.</td>
</tr>
<tr>
<td>3. Confirmation</td>
<td>Click &quot;Yes&quot;.</td>
</tr>
<tr>
<td>4. Inventory</td>
<td>Click &quot;Close&quot;.</td>
</tr>
<tr>
<td>5. Oracle Universal Installer</td>
<td>Click Exit&quot;.</td>
</tr>
<tr>
<td>6. Exit</td>
<td>Click &quot;Yes&quot;.</td>
</tr>
</tbody>
</table>

8. Oracle Procedural Gateway for APPC is now deinstalled.

When the Oracle Universal Installer confirms that the deinstallation has ended, verify that the deinstallation procedure was successful. To do this, read the contents of the deinstallation log file, which is located in the $ORACLE_HOME/install directory. The default file name is install.log.

9. The only files that are removed are those that were copied to the $ORACLE_HOME directory during the installation of Oracle Procedural Gateway for APPC. You must remove any other related files manually, including deleting listener.ora and tnsnames.ora entries relating to the gateway, dropping database link(s) and the PGAADMIN user ID and deleting the TIPs.
The gateway architecture involves multiple systems, database servers, and communications facilities, each having distinct security capabilities and limitations. To effectively plan and implement your security scheme, you must understand these capabilities and limitations, in addition to knowing your installation’s security requirements.

Read this chapter to learn about the capabilities and limitations of Oracle Procedural Gateway for APPC. This chapter contains the following sections:

- **Overview of Security Requirements** on page 5-2
- **Authenticating Application Logons** on page 5-3
- **Defining and Controlling Database Links** on page 5-4
- **Passwords in the Gateway Initialization File** on page 5-5
- **AIX-Based Systems Only: Using the pg4rrmpwd Utility** on page 5-6
Overview of Security Requirements

Before implementing your security scheme, you must understand the existing security requirements and expectations in your environment. Because you are enabling application access to different databases on different systems, you must merge multiple security cultures. When developing your security scheme, the most stringent security requirements prevail. When you connect several different systems into an operating whole, the system with the strictest security requirements generally dictates what the other systems can and cannot do.

Gateway security includes two main concerns:

- users and applications that are permitted access to a particular gateway instance and OLTP (online transaction processor)
- OLTP transactions that users and applications are able to execute

You can control access at several points in the gateway architecture. The primary options are discussed in the following sections. Control over remote transaction program access is provided by each OLTP with native authorization mechanisms based on user ID. These facilities are described in the product documentation for your OLTP. Information in this chapter includes how the gateway facilities determine the user ID that is in effect for a particular OLTP connection.

When the gateway is involved in an RPC request, security mechanisms are in effect for each system component encountered by the gateway. The first system component that is encountered is the application tool or 3GL program. The last system component that is encountered is the OLTP.

Each of the following sections identifies the component and the type of security processing that is available in that component. Each section offers a summary of key features and parameters. Refer to product-specific documentation for detailed information about the non-gateway components for Oracle and non-Oracle products.
Authenticating Application Logons

An application must connect to an Oracle Integrating Server before using Oracle Procedural Gateway for APPC. The type of logon authentication that you use determines the resulting Oracle user ID and can affect gateway operation.

Two basic types of authentication are available:

1. Oracle authentication.

   With Oracle authentication, each Oracle user ID has an associated password that is known to Oracle. When an application connects to the server, it supplies a user ID and password. Oracle confirms that the user ID exists and that the password matches the one stored in the database.

2. Operating system authentication.

   With operating system authentication, the server’s underlying operating system is responsible for authentication. An Oracle user ID that is created with the IDENTIFIED EXTERNALLY attribute (instead of a password) is accessed with operating system authentication. To log on to such a user ID, the application supplies a forward slash (/) for a user ID and does not supply a password.

   To perform operating system authentication, the server determines the requester’s operating system user ID, optionally adds a fixed prefix to it, and uses the result as the Oracle user ID. The server confirms that the user ID exists and is IDENTIFIED EXTERNALLY, but no password checking is done. The underlying assumption is that users were authenticated when they logged on to the operating system.

   Operating system authentication is not available on all platforms and is not available in some Oracle Net (client-server) and multi-threaded server configurations. Refer to your platform-specific Oracle server documentation and Oracle Net Administrator’s Guide to determine the availability of this feature in your configuration.

For more information about authenticating application log ons, see the Oracle9i Administrator’s Guide.
Defining and Controlling Database Links

The following sections discuss database links.

Link Accessibility

The first point of control for a database link is simply whether it is accessible to a given user. A public database link can be used by any user ID. A private database link can be used only by the user who created it. Database link usability is determined by its ability to open a session to the gateway. The Oracle Integrating Server makes no distinction as to the type of use (such as read-only versus update or write) or which remote objects can be accessed. These distinctions are the responsibility of the OLTP that is accessed.

Links and CONNECT Clauses

The CONNECT clause is another security-related attribute of a database link. You can use the CONNECT clause to specify an explicit user ID and password, which can differ from the user’s Oracle user ID and password. This CONNECT user ID and password combination is sent to the gateway when the database link connection is first opened. Depending on gateway-specific options, the gateway might send that user ID and password to the OLTP to be validated.

If a database link is created without a CONNECT clause using Oracle authentication, then the user’s Oracle user ID and password are sent to the gateway when the connection is opened. If the user logs on to the Oracle Integrating Server with operating system authentication, then the gateway receives no user ID or password from the Oracle Integrating Server. It is impossible for operating-system-authenticated Oracle users to use a gateway database link defined without a CONNECT clause. However, if your OLTP provides user ID mapping facilities based on the gateway LU name from which the user is connecting, then such a connection is possible if all users on the same gateway instance can use the same OLTP user ID.

For more information about database links, see the Oracle9i Administrator’s Guide.
Passwords in the Gateway Initialization File

Oracle Procedural Gateway for APPC uses user IDs and passwords to access the information on the remote database on the gateway server. For functions to be handled correctly, some user IDs and passwords must be defined in the gateway initialization file. (For an example, refer to the PGA_LOG_PASS parameter in Appendix A.) Because it is not secure to have plain-text passwords accessible in the initialization file, a new encryption feature, pg4pwd, has been added to the gateway. With this feature, passwords are no longer stored in the initialization file but are stored instead in an encrypted form in the password file, making the information more secure. The pg4pwd utility is an optional feature, but Oracle Corporation strongly recommends that you use it. The following section describes how to use it.

Using the pg4pwd Utility

The pg4pwd utility encrypts passwords that are normally stored in the gateway initialization file. The pg4pwd utility searches the initialization file for parameters with an asterisk, *. The asterisk denotes that the parameter value is stored in encrypted form in another file. The following is a sample section of the initialization file with this value:

```
SET PRIVATE PGA_LOG_PASS=* 
```

To use the pg4pwd utility:

1. Edit the initialization file to set the parameter value to *.
2. Run the pg4pwd utility, specifying the gateway SID on the command line. The utility reads the initialization file, and prompts you to enter the values to be encrypted.

   The syntax of the command, where gateway_sid is the SID of the gateway, is:

   `$ pg4pwd gateway_sid`

   For example, if the gateway SID is PGA, enter:
   
   `$ pg4pwd PGA`

   ORACLE Gateway Password Utility (pg4appc)
   Constructing password file for Gateway SID PGA
   Enter the value for PGA_LOG_PASS
   pgaadmin

   In the preceding example, the PGA_LOG_PASS parameter is identified as requiring encryption. The user enters the value, pgaadmin, and presses enter. If there are
more parameters requiring encryption, they are prompted for in turn. The encrypted data is stored in the $ORACLE_HOME/pg4appc/admin/init.sid.pwd file.

---

**Note:** It is important that the ORACLE_HOME environment variable points to the gateway Oracle home directory to ensure that the correct gateway initialization file is read.

---

### AIX-Based Systems Only: Using the pg4rrmpwd Utility

The pg4rrmpwd utility encrypts the password, LOG_PASS, which is normally stored in the initialization file for the local LU. The utility searches the local LU initialization file for the LOG_PASS parameter with an asterisk, *. The asterisk denotes that this parameter value is stored in encrypted form in another file. A sample section of the initialization file for the local LU with this value is as follows:

```plaintext
LOG_PASS=*  
```

To use the pg4rrmpwd utility:

1. Edit the initialization file for the local LU by setting the LOG_PASS parameter value to *.

2. Run the pg4rrmpwd utility, specifying the initialization file for local LU on the command line. The pg4rrmpwd utility reads the initialization file, and prompts you to enter the passwords to be encrypted.

The syntax of the command, where `LU_NAME.ini` is the initialization file for local LU, is:

```
$ pg4rrmpwd LU_NAME.ini  
```

For example:

```
$ pg4rrmpwd HQWM920.ini  
```

ORACLE Gateway Password Utility PG4ARRM 9.2.0.1.0
Constructing password file for LOG_PASS
Enter the value for LOG_PASS
pgaadmin

In the preceding example, the LOG_PASS parameter is identified as requiring encryption. The user enters the value, pgaadmin, and presses enter. The encrypted data is stored in the $ORACLE_HOME/pg4appc/admin directory.
Note: It is important that the ORACLE_HOME environment variable points to the gateway Oracle home directory to ensure that the correct gateway initialization file is read.
Oracle Procedural Gateway for APPC uses the SNA Advanced Program to Program Communication (APPC/LU6.2) protocol to communicate with an OLTP. AIX system support for APPC is provided by IBM’s SNA server product. This product requires a stored set of definitions, called profiles, to support connections between the gateway and OLTPs (online transaction processors). Each profile consists of a profile name and a profile type and set of fields describing the profile. The fields in a profile type are generally a mixture of operating parameter values and names of other SNA profiles relevant to the profile.

Read this chapter to learn about creating and activating SNA server profiles. This chapter contains the following sections:

- Using SNA Security Validation on page 6-2
- Specifying SNA Conversation Security on page 6-2
- Processing Inbound Connections on page 6-4
- Independent Versus Dependent LUs on page 6-4
- Creating SNA Profiles for the Gateway on page 6-5
- Profile Types on page 6-5
- SNA Server Profiles on page 6-5
- Activating Profiles on page 6-12
Using SNA Security Validation

When an RPC request to start a remote transaction program is received by the gateway, the gateway attempts to start an APPC conversation with the OLTP. Before the conversation can begin, a session must start between the AIX Logical Unit (LU) and the OLTP LU.

SNA and its various access method implementations (including SNA server and VTAM) provide security validation at session initiation time, allowing each LU to authenticate its partner. This validation is carried out entirely by network software before the gateway and OLTP application programs begin their conversation and process conversation-level security data. If session-level security is used, then correct password information must be established in the AIX SNA profiles and in similar parameter structures in the OLTP to be accessed. Refer to the appropriate communications software product documentation for detailed information about this subject.

Specifying SNA Conversation Security

The PGA_SECURITY_TYPE parameter of the gateway initialization file allows you to specify either of three options that determine the security conduct of the LU6.2 conversation that is allocated with the OLTP. These options are part of the SNA LU6.2 architecture, but their precise behavior might vary depending on the particular OLTP system.

SNA Security Option SECURITY=NONE

If you specify PGA_SECURITY_TYPE=NONE, then the gateway performs no processing of the client user ID and password. The conversation is allocated with SNA option SECURITY=NONE.

SNA Security Option SECURITY=PROGRAM

If you specify PGA_SECURITY_TYPE=PROGRAM, then the gateway allocates the conversation with SNA option SECURITY=PROGRAM, and the following information is sent to the OLTP:

- If the TIP user ID and password overrides are used, then the specified user ID and password are sent regardless of the database link specification.
- If the database link has explicit CONNECT information, then the specified user ID and password are sent.
If the database link has no CONNECT clause, and if the application logged on to Oracle with an explicit user ID and password, then the Oracle user ID and password are sent.

If the application logs on to Oracle with operating system authentication, and if the database link lacks explicit CONNECT information, then no user ID and password are sent. If no user ID and password are sent, and if the OLTP is not configured to assign a default user ID, then the connection fails.

In general, SNA option SECURITY=PROGRAM tells the OLTP to authenticate the user ID/password combination using whatever authentication mechanisms are available. For example, if CICS/ESA is the OLTP, then RACF can be used. This is not always the case, however, because each OLTP can be configured to process inbound user IDs in other ways.

**SNA Security Option SECURITY=SAME**

If you specify PGA_SECURITY_TYPE=SAME, the gateway allocates the conversation with SNA option SECURITY=SAME and sends only a user ID, without a password, to the OLTP. In this case, SNA server sends the owning user ID of the gateway server executable, $ORACLE_HOME/bin/pg4asrv, as the user ID. The user ID that is sent is not the Oracle user ID. This user ID can be viewed with the UNIX \texttt{ls} command and can be changed by an authorized user with the \texttt{chown} command. Because this user ID is the same for all users of a given gateway instance, this option is of limited use.

**Note:** The user ID sent is not translated to uppercase by the SNA server. If your OLTP is running on a system which does not allow lowercase user IDs (MVS, for example), you must set up an uppercase user ID on AIX to be the owner of the gateway executable file.

SECURITY=SAME is similar to the AIX operating system authentication. It tells the OLTP that the user has already been authenticated at the originating side of the conversation. There might be configuration parameters or options on the server side that affect whether SECURITY=SAME conversations are accepted. When properly configured, the OLTP only confirms that the user ID itself is valid and then accepts the connection. As with SECURITY=PROGRAM, you can change this using configuration options in many OLTPs.
Processing Inbound Connections

Many OLTPs provide options for manipulating the security conduct of an inbound (client) APPC session request. Refer to the appropriate OLTP documentation for detailed information about this topic.

Note that for CICS, one security option is not supported by the gateway.

ATTACHSEC=PERSISTENT, specified on the CICS CONNECTION definition, requires capability that is not yet available in the gateway.

ATTACHSEC=LOCAL, ATTACHSEC=IDENTIFY, ATTACHSEC=VERIFY, and ATTACHSEC=MIXIDPE are fully supported by the gateway.

Independent Versus Dependent LUs

Oracle Corporation recommends independent LUs for Oracle Procedural Gateway for APPC because they support multiple parallel sessions or conversations. This means that multiple Oracle client applications can be active simultaneously with the same OLTP through the independent LU.

Dependent LUs support only a single active session. The CP (SNA server for AIX, in this case) queues additional conversation requests from the Procedural Gateway server behind an already active conversation. In other words, conversations are single-threaded for dependent LUs.

If a dependent LU is correctly defined, then no alterations to Oracle Procedural Gateway for APPC configuration are needed, nor should any changes be needed to the host transaction or how the OLTP is started.

The operational impact of dependent LUs is that the first client application can initiate a conversation through the Procedural Gateway with the OLTP. While that transaction is active (which could be seconds to minutes to hours depending on how the client application and transaction are designed), any other client application initiating a conversation with the same OLTP instance appears to hang as it waits behind the previous conversation.

If a production application really uses only a single conversation or transaction at any one time, then there should be no impact.

However, additional concurrent conversations or transactions might be required for testing or for other application development. Each requires that additional dependent LUs be defined on the remote host, plus additional SNA server profiles, which define the additional dependent LUs on the RS/6000 workstation. The TIP that initiates the conversation must specify the different Partner LU through a

Creating SNA Profiles for the Gateway

You can create and modify SNA server profile definitions using menus in the AIX System Management Interface Tool (smit).

Maintenance of SNA server profiles is normally done by a user with root access. The following information is intended for the person creating profiles for the gateway. You should have some knowledge of SNA before reading this section.

By using smit, you should be able to accept most of the defaults. The default values assigned to many of the fields in a new set of profiles are acceptable for the gateway.

The $ORACLE_HOME/pg4appc/sna subdirectory contains a sample set of profiles for the gateway in the pgasna.export file.

Before building the SNA Server profiles, examine the appropriate sample export file to determine the profiles needed, their contents, and their interrelationships. The export file format is text-oriented, and each field of each profile is clearly labeled. You can print a copy of the export file to use while working with your profiles in a smit session.

Profile Types

Several types of SNA server profiles are relevant to gateway APPC/LU6.2 operation. Each profile can be created and edited using a corresponding smit menu that can be reached from the Communications Applications and Services primary menu choice.

The profiles are presented in hierarchical order. Those profile types that are lowest in the hierarchy are discussed first. This matches the logical sequence in which to create the profiles. You can use smit’s list pop-up menu to fill in profile names.

SNA Server Profiles

Refer to the appropriate vendor documentation for a complete discussion of SNA profiles. This section is an overview of SNA server profiles in relation to the Oracle Procedural Gateway for APPC.
SNA Node Profile

The SNA node profile defines miscellaneous SNA system defaults. Set the "Maximum number of sessions" and "Maximum number of conversations" fields to values large enough to handle the maximum number of concurrent gateway conversations anticipated, plus any non-gateway sessions and conversations that are in use on your system by other applications.

If you do not plan to use two-phase commit, then set the "Recovery resource manager (RRM) enabled?" field to "no", unless you already have other applications running on your AIX system that require this field to be set to "yes". For example, CICS/6000 and Encina both need this field set to "yes".

If you plan to use two-phase commit, set the "Recovery resource manager (RRM) enabled?" field to "yes". This is required to allow the gateway resource manager, pg4arrm, to execute successfully. You must change it to "no" if you do not plan to use two-phase commit and if you plan to import the sample file into SNA server.

Link Station Profile

The Link Station Profile and the related DLC profile describe and control the connection of the RS/6000 to the network. Oracle Procedural Gateway for APPC does not impose special requirements on these profiles, so details on their contents are not discussed here. The sample profile distributed in pgasna.export includes a profile created for a Token-Ring network connection.

The Link Station Profile name is specified later in the Partner LU Location Profile, if one is necessary.

Mode Profile

The Mode Profile specifies parameters that determine:

- APPC/LU6.2 parallel session limits
- send and receive pacing values
- SNA RU size
- the mode name that is sent to the OLTP at session initiation

The mode name that you specify must be defined to the OLTP communications software. Choose the mode name in addition to the other mode parameters after consulting the person responsible for configuring the OLTP communications software.
The parameters that are related to parallel session limits play a role in determining the maximum number of concurrent conversations allowed between a gateway instance and the OLTP. This equates to the maximum number of concurrently active remote transaction program invocations through the gateway instance.

The mode name, for example, ORAPLU62, is specified later in the side information profile.

---

**Note:** Do not confuse the Mode Profile name with the mode name.

---

**Local LU Profile**

The Local LU (Logical Unit) Profile describes the SNA LU through which the gateway communicates.

An LU name must be assigned to the gateway. The LU name assigned to the gateway might be required elsewhere in the SNA network. Contact the person responsible for your SNA network to determine the correct LU name to specify in the profile.

Set the "Local LU name" to the LU name assigned to the gateway.

An alias should be assigned to the LU using the "Local LU alias" field. This alias is used later in the side information profile.

Set the "Local LU is dependent" field to "no".

The Local LU Profile name is specified later in the side information profile.

---

**Note:** If you plan to use two-phase commit, then reserve the local LU exclusively for the use of the gateway.

---

If you plan to use two-phase commit, then the "Recovery resource manager (RRM) enabled?" field must be set to "yes". This is required to allow the gateway’s resource manager, *pg4arm*, to execute successfully. In the sample file *pgasna.export*, this field is set to "no". You must change it to "yes" if you plan to use two-phase commit and if you plan to import the sample file into SNA server.
Partner LU Profile

The Partner LU Profile describes the SNA LU of the OLTP system with which the gateway communicates. The name of the OLTP LU and the name of your SNA network must be specified in this profile. Contact the person responsible for your SNA network to determine the correct LU and network names.

Set the "Fully qualified partner LU name" field to the network name, followed by a period, followed by the OLTP LU name, as in "network.oltplu".

You can assign an alias to the partner LU name by setting the "Partner LU alias" to the value of your choice. This allows you to reference the partner LU without having to know the fully-qualified partner LU name and minimizes the change if the partner LU name is changed.

Set the "Parallel sessions supported?" field to "yes" unless your OLTP does not support parallel sessions.

If you plan to use SNA session or conversation security, then set the "Session security supported?" and "Conversation security supported?" fields as required. These settings require the Session Security and Conversation Security Profiles. Refer to the appropriate vendor documentation for more information.

Partner LU Location Profile

The Partner LU Location Profile is used when the target system where the Partner LU resides is not an APPN-capable node. Many mainframe systems do not have APPN capability. For example, MVS systems running VTAM versions before version 4 do not support APPN. Also, if your hardware connection is through a front-end processor running NCP versions before version 5, then APPN is not supported. In these cases, the Partner LU Location Profile can be used to specify the name of the System Services Control Point (SSCP) or Control Point (CP) which owns the network connection to the partner LU.

Set the "Fully qualified partner LU name" field to the network name, followed by a period, followed by the OLTP LU name, as in "network.oltplu".

Set the "Partner LU location method" and associated fields as required by your network configuration. If you use the "owning cp" option, then the "Fully qualified owning Control Point (CP) name" field should be set to the SSCP or CP name which owns the network connection to the partner LU. For VTAM, the SSCP name is the value of the VTAM "NETID" start parameter, usually found in VTAMLST member ATCSTR00.
Side Information Profile

The side information profile is a required profile which is used to identify target OLTP systems to be accessed through Oracle Procedural Gateway for APPC.

The side information profile identifies the following:

- the local LU alias
- the partner LU alias or fully-qualified name
- the remote transaction program name (optional)
- the mode name

Set the profile information as follows for each side information profile field:

- Set the Local LU or Control Point alias to the alias assigned to the local LU in the Local LU Profile.
- Set the Mode name to the actual mode name as specified in the Mode Profile.
- Set the Remote transaction program name (RTPN) using the actual remote TP name, or a dummy name to be overridden at execution time.
- Set the RTPN in hexadecimal? field to yes, if the remote TP name is hexadecimal.
- If there is a field for "Partner LU alias", then add the alias assigned to the partner LU in the Partner LU Profile.
- If there is a field for "Fully qualified partner LU name", then add the fully-qualified partner LU name of the partner LU.

Transaction Program Name Profile

You need the Transaction Program Name Profile only if you are implementing two-phase commit with the gateway. In that case, this profile is needed by the gateway resource manager to define some of its startup parameters. The profile name is provided to the resource manager by an initialization parameter. Set the transaction program name profile information as follows for each field:

- Set the "PIP data?" field to "no". The resource manager does not use PIP data.
- Set the "conversation type" field to "basic". The resource manager supports only basic mode.
- Set the "Sync level" field to "none/confirm". The resource manager uses confirm.
Set the "Full path to TP executable" field to "pg4arrm".

All other fields have no effect on the operation of the resource manager and can retain their default values.

Note that this profile is not used by SNA server to start the resource manager dynamically. It is used by the resource manager to provide some TP characteristics to SNA server when it starts up.

Figure 6–1, "Relationship Between SNA Profiles and Host VTAM Definitions", shows the relationship between SNA server profiles and the VTAM definitions on the host.
Figure 6–1  Relationship Between SNA Profiles and Host VTAM Definitions
Activating Profiles

After you have built all the necessary SNA Server profiles for communicating with the remote system, you must verify the profiles. Use the "Verify Configuration Profiles" option under the "Advanced Configuration" option of the smit SNA server menu. Then use smit to start the link station profile. Refer to the appropriate vendor documentation for more information about using smit to start link stations.
Oracle Procedural Gateway for APPC uses the SNA Advanced Program to Program Communication (APPC/LU6.2) protocol to communicate with an OLTP. HP-UX system support for APPC is provided by the SNAplus2 product.

Read this chapter to learn how to set up and configure SNAplus2 on a HP-UX system to run Oracle Procedural Gateway for APPC. This chapter contains the following sections:

- Using SNA Security Validation on page 7-2
- The SNAplus2 Configuration Tool on page 7-4
- Creating SNAplus2 Profiles for the Gateway on page 7-4
- Independent Versus Dependent LUs on page 7-5
- Creating SNA definitions for the Gateway on page 7-6
- Sample SNAplus2 Definitions on page 7-6
- Configuring SNAplus2 on page 7-7
- Testing the Connection on page 7-10
Using SNA Security Validation

When an RPC request to start a remote transaction program is received by the gateway, the gateway attempts to start an APPC conversation with the OLTP. Before the conversation can begin, a session must start between the HP-UX Logical Unit (LU) and the OLTP LU.

SNA and its various access method implementations (including SNAplus2 and VTAM) provide security validation at session initiation time, allowing each LU to authenticate its partner. This validation is carried out entirely by network software before the gateway and OLTP application programs begin their conversation and process conversation-level security data. If session-level security is used, then correct password information must be established in the HP-UX SNA profiles and in similar parameter structures in the OLTP to be accessed. Refer to the appropriate communications software product documentation for detailed information about this subject.

Specifying SNA Conversation Security

The PGA_SECURITY_TYPE parameter of the gateway initialization file allows you to specify either of three options that determine the security conduct of the LU6.2 conversation that is allocated with the OLTP. These options are part of the SNA LU6.2 architecture, but their precise behavior might vary depending on the particular OLTP system.

SNA Security Option SECURITY=None

If PGA_SECURITY_TYPE=None is specified, then the gateway performs no processing of the client user ID and password. The conversation is allocated with SNA option SECURITY=None.

SNA Security Option SECURITY=PROGRAM

If PGA_SECURITY_TYPE=PROGRAM is specified, then the gateway allocates the conversation with SNA option SECURITY=PROGRAM, and the following information is sent to the OLTP:

- If the TIP user ID and password overrides are used, then the specified user ID and password are sent regardless of the database link specification.
- If the database link has explicit CONNECT information, then the specified user ID and password are sent.
If the database link has no CONNECT clause, and if the application logged on to Oracle with an explicit user ID and password, then the Oracle user ID and password are sent.

If the application logs on to Oracle with operating system authentication, and if the database link lacks explicit CONNECT information, then no user ID and password are sent. If no user ID and password are sent, and if the OLTP is not configured to assign a default user ID, then the connection fails.

In general, SNA option SECURITY=PROGRAM tells the OLTP to authenticate the user ID/password combination using whatever authentication mechanisms are available. For example, if CICS/ESA is the OLTP, then RACF can be used. This is not always the case, however, because each OLTP can be configured to process inbound user IDs in other ways.

### SNA Security Option SECURITY=SAME

If PGA_SECURITY_TYPE=SAME is specified, the gateway allocates the conversation with SNA option SECURITY=SAME and sends only a user ID, without a password, to the OLTP. In this case, SNAplus2 sends the owning user ID of the gateway server executable, `SORACLE_HOME/bin/pg4asrv`, as the user ID. The user ID that is sent is not the Oracle user ID. This user ID can be viewed with the HP-UX `ls` command and can be changed by an authorized user with the `chown` command. Because this user ID is the same for all users of a given gateway instance, this option is of limited use.

**Note:** The user ID sent is not translated to uppercase by SNAplus2. If your OLTP is running on a system which does not allow lowercase user IDs (MVS, for example), you must set up an uppercase user ID on HP-UX to be the owner of the gateway executable file.

SECURITY=SAME is similar to the HP-UX operating system authentication. It tells the OLTP that the user has already been authenticated at the originating side of the conversation. There might be configuration parameters or options on the server side that affect whether SECURITY=SAME conversations are accepted. When properly configured, the OLTP only confirms that the user ID itself is valid and then accepts the connection. As with SECURITY=PROGRAM, it is possible to change this behavior through configuration options in many OLTPs.
Processing Inbound Connections

Many OLTPs provide options for manipulating the security conduct of an inbound (client) APPC session request. Refer to the appropriate documentation for your OLTP for detailed information about this topic.

Note that for CICS, one security option is not supported by the gateway.

- ATTACHSEC=PERSISTENT, specified on the CICS CONNECTION definition, requires capability that is not yet available in the gateway.
- ATTACHSEC=LOCAL, ATTACHSEC=IDENTIFY, ATTACHSEC=VERIFY, and ATTACHSEC=MIXIDPE are fully supported by the gateway.

Steps for Configuring the Communications Interfaces

1. Create SNAplus2 profiles for the gateway.
2. Create SNA definitions for the gateway.
3. Test the configuration.

The SNAplus2 Configuration Tool

All SNAplus2 product configuration is done using the xsnapadmin program. The xsnapadmin program is an X-Windows application which provides a graphical interface that you can use to view and modify the current SNAplus2 configuration and the current running state of the host SNA node. Refer to vendor for more information on using xsnapadmin.

Creating SNAplus2 Profiles for the Gateway

Oracle Procedural Gateway for APPC requires a stored set of definitions, called Side Information Profiles, to support connections between the gateway and remote server. Each profile consists of a profile name and a profile type, which is a set of fields describing the profile. The fields in a profile type are generally a mixture of operating parameter values and names of other SNA profiles relevant to the profile. Each functional part of APPC, such as Mode, Remote Transaction Program name and Logical Unit (LU), is described by a distinct profile type.
Independent Versus Dependent LUs

Oracle Corporation recommends independent LUs for the gateway because they support multiple parallel sessions or conversations. This means multiple Oracle client applications can be active simultaneously with the same OLTP through the independent LU.

Dependent LUs only support a single active session. SNAplus2 queues additional conversation requests from Oracle Procedural Gateway for APPC behind an active conversation. That is, conversations are single-threaded for dependent LUs.

If a dependent LU is correctly defined, no alterations to the gateway configuration are needed, nor should any changes be needed to the host transaction or how the OLTP is started.

The operational impact of dependent LUs is that the first client application can initiate a conversation through the gateway with the OLTP, and while that transaction is active (which could be seconds, to minutes, to hours, depending on how the client application and transaction are designed) any other client application initiating a conversation with the same OLTP instance appears to hang as it waits behind the previous conversation.

If a production application really only uses a single conversation or transaction at any one time, there should be no impact.

However, additional concurrent conversations or transactions might be required for testing or other application development. Each requires that additional dependent LUs be defined on the remote host, plus an additional SNAplus2 configuration file entry which defines the additional dependent LUs on the HP-UX workstation. The TIP which initiates the conversation must specify the different SNAplus2 Partner LU through a different Side Information Profile. Refer to "PGAU DEFINE TRANSACTION SIDEPROFILE" parameter in Chapter 2 of the Oracle Procedural Gateway for APPC User's Guide, and the SNAplus2 Symbolic Destination Name discussed in the section, "Sym Dest Name" on page 7-6.

In some uses of the gateway, independent LUs cannot be used. For example, with the IMS LU6.1 Adapter for LU6.2, parallel sessions are not supported. In this case, multiple concurrent sessions with the IMS LU6.1 Adapter for LU6.2 can be achieved by defining a pool of dependent LUs. For each dependent LU, select the "LU in the pool of default LUs" option. When a conversation is requested, an available local LU from the default LU pool is assigned automatically by SNAplus2. For more information, refer to vendor documentation.
**Sym Dest Name**

This option lets you enter the side information associated with a particular symbolic destination name. You can use an alphanumeric string up to 8 characters as the "Sym Dest Name." The symbolic destination name is referred to as the side information profile in other parts of this guide. This name is specified by the SIDEPROFILE keyword in the DEFINE TRANSACTION statement used to define your transaction to PGAU.

The "Partner TP name" field specifies the name of the transaction to be executed on the OLTP side of the conversation. This field must be specified, but the TP name can be overridden by the gateway at conversation startup.

The "Partner LU" field specifies the LU name of the OLTP on the remote system. The "Mode Name" field specifies the mode name to be used for conversations with the specified OLTP.

The security information that can be specified in this menu is not usable for the gateway. The security parameters are always set by the gateway based on gateway initialization parameters.

---

### Creating SNA definitions for the Gateway

SNAplus2 definitions are stored in two files, located in SNAplus2 /etc/opt/sna directory:

- SNA node definitions: `sna_node.cfg`
- SNA domain definitions: `sna_domn.cfg`

These files are created and maintained with the xsnapadmin tool. Maintenance of the SNA definitions is normally done by a user with administrative authority. The following information is intended for the person creating SNA definitions for the gateway. You should have some knowledge of SNA before reading this section.

---

### Sample SNAplus2 Definitions

The gateway’s `$ORACLE_HOME/pg4appc/sna` subdirectory contains a set of sample SNAplus2 definitions for the gateway, created with the xsnapadmin. SNA definitions are very specific to the HP 9000 host and SNA network; Thus, these sample definitions will not work without being tailored for the local host and SNA network.
Configuring SNAplus2

This section describes the process of creating your SNA definitions for SNAplus2 using the xsnapadmin tool. All of the tasks described in this section are performed from within xsnapadmin.

All configuration is done using the various pull-down menus and panels in xsnapadmin. The configuration descriptions in the steps below follow the samples provided. You must tailor the various SNA values for your local host and SNA network.

**Step 1 Invoking xsnapadmin**

Use the following commands to invoke xsnapadmin. The $DISPLAY environmental variable must be set appropriately. If you are running xsnapadmin from the local HP 9000 console, then $DISPLAY should already be set. If you are running xsnapadmin from a remote X display, then set $DISPLAY to the host name or IP address of that display.

```
$ DISPLAY=xstation10.us.oracle.com:0
$ export DISPLAY
$ xsnapadmin &
```

Upon startup of xsnapadmin, the main screen will open and display the current configuration of the local SNA node.

**Step 2 Configuring the SNA node**

a. On the main screen of xsnapadmin, pull down the Services menu and select Configure Node Parameters.

b. In the Node Parameters dialog box, enter the APPN support type, the Control Point Name, Control Point and Node ID as needed. The Control Point Name is composed of the SNA Network Name and the CP name of the local host.

c. Click [OK].

**Step 3 Adding a Port**

a. From the Service menu, select Connectivity and then select Add Port.

b. In the Add to <nodename> dialog box, select the Port and type you are using and click [OK].

c. In the subsequent SAP dialog box, enter a Port name and network card number. The Port name will be used to logically name the physical network.
card you are using and to bind a Service Access Port to the card for SNA protocols. Normally you can accept the values provided in the dialog box. If a different network card is needed, however, enter the card number as reported with the `lanscan` command.

d. Click [OK].

**Step 4 Create a Link Station**

Once the Port has been defined, you need to create a Link Station. The Link Station represents the SNA node of the remote host of the APPC server. But before you can create the Link Station, you must create a Remote Node definition.

a. From the Services menu, select APPC and select Add Remote Node.

b. In the Remote Node dialog box enter the SNA CPNAME of the remote node and click [OK].

c. Now you are ready to create the Link Station. From the Services menu, select Connectivity and select Add Link Station. In the resulting dialog box, select the Port previously defined and click [OK].

d. In the Link Station dialog box enter a name for the Link Station, choose the SNA Port name and choose the type of link activation.

e. Choose the LU Traffic type. For maximum flexibility, choose the Any option.

f. For Independent LU traffic, specify the Remote Node name.

g. Click on [Remote Node] and select the node you previously created. Click [OK].

h. For Dependent LU traffic, specify the Local Node ID, and optionally, Remote Node ID. Then specify the Contact Information. Contact information contains the MAC address of the remote host as well as the SAP number. Press the [Advanced] button for additional parameters of the Link Station.

i. The Ethernet Parameters screen shows additional parameters of the Link Station. These parameters effect initial XID contact and retransmission times and limits. The defaults are normally sufficient. Click [OK].
Step 5 Create Local LUs
a. Once the Remote Node definitions have been made, create the Local LU names for the local host. From the Services menu select APPC and Add Local LU.
b. In the Local LU dialog box, enter the name of the local LU and an alias. This name must correspond to the VTAM definitions on the remote server host for the HP 9000 host.
c. Click [OK].

Step 6 Create Partner LUs
a. Now define a Partner LU which represents the LU that the remote server is using to communicate. From the Services menu select APPC and Add Partner LUs and Partner LU on Remote Node.
b. In the resulting dialog box, Enter the Partner LU name and characteristics. The Partner LU name will contain the SNA Network Name as well as the LU name of the remote LU.
c. Enable parallel session support. The location is the name as the Remote Node name. You may click on [Location] for a list.
d. Click [OK].

Step 7 Create Mode and CPI-C Profiles
a. Once the local and remote LU definitions have been made, create the necessary Mode and CPI-C definitions. From the Services menu, select APPC and Modes.
b. In the Modes dialog box click on Add to add a new mode.
c. In the subsequent Mode dialog box enter the Mode Name and other session parameters. The prescribed name for an APPC mode is "IBMRDB". Contact your Remote Host system administrator for appropriate mode parameter.
d. Click [OK].
e. Now that the Mode has been defined, create the CPI-C Side Information Profile, which the gateway will use as a connection name. From the menu, select APPC and CPI-C. In the CPI-C destination names dialog box, click on Add to add a new profile.
f. In the CPI-C destination dialog box enter the Profile name, Partner TP, Partner LU, mode and Security option. The default TP name of the mode remote server will typically be a Service TP named "07F6C4C2".

g. For the Partner LU, enter either the full LU name or the alias created previously. Enter "IBMRDB" for the mode name.

h. Lastly, choose the type of security these sessions will use. This will affect how session authorization is done.

i. Click [OK].

Testing the Connection

Ensure that your connection is working. Do this by starting the SNAplus2 Node and then starting the individual link stations.

Figure 7–1 shows the relationship between SNAplus2 definitions and the VTAM definitions on the remote host.
Figure 7–1  Relationship Between SNAplus2 Definitions and Remote Host Definitions
Oracle Procedural Gateway for APPC uses the SNA Advanced Program to Program Communication (APPC/LU6.2) protocol to communicate with an OLTP. APPC support on Sun Solaris is provided by the SunLink SNA Peer-to-Peer product.

Read this chapter to learn how to configure SunLink SNA Peer-to-Peer on a Sun Solaris system to run Oracle Procedural Gateway for APPC.

This chapter contains the following sections:

- Using SNA Security Validation on page 8-2
- Specifying SNA Conversation Security on page 8-2
- Processing Inbound Connections on page 8-4
- Configuring SunLink SNA Peer-to-Peer Version 9.0 on page 8-4
- Configuring SNAP-IX Version 6 on page 8-11
Using SNA Security Validation

When an RPC request to start a remote transaction program is received by the gateway, the gateway attempts to start an APPC conversation with the OLTP. Before the conversation can begin, a session must start between the Solaris Logical Unit (LU) and the OLTP LU.

SNA and its various access method implementations (including SunLink SNA Peer-to-Peer and VTAM) provide security validation at session initiation time, allowing each LU to authenticate its partner. This validation is carried out entirely by network software before the gateway and OLTP application programs begin their conversation and process conversation-level security data. If session-level security is used, then correct password information must be established in the Solaris SNA profiles and in similar parameter structures in the OLTP to be accessed. Refer to the appropriate communications software product documentation for detailed information about this subject.

Specifying SNA Conversation Security

The PGA_SECURITY_TYPE parameter of the gateway initialization file allows you to specify either of three options that determine the security conduct of the LU6.2 conversation that is allocated with the OLTP. These options are part of the SNA LU6.2 architecture, but their precise behavior might vary depending on the particular OLTP system.

SNA Security Option SECURITY=NONE

If PGA_SECURITY_TYPE=NONE is specified, then the gateway performs no processing of the client user ID and password. The conversation is allocated with SNA option SECURITY=NONE.

SNA Security Option SECURITY=PROGRAM

If PGA_SECURITY_TYPE=PROGRAM is specified, then the gateway allocates the conversation with SNA option SECURITY=PROGRAM, and the following information is sent to the OLTP:

- If the TIP user ID and password overrides are used, then the specified user ID and password are sent regardless of the database link specification.
- If the database link has explicit CONNECT information, then the specified user ID and password are sent.
If the database link has no CONNECT clause, and if the application logged on to Oracle with an explicit user ID and password, then the Oracle user ID and password are sent.

- If the application logs on to Oracle with operating system authentication, and if the database link lacks explicit CONNECT information, then no user ID and password are sent. If no user ID and password are sent, and if the OLTP is not configured to assign a default user ID, then the connection fails.

In general, SNA option SECURITY=PROGRAM tells the OLTP to authenticate the user ID/password combination using whatever authentication mechanisms are available. For example, if CICS/ESA is the OLTP, then RACF can be used. This is not always the case, however, because each OLTP can be configured to process inbound user IDs in other ways.

**SNA Security Option SECURITY=SAME**

If PGA_SECURITY_TYPE=SAME is specified, the gateway allocates the conversation with SNA option SECURITY=SAME and sends only a user ID, without a password, to the OLTP. In this case, SunLink SNA Peer-to-Peer sends the owning user ID of the gateway server executable, `$ORACLE_HOME/bin/pg4asrv`, as the user ID. The user ID that is sent is not the Oracle user ID. This user ID can be viewed with the `ls` command and can be changed by an authorized user with the `chown` command. Because this user ID is the same for all users of a given gateway instance, this option is of limited usefulness.

**Note:** The user ID sent is not translated to uppercase by SunLink SNA Peer-to-Peer. If your OLTP is running on a system which does not allow lowercase user IDs (MVS, for example), you must set up an uppercase user ID on Solaris to be the owner of the gateway executable file.

SECURITY=SAME is similar to the Solaris operating system authentication. It tells the OLTP that the user has already been authenticated at the originating side of the conversation. There might be configuration parameters or options on the server side that affect whether SECURITY=SAME conversations are accepted. When properly configured, the OLTP only confirms that the user ID itself is valid and then accepts the connection. As with SECURITY=PROGRAM, it is possible to change this behavior through configuration options in many OLTPs.
Processing Inbound Connections

Many OLTPs provide options for manipulating the security conduct of an inbound (client) APPC session request. Refer to the appropriate documentation for your OLTP for detailed information about this.

Note that for CICS, one security option is not supported by the gateway.

- ATTACHSEC=PERSISTENT, specified on the CICS CONNECTION definition, requires capability that is not yet available in the gateway.
- ATTACHSEC=LOCAL, ATTACHSEC=IDENTIFY, ATTACHSEC=VERIFY, and ATTACHSEC=MIXIDPE are fully supported by the gateway.

Configuring SunLink SNA Peer-to-Peer Version 9.0

SunLink SNA Peer-to-Peer Version 9.0 contains two main components that are used by Oracle Procedural Gateway for APPC. They are the SunLink SNA PU2.1 Server and the SunLink P2P LU6.2 Server. The PU2.1 server provides the physical connectivity between the Sun SPARC workstation and the remote system(s), while the LU6.2 server provides all LU6.2 services required for establishing, transferring data on, and terminating APPC conversations. For more information on these components, refer to vendor documentation.

Overview of the Configuration Process

If you are configuring SunLink SNA Peer-to-Peer to run on your Sun SPARC workstation, it can be divided into three tasks:

- setting up a gateway name
- setting up a configuration
  - Sample files are provided in the $ORACLE_HOME/pg4appc/sna directory. For SunLink SNA Peer-to-Peer 9.0, the sample file is pgap2p9.cfg.
- setting up the Side Information profiles

The gateway name and the configuration file are needed to start the SunLink SNA Peer-to-Peer daemon process. Oracle Procedural Gateway for APPC, which uses SNA/APPC services, needs a side information profile to start a conversation.
Independent Versus Dependent LUs

Oracle Corporation recommends independent LUs for the gateway because they support multiple parallel sessions or conversations. This means multiple Oracle client applications can be active simultaneously with the same OLTP through the independent LU.

Dependent LUs only support a single active session. The CP (SunLink SNA Peer-to-Peer, in this case) queues additional conversation requests from Oracle Procedural Gateway for APPC behind an active conversation. That is, conversations are single-threaded for dependent LUs.

If a dependent LU is correctly defined, no alterations to the gateway configuration are needed, nor should any changes be needed to the host transaction or how the OLTP is started.

The operational impact of dependent LUs is that the first client application can initiate a conversation through the gateway with the OLTP, and while that transaction is active (which could be seconds to minutes to hours, depending on how the client application and transaction are designed) any other client application initiating a conversation with the same OLTP instance appears to hang as it waits behind the previous conversation.

If a production application really only uses a single conversation or transaction at any one time, there should be no impact.

However, additional concurrent conversations or transactions might be required for testing or other application development. Each requires that additional dependent LUs be defined on the remote host, plus an additional SunLink SNA Peer-to-Peer configuration file entry which defines the additional dependent LUs on the Sun SPARC workstation. The TIP which initiates the conversation must specify the different SunLink SNA Peer-to-Peer Partner LU through a different Side Information Profile. Refer to the "PGAU DEFINE TRANSACTION SIDE PROFILE" section in Chapter 2 of the Oracle Procedural Gateway for APPC User's Guide, and the SunLink SNA Peer-to-Peer Side Information file discussed in the section, "Side Information Files" on page 8-8.

Setting up a SunLink P2P LU6.2 Server Name

The SunLink P2P LU6.2 server name is the name by which Oracle Procedural Gateway for APPC identifies which SunLink P2P LU6.2 server is to be used. The SunLink P2P LU6.2 server name is always the host name of the Sun SPARC workstation on which the SunLink SNA PU2.1 and SunLink P2P LU6.2 servers are running.
The SunLink P2P LU6.2 server name is made known to Oracle Procedural Gateway for APPC through the environment variable APPC_GATEWAY. This environment variable must be specified in the pg4hoa1.sh file for the Procedural Gateway instance. For example, if your Sun SPARC workstation has a hostname of sun-sparc, then you would put the following in your pg4hoa1.sh file:

APPC_GATEWAY=sun-sparc

### Setting up a Configuration File

To enable communication between SunLink SNA Peer-to-Peer and a remote SNA host, you must specify the precise SNA configuration to the SunLink SNA PU2.1 server on your Sun SPARC workstation. The information is contained in a flat ASCII file that is read by the sunpu2.1 daemon process at startup. The configuration file might exist anywhere on the system and the file name is not restricted. The default name of the SunLink SNA PU2.1 configuration file is sunpu2.config, and it resides in the directory from which the sunpu2.1 daemon process is started, which is normally /opt/SUNWpu21. The configuration file contains directives and parameters that describe the network to the SunLink SNA PU2.1 server.

Refer to the vendor documentation for a detailed description of each directive in the input file and its associated parameters.

Oracle Corporation ships a sample SunLink SNA PU2.1 server sample configuration file for a token-ring connection to MVS. After you have successfully installed Oracle Procedural Gateway for APPC, you can find the sample file in $ORACLE_HOME/pg4appc/sna/pgap2p9.cfg.

A minimal configuration for the SunLink SNA PU2.1 server to be used with Oracle Procedural Gateway for APPC must include the following directives:

- CP
- TRLINE or SDLCLINE
- DLC
- LU
- PTNR_LU
- MODE
CP Directive
This directive defines the control point of the local PU2.1 node. It is required to support independent LUs.

TRLINE or SDLCLINE Directives
These directives define either a Token Ring connection or an SDLC connection to the remote host system.

DLC Directive
This directive defines the data link control characteristics of the link station for the specified TRLINE or SDLCLINE. When using SDLC, the ADDR parameter must match the ADDR parameter in the remote system’s definition for the SDLC line. If the remote system is an IBM mainframe host, this would be the ADDR parameter on the PU statement in the NCP gen for the line.

When using Token Ring, the RMTMACADDR parameter must match the Token-Ring address of the remote system. If the remote system is an IBM mainframe host, this would be the LOCADD parameter of the LINE statement in the NCP gen for the line. The TERMID parameter must match the combined IDBLK and IDNUM parameters on the remote system. If the remote system is an IBM mainframe host, this would be the IDBLK and IDNUM values from the PU statement in the VTAM switched definition for the Sun SPARC workstation.

LU Directive
This directive defines the local LU to be used for sessions with partner LUs on remote systems. The local LU name must match an LU definition on the remote system. If the remote system is an IBM mainframe host, this would be the name field of the LU statement in either the NCP gen for an SDLC line or the VTAM switched definition for a Token-Ring line.

For use with Oracle Procedural Gateway for APPC, the LUTYPE parameter must be set to 6.2, and the LOCALLADDR parameter must be set to 0 for independent LUs or to the corresponding ADDR parameter value from the remote LU definition for dependent LUs. Independent LUs are required if you want to support multiple concurrent sessions between a local LU and remote systems. The SESS_LMT parameter should be set to the maximum number of concurrent sessions expected when the LU is an independent LU.
PTNR_LU Directive

This directive defines a partner LU with which a local LU will communicate. The LOC_LU_NAME parameter specifies which local LU will be used for communications with this partner LU. The NQ_LU_NAME parameter specifies the fully-qualified SNA LU name of the partner LU.

For use with the gateway, the partner LU represents a remote OLTP server with which Oracle Procedural Gateway for APPC will communicate. The PAR_SESS_SUP parameter should be set to YES if parallel sessions are desired and supported by the remote OLTP server.

MODE Directive

This directive defines parameters for sessions between a local LU and a partner LU. The NAME parameter must specify a name that is defined to the partner LU on the remote system. Other parameters determine such values as LU6.2 parallel session limits, send and receive pacing values, and send and receive SNA RU sizes.

For use with Oracle Procedural Gateway for APPC, the NAME parameter must specify the name of a mode table entry defined to the OLTP communication software. Consult your system programmer responsible for configuring the OLTP communication software before choosing the mode name and parameters.

The LCL_MAX_SESS_LMT parameter is the maximum number of parallel sessions allowed between the local and partner LUs under this mode name. A value greater than 1 requires that parallel sessions are supported by both LUs, and reflects the maximum number of concurrent Oracle Procedural Gateway for APPC conversations allowed with the partner LU.

The CW_AUTOACT_LMT parameter should be set to at least 1 to improve performance, by pre-initializing sessions over which APPC conversations can be carried. The maximum value for this parameter is the value of the LCL_MAX_SESS_LMT parameter. There is a trade-off between resource utilization and performance when this parameter is set to a high value; performance will be better, because there will usually be an available session for each new conversation request, but resource utilization will be higher on both the local system and the remote system due to the presence of unused sessions. This parameter should be tuned according to your system’s capabilities and needs.

Side Information Files

Before starting an APPC conversation with a partner program, the Oracle Procedural Gateway for APPC requires certain information. The following list describes the required information and its purpose:
**Partner LU name** identifies the remote partner LU associated with the OLTP server. This name must be defined in the PU2.1 Server configuration file by the NAME parameter of a PTNR_LU directive.

**Mode Name** identifies the logmode entry for the partner LU to be used in session establishment. This name must be defined in the OLTP server’s communication software, and in the PU2.1 Server configuration file by the NAME parameter of the MODE directive.

**TP Name** identifies the transaction program to be executed on the OLTP server side of the conversation. This name must be defined in the OLTP server as a valid transaction program name. It does not need to be defined in the PU2.1 Server configuration file.

The TP name must be specified, but can be overridden by Oracle Procedural Gateway for APPC at execution time, using the value specified by the TPNAME keyword of the PGAU DEFINE TRANSACTION STATEMENT (refer to "DEFINE TRANSACTION" in Chapter 2 of the Oracle Procedural Gateway for APPC User’s Guide). This allows the use of a single side information profile for communication with multiple transaction programs at a particular OLTP.

All of the preceding information is provided to Oracle Procedural Gateway for APPC through the use of a Symbolic Destination Name. This name is specified by the SIDEPROFILE keyword of the PGAU DEFINE TRANSACTION statement (refer to "DEFINE TRANSACTION" in Chapter 2 of the Oracle Procedural Gateway for APPC User’s Guide). The Symbolic Destination Name identifies a Side Information file which contains the partner LU name, mode name, and TP name to be used to establish a conversation. The Side Information file must be located in the current search path for Oracle Procedural Gateway for APPC process, and its file name must be from 1 to 8 characters in length. The search path is specified by the PATH environment variable in the pg4hoa1.sh file for the Procedural Gateway instance. The PATH specification should add the directory path where your side information files are located to the existing search path, as in the following example:

```
PATH=$PATH:/oracle/pg/9.2.0/pg4appc/sna
```

The format of the value for the PATH environment variable is standard UNIX, with directory paths separated by colons (:). The search is done in the order in which the directories appear in the PATH setting.

Sample side information files are provided in the $ORACLE_HOME/pg4appc/sna directory. The following files are provided: APPCMVS, CICSPGA, IDMSPGA, IMSPGA, and IMSPGA61. These profiles are the ones used in all of the
demonstration transactions that are distributed with Oracle Procedural Gateway for APPC.

The following example shows the contents of the CICSPGA file:

```
# Sample side information file for access to CICS
#
PTNR_LU_NAME  = CICSPGA
MODE_NAME     = ORAPLU62
TP_NAME       = FLIP
```

The other files have similar contents. For further information on the side information file format, refer to vendor documentation.

**Starting the SunLink SNA PU2.1 Server**

To start the SunLink SNA PU2.1 Server, you must be logged in as the superuser. After you have logged in, change directory to the /opt/SUNWpu21 directory and enter the following command:

```
sunpu2.1 -f config.file
```

where `config.file` is the full path to your SunLink SNA PU2.1 configuration file. If you use the default file name, `sunpu2.config`, and it resides in the /opt/SUNWpu21 directory, then the parameter `-f config.file` is not required.

For detailed information on operating the SunLink SNA PU2.1 Server, refer to vendor documentation.

**Test the Connection**

Before proceeding with the installation of Oracle Procedural Gateway for APPC, ensure that your connection is working. For instructions on how to test this connection, refer to vendor documentation.
Configuring SNAP-IX Version 6

The following sections describe how to configure SNAP-IX version 6.

Before You Begin

This section requires you to input parameters unique to your system to configure SNAP-IX version 6 properly. Before you begin, request these parameters from your network administrator.

SNAP-IX Configuration Tool

All of the SNAP-IX product configuration is done using the `xsnaadmin` program. This tool is an X-Windows application which provides a graphical interface so that you can view and modify the current SNAP-IX configuration and the current running state of the host SNA node.

Creating SNAP-IX Profiles for the Gateway

Oracle Procedural Gateway for APPC requires a stored set of definitions, called Side Information Profiles, to support connections between the gateway and gateway servers. Each profile consists of a profile name and a profile type, which is a set of fields describing the profile. The fields in a given profile type are generally a mixture of operating parameter values and names of other SNA profiles relevant to the profile. Each functional part of APPC, such as the Mode, Remote Transaction Program name, and Logical Unit (LU), is described by a distinct profile type.

Independent Versus Dependent LUs

The Gateway configuration can accommodate either independent or dependent LUs. If you choose to use dependent LUs, or are restricted to using dependent LUs, the Gateway functions properly; if a dependent LU is correctly defined, then you do not need to make changes to the configuration of Oracle Procedural Gateway for APPC, nor should any changes be needed to the gateway server. However, Oracle Corporation recommends that you use independent LUs for Oracle Procedural Gateway for APPC because they support multiple parallel sessions or conversations. This means that multiple Oracle client applications can be active simultaneously with the same gateway server through the independent LU.

In contrast to independent LUs, dependent LUs support only a single active session. The CP (Control Point for the Node) queues each additional conversation request
from the gateway behind an already active conversation. In other words, conversations are single-threaded for dependent LUs.

The operational impact of dependent LUs is that the first client application can initiate a conversation through the gateway with the gateway server, but while that session is active (which could be seconds, minutes or hours, depending on how the client application and transaction are designed), any other client application initiating a session with the same gateway server appears to hang as it waits behind the previous session.

If a production application really uses only a single conversation at any one time, then there should not be a problem. However, at some point you might require additional concurrent conversations for testing or for other application development. Having more than one conversation requires that additional dependent LUs be defined on the remote host. Additional configuration entries must be added to SNAP-IX. Additional Side Information Profiles should be defined to use the new dependent LUs. Oracle Procedural Gateway for APPC instances should be created and configured to use these new Side Information Profiles.

Creating SNA Definitions for the Gateway

SNAP-IX definitions are stored in the following two files, located in the `/etc/opt/sna` directory:

- `sna_node.cfg` - SNA node definitions
- `sna_domn.cfg` - SNA domain definitions

These files are created and maintained with the `xsnaadmin` tool. Maintenance of SNA definitions is normally done by a user with administrative authority. The following information is intended for the person creating SNA definitions for the gateway. You should have some knowledge of SNA before reading this section.

Sample SNAP-IX Definitions

The `$ORACLE_HOME/pg4appc/sna` subdirectory contains a set of sample SNAP-IX definition files for the gateway, created with the `xsnaadmin`. These sample files are `sna_domn.cfg` and `sna_node.cfg`. SNA definitions are very specific to the host and SNA network. As such, the sample definitions provided will not work without being tailored for the local host and SNA network.
Configuring SNAP-IX

This section describes the process of creating your SNA definitions for SNAP-IX, using xsnaadmin. All of the tasks described in this section are performed from within xsnaadmin. All configuration is done using the various pull-down menus and panels in xsnaadmin. The following configuration descriptions follow the samples provided. Please tailor the various SNA values for your local host and SNA network.

Invoking xsnaadmin

Use the following commands to invoke xsnaadmin. The $DISPLAY environmental variable must be set appropriately. If you are running xsnaadmin from the local console, then $DISPLAY should already be set. If you are running xsnaadmin from a remote X display, then set $DISPLAY to the host name or IP address of that display.

$ DISPLAY=<your_display>:0
$ export DISPLAY
$ xsnaadmin &

On startup of xsnaadmin, the main screen opens and displays the current configuration of the local SNA node.

Configuring the SNA node

From the Services menu select Configure Node Parameters. In the Node Parameters dialog box enter the APPN support type, Control Point Name, Control Point Alias and Node ID as needed. The Control Point Name is composed of the SNA Network Name and the CP name of the local host. Click [OK].

Adding a Port

From the Services menu select Connectivity and New Port. In the Add to <nodename> dialog box, select the Port type and click [OK].

In the SAP dialog box, enter a Port name and network card number. The Port name will be used to logically name the physical network card that you are using and will be used to bind a Service Access Port to the card for SNA protocols. Normally you can accept the values provided in the dialog box. If a different network card is needed, however, enter the card number as reported with the dmesg command. Click [OK].
Create a Link Station

When the Port has been defined, you must create a Link Station. The Link Station represents the SNA node of the remote host of the gateway server. But before you can create the Link Station, you must create a Remote Node definition. From the Services menu select APPC and Add Remote Node. In the dialog box, enter the SNA CPNAME of the remote node and click [OK].

Now you are ready to create the Link Station. From the Services menu, select Connectivity and New Link Station. In the dialog box, select the Port previously defined and click [OK].

In the Link Station dialog box, enter a name for the Link Station, choose the SNA Port name and choose the type of link activation. Choose the LU Traffic type. For maximum flexibility, choose the Any option. For Independent LU traffic, specify the Remote Node name. Click on [Remote Node] and select the node you previously created. Click [OK]. Choose the type of the Remote node, typically a Network node. For Dependent LU traffic, specify the role of the Remote node, typically 'host', the Local Node ID, and optionally, Remote Node ID. Then specify the Contact Information.

Contact information contains the MAC address of the remote host as well as the SAP number. Press the [Advanced] button for additional parameters of the Link Station.

The Token Ring Parameters dialog box shows additional parameters of the Link Station. These parameters effect initial XID contact and retransmission times and limits. The defaults are normally sufficient. Click [OK].

Create Local LUs

When the Remote Node definitions have been made, create the Local LU names for the local host. From the Services menu select APPC and New Local LU. In the local LU dialog box, enter the name of the local LU and an alias. This name must correspond to the VTAM definitions on the remote gateway server host for the UNIX host. Click [OK].

Create Partner LUs

Now define a Partner LU which represents the LU that the gateway server is using to communicate. From the Services menu select APPC and New Partner LUs and Partner LU on Remote Node. In the Partner LU dialog box, enter the Partner LU name and characteristics. The Partner LU name contains the SNA Network Name as well as the LU name of the remote LU. Enable parallel session support. The
location is the name as the Remote Node name. You can click on [Location] for a list. Then click [OK].

**Create Mode and CPI-C Profiles**

When the local and remote LU definitions have been made, create the necessary Mode and CPI-C definitions. From the Services menu select APPC and Modes. In the Modes dialog box click on New to add a new mode.

In the Mode dialog box enter the Mode Name and other session parameters. The prescribed name for a gateway mode is "CICSPGA". Contact your Remote Host system administrator for appropriate mode parameters. Click [OK].

Now that the Mode has been defined, create the CPI-C Side Information Profile, which the gateway will use as a connection name. From the menu select APPC and CPI-C. In the CPI-C destination names dialog box click on New to add a new Profile.

In the CPI-C destination dialog box enter the Profile name, Local LU name, Partner TP, Partner LU and mode, and Security option. The default TP name of the mode gateway server will typically be a Service TP. For the Local LU, you may specify a specific LU or choose the default LU. For the Partner LU, enter either the full LU name or the alias created previously. Enter "CICSPGA" for the mode name. Choose the type of security these sessions will use. This will affect how session authorization is done. Click [OK].

**Testing the Connection**

Before proceeding with the gateway configuration tasks, ensure that your connection is working. Perform this by starting the SNAP-IX Node and then starting the individual link stations.

*Figure 8–1* shows the relationship between SNAP-IX definitions and the VTAM definitions on the remote host.
Figure 8–1  Relationship between SNAP-IX Definitions and Host VTAM Definitions

Local LU Definition

Local LU Alias

Local LU Name

...

Side Information

Local LU Alias

Partner LU Alias

or

Fully-qualified Partner LU Name

Mode Definition

Mode Name

Remote TP Name

...

Connection Definition

Connection Name

...

Partner LU Definition

Fully-qualified Partner LU Name

netname.pluname

Partner LU Alias

Connection

...

VTAMLST

APPL Definition
pluname
MODSTAB=mtname

ATCSTR00
NETWORK=netname
SSCPNAME=xpname

VTAM Mode Table

netname MODSTAB
MODBSET
LOGMODE=xmode
name
Two-phase commit enables you to update local Oracle resources in the same Oracle transaction as updating of non-Oracle resources accessed through Oracle Procedural Gateway for APPC. This chapter assumes that you are familiar with the basic concepts of two-phase commit as discussed in *Oracle9i Database Concepts*.

This chapter contains the following sections:

- **Introduction** on page 9-2
- **Supported OLTPs (Online Transaction Processors)** on page 9-2
- **Required Components** on page 9-3
- **Two-Phase Commit Architecture** on page 9-2
- **PreConfiguration Tasks** on page 9-10
- **Configuring Two-Phase Commit** on page 9-11
- **Post-Configuration Tasks** on page 9-18
- **RRM Recovery** on page 9-20
- **Two-Phase Commit With CICS/ESA** on page 9-23
- **Application Design Requirements** on page 9-24
- **Manual Recovery of In-Doubt Transactions** on page 9-26
- **Logging Two-Phase Commit Activity from CICS** on page 9-29

AIX-Based Systems Only: Implementing Two-Phase Commit
**Introduction**

Oracle Procedural Gateway for APPC implements two-phase commit using LU6.2 synclevel 2. This is a method fully developed in the architecture of SNA to ensure that resources updated on both sides of an APPC conversation are kept in synchronization. For complete details on SNA LU6.2 synclevel 2 architecture, refer to vendor documentation.

**Supported OLTPs (Online Transaction Processors)**

Two-phase commit is supported by the gateway across APPC conversations with CICS/ESA.

The gateway supports two-phase commit with CICS/ESA within the constraints imposed by the design of CICS/ESA. These constraints are described in vendor documentation. Pay special attention to the discussion of the INDOUBT option of the CICS TRANSACTION definition.

It is important to understand the CICS/ESA implementation of synclevel 2 to work within the CICS/ESA environment. Refer to vendor documentation for information on synclevel 2 processing in the CICS/ESA environment.

---

**Note:** Read "Two-Phase Commit With CICS/ESA" on page 9-23 for a complete discussion of the gateway operation with CICS/ESA.

IBM IMS/TM and APPC/MVS do not support LU6.2 synclevel 2, and therefore do not support two-phase commit across APPC conversations.

**Two-Phase Commit Architecture**

The architecture of the two-phase commit implementation in Oracle Procedural Gateway for APPC consists of four main components:

- the Oracle Integrating Server
- the Oracle Procedural Gateway for APPC server (gateway server)
- the Oracle Procedural Gateway for APPC Resource Recovery Manager (RRM)
- the Oracle logging server
The following components are required to support two-phase commit:

- **Oracle Procedural Gateway for APPC Server**
  
  The gateway server supports two-phase commit when you specify PGA_CAPABILITY=2_PHASE in the gateway initialization file. When the gateway server is running with two-phase commit enabled, then it connects to a local Oracle Integrating Server where it maintains an LU6.2 transaction log, similar to the Oracle two-phase commit log stored in the DBA_2PC_PENDING table. The gateway’s transaction log is stored in the PGA_2PC_PENDING table. An entry is stored in this table for each in-flight transaction and remains there until the transaction has completed.

- **Oracle Procedural Gateway for APPC Resource Recovery Manager (RRM)**
  
  This is a separate process that must be running at all times. It acts as an LU6.2 service transaction program supporting the Exchange Log Names communication with partner LUs, and functions as a local LU6.2 log manager. The RRM maintains a table of LU names and their corresponding LU6.2 log names, with an entry for the local LU on which the RRM is running and for each partner LU with which the gateway establishes conversations. This table is stored in the same Oracle server (Oracle logging server) that the gateway server uses to store its transaction log, and is called PGA_2PC_LUS.

- **Oracle logging server**
  
  An Oracle server must be available to the gateway server and by RRM for storing their respective tables, PGA_2PC_PENDING and PGA_2PC_LUS. This Oracle server is referred to as the Oracle logging server. For maximum performance, Oracle Corporation recommends that this Oracle server reside on the same system as the gateway server and the RRM. The Oracle logging server can be the same Oracle server as the Oracle Integrating Server, but is not a requirement.

- **Dedicated Local LU**
  
  A dedicated local SNA LU must be available for use by the gateway server and the RRM. No other applications should be using this LU on the system, to avoid any interference with the operation of the gateway server and RRM SNA interfaces.

The next section describes what role each component plays in the operation of two-phase commit and how these components interact.
Roles of the Components of Two-Phase Commit

The Oracle Integrating Server is the controlling component in the two-phase commit architecture. It tells the gateway server when to prepare a transaction, when to commit, and when to rollback. It does the same with all other servers participating in a distributed transaction. When a failure has occurred, the Oracle Integrating Server drives the recovery process in each participating server, including the gateway server.

The Oracle logging server is an Oracle server available to the gateway server and to the RRM (Resource Recovery Manager) for storing and accessing their LU6.2 log information. The logging server is not required to be the same Oracle server as the Oracle Integrating Server, but it can be. Because the logging server is an integral component of the gateway two-phase commit operations, the best place for the logging server to reside is on the same system as the gateway server and the RRM. This allows the communication between the gateway server and RRM and the logging server to use Inter-Process Communications (IPC), providing a high-speed, low overhead, local connection between the components.

The gateway server performs the task of converting the Oracle Integrating Server’s instructions into LU6.2 operations. It also performs its own logging of the transaction into the Oracle logging server. The log information that the gateway server stores is part of the LU6.2 log information, and is stored in a table in the logging server called PGA_2PC_PENDING. If a failure occurs in the LU6.2 processing of the transaction, then the gateway server determines what error should be returned to the Oracle Integrating Server.

The RRM (Resource Recovery Manager) is a long-running, stand-alone process that functions as an LU6.2 service transaction program (TP name x’06f2’) supporting the Exchange Log Names (XLN) request exchange with partner RRMs on remote partner LUs. The RRM must be running for synclevel 2 conversations to be established from the local LU to any partner LUs. The RRM performs its own logging of partner LU information into the Oracle logging server. The log information that the RRM stores is part of the LU6.2 log information, and it is stored in a table in the logging server called PGA_2PC_LUS. The RRM uses this table to keep track of which partner LUs it has communicated with and their LU6.2 log names. It also keeps its own LU6.2 log name in this table.
Interactions

A specific set of interactions occur between the two-phase commit components as follows:

- **Oracle Integrating Server <--> gateway server**
  
The Oracle Integrating Server drives all actions by the gateway server. At the request of the Oracle application, the Oracle Integrating Server can instruct the gateway server to begin a new Oracle transaction, start a prepare sequence, start a commit sequence, or start a rollback sequence. It can also call gateway RPC functions (PGAINIT, PGAXFER, PGATERM) on behalf of the Oracle application.

- **gateway server <--> Oracle logging server**
  
The gateway server calls the Oracle logging server to insert, update, and delete rows from its PGA_2PC_PENDING table. This is actually done by calling a PL/SQL stored procedure in the logging server to reduce the number of open cursors required by the gateway server for performing its logging. Only a single cursor is needed by the gateway server for logging.

- **RRM <--> Oracle logging server**
  
The RRM calls the Oracle logging server to insert, select, update, and delete rows from its PGA_2PC_LUS table. It also calls the Oracle logging server to select and delete rows from the PGA_2PC_PENDING table.

Note that the RRM never interacts directly with the gateway server. The RRM interacts only with the partner LUs using LU6.2, and therefore never needs to interact with the gateway server.

---

**Caution**: Oracle Corporation strongly urges you to comply with the design requirements described in this chapter. Failure to do so results in complex recovery situations.
Two-Phase Commit Flow

The flow of control for a successful two-phase commit between an Oracle application and a CICS transaction is outlined in the following steps. These flow elements are illustrated in Figure 9–1.

Two-Phase Commit Flow of Control, Step by Step

1. The application issues a COMMIT to the Oracle Integrating Server.
2. The Oracle Integrating Server sends PREPARE to the gateway.
3. The gateway receives a PREPARE from the Oracle Integrating Server, and creates a pending transaction row in the PGA_2PC_PENDING table with the status set to COLLECTING.
4. The gateway sends an LU6.2 PREPARE command to CICS. CICS presents a status to the CICS application indicating that a SYNCPOINT should be initiated.
5. The CICS application issues EXEC CICS SYNCPOINT, and CICS prepares any data stores that the application has updated (for example, DB2).
6. Once the data stores have been prepared, CICS sends an LU6.2 REQUEST-COMMIT command back to the gateway.
7. The gateway updates the pending transaction row in the PGA_2PC_PENDING table with a status of PREPARED, and then returns PREPARE OK to the Oracle Integrating Server. The Oracle Integrating Server then proceeds to PREPARE each of the other participants in the distributed transaction.
8. After all of the participants in the distributed transaction have responded with PREPARE OK, the Oracle Integrating Server sends a COMMIT to the gateway.
9. The gateway sends an LU6.2 COMMITTED command to CICS.
10. On receipt of the COMMITTED, CICS commits all prepared data stores, and sends an LU6.2 FORGET back to the gateway.
11. CICS returns control back to the CICS application program with a return code indicating that the EXEC CICS SYNCPOINT completed successfully.
12. When the gateway receives the FORGET from CICS, it deletes the pending transaction row from the PGA_2PC_PENDING table to indicate that the transaction has been committed, and then returns COMMIT OK back to the Oracle Integrating Server. The Oracle Integrating Server then proceeds to send COMMIT to each of the other participants in the distributed transaction, and waits until all participants have responded COMMIT OK.
13. The Oracle Integrating Server returns control to the Oracle application. Note that this is the flow if the application was designed to meet all of the requirements discussed in “Application Design Requirements” on page 9-24. If a COMMIT-CONFIRM gateway or another two-phase commit-capable Oracle Procedural Gateway for APPC is participating in the distributed transaction, or if the gateway initialization file does not specify COMMIT_POINT_STRENGTH=255, then the gateway is not guaranteed to be the commit point, and it might not be the first participant to receive COMMIT from the Oracle Integrating Server.

The preceding description of two-phase commit flow is illustrated in Figure 9–1.

**Figure 9–1 Two-phase Commit Flow with Synclerl 2**

**Gateway Server and RRM LU6.2 Log Tables**

The LU6.2 log information consists of two tables, PGA_2PC_LUS and PGA_2PC_PENDING. The PGA_2PC_LUS table contains a row for the local LU on which the RRM is running, and a row for each partner LU with which the RRM has communicated. This table is maintained completely by the RRM. The PGA_2PC_PENDING table contains a row for each pending gateway transaction. These rows are temporary and are deleted when the transaction is complete. This table is the gateway’s version of the Oracle server DBA_2PC_PENDING table and is maintained primarily by the gateway server, but in certain situations can be updated by the RRM.
The PGA_2PC_LUS table contains the following columns:

- **LU_NAME**
  This is the fully-qualified LU name for which this row was created. It can be either the local LU or a partner LU.

- **LU_TYPE**
  This is the type of LU, either "L" for the local LU or "P" for a partner LU. There can be only one local LU row, but any number of partner LU rows can exist.

- **LOG_NAME**
  This is the LU6.2 log name for the LU. For the local LU, this field is initialized when the RRM is cold started, and does not change until the RRM is cold started again. For a partner LU, this field is initialized when the RRM receives an Exchange Log Names (XLN) from the partner LU that includes an indication that the partner LU is cold starting, and this field does not change until the partner LU sends another XLN with the cold-start indicator set. This field is stored as a RAW because the LU6.2 log name does not necessarily contain printable characters.

- **LOG_STATUS**
  This is the LU6.2 log status for the LU’s log. It is either "W" for warm or "C" for cold. It is initialized to "C" when the LU cold starts, and it is changed to "W" the first time the LU warm starts, following a cold start. After that, it is not changed until the LU cold starts again.

- **CS_LUNAME**
  This is an indicator as to whether the LU supports the sending of its LU name as part of a Compare States (CS) request during recovery processing. This field can be either "Y" or "N".

The PGA_2PC_PENDING table contains the following columns:

- **GLOBAL_TRAN_ID**
  This is the Oracle Global Transaction ID for the transaction, and is identical to the corresponding column in the DBA_2PC_PENDING table.

- **STATE**
  This is the transaction state, and is identical to the corresponding column in the DBA_2PC_PENDING table.

- **ADVICE**
This is the suggested action to take ("C" for commit, "R" for rollback), and is identical to the corresponding column in the DBA_2PC_PENDING table.

- **TRANCOMMENT**
  This is the Comment provided in the COMMIT WORK COMMENT statement, and is identical to the corresponding column in the DBA_2PC_PENDING table.

- **SID**
  This is the SID of the gateway instance that logged this transaction.

- **OS_TERMINAL**
  This is the OS terminal ID for the user who originated this transaction, and is identical to the corresponding column in the DBA_2PC_PENDING table.

- **DB_USER**
  This is the Oracle database user ID used at the commit point site for the transaction, and is identical to the corresponding column in the DBA_2PC_PENDING table.

- **SIDE_NAME**
  This is the side information profile name that was used by the gateway to allocate the APPC conversation with the target LU. This corresponds to the SIDENAME parameter passed to the PGAINIT gateway function.

- **LU_NAME**
  This is the fully-qualified partner LU name of the target LU. This value is either the LU name from the side information profile, or the LUNAME parameter passed to the PGAINIT gateway function. This name fully identifies the CICS system on which the transaction was executed.
PreConfiguration Tasks

- **TP_NAME**
  This is the transaction program name executed at the target LU. This value is either the TP name from the side information profile, or the TPNAME parameter passed to the PGAINIT gateway function. This name fully identifies the CICS transaction program that was executed.

- **LUW_ID**
  This is the LU6.2 Logical Unit of Work Identifier for the APPC transaction. It is generated by the gateway during BEGIN TRANSACTION processing. The low-order two bytes contain a sequence number that is incremented after each COMMIT or ROLLBACK operation, so that each discrete unit of work has a unique LUW_ID value. This column is stored as a RAW.

- **CONV_CORR**
  This is the LU6.2 Conversation Correlator for the APPC conversation. It is generated by SNA and provided to the gateway after the conversation has been allocated.

- **SESS_ID**
  This is the LU6.2 Session Instance Identifier for the APPC conversation. It is generated by SNA and provided to the gateway after the conversation has been allocated.

PreConfiguration Tasks

Perform these steps after the installation of the gateway software and before the two-phase commit configuration.

Choose a Dedicated Local LU

Oracle Procedural Gateway for APPC server and RRM must execute on a dedicated local LU to prevent any interference by other applications with their SNA interfaces. You must define a local LU to be used by the gateway and RRM as described in Chapter 6, "Configuring the SNA Communication Package on AIX-Based Systems". The local LU name that you plan to use has a fully-qualified name consisting of a network name and an LU name. These names are required in some of the tasks you perform in this section and in the configuration section. Make a note of the network and LU names for reference when performing these tasks.
Set Up RRM Execution Group

Oracle Procedural Gateway for APPC RRM must execute under the system group to declare itself an RRM to SNA server. The group of the pg4arrm executable must be changed to system in order for it to execute under the system group.

Perform the following steps to make this change:

1. Change to superuser:
   
   ```
   $ su
   
   You are prompted by AIX for the root password.
   ```

2. Change directory to the gateway executable directory. Enter:
   
   ```
   # cd /oracle/pga/9.2.0/bin
   ```

3. Change the group of the RRM executable to the system group:
   
   ```
   # chgrp system pg4arrm
   ```

4. Exit from superuser:
   
   ```
   # exit
   ```

Configuring Two-Phase Commit

The steps for configuring two-phase commit include:

- configuring the Oracle logging server, where the gateway server and RRM store their LU6.2 log information
- configuring the gateway initialization parameters
- configuring the RRM initialization parameters
- configuring SNA Server
- configuring CICS/ESA

Perform all of the preceding steps before using any applications that use two-phase commit.
Configuring Two-Phase Commit

Configuring the Oracle Logging Server

The Oracle Integrating Server where the gateway server and RRM store their LU6.2 log information should ideally be on the same system where the gateway and RRM run. The configuration of the server consists of creating the gateway DBA user, creating the LU6.2 log tables, and creating the PL/SQL stored procedure used by the gateway server for logging transactions.

1. Use SQL*Plus to connect to the Oracle logging server as user SYSTEM.
2. From SQL*Plus, run the `pga2pcau.sql` script from the `$ORACLE_HOME/pg4appc/admin` directory.
   This script creates the PGADBA user ID. The initial password defined for PGADBA is PGADBA. You can use the ALTER USER command to change the password. For further information, refer to Oracle9i Server SQL Reference.
3. Use SQL*Plus to connect to the Oracle logging server as user PGADBA.
4. From SQL*Plus, run the `pga2pcpn.sql` script from the `$ORACLE_HOME/pg4appc/admin` directory.
   This script creates the PGA_2PC_PENDING table used by the gateway server for its LU6.2 transaction log.
5. From SQL*Plus, run the `pga2pclu.sql` script from the `$ORACLE_HOME/pg4appc/admin` directory.
   This script creates the PGA_2PC_LUS table used by the RRM for its LU6.2 luname/logname information.
6. Using SQL*Plus, run the `pga2pclg.sql` script from the `$ORACLE_HOME/pg4appc/admin` directory.
   This script creates the PGA_2PC_LOG PL/SQL stored procedure used by the gateway server for updating the PGA_2PC_PENDING table.
7. Disconnect from the Oracle logging server.

Configuring the Gateway Initialization Parameters

The gateway initialization parameters are discussed in Appendix A, "Gateway Initialization Parameters". The parameters that are necessary for two-phase commit support in the gateway are:

- PGA_CAPABILITY
- PGA_LOCAL_LU
Each gateway initialization parameter necessary for two-phase commit support is discussed briefly below. Add the parameters to your `initsid.ora` file, where `sid` is the gateway SID for your two-phase commit gateway.

Caution: Because the logging user ID and password are specified in the `initsid.ora` file, Oracle Corporation recommends that the file permissions be set to remove read permissions for non-DBA users.

Set `PGA_CAPABILITY` to `2_PHASE` or `2P` to enable two-phase commit.

`PGA_LOCAL_LU` specifies the local SNA LU on which the gateway server and RRM run. This local LU should not be used by any other gateways or other applications.

`PGA_LOG_DB` specifies the Oracle Net service name that is to be used by the gateway server to connect to the Oracle logging server where the `PGA_2PC_PENDING` table and the `PGA_2PC_LOG` PL/SQL procedure are stored. The service name must be defined in a `tnsnames.ora` file that is accessible to the gateway server. The gateway server accesses the `tnsnames.ora` file using the `TNS_ADMIN` environment variable setting. `TNS_ADMIN` must specify the full path name of the directory in which the `tnsnames.ora` file is stored.

`PGA_LOG_PASS` specifies the Oracle password that is used by the gateway when connecting to the Oracle logging server that is identified by the `PGA_LOG_DB` parameter.

`PGA_LOG_USER` specifies the Oracle user ID that is used by the gateway when connecting to the Oracle logging server that is identified by the `PGA_LOG_DB` parameter. This user ID is the PGADBA user ID created by the `pga2pcau.sql` script. The user ID that is specified must be the user ID under which the `pga2pcpn.sql`, `pga2pclu.sql`, and `pga2pcli.sql` scripts were run.
Configuring the RRM Initialization Parameters

The RRM initialization parameters control the operation of the RRM process. They are supplied to the RRM in a file in the $ORACLE_HOME/pg4appc/admin directory. The file name is composed of the unqualified local LU name on which the RRM runs, plus the suffix ".ini". For example, if the RRM is running on local LU ORACLE.HQWM920, then the unqualified local LU name is HQWM920, so the RRM initialization parameter file name is HQWM920.ini.

Table 9–1 lists the RRM parameters and their descriptions.

<table>
<thead>
<tr>
<th>RRM Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG_DB</td>
<td>Specifies an Oracle Net service name that is used by the RRM to connect to the Oracle Integrating Server where the PGA_2PC_LUS table is stored. The service name must be defined in a tnsnames.ora file that is accessible to the RRM. Note that the Oracle server that is identified by this parameter must be the same Oracle server that is identified to the gateway server by the PGA_LOG_DB parameter. The value specified can be from 1 to 255 characters long. This parameter is required and there is no default value.</td>
</tr>
<tr>
<td>LOG_DIRECTORY</td>
<td>Specifies the full directory path to the RRM log file, where messages that are issued by the RRM are written. The value specified can be from 1 to 255 characters long. If this parameter is not specified, then the default value is $ORACLE_HOME/pg4appc/log.</td>
</tr>
<tr>
<td>LOG_FILE</td>
<td>Specifies the file name of the RRM log file, where messages that are issued by the RRM are written. This file is written into the directory specified by the LOG_DIRECTORY parameter or by its default value. The value specified can be from 1 to 63 characters long. If this parameter is not specified, then the default value is the unqualified local LU name on which the RRM executes, plus the suffix &quot;.log&quot;. For example, if the RRM is running on local LU ORACLE.HQWM920, then the unqualified local LU name is HQWM920, so the default log file name is HQWM920.log. Appendix A contains a sample HQWM920.ini file. Note that this file is opened in append mode, so it contains the log messages for all executions of the RRM since the file was first created by the RRM.</td>
</tr>
</tbody>
</table>
### Table 9–1 Resource Recovery Manager Parameters

<table>
<thead>
<tr>
<th>RRM Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG_ERRORS</td>
<td>Specifies the maximum number of Oracle errors on the connection with the Oracle Integrating Server before the RRM shuts down. The value specified must be an integer from 0 to 65535. A value of 0 means that the RRM never shuts down due to Oracle errors on the connection with the Oracle Integrating Server. The default value is 1.</td>
</tr>
<tr>
<td>LOG_PASS</td>
<td>Specifies the Oracle password that is used by the RRM when connecting to the Oracle Integrating Server that is identified by the LOG_DB parameter. The value specified can be from 1 to 30 characters long. This parameter is required, and there is no default value. Note that this value must be the same as that specified by the PGA_LOG_PASS gateway initialization parameter, because the gateway and the RRM must both use the same PGA_2PC_PENDING table. For more information, refer to &quot;Using the pg4pwd Utility&quot; on page 5-5.</td>
</tr>
<tr>
<td>LOG_USER</td>
<td>Specifies the Oracle user ID that is used by the RRM when connecting to the Oracle Integrating Server that is identified by the LOG_DB parameter. The value specified can be from 1 to 30 characters long. This parameter is required, and there is no default value. Note that this value must be the same as that specified by the PGA_LOG_USER gateway initialization parameter, because the gateway and the RRM must both use the same PGA_2PC_PENDING table.</td>
</tr>
<tr>
<td>SIDE_PROFILE</td>
<td>Specifies the side profile for snalloc. The default is CICSPGA2.</td>
</tr>
<tr>
<td>TP_PROFILE</td>
<td>Specifies the TP profile for snactl. The default is PG4ARRM.</td>
</tr>
<tr>
<td>TRACE_DIRECTORY</td>
<td>Specifies the full directory path to the RRM trace file, where trace information from the RRM is written. The value specified can be from 1 to 255 characters long. If this parameter is not specified, then the default value is $ORACLE_HOME/pg4appc/trace. NOTE: Make sure that the directory has been created. If it has not, create it.</td>
</tr>
</tbody>
</table>

---

NOTE: Make sure that the directory has been created. If it has not, create it.
Configuring the SNA Server

The following items are required for the gateway and RRM to function correctly in the AIX environment:

- **Local LU Profile**
  
  A dedicated local independent LU must be defined for exclusive use by the gateway. This is necessary to prevent non-gateway applications from being affected by the actions of the gateway RRM. The RRM processes incoming Exchange Log Names requests from CICS that are directed to this LU only. The local LU is defined by a Local LU profile. For details on defining this profile, refer to Chapter 6, "Configuring the SNA Communication Package on AIX-Based Systems".

- **Mode Profile**
  
  A mode profile must be defined that references a mode name defined on the target CICS system with sync point capability enabled. The sample mode entry ($ORACLE_HOME/pg4appc/sna/orasyn62.asm) shipped with the gateway, is

---

**Table 9–1 Resource Recovery Manager Parameters**

<table>
<thead>
<tr>
<th>RRM Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRACE_FILE</td>
<td>Specifies the file name of the RRM trace file, where trace information from the RRM is written. This file is written into the directory that is specified by the TRACE_DIRECTORY parameter or its default value. The value specified can be from 1 to 63 characters long. If this parameter is not specified, then the default value is the unqualified local LU name on which the RRM runs, plus the suffix &quot;.trc&quot;. For example, if the RRM is running on local LU ORACLE.HQWM920, then the unqualified local LU name is HQWM920, so the default trace file name is HQWM920.trc. Note that this file is opened in replace mode, so it contains trace information only for the most recent execution of the RRM. Also note that this file is written only if TRACE_LEVEL specifies a value other than 0.</td>
</tr>
</tbody>
</table>
| TRACE_LEVEL   | Specifies the level of RRM tracing to be performed. The value specified must be an integer from 0 to 255, and specifies the sum of the desired trace values, as follows:  
  
  16 dump trace data in EBCDIC instead of ASCII  
  32 trace all LU6.2 data flow to and from the RRM  
  64 trace the return from each SNA API call  
  128 trace the issuing of each SNA API call  

set up for sync point enablement. This mode, or one like it, should be defined on the target CICS system. For details on defining this profile, refer to Chapter 6, "Configuring the SNA Communication Package on AIX-Based Systems."

- Side Information Profile

A side information profile must be defined for the RRM to use. The profile can be the same one used to define the target CICS system with which the gateway communicates. The purpose of this profile for the RRM is only to define on which local LU the RRM is running. By using the same side information profile that is used to define the target CICS system to the gateway, the RRM is ensured to run on the same local LU on which the gateway runs. For details on defining this profile, refer to Chapter 6, "Configuring the SNA Communication Package on AIX-Based Systems."

- Transaction Program Name Profile

A transaction program name profile must be defined for the RRM to use. This profile is used to provide some TP characteristics to SNA Server when the RRM starts. Note that the RRM is not dynamically started by SNA Server, so not all of the fields in the profile are used. For details on defining this profile, refer to Chapter 6, "Configuring the SNA Communication Package on AIX-Based Systems."

### Configuring CICS/ESA

The configuration tasks necessary for CICS/ESA are described in "Configuring CICS/ESA" on page 4-27. Be sure the VTAM logmode table in your CICS system contains an entry like the one in the $ORACLE_HOME/pg4appc/sna/orasyn62.asm file in the directory. Also be sure the CICS SESSION profile that is defined in the $ORACLE_HOME/pg4appc/demo/CICS directory is set up to use the logmode entry with sync level 2 supported. Note that CICS allows multiple SESSION profiles to be defined for the same CONNECTION profile. A second SESSION profile that specifies MODENAME(ORASYN62) can be defined so that both mode names are available for use with the gateway.
Post-Configuration Tasks

Perform these tasks after the configuration of the two-phase commit components.

Set Up RRM Startup and Shutdown Procedures

The RRM is a long-running detached process that must be started before any applications that use two-phase commit can be executed. The RRM is started up and shut down by two shell scripts, `rrmstart` and `rrmstop`, respectively. These scripts are located in the `/oracle/pga/9.2.0/bin` directory.

The `rrmstart` and `rrmstop` scripts must reference the same local LU name, because the LU name is unique to a particular RRM process. The RRM creates a file in the `/var/tmp` directory with a file name `luname.rmm`, where `luname` is the unqualified local LU name for which the RRM was started. This file is used to prevent multiple RRM processes from being started for the same local LU, and is also used by the `rrmstop` shell script to obtain the UNIX process ID of the RRM process.

Customizing the rrmstart script

You must customize the `rrmstart` script, which is in your `/oracle/pga/9.2.0/bin` directory, to set the ORACLE_HOME, TNS_ADMIN, and NLS_LANG environment variables as required for your installation. The `rrmstart` script contains comments describing how to do this. A sample of the `rrmstart` script is in Appendix A, "Gateway Initialization Parameters". Complete this customization before proceeding.

---

**Note:** If your `pg4arrm` executable is not working properly, refer to the security section of vendor documentation for information on how to set your security trusted groups names.

---

The RRM startup and shutdown can be set up in two ways. The first way is completely manual operation, where a system operator must issue the commands to start up the RRM after the system is started up and to shut down the RRM before the system is shut down. The second, and preferred, way is automatic operation, where the RRM is started automatically during the system startup processing, and is shut down either automatically or manually during system shutdown processing. By including command lines that call the `rrmstart` and `rrmstop` scripts in the appropriate system startup and shutdown files, the RRM can be started up and shut down automatically each time your system is started up and shut down.
Whether you choose to start up the RRM manually or automatically, you must ensure that the startup command is issued from the AIX user ID under which the gateway was installed.

Enter the following commands to start up the RRM manually, where `netname.luname` is the fully-qualified local LU name on which the RRM is to run:

```
$ cd /oracle/pga/9.2.0/bin
$ rrmstart netname.luname
```

Note that the RRM cannot be started until the Oracle Integrating Server into which it stores its LU6.2 log information has been started, and any required Oracle Net listeners have also been started.

Enter the following commands to shut down the RRM manually where `netname.luname` is the fully-qualified local LU name on which the RRM is running:

```
$ cd /oracle/pga/9.2.0/bin
$ rrmstop netname.luname
```

Note that the RRM should be shut down before the Oracle Integrating Server and Oracle Net listeners are shut down.

If you plan to use automatic RRM startup, then you must also use automatic database startup to start the Oracle Integrating Server to be used by the RRM, along with the Oracle Net listener if needed. In general, the `rrmstart` script should be invoked at the end of the startup procedure after the Oracle Integrating Server and Oracle Net listener have been started. For further information on automatic database startup, refer to Oracle9i Installation Guide.

To ensure that your RRM starts up when you restart the system, you can add an entry to the `/etc/inittab` file as outlined below. The `rmrstart` script brings up the RRM. The `/etc/inittab` script should only be executed as part of the AIX boot procedure.

To set up for automatic RRM startup, perform the following steps:

1. Change to superuser:

   ```
   $ su
   ```

   You are prompted by AIX for the root password.
2. Add the **rrmstart** script to your `/etc/inittab` file by entering the following command:

```bash
# mkitab "pg4arrm:2:wait:/bin/su oradba -c /oracle/pga/9.2.0/bin/rrmstart
netname.luname"
```

where:

- **oradba** is the DBA user ID under which you installed the gateway, and
- **netname.luname** is the fully-qualified local LU name on which the RRM executes.

3. Exit from superuser:

```bash
# exit
```

Oracle recommends that you do not modify the `/etc/shutdown` script to shutdown the RRM. Instead, make the following command part of your standard system shutdown procedure performed before executing the `/etc/shutdown` script, and before shutting down the Oracle Integrating Server and Oracle Net listener where **netname.luname** is the fully-qualified local LU name on which the RRM is executing:

```bash
rrmstop netname.luname
```

### RRM Recovery

Certain error conditions in the RRM require that recovery action be taken before the RRM can be successfully started. Most of the time, this is as simple as manually updating the RRM local LU log table, PGA_2PC_LUS. If severe problems occur, the PGA_2PC_LUS table can be dropped and recreated using the `SORACLE_HOME/pg4appc/admin/pga2pclu.sql` script. This forces the RRM to cold start the next time it is started. In such a case, all records of any pending gateway transactions that have not yet been resolved are erased, because a cold start of the RRM deletes all rows from the PGA_2PC_PENDING table.

---

**Note:** If you need to cold start the RRM, then first ensure that all pending transactions have been resolved.

---

The most common error situations are listed below, along with the recovery actions required.
RRM issued message PGA-21215

This message is issued when a partner LU sends an Exchange Log Names (XLN) request containing an LU6.2 log name that is different from that stored in the RRM’s PGA_2PC_LUS row for that partner LU. This can happen if, while the RRM was not active, the partner LU (CICS/ESA for example) was cold started and later warm started. If the RRM is subsequently warm started, then it is unaware that the partner LU was ever cold started, and must assume that the partner LU is using the wrong LU6.2 log.

The partner LU name is identified by the PGA-21110 message in the RRM’s log file that precedes the PGA-21215 message.

In this situation, when you have verified that the partner LU is running with the correct LU6.2 log, perform the following steps:

1. Obtain the Oracle server database specification, user ID, and password used by the RRM to connect to the Oracle server where it stores its LU6.2 log. These are identified in the RRM initialization file luname.ini in the SORACLE_HOME/pg4appc/admin directory, by the parameters LOG_DB, LOG_USER, and LOG_PASS, respectively.

2. Use SQL*Plus to connect to the Oracle server identified in Step 1 with the user ID and password obtained in Step 1.

3. Enter the following SQL statements where luname is the fully-qualified partner LU name from message PGA-21110:

```
SQL>DELETE FROM PGA_2PC_LUS WHERE LU_NAME = 'LUNAME';
SQL>COMMIT;
```

The next time a session is started from the RRM local LU to the partner LU, the partner LU’s new log name is accepted and stored in the RRM PGA_2PC_LUS table.

RRM issued message PGA-21218

This message is issued when a partner LU sends an Exchange Log Names (XLN) request indicating a cold start, and the RRM finds that there are possibly unresolved pending transactions for that partner LU in the PGA_2PC_PENDING table. The RRM then rejects the cold start from the partner LU. This means that no synclenvel 2 conversations can be initiated by the gateway with that partner LU. The partner LU name is provided in the PGA-21218 message text, along with the number of pending transactions found in the PGA_2PC_PENDING table.
In this situation, you should check each pending transaction for the partner LU and ensure that it has been resolved. Then you can delete the rows from the PGA_2PC_PENDING table for that partner LU. The next time a conversation is initiated by the gateway with that partner LU, the RRM accepts the XLN that is received from the partner LU. To resolve the pending transactions and clean up the PGA_2PC_PENDING table, perform the following steps:

1. Obtain the Oracle server database specification, user ID, and password used by the RRM to connect to the Oracle server where it stores its LU6.2 log. These are identified in the RRM initialization file `luname.ini` in the `$ORACLE_HOME/pg4appc/admin` directory, by the parameters LOG_DB, LOG_USER, and LOG_PASS, respectively.

2. Use SQL*Plus to connect to the Oracle server identified in Step 1 with the user ID and password obtained in Step 1.

3. Obtain a list of the pending transactions for the partner LU. To do this, enter the following SQL statement where `luname` is the fully-qualified partner LU name from message PGA-21218:

   ```sql
   SQL> SELECT * FROM PGA_2PC_PENDING WHERE LU_NAME = 'luname';
   ```

4. See "Manual Recovery of In-Doubt Transactions" on page 9-26 for information on how to resolve the pending transactions. Do not proceed until all transactions have been resolved.

5. Delete the pending transactions for the partner LU from the PGA_2PC_PENDING table. To do this, execute the following SQL statements:

   ```sql
   SQL> DELETE FROM PGA_2PC_PENDING WHERE LU_NAME = 'luname';
   SQL> COMMIT;
   ```

   where `luname` is the fully-qualified partner LU name from message PGA-21218.

The next time a session is started from the RRM’s local LU to the partner LU, the partner LU’s cold-start XLN is accepted.
Two-Phase Commit With CICS/ESA

The gateway supports two-phase commit with CICS/ESA but without any automatic recovery. As mentioned earlier in "Supported OLTPs (Online Transaction Processors)" on page 9-2, CICS/ESA has design constraints that limit its ability to participate fully in a two-phase commit. Specifically, CICS/ESA does not leave any resource update in an in-doubt state, except for recoverable TS queues. A transaction failure always results in either a commit or a rollback of all resources except recoverable TS queues. This action is based on the setting of the INDOUBT parameter in the CICS transaction definition. INDOUBT=COMMIT results in a commit upon a failure, and INDOUBT=BACKOUT or INDOUBT=WAIT result in a rollback upon a failure. The difference between INDOUBT=BACKOUT and INDOUBT=WAIT is that with INDOUBT=WAIT, recoverable TS queues will remain in-doubt until LU6.2 recovery occurs. Refer to vendor documentation for details on this action.

Because the gateway has no way of knowing which of the INDOUBT parameters was used for a particular CICS transaction definition, it cannot make any logical decisions during recovery about the state of any resources updated by the CICS transaction. This problem is compounded because any LU6.2 log information that CICS had regarding the transaction is deleted when CICS performs its INDOUBT processing. When a transaction has been committed or backed out, it is deleted from the log. Consequently, the gateway server cannot use the LU6.2 recovery facilities to query CICS as to the status of the transaction, because CICS no longer has any record of it.

When the Oracle Integrating Server instructs the gateway to recover a transaction, the gateway must respond to the Oracle Integrating Server with an indication that CICS took heuristic action, and that action is unknown. The result is that the DBA must perform manual recovery. Manual recovery is discussed later in "Manual Recovery of In-Doubt Transactions" on page 9-26.
Application Design Requirements

When designing two-phase commit applications which include Oracle Procedural Gateway for APPC, there are some requirements you must meet to minimize the exposures inherent with the CICS/ESA implementation of LU6.2 synchlevel 2. Meeting these requirements ensures that your applications are less likely to result in unresolved in-doubt Oracle transactions.

- Ensure CICS supports two-phase commit with your data store
  To support a two-phase commit application, CICS must be supported by your data store as a resource manager. Refer to your data store documentation to find out whether it supports two-phase commit using CICS as the resource manager. If this requirement is not met, then your data store cannot participate in a two-phase commit using the gateway.

- Use INDOUBT=BACKOUT for the CICS transactions
  In the CICS transaction definitions, specify INDOUBT=BACKOUT (or let it default to that value) so that CICS always rolls back after a failure. This ensures consistent behavior of your CICS transactions and simplifies the task of resolving any in-doubt Oracle transactions. If you use INDOUBT=COMMIT, then it might be impossible to ensure the integrity of your data, because CICS could commit its part of a transaction while the Oracle database rolled back its part.

- Use COMMIT_POINT_STRENGTH=255 for the gateway
  In the init<database>sid.ora file for the gateway, make sure that you specify COMMIT_POINT_STRENGTH=255. This ensures that the gateway receives the commit first among the participants in the Oracle transaction, after all parties have successfully prepared. The intent here is to minimize the window during which the CICS transaction could be rolled back after a failure when part of the Oracle transaction has already been committed. Because CICS rolls back automatically after a failure (if you have used INDOUBT=BACKOUT), it should never be given the opportunity to do so after any other part of the Oracle transaction has been committed.

- Do not update using a COMMIT-CONFIRM gateway
  Avoid performing updates to a COMMIT-CONFIRM gateway, such as one of the Oracle Transparent Gateways, in the same Oracle transaction that updates CICS resources through Oracle Procedural Gateway for APPC. The reason is that the COMMIT-CONFIRM gateway will always be committed first, which
opens the window of exposure described under the COMMIT_POINT_STRENGTH discussion.

- Do not update using multiple procedural gateways

  Avoid using multiple Oracle Procedural Gateway for APPC instances in the same Oracle transaction. Only one of them is committed first, opening the same window of exposure described under the COMMIT_POINT_STRENGTH discussion. If possible, do all of your updates through a single CICS transaction.

- Log update activity from your application

  Because recovery of in-doubt Oracle transactions (in which the gateway is a participant) is manual, you should log update information from your CICS transaction to simplify the recovery task. The ability to correlate a change made by the CICS transaction with the Oracle transaction of which that change was a part assists the DBA in determining how to resolve the in-doubt Oracle transaction.

  Oracle Procedural Gateway for APPC provides a useful piece of information to the CICS transaction at startup time when the conversation is at syncllevel 2. The LU6.2 logical unit of work identifier (LUWID) is provided as PIP data to the CICS transaction. If the CICS transaction logs the LUWID along with the changes that it made, then the LUWID can be used at recovery time to identify the Oracle transaction of which that LUWID was a part.

  A discussion of how to obtain and use the LUWID is provided in "Logging Two-Phase Commit Activity from CICS" on page 9-29.

In general, the two-phase commit gateway SID should be reserved for use only to invoke update transactions. Some extra overhead is involved in the setup for logging when PGA_CAPABILITY is set to 2_PHASE. Read-only transactions should be invoked through a separate gateway SID with PGA_CAPABILITY set to READ_ONLY so that they will not incur the extra overhead.
Manual Recovery of In-Doubt Transactions

This section describes the general procedure to follow when Oracle transactions involving Oracle Procedural Gateway for APPC are in an in-doubt state and require manual recovery. For further information on Oracle facilities for resolving transactions, refer to the appropriate Oracle server documentation, such as *Oracle9i Administrator’s Reference*.

**Caution:** The procedure described in this section is provided here with the assumption that your application was written in strict accordance with the requirements discussed in "Application Design Requirements" on page 9-24. This recovery procedure is not sufficient for resolution of in-doubt transactions if your application has not met those requirements.

If an Oracle transaction in which Oracle Procedural Gateway for APPC was involved is left in an in-doubt state, then it must be resolved manually by the database administrator.

Table 9–2 lists failure points and their resulting Oracle states and resulting gateway states. The table uses the failure point numbers from Figure 9–1, “Two-phase Commit Flow with Synclevel 2” on page 9-7. Note that a gateway state of "none" means that the gateway has completed COMMIT processing with CICS and has deleted the PGA_2PC_PENDING row for that transaction.

**Table 9–2  In-Doubt Transactions: Failure Points, Pending Transaction States**

<table>
<thead>
<tr>
<th>Failure Points</th>
<th>Resulting Oracle State</th>
<th>Resulting Gateway State</th>
</tr>
</thead>
<tbody>
<tr>
<td>3, 4, 5, 6, 7</td>
<td>COLLECTING</td>
<td>COLLECTING</td>
</tr>
<tr>
<td>8, 9, 10</td>
<td>PREPARED</td>
<td>PREPARED</td>
</tr>
<tr>
<td>11, 12</td>
<td>PREPARED</td>
<td>None</td>
</tr>
<tr>
<td>13</td>
<td>COMMITTED</td>
<td>None</td>
</tr>
</tbody>
</table>

The transaction can be resolved using a step-by-step approach. The following steps provide all of the information necessary to resolve the transaction:

1. Obtain the Oracle Local Transaction ID of the in-doubt transaction from the ORA-02050 or ORA-02053 message received by the application at the time of the failure.
2. Use SQL*Plus to log on to the Oracle Integrating Server as user SYSTEM.

3. Enter the following SQL statement where \texttt{ltid} is the Oracle Local Transaction ID of the in-doubt transaction that you obtained in Step 1:

\begin{verbatim}
SQL>SELECT GLOBAL_TRAN_ID, STATE FROM DBA_2PC_PENDING
WHERE LOCAL_TRAN_ID = 'ltid';
\end{verbatim}

The query provides for you the Oracle Global Transaction ID of the in-doubt transaction, and the state of the transaction.

4. Obtain the Oracle logging server database specification, user ID, and password used by the gateway server for its LU6.2 logging. These items are identified by the PGA\_LOG\_DB, PGA\_LOG\_USER, and PGA\_LOG\_PASS parameters in the gateway initialization file \texttt{initsid.ora}.

5. Use SQL*Plus to log on to the Oracle logging server using the information from Step 4.

6. Enter the following SQL statement where \texttt{gtid} is the Oracle Global Transaction ID of the in-doubt transaction that you obtained in Step 3:

\begin{verbatim}
SELECT LUW_ID, STATE FROM PGA_2PC_PENDING
WHERE GLOBAL_TRAN_ID = 'gtid';
\end{verbatim}

The query provides for you a single LUW_ID value and a single STATE value. The LUW_ID value appears in printable hexadecimal, because the LUW_ID column is a RAW. This value is the LU6.2 Logical Unit of Work Identifier (LUWID) of the gateway portion of the transaction. The STATE value is a character string, and is either "COLLECTING" or "PREPARED". If this query returns no data, then the gateway portion of the transaction was successfully committed.

7. Based on the results of the queries in Steps 3 and 6, perform the appropriate recovery action using Table 9–3 below. The Oracle state is the value of the STATE column from the DBA\_2PC\_PENDING table; the gateway state is the value of the STATE column from the PGA\_2PC\_PENDING table. The LUWID you obtained in Step 6 is necessary if you need to check your CICS application’s log information to determine what action CICS performed on the transaction.

\textbf{Table 9–3 Recovery Actions for In-Doubt Transactions}

<table>
<thead>
<tr>
<th>Oracle State</th>
<th>Gateway State</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLLECTING</td>
<td>COLLECTING</td>
<td>Issue ROLLBACK FORCE on the Oracle Integrating Server.</td>
</tr>
</tbody>
</table>
Note that the CICS application’s log information is required when both the Oracle state and the gateway state for the transaction are PREPARED. The following section, "Logging Two-Phase Commit Activity from CICS" provides information on how to log this information from the CICS application.
Logging Two-Phase Commit Activity from CICS

This section provides all the information necessary for implementation of a simple two-phase commit logging scheme for your CICS applications. Every sample file referenced in this section resides in the `$ORACLE_HOME/pg4appc/demo/CICS` directory of the gateway installation. The DB2 sample update transaction, DB2, that is provided with the gateway implements this logging scheme. Refer to the source for that transaction in `pgadb22.cob` for a complete example.

Within an LU6.2 conversation running at synclevel 2, all activity that occurs between the start of the transaction and the first COMMIT or ROLLBACK, or between a COMMIT or ROLLBACK and the next COMMIT or ROLLBACK, is known as a Logical Unit of Work (LUW). Within the execution of the transaction, a new LUW begins after each COMMIT or ROLLBACK. Some transactions consist of a single LUW, while others consist of many LUWs. Each LUW has a unique identifier known as the Logical Unit of Work Identifier (LUWID). The format of the LU6.2 LUWID is defined by the SNA architecture. Following is the format of the LUWID:

- **Byte 0** is the length of fully-qualified LU name (1 byte);
- **Bytes 1 - n** is a fully-qualified LU name (3 to 17 bytes);
- **Bytes n+1 - n+6** is an LUW instance number (6 bytes), and
- **Bytes n+7 - n+8** is LUW sequence number (2 bytes)

The minimum length of the LUWID is 12 bytes, and the maximum is 26 bytes. The last field, the LUW sequence number, is of special interest. Because the LUWID is generated at the start of a conversation, and because a conversation might contain more than one LUW, there must be a way to generate a unique LUWID for each LUW. The LUW sequence number is designed to do that. After each COMMIT or ROLLBACK done in the application, the LUW sequence number must be incremented to ensure that the next LUW has a unique LUWID.

Logging of the LUWID from your CICS application is the only possible way to relate your CICS activity back to the Procedural Gateway and Oracle server two-phase commit logs, `PGA_2PC_PENDING` and `DBA_2PC_PENDING`. The goal is to make the logging as simple as possible, and to make recovery as simple and straightforward as possible. The logging scheme described in this section requires the following:

- **LUWID log table or file**: A new table or file must be defined in your data store to be used for logging LUWID information. This is referred to as the LUWID log.
The LUWID log in your data store needs to contain only two fields, a date/time stamp and the actual LU6.2 LUWID. The date/time stamp is used for deleting old entries, and it can be in any format you choose. The LUWID is a variable-length character string from 24 to 52 bytes in length, and must be the key. Its format is a printable hex string.

You can use a single LUWID log for all of your applications, or a different one for each application. If you use a different one for each application, then you need to know to which application a particular in-doubt transaction belongs. You can use the TP_NAME and LU_NAME columns of the PGA_2PC_PENDING table to help identify the application. Using a single LUWID log for all applications might be a simpler approach, because there would be a single place to look for recovery information on the CICS side of the transaction.

**EXEC CICS EXTRACT PROCESS:** An EXEC CICS EXTRACT PROCESS call must be issued in your CICS application initialization to obtain a pointer to the PIP data provided by the gateway server. The sample DB2 update transaction provided in the `pgadb22.cob` sample file contains an example of using this call. For further information on this call, refer to IBM documentation.

**Extract LU6.2 LUWID:** The LU6.2 LUWID must be extracted from the PIP data provided by the gateway server. A sample assembler subroutine, `PGALUWID`, is provided in the `pgaluwid.asm` sample file. This subroutine can extract the LU6.2 LUWID from the PIP data, and should be called immediately after the EXEC CICS EXTRACT PROCESS call has successfully completed. An example of how to call `PGALUWID` to extract the LUWID is in the sample DB2 update transaction provided in the `pgadb22.cob` sample file.

**Insert Into LUWID Log:** You must add code to your CICS application to insert a new row or record into your LUWID log as part of each unit of work. You must ensure that this insert operation is part of the same LUW as the update to your data store. This guarantees that the presence or absence of a LUWID log entry for a particular LUWID is enough to determine the state of the LUW. If the log entry does not exist, then the LUW was rolled back. If the log entry exists, then the LUW was successfully committed. The sample DB2 update transaction in the `pgadb22.cob` sample file contains an example of logging the LUWID for DB2 updates.

**Increment LU6.2 LUW Sequence Number:** If your CICS transaction consists of only a single LUW, then this step is not necessary. Otherwise, the LUW sequence number must be incremented after each EXEC CICS SYNCPOINT or EXEC CICS ROLLBACK call. This can be done by calling `PGALUWID`. An
example of how to call PGALUWID to increment the LUW sequence number is in the sample DB2 update transaction in the pgadb22.cob sample file.

PGALUWID Subroutine

A sample assembler subroutine, PGALUWID, is provided in the pgaluwid.asm sample file. This subroutine can be called by a CICS transaction during initialization to extract the LU6.2 LUWID from the PIP data received from Oracle Procedural Gateway for APPC server, and later after each EXEC CICS SYNCPOINT or EXEC CICS ROLLBACK to increment the LUW sequence number as required by LU6.2. This subroutine is fully reentrant and reusable, so it is compatible with any type of CICS transaction program.

The JCL to assemble this subroutine and linkedit it into your CICS library is provided in the pgaluwid.jcl sample file. There are comments in the JCL to guide you in tailoring the JCL for your system.

The parameter to this subroutine is a pointer to the COBOL structure defined in the pgaluwid.cob sample file. This file is a COBOL copybook that should be included in each CICS transaction program that is using two-phase commit and that calls the PGALUWID subroutine to maintain logging information. The pgadb22.cob sample file uses this copybook.

The structure is used by PGALUWID for both input and output. The following example shows a listing of the COBOL structure definition:

```cob
01 PGALUWID.
   05 FILLER       PIC X(80).  subroutine work area
   05 PIPPTR       POINTER.   pointer to PIP data
   05 PIPLEN       PIC S9(4). length of PIP data
   05 REQUEST      PIC X(1).  request code
   05 FILLER       PIC X(3).  for alignment
   05 LUWSEQNO     PIC S9(4). LUW sequence number
   05 LUWSEQPT     POINTER.   pointer to LUW sequence
   05 LUWIDLEN     PIC S9(4). length of LUWID
   05 LUWIDDAT     PIC X(52). LUWID (printable hex)
```

The two calls to PGALUWID are "get PIP data" and "increment LUW sequence". A discussion of input and output fields follows. Note that the output fields must not be changed by the CICS application, or subsequent calls to PGALUWID will fail.
The "get PIP data" call copies the PIP data received from Oracle Procedural Gateway for APPC server into an area usable by the CICS application for logging the LUWID. This call can be used only once during a single execution of the CICS transaction, and must be used before any two-phase commit activity is performed. Table 9–4 lists the input and output information necessary for each field in the PGALUWID "get PIP data" call.

Table 9–4  PGALUWID Structure Fields Usage on "get PIP data" Call

<table>
<thead>
<tr>
<th>Direction</th>
<th>Field</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>REQUEST</td>
<td>Must be set to &quot;G&quot;.</td>
</tr>
<tr>
<td>Input</td>
<td>PIPPTR</td>
<td>Must be set to the address of the PIP data. This is obtained with the EXEC CICS EXTRACT PROCESS call, with PIPLIST(PIPPTR) specified to set PIPPTR to the address of the PIP data.</td>
</tr>
<tr>
<td>Input</td>
<td>PIPLEN</td>
<td>Must be set to the length of the PIP data. This is obtained with the EXEC CICS EXTRACT PROCESS call, with PIPLength (PIPLEN) specified to set PIPLEN to the length of the PIP data.</td>
</tr>
<tr>
<td>Output</td>
<td>LUWSEQNO</td>
<td>Must be set to the address of the PIP data. This is obtained with the EXEC CICS EXTRACT PROCESS call, with PIPLIST(PIPPTR) specified to set PIPPTR to the address of the PIP data.</td>
</tr>
<tr>
<td>Output</td>
<td>LUWSEQPT</td>
<td>Must be set to the address of the last two bytes of the LUWID field, which contain the LUW sequence number in printable hex.</td>
</tr>
<tr>
<td>Output</td>
<td>LUWIDLEN</td>
<td>Must be set to the length of the LUWID field.</td>
</tr>
<tr>
<td>Output</td>
<td>LUWIDDAT</td>
<td>Must be set to the LUWID in printable hex.</td>
</tr>
</tbody>
</table>
The "increment LUW sequence" call increments the LUW sequence number by one, and stores the resulting value in the LUWID to be logged by the CICS application. This call must be used after each EXEC CICS SYNCPOINT and EXEC CICS ROLLBACK call in the CICS transaction so that each new LUW has a unique LUWID. Failure to do this results in LU6.2 errors. The following table lists the input and output information necessary for each field in the PGALUWID "increment LUW sequence" call.

<table>
<thead>
<tr>
<th>Direction</th>
<th>Field</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>REQUEST</td>
<td>Must be set to 'I'.</td>
</tr>
<tr>
<td>Input</td>
<td>LUWSEQNO</td>
<td>The value that was set by the &quot;get PIP data&quot; call.</td>
</tr>
<tr>
<td>Input</td>
<td>LUWSEQPT</td>
<td>The value that was set by the &quot;get PIP data&quot; call.</td>
</tr>
<tr>
<td>Input</td>
<td>LUWIDDAT</td>
<td>The value that was set by the &quot;get PIP data&quot; call.</td>
</tr>
<tr>
<td>Output</td>
<td>LUWSEQNO</td>
<td>Incremented by one.</td>
</tr>
<tr>
<td>Output</td>
<td>LUWIDDAT</td>
<td>Has its last two bytes set to the new LUW sequence number in printable hex.</td>
</tr>
</tbody>
</table>

**Periodic Log Cleanup**

Because every successfully committed update by your CICS transaction results in another row being inserted into your LUWID log, you need to implement a periodic cleanup scheme for the LUWID log. The easiest way to do this is to design your log, as suggested previously in this section, to include a time/date stamp. Your cleanup procedure can use the time/date stamp to remove old log entries. You can define any time criteria you want for this. If your applications are monitored closely and in-doubt transactions are resolved quickly, five days should be a sufficient criterion for aging off log entries. You can set up a job to run nightly to delete all log entries older than five days.
Migration and Coexistence with Existing Gateways

Migrating to new instances of Oracle Procedural Gateway for APPC from an existing installation is straightforward, provided you follow some guidelines. This chapter provides information to make these new installations as easy as possible.

This chapter provides information that is specific to this release of Oracle Procedural Gateway for APPC for UNIX. It contains the following sections:

- Migrating An Existing Gateway Instance to New Release on page 10-2
- Backout Considerations When Migrating to New Releases on page 10-2
- Parameter Changes for Version 4 to Release 9i of the Gateways on page 10-2
- Parameter Changes for Version 8 or Earlier to Oracle9.2.0 Gateway on page 10-5
Migrating An Existing Gateway Instance to New Release

Follow these steps to migrate an existing gateway to Oracle Procedural Gateway for APPC, Release 9.2.0.1.0.

**Step 1: Install the new Release**

Install the new release of the gateway in a separate directory as outlined in Chapter 4, "Installing and Configuring the Gateway".

---

Caution: Do not install the Gateway over a previously existing Gateway installation. Doing so will corrupt the existing installation.

---

**Step 2: Transferring init$SID.ora Gateway Initialization File Parameters**

Copy the *init$SID.ora* file from the old gateway instance to the new instance. PGA_TRACE is not supported by Oracle Procedural Gateway for APPC server Release 9.2.0.1.0; use TRACE_LEVEL instead.

---

Note: If you use TRACE_LEVEL, you must set the path for the LOG_DESTINATION parameter.

---

**Backout Considerations When Migrating to New Releases**

Oracle Corporation recommends that you keep the old gateway Oracle home directory and instance configurations intact and operational when you are installing a new release of the gateway and upgrading existing instances, in case there are problems with the upgrade. This will help ensure minimal downtime between changes to different gateway instances.

**Parameter Changes for Version 4 to Release 9i of the Gateways**

This release of Oracle Procedural Gateway for APPC introduces new and changed initialization parameters if you are migrating from a Version 4 gateway to Oracle 9i.
This release of Oracle Procedural Gateway for APPC introduces new initialization parameters. The following section lists the new parameters relevant to migration from Version 4 gateways.

**New Startup Shell Script Parameters**
The following parameters are in the startup shell script `pg4hoa1.sh`:
- FDS_CLASS
- FDS_INSTANCE

**New Gateway Initialization Parameters**
The following parameters are in the gateway initialization file (`init<sid>.ora`):
- HS_RPC_FETCH_REBLOCKING
- HS_FDS_FETCH_ROWS
- HS_RPC_FETCH_SIZE
- HS_NLS_NCHAR
- LOG_DESTINATION
- TRACE_LEVEL

**Note:** The "Parameter Changes for Version 4 to Release 9i of the Gateways" section does not apply to you if you are migrating to Oracle 9i from Version 8 of Oracle Procedural Gateway for APPC.

**Note:** The "HS_" parameters are specific to Oracle Heterogeneous Services. For details on HS parameters, refer to the *Oracle9i Database Administrator’s Guide*. 
Renamed Startup Shell Script Parameters
The following table lists the startup shell script \texttt{(pg4hoa1.sh)} parameters that have been renamed in this release of the gateway, and shows their old names:

<table>
<thead>
<tr>
<th>New Name</th>
<th>Old Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>HS_COMM_STRENGTH_POINT</td>
<td>COMMIT_STRENGTH_POINT</td>
</tr>
<tr>
<td>HS_DB_DOMAIN</td>
<td>DB_DOMAIN</td>
</tr>
<tr>
<td>HS_DB_INTERNAL_NAME</td>
<td>DB_INTERNAL_NAME</td>
</tr>
<tr>
<td>HS_DB_NAME</td>
<td>DB_NAME</td>
</tr>
<tr>
<td>HS_DESCRIBE_CACHE_HWM</td>
<td>DESCRIBE_CACHE_HWM</td>
</tr>
<tr>
<td>HS_LANGUAGE</td>
<td>LANGUAGE</td>
</tr>
</tbody>
</table>

Renamed Gateway Initialization File Parameters
The following table lists the gateway initialization file \texttt{(init\_sid.ora)} parameters that have been renamed in this release of the gateway, and shows their old names:

<table>
<thead>
<tr>
<th>New Name</th>
<th>Old Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>HS_NLS_DATE_FORMAT</td>
<td>NLS_DATE_FORMAT</td>
</tr>
<tr>
<td>HS_NLS_DATE_LANGUAGE</td>
<td>NLS_DATE_LANGUAGE</td>
</tr>
<tr>
<td>HS_OPEN_CURSORS</td>
<td>OPEN_CURSORS</td>
</tr>
<tr>
<td>HS_ROWID_CACHE_SIZE</td>
<td>ROWID_CACHE_SIZE</td>
</tr>
</tbody>
</table>

Obsolete Parameters
The following parameters are now obsolete. Please remove them from your configuration files:

- MODE
- SERVER\_PATH
- ERROR\_LOGGING
- ERROR\_REPORTING
- ERRORTAG
- GATEWAY\_SID
Parameter Changes for Version 8 or Earlier to Oracle9.2.0 Gateway

The following startup shell script parameter must be added to `pg4hoa1.sh` if you are migrating from a Version 4 or Version 8 gateway to Release 9i of Oracle Procedural Gateway for APPC:

- **FDS_CLASS_VERSION**
Migrating from Oracle9.0.1 to Oracle9.2.0

No new parameters were added between release 9.0.1 and release 9.0.2.

Oracle Net Considerations

The gateway uses the Heterogeneous Services (HS) facilities of Oracle and Oracle Net. If you are upgrading from a version 4 gateway, then you need to slightly modify the gateway service name entries in the `tnsnames.ora` file. Add an `(HS=)` clause to tell Oracle Net that the gateway uses HS facilities. For more information, refer to "Configuring Your Network" on page 4-24.
This appendix describes the gateway initialization file location and lists the gateway initialization parameters that are supported by Oracle Procedural Gateway for APPC. These parameters are fully documented in Chapter 10, "Migration and Coexistence with Existing Gateways". This appendix also describes the PGA parameters which control the APPC interface portion of the gateway and it contains sample pg4hoa1.sh and initpga.ora files. It also contains sample AIX-Based Systems only rrmstart and HQWM920.ini files.

This appendix contains the following sections:

- Gateway Initialization Parameter File on page A-2
- PGA Parameters on page A-2
- Sample pg4hoa1.sh (Boot Parameter) File on page A-15
- Sample initPGA.ora File on page A-18
- AIX-Based Systems Only: Sample rrmstart Script on page A-19
- AIX-Based Systems Only: Sample HQWM920.ini File on page A-20
The parameter file for Oracle Procedural Gateway for APPC is located in the $ORACLE_HOME/pg4appc/admin directory and is called initсид.ora.

PGA Parameters

The PGA parameters control the APPC interface portion of the gateway. PGA parameters are specified using the SET gateway initialization parameter. For example:

SET pga_parm=value

where:

- pga_parm is one of the PGA parameter names in the list that follows
- value is a character string with contents that depend on pga_parm

PGA Parameters on AIX

Table A–2 provides a list of PGA parameters and their descriptions for AIX.

Table A–1  PGA Parameters on AIX

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG_DESTINATION=logpath</td>
<td>logpath specifies the destination at which STDERR is reopened. LOG_DESTINATION specifies a directory only and STDERR is reopened to logpath/sid_pid.log where:</td>
</tr>
<tr>
<td></td>
<td>■ sid is the sid name</td>
</tr>
<tr>
<td></td>
<td>■ pid is the process ID assigned to the gateway</td>
</tr>
</tbody>
</table>
PGA Parameters

Gateway Initialization Parameters

Table A–1 (Cont.) PGA Parameters on AIX

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| PGA_CAPABILITY  | PGA transaction capability. This controls whether updates are allowed through the gateway. The following are valid values: READ_ONLY or RO - read-only capabilities. SINGLE_SITE or SS - single-site update only. This indicates that in a distributed environment, only the gateway can perform updates. No other database updates can occur within the Oracle transaction. COMMIT_CONFIRM or CC - commit-confirm. This indicates that in a distributed environment, updates can be performed by both the gateway and other participants within the Oracle transaction. The gateway is always committed first in this mode, and no other commit-confirm sites are allowed to participate in the Oracle transaction.
| PG A_CONFIRM    | Incoming APPC CONFIRM request handling option. This controls what the gateway does when an APPC CONFIRM request is received from the remote transaction program. This parameter has meaning only when the conversation is running with SYNCLEVEL > 0. The following are valid values: ACCEPT - respond to incoming APPC CONFIRM requests with APPC CONFIRMED responses. REJECT - treat incoming APPC CONFIRM requests as errors causing the conversation to be de-allocated and an error message to be issued. |
| PGA_LOCAL_LU    | The SNA local LU name on which the gateway executes. This is the fully-qualified local LU name as defined in the local LU definition, and can be from 3 to 17 characters long in the form network.luname, where network is the SNA network name and luname is the local LU name. This parameter is required only when PGA_CAPABILITY is set to 2_PHASE. There is no default value. |
### Table A–1 (Cont.) PGA Parameters on AIX

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PGA_LOG_DB</td>
<td>The Oracle Net service name for the Oracle server in which the gateway maintains its transaction log. This parameter can be from 1 to 255 characters long. This parameter is required only when PGA_CAPABILITY is set to 2_PHASE or COMMIT_CONFIRM. There is no default value.</td>
</tr>
<tr>
<td>PGA_LOG_PASS</td>
<td>The Oracle password to be used by the gateway when connecting to the Oracle server specified by the PGA_LOG_DB parameter. The password can be from 1 to 30 characters long. This parameter is required only when PGA_CAPABILITY is set to 2_PHASE or COMMIT_CONFIRM. For more information, refer to “Using the pg4pwd Utility” on page 5-5. There is no default value.</td>
</tr>
<tr>
<td>PGA_LOG_USER</td>
<td>The Oracle user ID to be used by the gateway when connecting to the Oracle server specified by the PGA_LOG_DB parameter. The user ID can be from 1 to 30 characters long. This parameter is required only when PGA_CAPABILITY is set to 2_PHASE or COMMIT_CONFIRM. There is no default value.</td>
</tr>
<tr>
<td>PGA_RECOVERY_PASS</td>
<td>The password to be used by the gateway when allocating an APPC conversation with the transaction specified by the PGA_RECOVERY_TPNAME parameter. The password can be from 1 to 8 characters long. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM and PGA_SECURITY_TYPE is set to PROGRAM. For more information, refer to “Using the pg4pwd Utility” on page 5-5. There is no default value.</td>
</tr>
<tr>
<td>PGA_RECOVERY_TPNAME</td>
<td>The TP name of the transaction installed in the OLTP for commit-confirm FORGET and RECOVERY processing. The TP name can be from 1 to 64 characters long. For CICS/ESA, the TP name is limited to 4 characters. For IMS/TM, the TP name is limited to 8 characters. Other OLTPs might have other limits on the length of the TP name. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM. The default value is RECO.</td>
</tr>
</tbody>
</table>
### Table A–1 (Cont.) PGA Parameters on AIX

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PGA_RECOVERY_USER</td>
<td>The user ID to be used by the gateway when allocating an APPC conversation with the transaction specified by the PGA_RECOVERY_TPNAME parameter. The user ID can be from 1 to 8 characters long. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM and PGA_SECURITY_TYPE is set to PROGRAM or SAME. There is no default value.</td>
</tr>
<tr>
<td>PGA_SECURITY_TYPE</td>
<td>APPC conversation security option. This controls what security parameters are sent to the OLTP in the FMH-5 at conversation allocation. The following are valid values: NONE - which sends no security parameters SAME - which sends only a user ID PROGRAM - which sends a user ID and password The default is NONE. For further information on these options, refer to Chapter 5, &quot;Security Requirements&quot;.</td>
</tr>
<tr>
<td>PGA_SIGDANGER</td>
<td>Action to take upon receipt of a SIGDANGER signal from the system indicating a shortage of paging space. The following are valid values: DEALLOCATE: - which de-allocates all active conversations IGNORE - which ignores the signal The default is IGNORE.</td>
</tr>
<tr>
<td>TRACE_LEVEL</td>
<td>PGA trace level. This controls tracing output written to stderr (the target of the LOG_DESTINATION parameter.) The value must be an integer from 0 to 255. The default is 0, indicating no tracing.</td>
</tr>
</tbody>
</table>
PGA Parameters on HP-UX and Solaris

Table A-2 provides a list of PGA parameters and their descriptions, for HP-UX and Solaris systems.

Table A-2 PGA Parameters on HP-UX and Solaris

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| LOG_DESTINATION=\texttt{logpath} | \texttt{logpath} specifies the destination at which STDERR is reopened. LOG_DESTINATION specifies a directory only and stderr is reopened to \texttt{logpath/sid\_pid.log} where:  
  \begin{itemize}
    \item \texttt{sid} is the sid name
    \item \texttt{pid} is the process ID assigned to the gateway
  \end{itemize} |
| PGA_CAPABILITY     | PGA transaction capability. This controls whether updates are allowed through the gateway. The following are valid values:  
  \begin{itemize}
    \item READ\_ONLY or RO - read-only capabilities.
    \item SINGLE\_SITE or SS - single-site update only. This indicates that in a distributed environment, only the gateway can perform updates. No other database updates can occur within the Oracle transaction.
    \item COMMIT\_CONFIRM or CC - commit-confirm. This indicates that in a distributed environment, updates can be performed by both the gateway and other participants within the Oracle transaction. The gateway is always committed first in this mode, and no other commit-confirm sites are allowed to participate in the Oracle transaction.
  \end{itemize}  
The default is SINGLE\_SITE. |
Table A–2 (Cont.) PGA Parameters on HP-UX and Solaris

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| PGA_CONFIRM   | Incoming APPC CONFIRM request handling option. This controls what the gateway does when an APPC CONFIRM request is received from the remote transaction program. This parameter has meaning only when the conversation is running with SYNCELEVEL > 0. The following are valid values:  
  ACCEPT - respond to incoming APPC CONFIRM requests with APPC CONFIRMED responses.  
  REJECT - treat incoming APPC CONFIRM requests as errors causing the conversation to be de-allocated and an error message to be issued.  
  The default is REJECT.                                                      |
| PGA_LOG_DB    | The Oracle Net service name for the Oracle server in which the gateway maintains its transaction log. This parameter can be from 1 to 255 characters long. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM.  
  There is no default value.                                                   |
| PGA_LOG_PASS  | The Oracle password to be used by the gateway when connecting to the Oracle server specified by the PGA_LOG_DB parameter. The password can be from 1 to 30 characters long. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM.  
  For more information, refer to "Using the pg4pwd Utility" on page 5-5.     
  There is no default value.                                                   |
| PGA_LOG_USER  | The Oracle user ID to be used by the gateway when connecting to the Oracle server specified by the PGA_LOG_DB parameter. The user ID can be from 1 to 30 characters long. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM.  
  There is no default value.                                                   |
PGA Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PGA_RECOVERY_PASS</td>
<td>The password to be used by the gateway when allocating an APPC conversation with the transaction specified by the PGA_RECOVERY_TPNAME parameter. The password can be from 1 to 8 characters long. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM and PGA_SECURITY_TYPE is set to PROGRAM. For more information, refer to “Using the pg4pwd Utility” on page 5-5. There is no default value.</td>
</tr>
<tr>
<td>PGA_RECOVERY_TPNAME</td>
<td>The TP name of the transaction installed in the OLTP for commit-confirm FORGET and RECOVERY processing. The TP name can be from 1 to 64 characters long. For CICS/ESA, the TP name is limited to 4 characters. For IMS/TM, the TP name is limited to 8 characters. Other OLTPs might have other limits on the length of the TP name. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM. The default value is RECO.</td>
</tr>
<tr>
<td>PGA_RECOVERY_USER</td>
<td>The user ID to be used by the gateway when allocating an APPC conversation with the transaction specified by the PGA_RECOVERY_TPNAME parameter. The user ID can be from 1 to 8 characters long. This parameter is required only when PGA_CAPABILITY is set to COMMIT_CONFIRM and PGA_SECURITY_TYPE is set to PROGRAM or SAME. There is no default value.</td>
</tr>
<tr>
<td>PGA_SECURITY_TYPE</td>
<td>APPC conversation security option. This controls what security parameters are sent to the OLTP in the FMH-5 at conversation allocation. The following are valid values: NONE -which sends no security parameters SAME -which sends only a user ID PROGRAM -which sends a user ID and password The default is NONE. For further information on these options, refer to Chapter 5, &quot;Security Requirements&quot;.</td>
</tr>
</tbody>
</table>
PGA Parameters

Table A–2  (Cont.) PGA Parameters on HP-UX and Solaris

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRACE_LEVEL</td>
<td>PGA trace level. This controls tracing output written to stderr (the target of the LOG_DESTINATION parameter.) The value must be an integer from 0 to 255. The default is 0, indicating no tracing.</td>
</tr>
</tbody>
</table>

PGA_CAPABILITY Parameter Considerations: AIX-Based Systems

When choosing a setting for the PGA_CAPABILITY parameter, take care to ensure that the correct setting is used based on what the remote transaction programs will be doing.

The READ_ONLY setting should always be used when the remote transaction programs are read-only, that is, when the remote transaction programs perform no database updates. READ_ONLY should never be used when the remote transaction programs perform database updates. For example, if the READ_ONLY setting is chosen, and if a remote transaction program invoked by the gateway performs updates to a foreign database, then the Oracle Integrating Server does not provide any integrity protection for those updates. Furthermore, READ_ONLY mode allows a gateway transaction to be part of a distributed transaction that might update several other databases. If the gateway invokes a remote transaction program that performs updates in this situation, and if a failure occurs, then the database updated by the remote transaction program is out of synch with the other databases.

In cases where the remote transaction programs perform updates to foreign databases, there are three options for PGA_CAPABILITY:

- SINGLE_SITE
- COMMIT_CONFIRM
- 2_PHASE

Each of these options provides protection against data integrity problems by allowing COMMIT and ROLLBACK requests to be forwarded to the remote transaction program, and by informing the Oracle Integrating Server about the distributed update and recovery capabilities of the gateway. The particular option chosen depends upon the design of the remote transaction programs and upon the capabilities of the OLTP (online transaction processor) where they execute.

If the OLTP has LU6.2 SYNCLEVEL 2 support, then the 2_PHASE capability provides the most flexible solution, allowing full two-phase commit between the
Oracle Integrating Server and the OLTP. If the OLTP has only LU6.2 SYNCLEVEL 1 support, then the COMMIT_CONFIRM capability provides limited two-phase commit between the Oracle Integrating Server and the OLTP, with the restriction that no other commit-confirm site (gateway or Oracle) can be part of the distributed transaction. Only full two-phase commit sites can join a distributed transaction that includes a commit-confirm gateway. If it is not possible to use COMMIT_CONFIRM, then the SINGLE_SITE capability provides update capability between the Oracle Integrating Server and the OLTP, with the restriction that only the OLTP can perform updates, and no updates can occur on the Oracle side.

Each of the PGA_CAPABILITY options for update control imposes specific requirements on the remote transaction program and on the OLTP. For 2_PHASE capability, these requirements are discussed in detail in Chapter 9, “AIX-Based Systems Only: Implementing Two-Phase Commit”. For COMMIT_CONFIRM capability, these requirements are discussed in detail in Chapter 5, “Implementing Commit-Confirm,” of the Oracle Procedural Gateway for APPC User’s Guide. For SINGLE_SITE capability, the remote transaction program is responsible for performing the appropriate tasks in response to COMMIT and ROLLBACK requests received from the gateway on behalf of the Oracle Integrating Server. The gateway uses the APPC CONFIRM and SEND_ERR requests to implement COMMIT and ROLLBACK, respectively. Upon receipt of a CONFIRM, the remote transaction program must perform COMMIT processing and then respond to the gateway with an APPC CONFIRMED response. Upon receipt of a SEND_ERR, the remote transaction program must perform ROLLBACK processing.

Because the distributed transaction capability of the Oracle Integrating Server is affected by the PGA_CAPABILITY option used by the gateway, it is desirable to separate inquiry and update applications by using different gateway instances for each. One gateway can be defined with PGA_CAPABILITY set to READ_ONLY and others with PGA_CAPABILITY set to SINGLE_SITE, COMMIT_CONFIRM, or 2_PHASE.

This allows read-only transaction programs to participate in distributed transactions under the control of the Oracle Integrating Server. For example, data from DB2 can be retrieved through the READ_ONLY gateway by an inquiry-only remote transaction program, and can then be used as input to database updates on the Oracle Integrating Server, all in one Oracle transaction. A SINGLE_SITE gateway can be used only for accessing remote transaction programs which perform updates to foreign databases outside the scope of the Oracle Integrating Server’s control. Data can be read from any databases accessible to the Oracle Integrating Server, and that data can be used to perform updates through the gateway.
When it is necessary to update resources on both the Oracle side and the OLTP side, a COMMIT_CONFIRM or 2_PHASE gateway can be used, provided that the OLTP and the remote transaction programs are set up to implement commit-confirm.

All that is necessary to set up multiple gateway instances is to set up the following for each instance:

- an entry in the `listener.ora` file defining the `sid` of the gateway instance
- an entry in the `tnsnames.ora` file defining an alias to be used to connect to the gateway instance defined in `listener.ora`
- a database link in the Oracle Integrating Server that specifies the alias defined in the `tnsnames.ora` file in its USING parameter
- initialization variables in the bootstrap shell script (for example, `pg4hoa1.sh`)

Note that the gateway instances can share one common directory structure, and use the same executables.

For example, to set up two gateways, PGAI and PGAU (for inquiry and update use, respectively) the following steps are required:

1. Define entries in `listener.ora` for two `sids`, PGAI and PGAU.
2. Define two aliases in `tnsnames.ora` that connect to the two new `sids`, PGAI and PGAU.
3. Define two database links in the Oracle Integrating Server, one connecting to PGAI and the other connecting to PGAU.
4. Create the bootstrap shell scripts based on the sample `pg4hoa1.sh` file that is supplied in the gateway `$ORACLE_HOME/bin` directory.
5. Finally, create the initialization files `initPGAI.ora` and `initPGAU.ora`.

   In `initPGAI.ora`, set `PGA_CAPABILITY` to `READ_ONLY`, and in `initPGAU.ora`, set `PGA_CAPABILITY` to `SINGLE_SITE`, `COMMIT_CONFIRM`, or `2_PHASE`. Then, use the PGAI gateway for inquiry-only transactions, and use the PGAU gateway for update transactions.

   The same steps can be used to set up additional gateway instances.

**PGA_CAPABILITY Parameter Considerations: HP-UX and Solaris**

When choosing a setting for the `PGA_CAPABILITY` parameter, take care to ensure that the correct setting is used based on what the remote transaction programs will be doing.
The READ_ONLY setting should always be used when the remote transaction programs are read-only, that is, when the remote transaction programs perform no database updates. READ_ONLY should never be used when the remote transaction programs perform database updates. For example, if the READONLY setting is chosen, and if a remote transaction program invoked by the gateway performs updates to a foreign database, then the Oracle Integrating Server does not provide any integrity protection for those updates. Furthermore, READ_ONLY mode allows a gateway transaction to be part of a distributed transaction that might update several other databases. If the gateway invokes a remote transaction program that performs updates in this situation, and if a failure occurs, then the database updated by the remote transaction program is out of synch with the other databases.

In cases where the remote transaction programs perform updates to foreign databases, there are two options for PGA_CAPABILITY:

- **SINGLE_SITE**
- **COMMIT_CONFIRM**

Each of these options provides protection against data integrity problems by allowing COMMIT and ROLLBACK requests to be forwarded to the remote transaction program, and by informing the Oracle Integrating Server about the distributed update and recovery capabilities of the gateway. The particular option chosen depends upon the design of the remote transaction programs and upon the capabilities of the OLTP (online transaction processor) where they execute.

If the OLTP has LU6.2 SYNCELEVEL 1 or 2 support, then the COMMIT_CONFIRM capability provides limited two-phase commit between the Oracle Integrating Server and the OLTP, with the restriction that no other commit-confirm site (gateway or Oracle) can be part of the distributed transaction. If it is not possible to use COMMIT_CONFIRM, then the SINGLE_SITE capability provides update capability between the Oracle Integrating Server and the OLTP, with the restriction that only the OLTP can perform updates, and no updates can occur on the Oracle side.

Each of the PGA_CAPABILITY options for update control imposes specific requirements on the remote transaction program and on the OLTP. For COMMIT_CONFIRM capability, these requirements are discussed in detail in Chapter 5, “Implementing Commit-Confirm,” of the *Oracle Procedural Gateway for APPC User’s Guide*. For SINGLE_SITE capability, the remote transaction program is responsible for performing the appropriate tasks in response to COMMIT and ROLLBACK requests received from the gateway on behalf of the Oracle Integrating Server. The gateway uses the APPC CONFIRM and SEND_ERR requests to
implement COMMIT and ROLLBACK, respectively. Upon receipt of a CONFIRM, the remote transaction program must perform COMMIT processing and then respond to the gateway with an APPC CONFIRMED response. Upon receipt of a SEND_ERR, the remote transaction program must perform ROLLBACK processing.

Because the distributed transaction capability of the Oracle Integrating Server is affected by the PGA_CAPABILITY option used by the gateway, it is desirable to separate inquiry and update applications by using different gateway instances for each. One gateway can be defined with PGA_CAPABILITY set to READ_ONLY and others with PGA_CAPABILITY set to SINGLE_SITE or COMMIT_CONFIRM.

This allows read-only transaction programs to participate in distributed transactions under the control of the Oracle Integrating Server. For example, data from DB2 can be retrieved through the READ_ONLY gateway by an inquiry-only remote transaction program, and can then be used as input to database updates on the Oracle Integrating Server, all in one Oracle transaction. A SINGLE_SITE gateway can be used only for accessing remote transaction programs which perform updates to foreign databases outside the scope of the Oracle Integrating Server’s control. Data can be read from any databases accessible to the Oracle Integrating Server, and that data can be used to perform updates through the gateway.

When it is necessary to update resources on both the Oracle side and the OLTP side, a COMMIT_CONFIRM gateway can be used, provided that the OLTP and the remote transaction programs are set up to implement commit-confirm.

All that is necessary to set up multiple gateway instances is to set up the following for each instance:

- an entry in the listener.ora file defining the sid of the gateway instance
- an entry in the tnsnames.ora file defining an alias to be used to connect to the gateway instance defined in listener.ora
- a database link in the Oracle Integrating Server that specifies the alias defined in the tnsnames.ora file in its USING parameter
- initialization variables in the bootstrap shell script (for example: pg4hoa1.sh)

Note that the gateway instances can share one common directory structure, and use the same executables.

For example, to set up two gateways, PGAI and PGAU (for inquiry and update use, respectively) the following steps are required:

1. Define entries in listener.ora for two sids, PGAI and PGAU.
2. Define two aliases in `tnsnames.ora` that connect to the two new SIDs, PGAI and PGAU.

3. Define two database links in the Oracle Integrating Server, one connecting to PGAI and the other connecting to PGAU.

4. Create the bootstrap shell scripts based on the sample `pg4hoa1.sh` file that is supplied in the gateway `$ORACLE_HOME/bin` directory.

5. Finally, create the initialization files `initPGAI.ora` and `initPGAU.ora`.

   In `initPGAI.ora`, set `PGA_CAPABILITY` to `READ_ONLY`, and in `initPGAU.ora`, set `PGA_CAPABILITY` to `SINGLE_SITE` or `COMMIT_CONFIRM`. Then, use the PGAI gateway for inquiry-only transactions, and use the PGAU gateway for update transactions.

   The same steps can be used to set up additional gateway instances.

### PGA_CONFIRM Parameter Considerations

When deciding upon the setting for the `PGA_CONFIRM` parameter, it is important to understand the effects of each setting. First, keep in mind that this parameter affects only those conversations running at `SYNCLEVEL 1`. The default setting, `PGA_CONFIRM=REJECT`, is appropriate for most applications. With this setting, the gateway generates an error if a CONFIRM request is received from the remote transaction program. If you have a remote transaction that uses CONFIRM to verify that data was received by the gateway, then you must use `PGA_CONFIRM=ACCEPT` to allow the gateway to respond to those incoming CONFIRM requests with CONFIRMED responses. You must be aware that the gateway sends CONFIRM requests to the remote transaction when the Oracle application has issued a COMMIT. In order for the COMMIT processing to work correctly, the remote transaction must be written to perform its local commit processing whenever a CONFIRM request is received from the gateway, and respond to the gateway with CONFIRMED after the commit processing has successfully completed. If an error occurs during commit processing, then the remote transaction must respond to the gateway with SEND_ERR to indicate that the commit failed.

One special case for the use of `PGA_CONFIRM=ACCEPT` is with IMS/TM version 6. When using the "implied APPC" support that is provided by IMS/TM version 6, conversations that run at `SYNCLEVEL 1` are handled differently than conversations that run at `SYNCLEVEL 0`. IMS/TM automatically generates CONFIRM requests after each APPC SEND when the conversation is at `SYNCLEVEL 1`. On the gateway side, if `PGA_CONFIRM=ACCEPT` is not specified,
then the CONFIRM requests sent by IMS/TM result in errors generated by the gateway. Using PGA_CONFIRM=ACCEPT alleviates this problem, allowing the gateway to respond to incoming CONFIRM requests with CONFIRMED responses. The only limitation with running this way is that the implied APPC support provided by IMS does not notify the application when a CONFIRM is received from the gateway. This means that the gateway cannot use CONFIRM to implement COMMIT, thereby disabling the use of COMMIT/ROLLBACK to control updates on the IMS side of the conversation.

Sample pg4hoa1.sh (Boot Parameter) File

The following sections contain a sample pg4hoa1.sh file for each platform.

AIX-Based Systems

The following is the sample pg4hoa1.sh file for AIX-Based Systems:

```bash
#!/bin/ksh
FDS_CLASS='PG4APPC_9I';export FDS_CLASS
FDS_CLASS_VERSION='2';export FDS_CLASS_VERSION
FDS_INSTANCE='PGA';export FDS_INSTANCE
exec /oracle/pga/9.2.0/bin/pg4asrv $*
```

**Note:** Make sure that pg4hoa1.sh has execute permissions.

HP-UX

The following is the sample pg4hoa1.sh file for HP-UX:

```bash
#!/bin/sh
FDS_CLASS='PG4APPC_92';export FDS_CLASS
FDS_CLASS_VERSION='2';export FDS_CLASS_VERSION
FDS_INSTANCE='PGA';export FDS_INSTANCE
# Make sure your ORACLE_HOME is set.
exec $ORACLE_HOME/bin/pg4asrv $*
```

**Note:** Make sure that pg4hoa1.sh has executable permissions.

Solaris

The following is the sample pg4hoa1.sh file for Solaris:

#!/bin/ksh
# Sample Procedural Gateway for APPC boot parameter file
#
# stalmoud 02/09/02 - modified for 9.2.0
#
# These are the boot parameters for the Procedural Gateway driver. See the
# Oracle Procedural Gateway for APPC Installation and configuration for
# SUN Solaris for details on these parameters.
# These are environment variable settings required by the SNA software.
#
# When running with SunLink SNA Peer-to-Peer 9.0, the following parameters are
# required:
#
# APPC_GATEWAY
# Specifies the name of the SunLink PU2.1 server which will be
# used for all SNA services. This name is always the machine name.
# PATH
# Specifies the search path to be used to locate Side Information
# profile files for starting APPC conversations. This must include
# the directory where you have created your Side Information profile
# files. Each profile is contained in a single file. Your files must
# have names no longer than 8 characters, with only alphanumeric
# characters allowed.
#
# If you are running SunLink SNA Peer-to-Peer 9.0, uncomment the following
# lines and tailor them for your system:
#
# APPC_GATEWAY=<gateway name>; export APPC_GATEWAY
# PATH=$PATH:/oracle/pga/9.2.0/pg4appc/sna
#
# If you are using SNAP-IX V6, you do not need to use the above lines and
# tailor them for your system.
#
# you need to specify the TRACE_LEVEL and LOG_DESTINATION,
# if you need to run Procedural Gateway for APPC with the debug option on.
#
# FDS_CLASS='PG4APPCC_9I';export FDS_CLASS
# FDS_CLASS_VERSION='2';export FDS_CLASS_VERSION
# FDS_INSTANCE='PGA';export FDS_INSTANCE
#
# In addition, you need to specify the full path and the Procedural Gateway
# executable file.
/oracle/pga/9.2.0/bin/pg4asrv $*
exit 0
Note: Make sure that pg4hoa1.sh has executable permissions.
Sample initPGA.ora File

Following is a sample `initPGA.ora` file:

```
# SAMPLE initPGA.ora file for PG4APPC
#
# SET TRACE_LEVEL=0
# SET LOG_DESTINATION=/oracle/pga/9.2.0/pg4appc/log
# HS_COMMIT_POINT_STRENGTH=255
# HS_DB_NAME=PGA
# HS_DB_DOMAIN=WORLD
# HS_DB_INTERNAL_NAME=504741
# SET PGA_CAPABILITY=SINGLE_SITE
# SET PGA_SECURITY_TYPE=NONE
```
AIX-Based Systems Only: Sample rrmstart Script

# Copyright (c) Oracle Corporation 1996,2001. All rights reserved.
#
# NAME
#   rrmstart
#
# DESCRIPTION
#   This is a shell script to start the Procedural Gateway for APPC Resource
#   Recovery Manager, pg4arrm.
#
# NOTES
#
#   Parameters:
#
#   The only parameter to this script is the fully-qualified local LU
#   name on which the RRM should run.
#
#   The following fields should be customized for your installation:
#
#   ORACLE_HOME should be set to the directory into which the gateway
#   software is installed.
#   TNS_ADMIN should be set to the directory where the Net8
#   tnsnames.ora file resides. If it resides in your gateway's
#   ORACLE_HOME/network/admin directory, this variable is not
#   required. You can also set up a soft link called
#   tnsnames.ora in your gateway's ORACLE_HOME/network/admin
#   directory that points to a shared system-wide tnsnames.ora
#   file if you prefer. In that case also, this variable is
#   not required.
#   NLS_LANG should be set to the NLS language specification to be used
#   for messages from the RRM. Refer to the Oracle Procedural
#   Gateway for APPC Installation & User's Guide for information
#   on supported languages.
#
# MODIFIED (MM/DD/YY)
#   stalmoud 12/13/2001 - updated for 9.2.0
#   stalmoud 06/25/2001 - updated for 9.0.1
#   stalmoud 01/13/2000 - updated for 8.0.4
#   mwhite 03/07/96 - created
#   mwhite 09/05/97 - updated for V4.0.1
#
export ORACLE_HOME=/oracle/pga/9.2.0
export TNS_ADMIN=$ORACLE_HOME/network/admin
export NLS_LANG=american_america.us7ascii
$ORACLE_HOME/bin/pg4arrm $1 &

AIX-Based Systems Only: Sample HQWM920.ini File

#
# Procedural Gateway for APPC Resource Manager (PG4ARRM)
# Initialization file for local LU HQWM920
#
# Values specified here assume ORACLE_HOME is /oracle/pga/9.2.0 Values that
# are the default values are
# LOG_DIRECTORY, LOG_FILE, LOG_ERRORS, TRACE_DIRECTORY, TRACE_FILE, and
# TRACE_LEVEL.
#
# Required parameters that have no default values are
# LOG_DB, LOG_PASS, LOG_USER, SIDE_PROFILE, and TP_PROFILE.
#
# NOTE: You must modify this file to work on your system.
#
# stalmoud    02/27/02    - modified for 9.2.0.1.0 (aix)
#
# LOG_DIRECTORY=/oracle/pga/9.2.0/pg4appc/log
LOG_FILE=HQWM920.log
LOG_DB=tnsalias
LOG_ERRORS=1
LOG_PASS=PGADBA
LOG_USER=PGADBA
TRACE_DIRECTORY=/oracle/pga/9.2.0/pg4appc/trace
TRACE_FILE=HQWM920.trc
TRACE_LEVEL=0
#
# OS-specific parameters for AIX
#
SIDE_PROFILE=CICSPGA2
TP_PROFILE=PG4ARRM

---

Caution: Make sure to not place any blanks between the parameters and its values in the initialization file for local LU.

---
This appendix describes the gateway messages specific to your platform. It contains the following section:

- AIX-Based Systems Specific Messages on page B-2
- HP-UX Specific Messages on page B-5
- Solaris Specific Messages on page B-6
AIX-Based Systems Specific Messages

The following sections provide information on messages specific to AIX-Based Systems.

Oracle Procedural Gateway for APPC Server Messages

PGA-20910 communication error: CPI-C *func* failed, *rc = rc*, *errno = errno*

**Cause:** An unexpected communication error occurred while executing the specified CPI-C function *func*. The CPI-C function return code *rc* and system error number *errno* are provided in the message text. The CPI-C function return codes and system error numbers are described in vendor documentation. They are listed both by number and by symbolic name in the documentation. Also, the CPI-C return codes can be found in the */usr/include/cmc.h* header file, and the AIX SNA system error numbers can be found in the */usr/include/luxsna.h* header file.

Table B–4 lists some commonly received *rc* and *errno* combinations, and their possible causes.

<table>
<thead>
<tr>
<th>Table B–1</th>
<th>Error Message PGA-20910</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>rc</strong></td>
<td><strong>errno</strong></td>
</tr>
<tr>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>1</td>
<td>25</td>
</tr>
<tr>
<td>6</td>
<td>133</td>
</tr>
<tr>
<td>9</td>
<td>127</td>
</tr>
<tr>
<td>10</td>
<td>147</td>
</tr>
<tr>
<td>17</td>
<td>121</td>
</tr>
<tr>
<td>19</td>
<td>9</td>
</tr>
</tbody>
</table>
AIX-Based Systems Specific Messages

Table B–1 (Cont.) Error Message PGA-20910

<table>
<thead>
<tr>
<th>rc</th>
<th>errno</th>
<th>Possible Cause(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>125</td>
<td>The SNA session on which the conversation was running has been terminated, or SNA Communication Package is shutting down.</td>
</tr>
<tr>
<td>20</td>
<td>146</td>
<td>The target LU name or alias is not defined locally to SNA Server.</td>
</tr>
<tr>
<td>20</td>
<td>160</td>
<td>SNA Server is not currently running.</td>
</tr>
</tbody>
</table>

Action: Determine the cause of the communications error, correct it, and rerun the transaction.

PGA-20995 communication error: ‘func’ failed, rc = rc, errno = errno

Cause: An unexpected communications error occurred while executing the AIX SNA Server API func function. The function return code is rc and the system error number is errno. The AIX SNA Server LU6.2 API function return codes and system error numbers are described in vendor documentation. The error codes are listed both by number and by symbolic name in the documentation. The SNA system error numbers can also be found in the /usr/include/luxsna.h header file on AIX-Based systems.

Table B–2 lists some commonly received rc and errno combinations, and their possible causes.

Table B–2 Error Message PGA-20995

<table>
<thead>
<tr>
<th>rc</th>
<th>errno</th>
<th>Possible Cause(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>112</td>
<td>When received at conversation startup time, either the target LU is not active or has not enabled SNA communications, or the target LU name is not defined on the target system. The target LU name is the fully-qualified LU name specified in either the side information profile or the Partner LU Location Profile. When received during an active conversation, the SNA session on which the conversation was running has been terminated.</td>
</tr>
<tr>
<td>-1</td>
<td>115</td>
<td>A conversation was requested at synclevel 2, but the required LU6.2 Resource Recovery Manager, PG4ARRM, is not active on the local LU, or the mode name specified does not support synclevel 2.</td>
</tr>
<tr>
<td>-1</td>
<td>121</td>
<td>The transaction program terminated abnormally on the target LU, or issued a DEALLOCATE_ABEND.</td>
</tr>
</tbody>
</table>
Oracle Procedural Gateway for APPC Installation and Configuration Guide

**Table B–2 (Cont.) Error Message PGA-20995**

<table>
<thead>
<tr>
<th>rc</th>
<th>errno</th>
<th>Possible Cause(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>123</td>
<td>The mode name specified is either not defined locally to SNA Server, or it is not defined to the target LU.</td>
</tr>
<tr>
<td>-1</td>
<td>127</td>
<td>The transaction program requested is not defined to the target LU.</td>
</tr>
<tr>
<td>-1</td>
<td>133</td>
<td>A security violation occurred on the target system. Either the user ID/password is not valid on that system, or the user ID is not authorized to execute the requested transaction.</td>
</tr>
<tr>
<td>-1</td>
<td>146</td>
<td>The target LU name or alias is not defined locally to SNA Server.</td>
</tr>
<tr>
<td>-1</td>
<td>147</td>
<td>The transaction program requested is defined to the target LU, but could not be found.</td>
</tr>
<tr>
<td>-1</td>
<td>160</td>
<td>SNA Communication Package is not active.</td>
</tr>
</tbody>
</table>

**Action:** Determine the cause of the communications error, correct it, and rerun the transaction. If you need assistance, then contact your system administrator.

---

**Oracle Procedural Gateway for APPC Resource Recovery Manager Messages**

PGA-21450 communication error: *func* failed, *errno = errno*

**Cause:** The AIX SNA function *func* failed with a system error number of *errno*.

Table B–3 lists some commonly received *errno* values, and their possible causes.

**Table B–3 Error Message PGA-21450**

<table>
<thead>
<tr>
<th>errno</th>
<th>Possible Cause(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>112</td>
<td>The session was terminated by the system operator</td>
</tr>
<tr>
<td>121</td>
<td>The RRM on the partner LU de-allocated the conversation abnormally. This usually occurs when the partner LU’s RRM detected an error in the XLN received from the local RRM.</td>
</tr>
</tbody>
</table>

**Action:** The AIX SNA error codes are documented in vendor documentation. Use the documentation to determine the cause of the problem, and then correct it. AIX SNA system error numbers are also documented in the `/usr/include/luxsna.h` header file on AIX-Based systems.
PGA-21451 SNA Server is shutting down

**Cause:** SNA Communication Package is being shut down, and it sent a SIGUSR1 to the RRM.

**Action:** The RRM must terminate when this occurs. Contact the system administrator to determine the cause of the SNA Communication Package shutdown. Restart the RRM after SNA Communication Package has been restarted.

---

**HP-UX Specific Messages**

The following sections provide information on messages specific to HP-UX.

**Oracle Procedural Gateway for APPC Server Messages**

PGA-20910 communication error: CPI-C `func` failed, `rc = rc`, `errno = errno`

**Cause:** An unexpected communication error occurred while executing the specified CPI-C function `func`. The CPI-C function return code `rc` and system error number `errno` are provided in the message text. The CPI-C function return codes and system error numbers are described in vendor documentation. They are listed both by number and by symbolic name in the documentation.

Some commonly received `rc` and `errno` combinations, and their possible causes, are listed in Table B–4 below.

<table>
<thead>
<tr>
<th><code>rc</code></th>
<th><code>errno</code></th>
<th>Possible Cause(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>n/a</td>
<td>The target LU is not active or has not established communications with SNA Communication Package.</td>
</tr>
<tr>
<td>6</td>
<td>n/a</td>
<td>A security violation occurred on the target system; either the user ID or password is not valid on that system or the user ID is not authorized to execute the requested transaction on the target OLTP.</td>
</tr>
<tr>
<td>9</td>
<td>n/a</td>
<td>The transaction program requested is not defined to the target OLTP.</td>
</tr>
<tr>
<td>10</td>
<td>n/a</td>
<td>The transaction program requested is defined to the target OLTP, but could not be found.</td>
</tr>
<tr>
<td>17</td>
<td>n/a</td>
<td>The transaction program terminated abnormally on the target OLTP, or issued a DEALLOCATE_ABEND.</td>
</tr>
</tbody>
</table>
Action: Determine the cause of the communications error, correct it, and rerun the transaction.

Solaris Specific Messages

The following sections provide information on messages specific to Solaris.

Oracle Procedural Gateway for APPC Server Messages

**PGA-20910 communication error: CPI-C *func* failed, rc = *rc*, errno = *errno***

**Cause:** An unexpected communication error occurred while executing the specified CPI-C function *func*. The CPI-C function return code *rc* and system error number *errno* are provided in the message text. For SNA Communication Package 9.0, CPI-C function return codes are described in vendor documentation.

The following table lists some commonly received *rc* and *errno* combinations, and their possible causes:

<table>
<thead>
<tr>
<th>rc</th>
<th>errno</th>
<th>Possible Cause(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>The mode name specified is not defined in the mode table of the target LU (P2P 9.0 only).</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>A security violation occurred on the target system; either the user ID or password is not valid on that system or the user ID is not authorized to execute the requested transaction on the target OLTP.</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>The transaction program requested is not defined to the target OLTP.</td>
</tr>
</tbody>
</table>
Solaris Specific Messages

Platform-Specific Gateway Messages

<table>
<thead>
<tr>
<th>rc</th>
<th>errno</th>
<th>Possible Cause(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0</td>
<td>The transaction program requested is defined to the target OLTP, but could not be found (P2P 9.0 only).</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>The transaction program terminated abnormally on the target OLTP, or issued a DEALLOCATE_ABEND.</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>The target LU is not active or has not established communications with SNA Communication Package, or the LU name specified is not defined to SNA Communication Package, or the mode name specified is not defined to SNA Communication Package, or the LU name specifies is not defined on the target system (P2P 9.0 only).</td>
</tr>
<tr>
<td>26</td>
<td>146</td>
<td>SNA Communication Package is not active (P2P 9.0 only).</td>
</tr>
</tbody>
</table>

Action: Determine the cause of the communications error, correct it, and rerun the transaction.
This appendix provides a summary of changes in previous versions of Oracle Procedural Gateway for APPC.

This appendix contains the following sections:

- Changes and Enhancements in Previous Versions on page C-2
- Corrected Problems in Previous Versions on page C-11
- Known Problems in Previous Releases on page C-16
Changes and Enhancements in Previous Versions

The following sections list changes and enhancements that were made to previous releases of the gateway.

Release 9.0.1.0.1

**PGA_TRACE Parameter No Longer Supported**
With this release of Oracle Procedural Gateway for APPC, the PGA_TRACE parameter is no longer supported. If you are copying the `initsid.ora` file from an older version of the gateway, you must remove this parameter.

**TRACE_LEVEL and LOG_DESTINATION Parameters Moved**
With this release of Oracle Procedural Gateway for APPC, the TRACE_LEVEL and LOG_DESTINATION parameters have been moved from the `pg4hoa1.sh` file to the `initsid.ora` file.

**FDS_CLASS_VERSION Parameter Added**
You will need to add the FDS_CLASS_VERSION parameter to your startup shell script file (`pg4hoa1.sh`). A default value is specified in `pg4hoa1.sh`.

Release 8.0.4.1.0/ 8.0.6.1.0

- **Heterogeneous Services architecture**
  This release of Oracle Procedural Gateway for APPC utilizes the Oracle Heterogeneous Services component within the Oracle8 server. Heterogeneous Services is the building block for the next generation of Oracle Open Gateways.

- **Performance enhancements**
  Oracle Procedural Gateway for APPC contains several internal performance enhancements. This product has shown major improvements over the version 4 gateways, particularly in response time and CPU utilization for all relevant address spaces for a variety of workloads. The actual performance improvement at your site may vary, depending on your installation type and workload.

- **PL/SQL V3 compatibility**
Before Release 8.0.6.1.0, you had to use the PLSQL_V2_COMPATIBILITY =TRUE parameter to compile PGAU-generated TIP specifications. This is no longer necessary.

- **Oracle server dependencies**

This release of Oracle Procedural Gateway for APPC requires that the Oracle Integrating Server be an Oracle8 server Version 8.0.6.2.0 (Oracle database server release 8.0.6.2.0) or later.

- **Gateway initialization parameters**

In previous versions of the gateway the initialization parameters were stored in files named `initsid.gtwboot` and `initsid.ora`, both found in the gateway instance directories. With version 8, most parameters that were in `initsid.gtwboot` have been moved to the `initsid.ora` file. The syntax of `initsid.ora` has been simplified.

In addition, Oracle Procedural Gateway for APPC’s executable (`pg4asrv`) calls the gateway indirectly through a startup shell script sample file called `pg4hoa1.sh`. This startup shell script is also used for setting SNAplus environmental variables, such as APPCLU.

- **tnsnames.ora**

The service name definition (from the Oracle Integrating Server to the gateway) must contain an extra parameter, (HS=).

### Release 4.0.1.1.0

- **Gateway components now based on Oracle7 Server release 7.3.2**

The gateway components are now built using the Oracle7 server release 7.3.2 product libraries as a base. All product components shipped with the gateway, including SQL*Net, are now at the release 7.3.2 level.

- **Gateway supports oracle8 Server release 8.0.3**

The gateway now supports Oracle8 server release 8.0.3 as an integrating server.

- **UTL_PG now shipped with Oracle Server**

The scripts for installing the UTL_PG PL/SQL package are now shipped with the Oracle7 or Oracle8 server and are no longer shipped with the gateway.

- **Gateway no longer supports Oracle7 Server release 7.1.6 and 7.2**

The gateway no longer supports Oracle7 server release 7.1.6 and 7.2 as integrating servers.
Changes and Enhancements in Previous Versions

- User ID and password overrides now supported by gateway (Enh. No. 423300)
  A new TIP override is now available to allow the user ID and password sent by
  the gateway to the OLTP (online transaction processor) to be specified by the
  user. For complete information on using this override, refer to "Overriding TIP
  Initializations" in Chapter 2 of the Oracle Procedural Gateway for APPC User’s
  Guide.

- New "LENGTH IS" clause supported by PGAU (Enh. No. 428783)
  A new "LENGTH IS" clause in COBOL data definitions is now recognized by
  PGAU and allows the definition of variable-length character fields without
  using OCCURS DEPENDING ON. For complete information on using this
  clause, refer to "Format Conversion" in Appendix G of the Oracle Procedural
  Gateway for APPC User’s Guide.

- Date/time stamp in trace output
  All trace and debug output from the gateway server now contains a date/time
  stamp in the beginning of each line.

- Installer improvements
  The installer prompts not relevant to the gateway have been eliminated to
  streamline the installation process.

Release 4.0.0.1.0

- Gateway components now based on Oracle7 server release 7.2.3
  The gateway components are now built using the Oracle7 server release 7.2.3
  product libraries as a base. All product components shipped with the gateway,
  including SQL*Net, are now at the V7.2.3 level.

- SQL*Net version 1 no longer supported
  SQL*Net version 1 is no longer shipped with, nor supported by, the gateway.
  Only SQL*Net version 2 is included with, and supported by, the gateway.

- Simple network management protocol (SNMP) now supported
  The gateway server now supports the use of SNMP for collecting information
  about the gateway, including configuration, execution, and performance data.
  A new gateway SNMP subagent and a program for controlling the subagent are
  provided with the gateway. Refer to the Oracle Open Gateway Release Notes for
  SNMP for further information on the gateway SNMP support.

- Gateway server now supports COMMIT-CONFIRM
The gateway server now supports COMMIT-CONFIRM capability, allowing the gateway to participate in a distributed update transaction where resources on both the Oracle side and the gateway (OLTP) side are updated. This capability is enabled by specifying PGA_CAPABILITY=COMMIT_CONFIRM in the gateway initialization parameters. For complete information on the use of COMMIT-CONFIRM with the gateway and the OLTP, refer to Chapter 4, “Implementing Commit-Confirm,” in the Oracle Procedural Gateway for APPC User's Guide.

- PGAU now supports multi-byte character set data (Enh. No. 377056)
  PGAU now supports the definition of multi-byte character set (MBCS) data items, and allows the character set to be specified on a field-by-field basis. Generated TIPs perform all necessary character set conversions.

- Sample DFSASYNC transaction for IMS now included (Enh. No. 308592)
  A sample DFSASYNC transaction is now provided for use with IMS/TM. The sample is a Pro*C CPI-C application that receives messages from IMS/TM and writes them into an Oracle table. A sample PL/SQL application is also provided which will format, list, and delete the rows in the table. For complete information on the DFSASYNC sample, refer to the dfsasync.doc file in the $ORACLE_HOME/pg4appc/demo/IMS directory.

- Sample application to copy data from DB2 to Oracle
  A sample PL/SQL application is now provided to show the use of the gateway to copy data from a DB2 table to an Oracle table. The sample uses the CICS DB2 multi-row inquiry transaction provided with the gateway. For complete information on this sample, refer to the README.doc file in the $ORACLE_HOME/pg4appc/demo/CICS directory.

**Release 3.4.0.1.0**

- Support for release 7.3 Oracle7 server
  Oracle7 servers at release 7.3 are now supported by the gateway.

- C compiler no longer required
  The HP9000 Series 700/800 compiler is no longer required for installation of the gateway. The HP9000 Series 700/800 linker, ld, is now invoked directly to relink the gateway executables.

- APPC conversation sharing (Enh. No. 287657)
PGAU-generated TIP specifications can now share a single APPC conversation. This is useful when the target system allows transaction programs to pass an APPC conversation among one another, or when TIPs are too large for PL/SQL to compile. To take advantage of this change, the TIPs must be regenerated and recompiled.

- An Oracle Call Interface sample client application is provided

  This sample C program and make file demonstrate use of an OCI-C client application to drive the sample TIP for CICS to DB2 update, and performs the inquiry, update, diagnostic, initialization, and termination calls. Refer to the $ORACLE_HOME/pg4appc/demo/CICS/README.doc, pgadb2ud.c, and pgadb2ud.mk files for details.

- Certification with CICS/VSE

  The gateway has been certified to function fully with CICS/VSE.

- Additional languages for PGA and PGU messages

  The PGA and PGU messages are now provided in Brazilian Portuguese, Dutch, Japanese, Korean, simplified Chinese, and Slovak.

**Release 3.3.1.0.1**

- Installation changes

  The gateway must now be installed in a separate Oracle home directory from the integrating server Oracle home directory. This is required to isolate the gateway from integrating server upgrades that might cause incompatibilities if the gateway executables were relinked with later versions of the Oracle7 server libraries. The result of this change is that more disk space will be required when installing the gateway on the same system as the integrating server.

- Improved parameter validation in PGAXFER

  The PGAXFER gateway RPC function has better checking of input parameters to simplify debugging of calling PL/SQL code.

- Improved trace control in PGATCTL (Enh. No. 227397)

  The PGATCTL gateway RPC trace control function provides a way to update trace flags for all currently active conversations with a single call to PGATCTL.

- Integrating server restriction lifted
It is no longer required that the integrating server reside on UNIX. The integrating server can reside on any supported platform as long as the Oracle server and PL/SQL requirements are met.

- Full SQL*Net protocol support
  All SQL*Net protocols available for the UNIX are supported by the gateway server and the Administration Utility. You select which protocols are enabled during gateway installation.

- National Language Support in Administration Utility
  The Administration Utility messages are now provided in the following languages:
  - American English
  - Czech
  - Danish
  - Finnish
  - French
  - German
  - Greek
  - Hungarian
  - Italian
  - Norwegian
  - Polish
  - Portuguese
  - Russian
  - Spanish
  - Swedish
  - Turkish

- IMS/TM and APPC/MVS support enhanced
  IMS/TM Releases 3.1 and 4.1, and APPC/MVS with MVS/ESA release 4.2 and higher, support has been enhanced. Configuration and IVP procedures for these OLTP systems are provided in this guide.
Changes and Enhancements in Previous Versions

- Gateway accepts incoming APPC CONFIRM (Enh. No. 267776)
  A new gateway initialization parameter, PGA_CONFIRM, controls whether the gateway accepts and responds to incoming APPC CONFIRM requests sent by the remote transaction program (PGA_CONFIRM=ACCEPT), or treats incoming APPC CONFIRM requests as errors (PGA_CONFIRM=REJECT). When using the gateway with IMS/TM transactions using "implied APPC support" and SYNCLEVEL 1, IMS/TM automatically generates a CONFIRM request to the gateway following each SEND. Setting PGA_CONFIRM=ACCEPT allows the gateway to work with such transactions.

- IMS/TM sample applications
  IVP and sample applications for IMS/TM are included in the sample suite. Full documentation is provided in the README.doc file in the pg4appc/demo/IMS directory.

- APPC/MVS sample applications
  IVP and sample applications for APPC/MVS are included in the sample suite. Full documentation is provided in the README.doc file in the pg4appc/demo/MVS directory.

- UTL_PKG NUMBER_TO_RAW and RAW_TO_NUMBER format buffer size reduced
  The size of the format buffers used by the NUMBER_TO_RAW and RAW_TO_NUMBER routines of the UTL_PKG PL/SQL package has been reduced from 2048 to 512 bytes. This change is downward-compatible with PL/SQL procedures written for the existing UTL_PKG routines, so no TIP regeneration is necessary.

- New UTL_PKG and UTL_RAW exceptions
  The VALUE_ERROR exception is raised when UTL_PKG or UTL_RAW functions are passed invalid parameters. Previously, the functions returned null results when invalid parameters were passed.

- PG DD diagnostic SQL scripts provided
  New diagnostic SQL scripts for diagnosing data dictionary entry problems are provided with this release.

- PGAU flags location of error in statement
  Like SQL*Plus, PGAU now flags errors in PGAU statement syntax with an asterisk (*).
Changes and Enhancements in Previous Versions

- **PGAU open cursor reduction**
  The number of concurrently opened cursors used by PGAU has been reduced. The default value for the OPEN_CURSORS parameter in the Oracle server `init.ora` file is now sufficient for running PGAU and need not be increased.

- **Data items larger than 32K bytes handled by TIPs**
  TIPs generated by PGAU can handle data items larger than 32K bytes. The TIPs assemble the data items from multiple APPC data transfers of 32,763 bytes from the remote transaction program and break down items into multiple APPC data transfers to the remote transaction program.

- **PGAU UNDEFINE now cascades down**
  The PGAU UNDEFINE command cascades down when the WITH option is specified. For example, `UNDEFINE TRANSACTION tran WITH DATA` deletes all associated CALL and DATA entries along with the TRANSACTION entry as long as they are not referenced by other PG DD entries.

- **PGAU supports COBOL SYNCHRONIZED and JUSTIFIED options**
  PGAU supports the COBOL SYNCHRONIZED and JUSTIFIED picture mask options. Data alignment and justification are preserved as specified by these options.

- **Runtime TIP traces and warnings issued optionally**
  These four types of TIP traces are at the control of the TIP caller:
  - data conversion trace
  - data conversion warnings
  - function entry/exit trace
  - gateway exchange trace
  The data conversion trace and data conversion warnings are optionally included in the TIP at generation time. The function entry/exit and gateway exchange traces are always included in the TIP. The PKGEX keyword of the PGAU GENERATE DIAGNOSE option can be used to specify that data conversion trace and data conversion warnings be generated in the TIP.

- **PG DD references optionally generated Comments in TIPs**
  PGAU-generated TIP specifications now optionally contain Comments indicating which PG Data Dictionary entries were used to generate the TIP. This is controlled by the DIAGNOSE(PKGEX(DR)) option of the PGAU
GENERATE statement. It greatly improves the tracking of TIP changes by allowing them to be correlated with PG DD changes when used with the REPORT WITH DEBUG statement.

- PGAU generates a TIP content documentation file
  The PGAU GENERATE statement now generates a separate documentation file for each TIP. The file contains all information necessary for a client application to interface with the TIP, including information about the remote transaction program and data definitions.

- PGAU REPORT shows missing references
  The PGAU REPORT command now flags any missing references found in the PG Data Dictionary. For example, a CALL entry for which a referenced DATA entry is not found is flagged as a missing reference.

- New PGAU REPORT WITH DEBUG option
  The PGAU REPORT statement with the WITH DEBUG option produces a listing which includes PG Data Dictionary entry ID numbers for correlation with TIPS generated with PG DD diagnostic references. This helps identify TIP errors related to PG DD entries that are not properly defined.

- New PGAU REPORT ISOLATED option
  The PGAU REPORT command has a new option, ISOLATED, which reports on all PG Data Dictionary CALL and DATA entries that are not referenced by any TRANSACTION or CALL entries, respectively. This helps with data dictionary housekeeping and debugging.

- TIPS generated into separate package files
  To improve the ability to upgrade TIP body functions without affecting dependent client applications, TIPS are now generated into two output files. One file is a package specification, and the other is a package body. These can be separately compiled.

- TIPS can be initialized with overridden parameters
  To allow one TIP to be used for multiple OLTPs or remote host transactions, such as a test version and a production version, TIPS can be initialized to start conversations with different transaction attributes.
### Corrected Problems in Previous Versions

The following tables list the numbers and descriptions of each bug fixed in previous releases of the gateway.

#### Release 9.0.1.0.1

The following table lists the bugs that have been fixed in Release 9.0.1.0.1 of the gateway, along with their descriptions:

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1276298</td>
<td>When using an invalid value for receive parameter for the pgaxfer procedure, the user receives an ORA-28511 error message.</td>
</tr>
<tr>
<td>1302253</td>
<td>Oracle Procedural Gateway for APPC procedure statements executed on the Oracle integrating servers would hang in SQL*Plus, without giving any messages.</td>
</tr>
<tr>
<td>1336805</td>
<td>When the PGA_SECURITY_TYPE parameter was set to PROGRAM and the PGA_CAPABILITY parameter was set to COMMIT_CONFIRM, the user received an authentication error but could not see who was complaining about it.</td>
</tr>
<tr>
<td>1404454</td>
<td>The pg4appc log file should be improved and have the received buffer in the log file.</td>
</tr>
<tr>
<td>1411694</td>
<td>The user receives message ORA-28527 when the PGA_CAPABILITY is set to READ_ONLY.</td>
</tr>
<tr>
<td>1472800</td>
<td>Multi-row queries failed on the following error messages: ORA-01401 and ORA-06512.</td>
</tr>
<tr>
<td>1519088</td>
<td>User received sporadic abends when inserting CICS records to VSAM file.</td>
</tr>
<tr>
<td>1677939</td>
<td>Oracle Procedural Gateway for APPC would partially transfer low values to VSAM files.</td>
</tr>
<tr>
<td>1722467</td>
<td>When the PGA_SECURITY_TYPE parameter was set to PROGRAM and the user specified the user ID and password through database link explicit CONNECT information, the query failed with message PGA-20910 RC=6.</td>
</tr>
<tr>
<td>1724988</td>
<td>When the programmer used RPC PGAINIT_SEC, no matter what synclenci was being used, the following message was received: &quot;invalid SYNCHLEVEL, 152, specified; valid range is 0:1.&quot;</td>
</tr>
</tbody>
</table>
## Corrected Problems in Previous Versions

### Release 8.0.6.1.0

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1329386</td>
<td>PGAU core dumped on large packages. When users tried to generate large packages with PGAU, they got a core dump.</td>
</tr>
<tr>
<td>561128</td>
<td>The PLSTYPE attribute of binary integers (COBOL COMP data items) had the wrong precision when the data was defined without COMPOPTS('TRUNC(BIN)') specified.</td>
</tr>
<tr>
<td>599696</td>
<td>PGAU COBOL parser did not accept OCCURS clause appearing before datatype as valid syntax.</td>
</tr>
<tr>
<td>603934</td>
<td>When COBOL data definitions contained OCCURS and REDEFINE, the PGAU-generated .pkb file was invalid.</td>
</tr>
<tr>
<td>689304</td>
<td>This bug occurred when a CICS transaction abended between the time that the Oracle server called the gateway to perform a commit and the time that the CICS transaction was successfully committed. When this occurred, the Oracle recovery process (RECO) would call the gateway’s recovery function (GTARECO) to resolve the transaction. The GTARECO function mistakenly relayed that it had successfully rolled back the CICS transaction, but in fact it had failed to delete the row from PGA_CC_PENDING.</td>
</tr>
</tbody>
</table>

### Release 4.0.1.1.0

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>420391</td>
<td>PGAU generated invalid PL/SQL in the TIP when a COBOL definition contained an OCCURS clause and was defined as an IN OUT parameter. A PLS-103 error was generated when attempting to compile the TIP.</td>
</tr>
<tr>
<td>460960</td>
<td>ORA-9199 errors were generated by the gateway after some gateway exceptions due to a memory overlay.</td>
</tr>
<tr>
<td>473360</td>
<td>PGAU encountered a segment fault when executing a GENERATE command for a transaction with a very large number of fields defined in its COBOL data.</td>
</tr>
<tr>
<td>506777</td>
<td>An ORA-1017 error was generated by the gateway when RECO attempted to log onto the gateway during recovery of a distributed transaction, even though the gateway was running in READ-ONLY mode.</td>
</tr>
</tbody>
</table>
Corrected Problems in Previous Versions

Summary of Changes in Previous Versions

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>44471</td>
<td>The install of the gateway failed while linking the gateway executables if the C compiler was not installed on the UNIX system.</td>
</tr>
<tr>
<td>474889</td>
<td>The gateway server encountered a segment fault during installation verification if the patch for bug number 444771 was installed. The patch contained incorrect linker parameters.</td>
</tr>
</tbody>
</table>

Release 4.0.0.1.0

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>348132</td>
<td>COBOL PIC S9 (4) values greater than +9999 or less than -9999 were truncated to 4 digits by PGAU-generated TIPS.</td>
</tr>
<tr>
<td>359116</td>
<td>Error message ORA-3106 received when integrating server resided on a platform with different byte-ordering from the gateway platform.</td>
</tr>
<tr>
<td>359973</td>
<td>PL/SQL packages that called the gateway were marked invalid when one of them encountered an error during execution.</td>
</tr>
<tr>
<td>361966</td>
<td>PL/SQL procedures that called the gateway were marked invalid when other procedures that called the gateway were recompiled.</td>
</tr>
<tr>
<td>364538</td>
<td>SAVEPOINT caused error message ORA-3113 to be received from the gateway. The gateway server was taking a segmentation fault.</td>
</tr>
<tr>
<td>380431</td>
<td>Install did not copy sighold.o into SORACLE_HOME/lib, resulting in a failure in the gensnalib shell script and subsequent failures on the linking of the gateway executables.</td>
</tr>
<tr>
<td>381305</td>
<td>PGAU-generated TIP specifications sent invalid COBOL data when SIGN LEADING SEPARATE or SIGN TRAILING SEPARATE was used with USAGE IS DISPLAY. The fix for PL/SQL bug 383510 is also required to correct problems in the UTL_PG PL/SQL package.</td>
</tr>
<tr>
<td>388162</td>
<td>PGAU-generated TIP specifications sent more than 32K of data, causing error message ORA-6502 to be received from UTL_RAW.OVERLAY.</td>
</tr>
<tr>
<td>405129</td>
<td>RPAD call used EBCDIC blank (x '40') as the pad character when converting data regardless of the local or remote character set setting.</td>
</tr>
</tbody>
</table>
Corrected Problems in Previous Versions

Release 3.4.0.1.0

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>284743</td>
<td>PGAU GENERATE incorrectly rejected sequential OCCURS groups as being nested.</td>
</tr>
<tr>
<td>312840</td>
<td>PGAU GENERATE skipped generation of a unique nested OCCURS group that followed a duplicate nested OCCURS group.</td>
</tr>
<tr>
<td>318928</td>
<td>PGAU DEFINE TRANSACTION did not allow a fully-qualified LU name value in the LUNAME parameter.</td>
</tr>
<tr>
<td>320113</td>
<td>The sample SNAplus configuration file had the wrong partner LU in the SYM_DEST_NAME definition for IMSPGA61.</td>
</tr>
<tr>
<td>320635</td>
<td>PGAU GENERATE created a TIP with a missing output record descriptor when the input and output data definitions used the same COBOL copybook.</td>
</tr>
<tr>
<td>324540</td>
<td>PGAU-generated TIPs could not handle variable-length records and generated errors when the records were shorter than the maximum length.</td>
</tr>
<tr>
<td>329319</td>
<td>PGAU GENERATE corrupted COBOL 'REDEFINES WHEN' criteria with negative values.</td>
</tr>
<tr>
<td>337093</td>
<td>PGAU DEFINE TRANSACTION did not allow the TPNAME parameter to include path specification, even when enclosed in double quotes.</td>
</tr>
<tr>
<td>340832</td>
<td>The CICS FEPI demo COBOL source file pgadb2f.cob was corrupted.</td>
</tr>
</tbody>
</table>

Release 3.3.1.0.1

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>231644</td>
<td>PGAU core dumped if LANGUAGE environment variable was not set.</td>
</tr>
<tr>
<td>231768</td>
<td>PGAU did not handle COBOL variable names continued across multiple input lines.</td>
</tr>
<tr>
<td>231779</td>
<td>PGAU did not reject ambiguous references in COBOL definitions.</td>
</tr>
<tr>
<td>235026</td>
<td>PGAVSN procedure returned wrong version number.</td>
</tr>
<tr>
<td>Bug Number</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>236295</td>
<td>Misleading PGA-20910 message was issued when the remote transaction program de-allocated a conversation without sending expected data.</td>
</tr>
<tr>
<td>240908</td>
<td>Lowercase gateway SID was converted to uppercase when used in a log file name.</td>
</tr>
<tr>
<td>244606</td>
<td>PGAU converted the LUNAME specified in a DEFINE TRANSACTION statement to uppercase.</td>
</tr>
<tr>
<td>244612</td>
<td>PGAU converted the TPNAME specified in a DEFINE TRANSACTION statement to uppercase.</td>
</tr>
<tr>
<td>244654</td>
<td>PGAU converted the SIDEPROFILE specified in a DEFINE TRANSACTION statement to uppercase.</td>
</tr>
<tr>
<td>252135</td>
<td>PGAU failed to reject invalid trace options specified in the DIAGNOSE(TRACE()) parameter on a GENERATE statement.</td>
</tr>
<tr>
<td>257787</td>
<td>Message PGA-20905 was received from the gateway server if conversations were terminated in a different order than they were started.</td>
</tr>
<tr>
<td>257998</td>
<td>PGAADMIN roles could not be assigned to other users.</td>
</tr>
<tr>
<td>261238</td>
<td>PGAU incorrectly reported the SYNCELEVEL value for a transaction as 0 even when the DEFINE TRANSACTION statement specified SYNCELEVEL(1).</td>
</tr>
<tr>
<td>264615</td>
<td>PGAU encountered a segmentation violation when generating a TIP.</td>
</tr>
<tr>
<td>268122</td>
<td>PGAU generated incorrect PL/SQL if the COBOL REDEFINES WHEN construct was used.</td>
</tr>
<tr>
<td>274137</td>
<td>PGAU did not allow MODE or LU names to begin with $, #, or @.</td>
</tr>
<tr>
<td>278549</td>
<td>PGAU generated incorrect table keys for OCCURS group, resulting in an ORA-6502 message.</td>
</tr>
<tr>
<td>284247</td>
<td>PGAU encountered a segmentation violation when generating a TIP if a SIDEPROFILE was specified in the DEFINE TRANSACTION statement.</td>
</tr>
<tr>
<td>287169</td>
<td>PGAU GENERATE computed incorrect data offsets after multiple adjacent OCCURS groups.</td>
</tr>
<tr>
<td>312840</td>
<td>PGAU GENERATE skipped a unique nested group that followed a duplicate nested group.</td>
</tr>
</tbody>
</table>
Known Problems in Previous Releases

The following section lists problems that were known to exist in previous releases of the gateway.

Release 8.0.6.1.0

No known problems were documented in release 8.0.6.1.0.

Release 4.0.1.1.0

The following problems in other products are known to affect the operation of products in this release:

- UTL_PG Does Not Handle Numeric Data Values Between 0 and +1 (Bug No. 360427)
  The UTL_PG.NUMBER_TO_RAW function erroneously issues messages ORA-08414 and ORA-08467 when converting an Oracle number between 0 and +1 into a COBOL numeric value. Contact Oracle Support Services to order the patch correcting this bug.

- PGA Grows Excessively When Using Indexed PL/SQL Tables (Bug No. 373700)
  When TIPs are executed that use large PL/SQL tables, the memory usage grows excessively. Contact Oracle Support Services to order the patch correcting this bug.

- UTL_PG Does Not Handle Leading or Trailing Signs Correctly for DISPLAY Data (Bug No. 383510)
  The UTL_PG.RAW_TO_NUMBER and UTL_PG.NUMBER_TO_RAW functions do not handle leading or trailing signs correctly for numeric data defined with USAGE IS DISPLAY. Contact Oracle Support Services to order the patch correcting this bug.
Symbols

$ORACLE_HOME, 4-5, 4-13, 4-23, 4-27, 4-28, 4-29, 4-30, 4-31, 4-32, 4-33, 4-34, 4-35, 4-36, 4-38, 4-40, 6-3, 7-3, 8-3, 8-4, 8-6, 8-9

(HS=) (TNSNAMES parameter for Oracle Net)
with IPC protocol, 4-25
with TCP/IP protocol, 4-25

Numerics

2_PHASE, A-10, A-11
  capability, A-9
  or 2P, A-3

A

accessing the gateway from other servers, 4-38
action items
  for installing the gateway, 4-10
activating and verifying SNA server profiles, 6-12
ADDR parameter, 8-7
ADVICE column
  PGA_2PC_PENDING table, 9-8
AIX
  APPC support provided by SNA, 6-1
  System Management Interface Tool, 6-5
AIX only
  Two Phase Commit, 4-26
AIX-only
  password encryption utility, 1-3
ALTER USER command, 4-17
APPC, 2-2
  conversation security option, A-5, A-8

APPCCP log file
  changes to, 1-2

APPCCP PIP data, 9-25
  Also See Pip data
  restrictions, 1-5
appcdef.doc file, 4-29

APPCCP/MVS
  configuring for the gateway, 4-31
  installation verification, 4-36
  not supported by two-phase commit, 9-2
  verification of configuration, 4-36
architecture
  components of the gateway, 2-7
  two-phase commit, 9-2

ASCII
  automatic conversion, 2-3
  authentication
    operating system authentication, 5-3
    Oracle authentication, 5-3
  types, 5-3

B

backout possibilities during migration, 10-2
bootstrap shell scripts, creating, A-11, A-14
Bourne shell, 4-7, 4-13
bugs
  bypassing a bug in IDMS-DC, 4-28
  fixed in previous releases, C-12
  fixed in this release, 1-3

C

C shell, 4-7, 4-13
CDMSLIB DD statement, 4-30
CD-ROM drive requirements, 3-3
changes
  in previous releases, C-1
  in this release, 1-2
    PGA_TRACE desupported, C-2
    TRACE_LEVEL and LOG_DESTINATION parameters moved, C-2
releases
  release 3.3.1.0.1, C-6
  release 3.4.0.1.0, C-5
  release 4.0.0.1.0, C-4
  release 4.0.1.1.0, C-3
  release 8.0.6.1.0, C-2
changes and enhancements
  password encryption utility, 1-3
  to Oracle Call Interface, 1-2
CICS, 2-11, 3-5, 9-24, 9-31
  application log, 9-27
  installation verification, 4-33
  restriction, PF keys, 1-5
  security options, 6-4, 7-4, 8-4
    ATTACHSEC parameter, 6-4, 7-4, 8-4
    not supported by the gateway, 6-4, 7-4, 8-4
  transaction ID, 4-34
  two-phase commit logging scheme for CICS applications, 9-29
  verifying configuration, 4-33
CICS/ESA
  authentication mechanism, 6-3, 7-3, 8-3
  configuring for the gateway, 4-27
  configuring for two-phase commit, 9-17
  design constraints re
    two-phase commit, 9-23
  minimizing exposures on two-phase commit, 9-24
OLTP
  configuration verification, 4-33
  requirements, 3-5
  support for two-phase commit, 9-2, 9-23
  TP name length, A-4, A-8
CICS/MVS
  OLTP requirements, 3-5
CICSPGA file, 8-10
CICSPGA2, 9-15
CICS/VSE
OLTP requirements, 3-5
COBOL
  restrictions on COPY REPLACE clause, 1-6
cold start
  RRM recovery, 9-20
column
PGA_2PC_LUS table
    CS_LUNAME, 9-8
    LOG_NAME, 9-8
    LOG_STATUS, 9-8
    LU_NAME, 9-8
    LU_TYPE, 9-8
PGA_2PC_PENDING table, 9-8
    ADVICE, 9-8
    CONV_CORR, 9-10
    DB_USER, 9-9
    LU_NAME, 9-9
    LUW_ID, 9-10
    OS_TERMINAL, 9-9
    SESS_ID, 9-10
    SID, 9-9
    SIDE_NAME, 9-9
    STATE, 9-8
    TP_NAME, 9-10
    TRAN_COMMENT, 9-9
command
  SET, 4-24
  COMMIT, 9-6, 9-7, 9-9, 9-23, 9-26, 9-29, A-9, A-12
  commit, 9-5, 9-9
  COMMIT FORCE, 9-28
  COMMIT_CONFIRM, A-3, A-6, A-10, A-12
  and PGA_CAPABILITY parameter, A-12
capability, A-12
  COMMIT_POINT_STRENGTH, 9-24
  commit-confirm, A-10, A-12
  configuring the gateway, 4-26
  COMMIT/ROLLBACK, 3-5
communication between server, gateway and target system, 2-9
components
  in this release, 1-2
configuration
  verification of OLTP, 4-32
configuring
  APPC/MVS, 4-31
CICS/ESA, 4-27

gateway
  commit-confirm, 4-26
  optional steps to allow multiple users, 4-19
HP-UX for the gateway, 4-23
IDMS-DC MVS, 4-28
IMS/TM, 4-30
OLTP, 4-27
Oracle integrating server
  first-time installation steps, 4-19
  pre-configuration steps, 4-13
  upgrading from previous releases, 4-17
SNAplus2
  on the gateway for HP-UX, 4-23
SunLink SNA Peer-to-Peer, 8-4
the gateway, 4-13
two-phase commit, 9-11
upgrade steps, 4-19
your network, 4-24
CONNECT clause, 6-2, 7-2, 8-2
  for database link security, 5-4
CONV_CORR column
  PGA_2PC_PENDING table, 9-10
corrected problems
  release 3.3.1.0.1, C-14
  release 3.4.0.1.0, C-14
  release 4.0.0.1.0, C-13
  release 4.0.1.1.0, C-12
  release 8.0.6.1.0, C-12
CP directive, 8-7
CPI-C, 4-38
  function
    gateway error messages, B-2, B-5, B-6
CPI-C profiles, 7-9
create
  new product installation directory, 4-6
  public database link, 4-17
creating
  SNA definitions, 7-6
  SNAplus2 definitions, 7-6
  SNAplus2 profiles, 7-4
CS_LUNAME column
  PGA_2PC_LUS table, 9-8
CW_AUTOACT_LMT parameter, 8-8

data compression
  restrictions, 1-5
data dictionary, see PG DD
data exchange
  PGAXFER function, 2-10
database instance, 2-9
database link, 2-10, A-11, A-13
  connect clauses, 5-4
  creating, 4-17
    to access gateway from Oracle servers, 4-38
    when configuring integrating server, 4-17
  dropping, when deinstalling the gateway, 4-17
  in configuring the network, 4-24
  in verifying gateway installation, 4-33
  public and private, 5-4
  security, CONNECT clause, 5-4
database link name
  modifying .sql files, 4-32
datastores
  gateway access to, 2-2
DB_USER column
  PGA_2PC_PENDING table, 9-9
DBA_2PC_PENDING table, 9-7
DBMS_OUTPUT packages, 4-15, 4-38
DBMS_PIPE, 4-17, 4-19, 4-20
dedicated local LU
  required component of two-phase commit, 9-3
DEFINE TRANSACTION statement, 7-6
de-installing
  the gateway, 4-39
dependent LU, 4-30, 6-4, 7-5
  also see, LUs
  and OLTP, 7-5
describe statement, 4-15
  DBMS_OUTPUT, 4-15, 4-16, 4-38
  UTL_RAW, 4-15
dfhcsdup.jcl file, 4-27, 9-17
DFHRPL DD statement, 4-28
disk space requirements, 3-3
DISPLAY datatypes, 1-5
DLC directive, 8-7
documentation required, xv
EBCDIC
  automatic conversion,  2-3
  enhancements
  heterogeneous services,  C-2
  in this release,  1-2
  release 3.3.1.0.1,  C-6
  release 3.4.0.1.0,  C-5
  release 4.0.0.1.0,  C-4
  release 4.0.1.1.0,  C-3
  release 8.0.6.1.0,  C-2
  response time and CPU utilization
  performance,  C-2
  using PGAU to automatically upgrade PG DD
  entries,  4-22
environmental variables
  ORACLE_HOME,  4-7
  TNS_ADMIN,  9-13
error
  during commit processing,  A-14
  error conditions in the RRM,  9-20
  gateway server messages,  B-5, B-6
  in-doubt transaction message ORA-02050,  9-26
  in-doubt transaction message ORA-02053,  9-26
  LOG_ERRORS,  9-15
  LU6.2 errors
    caused by LUWID not unique,  9-33
    determined by gateway server,  9-4
  messages from the gateway,  B-1
  obsolete parameters,  10-4
  ORA-28509,  4-24
  parameter name misspelled,  4-24
  PGA-20910 communication error message,  B-2,
    B-5, B-6
  PGA-20995 communication error message,  B-3
  PGA-21110,  9-21
  PGA-21450 RRM communication error,  B-4
  PGA-21451 RRM message
    communication error,  B-5
  RRM message
    partner LU message PGA-21215,  9-21
  RRM message PGA-21218,  9-21
  system error number “errno”,  B-2, B-5
  system error number “errno”,  B-6
  treating incoming APPC CONFIRM requests as
  errors,  A-3, A-7, A-14
  EXEC CICS ROLLBACK,  9-31, 9-33

F
failure points,  9-26
FDS_CLASS parameter,  10-3
FDS_CLASS_VERSION
  parameter added,  10-5, C-2
FDS_INSTANCE parameter,  10-3
file
  appcdef.doc,  4-29
  CICSPGA,  8-10
  dfhcsdup.jcl,  4-27, 9-17
  HQWM804.trc,  9-16
  imsgen.asm,  4-31
  initPGAIora,  A-11, A-14
  initPGA.ora,  4-23
    sample,  A-18
  initPGAU.ora,  A-11, A-14
  initsid.ora,  2-4, 2-9, 4-23, 9-13, 9-24, 9-27, 10-2,
    A-2, C-2, C-3
  gateway parameters,  4-23
  new parameters,  10-3
  new parameters since V4,  10-4
  inittab,  9-19, 9-20
  install.log,  4-40
  listener.ora,  4-7, 4-24, 4-25, 4-26, 4-40, A-11,
    A-13
  luname.ini,  9-21, 9-22
  oralu62.asm,  4-30
  orapl62.asm,  4-27, 4-28, 4-30, 4-31
  orasyn62.asm,  9-17
  pg4hoa1.sh,  4-24, 8-6, 8-9, A-11, A-13, A-14, C-2
  new startup shell parameter,  10-5
  new startup shell parameters,  10-3
  sample,  A-15
  pga2pcau.sql,  9-12
  pga2pclg.sql,  9-12
  pga2pclu.sql,  9-12
  pga2pcpn.sql,  9-12
  pgacics.sql,  4-32, 4-33, 4-34
  pgadb22.cob,  9-29, 9-30, 9-31
  pgaecho.sql,  4-32
pgaflip.asm, 4-27, 4-28, 4-30, 4-31
pgaflip.jcl, 4-27, 4-28, 4-31
pgaidms.sql, 4-32, 4-34, 4-35
pgaim.sql, 4-36
pgaims.sql, 4-32, 4-35
pgaluwid.asm, 9-30, 9-31
pgaluwid.cob, 9-31
pgaluwid.jcl, 9-31
pgamvs.sql, 4-32, 4-36
pgasna.export, 6-5, 6-6, 6-7
PGAU control files, 4-3
pgavsn.sql, 4-32
pgddadev.sql, 4-21
pgddapub.sql, 4-21
prvtpgb.plb, 4-18
README.doc, 4-38
sunpu2.config, 8-6
tnsnames.ora, 4-7, 4-17, 4-24, 4-25, 4-26, 4-40, 4-13, 9-14, A-11, A-13, A-14
trandef.doc, 4-29
uname.rrm, 9-18
utlpq.sql, 4-18
utlrw.sql, 4-18
FLIP transaction
OLTP configuration
   and verification for APPC/MVS, 4-36
   and verification for CICS/ESA, 4-33
   and verification for IDMS-DC, 4-34
   and verification for IMS/TM, 4-31, 4-35
floating point datatype
   restrictions, 1-5
function
   put_line, 4-16
functions
   see RPC (remote procedural call)
   see UTL_PG
   see UTL_RAW

G

gateway
   access to IBM datastores, 2-2
   accessing from other Oracle servers, 4-38
   communications with all platforms, 2-2
   components, 2-7, 2-9, 4-13
   configuring, 4-13
      for HP-UX, 4-23
      for multiple users, 4-19
   creating SNA definitions for, 7-6
   creating SNAplus2 profiles, 7-4
   de-installing, 4-39
   directory see product installation directory, 4-6
   error messages, B-1
   factors affecting memory requirements, 3-2
   features, 2-2
      application transparency, 2-2
      code generator, 2-3
      fast interface, 2-2
      flexible interface, 2-3
      location transparency, 2-2
      Oracle server integration, 2-3
      Performs automatic conversions, 2-3
      site autonomy and security, 2-3
      support for tools, 2-3
   functions, 2-11
   initialization files, 2-4, 9-3
      sample initPGA.ora, A-18
   initialization parameters, 9-12, C-3
      also see PGA parameters
   configuring for two-phase commit, 9-12
      described, 4-23
      list, A-1
      location, A-2
      new, 10-2
      new and changed since Version 4
         gateway, 10-2
      renamed since V4, 10-4
      SET, 4-24, A-2
      stored, C-3
      to configure two-phase commit, 9-12
   installation
      first-time install, configuring the integrating server, 4-15
      pre-installation procedures, 4-4
      steps, 4-6
      verification, 4-32
   installing, 4-1
      with Oracle Universal Installer action items, 4-10
   overview, 2-2
parameter files, 4-23
   also see gateway initialization parameters, and
   PGA parameters
   initPGA.ora, 4-23
remote procedure, definition, 2-4
requirements
   hardware, 3-2
restoring to previous releases, 4-3
RPC functions, 9-5
security options and overview, 5-2
   also see, security
server
   description in two-phase commit, 9-4
   error messages, B-5, B-6
   interaction with Oracle logging server in
two-phase commit, 9-5
   restoring previous version, 4-3
   support for two-phase commit, 9-3
setting up multiple gateway instances, A-11, A-13
SNA
   configuration, 8-1
   security validation, 6-2, 7-2, 8-2
SNA profiles, see SNA server profiles, 6-5
starting, 2-9
startup shell parameters
   FDS_CLASS_VERSION, 10-5
steps to install, via Oracle Universal
   Installer, 4-10
upgrading
   from previous release, 4-2
   preparing to upgrade, 4-3, 4-14
gateway initialization parameters
   see gateway, also see, PGA parameters
global transaction ID, 9-8
GLOBAL_TRAN_ID column
   PGA_2PC_PENDING table, 9-8
gpglocal, 4-19
   needed to compile PGAU-generated TIP
   specifications, 4-19
gpglocal.pkb script, 4-19
gpglocal.pkh script, 4-19
grant
   access, 4-20
   authorization, 4-19
execute, 4-17
explicit, 4-20
private, 4-20
public, 4-20

H
   hardware requirements, 3-2
   Heterogeneous Services (HS), 10-3
      and Oracle Net considerations, 10-6
catalogs
      installing, 4-16
definition, C-2
HOST key, 4-26
HP-UX, 6-2, 7-2, 8-2
   and security, 6-2, 7-2, 8-2
   and SNA profiles, 6-2, 7-2, 8-2
   and SNA security validation, 6-2, 7-2, 8-2
   APPC support provided by SNAplus2, 7-1
configuration for the gateway, 4-23
for OLTP verification, 4-34
gateway components, 2-9
   in gateway architecture, 2-8
   in verifying CICs, 4-34
   operating system, 2-8
   starting the gateway, 2-9
HQWM804.trc file, 9-16
HS parameters, 10-3
   see also, (HS=)
   HS_COMMIT_STRENGTH_POINT
      parameter, 10-4
   HS_DB_DOMAIN parameter, 10-4
   HS_DB_INTERNAL_NAME parameter, 10-4
   HS_DB_NAME parameter, 10-4
   HS_DESCRIBE_CACHE_HWM parameter, 10-4
   HS_FDS_FETCH_ROWS parameter, 10-3
   HS_LANGUAGE parameter, 10-4
   HS_NLS_DATE_FORMAT parameter, 10-3
   HS_NLS_DATE_LANGUAGE parameter, 10-4
   HS_NLS_NCHAR parameter, 10-3
   HS_OPEN_CURSORS parameter, 10-4
   HS_ROWID_CACHE_SIZE parameter, 10-3
   HS_RPC_FETCH_REBLOCKING parameter, 10-3
   HS_RPC_FETCH_SIZE parameter, 10-3
using smit to start, 6-12
listener.ora file, 4-7, 4-24, 4-25, 4-26, 4-40, A-11, A-13
LOCADDR parameter, 8-7
local LU, 7-5
LOG_DB RRM parameter, 9-14
LOG_DESTINATION parameter, 10-2, 10-3
moved to initsid.ora, C-2
LOG_DIRECTORY RRM parameter, 9-14
LOG_ERRORS RRM parameter, 9-15
maximum number of Oracle errors before RRM shuts down, 9-15
LOG_FILE RRM parameter, 9-14
LOG_NAME column
PGA_2PC_LUS table, 9-8
LOG_PASS RRM parameter, 9-15
LOG_STATUS column
PGA_2PC_LUS table, 9-8
LOG_USER RRM parameter, 9-15
logging scheme
requirements for logging two-phase commit for CICS transactions, 9-29
logging server
See Oracle logging server
Logical Unit of Work (See LUW), 9-29
logmode entry name, 4-34
LTERM, 4-29
LU name
assigning to the gateway, 6-7
LU NAME column
PGA_2PC_LUS table, 9-8
PGA_2PC_PENDING table, 9-9
LU TYPE column
PGA_2PC_LUS table, 9-8
LU6.1 Adapter for LU6.2 applications, 4-30
LU6.2, 6-2, 7-2, 8-2
server, 8-4
LU6.2 log information, 9-7
luname.ini file, 9-21, 9-22
LUs, 6-2, 6-4, 7-2, 8-2
(Logical Units), 6-7
alias identified by side information profile, 6-9
and gateway security, 5-4
and Side Information Profiles, 7-4
choosing a dedicated local LU for two-phase commit, 9-10
defining a directive, 8-7
dependent, 6-4
for configuring LU6.2 for IMS/TM for the gateway, 4-30
independent
for optimal network performance, 3-2
in configuring APPC/MVS on the gateway, 4-31
in configuring CICS/ESA, 4-27
in configuring IDMS-DC MVS, 4-28
in configuring IMS/TM for the gateway, 4-30
vs. dependent, 8-5
local, 7-5
local LU
profile, 6-7, 9-16
required for two-phase commit, 9-3
partner, 7-5
defined by PTNR LU directive, 8-8
partner LU, 9-21
location profile, 6-8
profile, 6-8
partner LU name, 8-9
assigning alias, 6-8
LUTYPE parameter, 8-7
luw_id, 9-27
LUW_ID column
PGA_2PC_PENDING table, 9-10
LUWID, 9-27, 9-29
info provided to gateway at startup, 9-25
log, 9-30
log cleanup, 9-33
log table or file, 9-29

M
manual recovery, 9-23
of in-doubt transactions, 9-26
memory requirements, 3-2
message
ORA-02050, 9-26
ORA-02053, 9-26
RRM partner LU message PGA-21215, 9-21
RRM partner LU message PGA-21218, 9-21

Index-8
messages
  gateway server error messages, B-5, B-6
  PGA-21110, 9-21
  RRM error messages, 9-20, B-4
  see error
migrating
  an existing gateway instance to the new release, 10-2
  backout considerations when migrating to new release, 10-2
  from Version 4 to 9i special parameters, 10-2
  see error
  migrating an existing gateway instance to the new release, 10-2
  backout considerations when migrating to new release, 10-2
  from Version 4 to 9i special parameters, 10-2
Mode
  and Side Information Profiles, 7-4
  MODE directive, 8-8
Mode name, 8-9
mode profiles
  SNA server, 6-6, 9-16
  mounting the CD-ROM
    for AIX, 4-8
    for HP-UX, 4-8
    for Solaris, 4-9
multi-byte character set (MBCS) restrictions, 1-5
multi-conversational transactions, 2-11
MVS/ESA
  OLTP requirements, 3-5
N
NAME parameter, 8-8, 8-9
  network
    configuring, 4-24
    reconfiguring, 4-24
  node profile, 6-6
  NQ_LU_NAME parameter, 8-8
O
obsolete parameters, 10-4
OLTP, 4-23, 7-5, 7-6, 8-5, A-9, A-10, A-12
  accessing from the Oracle integrating server, 4-39
  and dependent LUs, 7-5
  and SECURITY=PROGRAM option, 6-2, 7-2, 8-2
  and SNA SECURITY=SAME option, 6-3, 7-3, 8-3
  configuration verification, 4-32
  APPC/MVS, 4-36
  CICS/ESA, 4-33
  IDMS-DC MVS, 4-34
  IMS/TM, 4-35
configuring, 4-27
  APPC/MVS for the gateway, 4-31
  CICS/ESA for the gateway, 4-27
  IDMS-DC MVS for the gateway, 4-28
  IMS/TM for the gateway, 4-30
  to allow gateway access, 4-2
currently supported types, 4-27, 4-33
defining PTERM and LTERMs (configuration of IDMS-DC MVS), 4-29
definition, 2-4
gateway architecture
  installation verification, 4-33
  mode name specification, 6-6
  post-installation steps, 4-37
  remote, 2-1
  requirements, 3-5
  security and inbound APPC session requests, 6-4, 7-4, 8-4
  security on the gateway, 5-2
  SNA security option, 6-3, 7-3, 8-3
  support for two-phase commit, 9-2
  user ID mapping, 5-4
  verifying configuration, 4-33
  one-shot transactions, 2-11
  online transaction processor
    See OLTP
operating system
  HP-UX, 2-8
Oracle Adapter
  network needs, 3-5
Oracle Call Interface
  changes to, 1-2
Oracle directory tree, 4-6
Oracle global transaction ID, 9-27
Oracle Heterogeneous Services
  see Heterogeneous Services
Oracle integrating server, 2-10, 4-2, 4-4, 4-5, 4-13,
accessing
  gateway from other servers, 4-38
  OLTPs from, 4-39
  other OLTPs, 4-39
Also called Oracle9i Server
and gateway security, 5-4
component of the gateway, 2-7
configuring
  for first-time gateway installations or upgrade, 4-15, 4-19
connecting to gateway server in two-phase commit, 9-3
controlling component in two-phase commit, 9-4
definition, 2-4
enabling DBMS_OUTPUT PL/SQL package, 4-15
function with RRM parameters, 9-14
installation steps, 4-6
interaction with gateway server in two-phase commit, 9-5
LOG_PASS RRM password parameter, 9-15
logging server locations, 9-4
logon authentication needed, 5-3
multiple servers on the gateway, 2-8
precompiles PL/SQL package, 2-2
pre-configuration steps, 4-13
READ_ONLY mode, A-9, A-12
rename links when upgrading the gateway, 4-14
role
  in gateway communication, 2-9
  in logon security, 5-4
  in starting the gateway, 2-9
separate directory from gateway, 4-5
shipped with PL/SQL packages, 4-20
shut down before shutting down RRM, 9-19
starting, before RRM, 9-19
stores PL/SQL, 2-5
system shutdown procedure, 9-20
upgrading
  from previous releases, 4-17
  verifying
APP/PC/MVS configuration, 4-37
gateway installation, 4-33
IMS/ TM, 4-36
OLTP configuration, 4-34
Oracle local transaction ID, 9-27
Oracle logging server, 9-4, 9-13
component of two-phase commit, 9-3
configuring for two-phase commit, 9-12
description of function in two-phase commit, 9-4
in manual transaction recovery, 9-27
in resolving in-doubt transactions, 9-27
interaction with gateway server, 9-5
interaction with RRM, 9-5
two-phase commit interactions, 9-5
Oracle Net, 2-4, 9-13, 9-14, A-4, A-7
accessing gateway from other Oracle servers, 4-38
function on the gateway, 2-7
Heterogeneous services/tnsnames.ora considerations, 10-6
listeners, 9-19
  shut down after shutting down RRM, 9-19
  starting before starting RRM, 9-19
protocol for accessing gateways, 4-5
release, 1-2
security considerations, 5-3
version needed, 1-2
Oracle Procedural Gateway for APPC
  also see gateway
de-installing, 4-39
development environment, 2-3
functions, 2-11
Oracle Universal Installer, 4-9
  starting for AIX, 4-9
  using
to de-install the gateway, 4-40
to install gateway, 4-10
ORACLE_HOME environmental variable, 4-7
Oracle9i Server
  and networking products needed, 3-5
  release 9.0.1.0.0 needed, 1-2
oralu62.asm file, 4-30
ORAPGA.APPCMVS.SAMPLIB, 4-31
oraplus2.asm file, 4-27, 4-28, 4-30, 4-31

Index-10
Index-11

orasy62.asm file, 9-17
OS_TERMINAL column
PGA_2PC_PENDING table, 9-9
override
user ID and password, 6-2, 7-2, 8-2

P

package
DBMS_OUTPUT, 4-15, 4-38
UTL_PG, 4-16, 4-20, 4-38
invaldated or deinstalled, 4-18
UTL_RAW, 4-15, 4-38
invaldated or deinstalled, 4-18
package specifications
avoid reinstalling, 4-18
PAR_SESS_SUP parameter, 8-8
parameter
RRM
initialization for two-phase commit, 9-14
LOG_DB, 9-14
LOG_DIRECTORY, 9-14
LOG_ERRORS, 9-15
LOG_FILE, 9-14
LOG_USER, 9-14
password parameter LOG_PASS used to
connect to Oracle integrating
server, 9-15
TRACE_DIRECTORY, 9-15
TRACE_FILE, 9-16
TRACE_LEVEL, 9-16
parameter files
see gateway initialization files
see PGA parameters
see RRM parameters
parameters
ADDR, 8-7
changed since Version 4 gateway, 10-2
CW_AUTOACT_LMT, 8-8
FDS_CLASS_VERSION, 10-5
gateway initialization
parameter, described, 4-23
gateway initialization parameters needed for
two-phase commit, 9-12
IDBLK, 8-7

IDNUM, 8-7
LCL_MAX_SESS_LMT, 8-8
LOCADDR, 8-7
LTYPE, 8-7
NAME, 8-8, 8-9
new
FDS_CLASS, 10-3
FDS_INSTANCE, 10-3
new initialization parameters since V4 gateway
release, 10-2
NQ_LU_NAME, 8-8
obsolete, 10-4
PAR_SESS_SUP, 8-8
PGA

described, 4-24
renamed startup shell script
since Version 4 gateway, 10-4
RMACADDR, 8-7
RRM
see RRM parameters
see PGA parameters and gateway
SESS_LMT, 8-7
TERMID, 8-7
USING, 4-24
partner LU, 7-5, 9-21
error message, 9-21
locations profile, 6-8
name, 8-9
profile, 6-8
password
change using ALTER USER command, 4-17
Oracle authentication, 5-3
Oracle password to be used by gateway, A-4,
A-7
overrides, 6-2, 7-2, 8-2
with operating system authentication, 5-3
Password Encryption Utility
description, 1-3
password encryption utility
AIX-only, 1-3
PDS (partitioned dataset), 4-31
persistent transactions, 2-11
PF keys, 1-5
PG Data Dictionary
See PG DD

Index-11
PG DD
after upgrade, 4-3
allowing multiple users, 4-19
creating public synonyms for multiple
users, 4-19
definition, 2-5
install script, 4-17
installing for gateway configuration, 4-17
no access from earlier PGAU versions, 4-3
packages
dropping after de-installing the
gateway, 4-39
restoring previous version, 4-3
tables, 4-20
upgrade when upgrading gateway, 4-18
using PGAU to upgrade existing entries, 4-22
pg4appc directory
product installation directory, 4-6
PG4ARRM, 9-15
pg4arrm executable, 9-11
pg4hoa1.sh file, 4-24, 8-6, 8-9, A-11, A-13, A-14, C-2
new parameters, 10-3
new startup shell parameters, 10-5
parameter added, C-2
parameters renamed since Version 4, 10-4
possible error message, 4-24
sample, A-15
setting up environmental variables, 8-6
PGA, A-5, A-8
definition, 2-4
initialization files
initPGAI.ora and initPGAU.ora, A-11, A-14
PGA parameters, A-2
described, 4-24
LOG_DESTINATION, A-2, A-6
PGA_CAPABILITY, 4-24, 9-12, 9-13, A-3, A-6
options for updating foreign databases, A-9, A-12
protections against data problems, A-9, A-12
PGA_CONFIRM, A-3, A-7
choosing settings, A-14
PGA_LOCAL_LU, 9-13
PGA_LOG_DB, 9-13, A-4, A-7
PGA_LOG_PASS, 9-13, A-4, A-7
PGA_LOG_USER, 9-13, A-4, A-7
PGA_RECOVERY_PASS, A-4, A-8
PGA_RECOVERY_TPNAME, A-4, A-8
PGA_RECOVERY_USER, A-5, A-8
PGA_SECURITY_TYPE, 6-2, 7-2, 8-2, A-5, A-8
PGA_SECURITY_TYPE=NONE, 6-2, 7-2, 8-2
PGA_SECURITY_TYPE=SAME, 6-3, 7-3, 8-3
TRACE_LEVEL, A-5, A-9
PGA_2PC_LUS table, 9-3
columns, 9-8
maintained by RRM, 9-7
modified by logging server, 9-5
with two-phase commit, 9-4
PGA_2PC_PENDING table, 9-3, 9-7, 9-20, 9-27
columns, 9-8
created, 9-12
maintained by gateway server, 9-7
modified by logging server, 9-5
with two-phase commit, 9-4, 9-6
PGA_CAPABILITY
See PGA parameters
PGA_CC_PENDING table
unsuccessful rollback, C-12
PGA_CONFIRM
See PGA parameters
PGA_LOCAL_LU
See PGA parameters
PGA_LOG_DB
See PGA parameters
PGA_LOG_PASS
See PGA parameters
PGA_LOG_USER
See PGA parameters
PGA_SECURITY_TYPE
See PGA parameters
PGA_SECURITY_TYPE=NONE
See PGA parameters
PGA_SECURITY_TYPE=SAME
See PGA parameters
PGA_TRACE parameter
no longer supported, C-2
PGA-20910 message
communication error, B-2, B-5, B-6
PGA-20995 message
communication error, B-3
PGA-21450 RRM message
communication error,  B-4
PGA-21451 RRM message
   SNA server/6000 is shutting down,  B-5
pga2pcau.sql
   file,  9-12
   script,  9-12, 9-13
pga2pclg.sql
   file,  9-12
   script,  9-12, 9-13
pga2pclu.sql
   file,  9-12
   script,  9-12, 9-13, 9-20
pga2pcpn.sql
   file,  9-12
   script,  9-12, 9-13
PGAAADMIN,  4-19, 4-33
   creating the gateway administrator user
   ID,  4-17
   granting access to additional users,  4-19
   granting execution privileges on DBMS_PIPE,  4-17
   initial password during creation,  4-17
pgacics.sql file,  4-32, 4-33, 4-34
pgacr8au.sql script,  4-17
pgadb22.cob
   file,  9-30
pgadb22.cob file,  9-29, 9-31
PGADBA user id, created,  9-12
pgaecho.sql file,  4-32
pgaflip.asm file,  4-27, 4-28, 4-30, 4-31
pgaflip.jcl file,  4-27, 4-28, 4-31
PGAI
   setting up,  A-11, A-13
pgaidms.sql file,  4-32, 4-34, 4-35
pgaims.sql file,  4-32, 4-35, 4-36
PGAINIT function,  2-4, 2-10, 9-5
PGALUWID,  9-31
   subroutine,  9-31
pgaluwid.asm file,  9-30, 9-31
pgaluwid.cob file,  9-31
pgaluwid.jcl file,  9-31
pgamvs.sql file,  4-32, 4-36
pgasna.export file,  6-5, 6-6, 6-7
PGATERM function,  2-4, 9-5
pgatiptr.sql script,  4-19

PGAU,  4-26
   access to Oracle integrating server,  4-14
   accesses definitions in PG DD,  2-5
   control files,  4-3
   deciding where to store TIPS and PGAU input
   files,  4-5
   DEFINE/REDEFINE DATA statements,  1-5
definition
   used to generate TIP specifications,  2-5
   execution in stand-alone installation,  4-5
   -generated TIP specifications,  2-9
   -generated TIP specifications use UTL_PG,  4-16
   -generated TIP specifications use UTL_RAW,  4-15
generating TIP specifications, PL/SQL v3
   compatibility,  C-3
   purpose of PGDL,  2-5
   release number,  1-2
   restoring previous versions,  4-3
   setting up,  A-11, A-13
   upgrading existing PG DD entries,  4-22
   USAGE(NULL) keyword,  1-5
PGAU commands
   DEFINE DATA
   - COBOL COPY REPLACE restrictions,  1-6
   - GENERATE,  4-21, 4-22
      produces TIP in output files,  4-22
      to upgrade existing TIPs,  4-21
   pgavsn.sql file,  4-32
   PGAXFER function,  2-4, 2-10, 9-5
   pgddadev.sql
      file,  4-21
      script,  4-21
   pgddapub.sql
      file,  4-21
      script,  4-21
   pgddcr8r.sql script,  4-20
   pgddcr8.sql script,  4-17
   pgddcr8s.sql script,  4-20
   PGDDDEF role,  4-20
   pgdddel.sql script,  4-39
   PGDDGEN role,  4-20
      adding a privilege for upgrading PG DD
      entries,  4-22
   pgddupgr.sql script,  4-18
PGDL (Procedural Gateway Definition Language)
definition, 2-5
PIP data, 6-9, 9-30, 9-31, 9-32
PL/SQL, 2-5
code generator, 2-3
datatypes, 2-9
classified to RAW, 2-10
function in the gateway, 2-2
function references in the gateway, 2-10
running pgateiptr.sql script to create
routines, 4-19
UTL_PG package function, 2-6
UTL_RAW function, 2-6
UTL_RAW package, installation, 4-15
PL/SQL package
definition, 2-5
developer access to, 4-20
enabled, 4-15
functions, 2-9
needed to access gateway from other Oracle
servers, 4-38
See TIP
PL/SQL stored procedure specification
also called "TIP"
See PL/SQL package
post-installation steps for OLTP, 4-37
privileges
needed to create TIPs, 4-21
Procedural Gateway Administration
see PGA
Procedural Gateway Administration Utility
also see PGAU, 2-9
processor requirements, 3-2
product installation directory, 4-6
naming, 4-6
see ORACLE_HOME and pg4appc directory
product publications, xv
product set in this release, 1-2
profiles
see side information profile, 9-17
see SNA server profiles
transaction program name profile, 9-17
protocol
IPC, 4-25
TCP, 4-25
prvtpgb.plb
file, 4-18
script, 4-16, 4-18
prvtrawb.plb script, 4-15, 4-17
PTERM, 4-29
PTNR_LU directive, 8-8
public synonyms for multiple PG DD users, 4-19
put_line function, 4-16

R

READONLY
PGA_COMPATIBILITY setting, A-12
PGA_COMPATIBILITYsetting, A-9
README.doc file, 4-38
RECO
Oracle recovery process, C-12
recompiling TIPs
See TIP
reconfiguring your network, 4-24
recovery actions (manual), 9-27
release-specific information
restrictions, 1-5
remote host transactions (RHT)
types, 2-11
Remote Node definition
creating, 7-8
remote procedural call
See RPC
remote procedure
definition, 2-4
remote transaction program
See RTP
renamed startup shell script parameters since
Version 4 gateway, 10-4
REQUEST-COMMIT, 9-6
requirements
CICS/ESA, 3-5
CICS/MVS, 3-5
CICS/VSE, 3-5
documentation, 3-6
hardware, 3-2
IDMS-DC, 3-5
IMS/TM, 3-5
MVS/ESA, 3-5
network attachments,  3-2  
OLTP,  3-5  
software,  3-3  
system,  3-1  
resolving an in-doubt transaction,  9-26  
Resource Recovery Manager - See RRM,  9-4  
Resource Recovery Manager on AIX  
sample HQWM920.ini file,  A-20  
restoring a previous release of the gateway,  4-3  
restrictions  
   APPC PIP data not supported,  1-5  
n cannot emulate CICS PF keys,  1-5  
CICS/ESA,  1-5  
COBOL COPY REPLACE processing,  1-6  
   floating point datatype conversion,  1-5  
in this release,  1-5  
   multi-byte character set support for numeric data,  1-5  
n   no compression and decompression facilities in gateway,  1-5  
PGAU USAGE null on DEFINE/REDEFINE data,  1-5  
RHDCCNOS multiple session service  
   manager,  4-29  
RMTMACADDR parameter,  8-7  
role  
   PGDDDEF,  4-20  
   PGDDGEN,  4-20  
ROLLBACK,  3-5, 9-10, 9-27, 9-29, 9-30, 9-31, 9-33,  
   A-9, A-12  
   rollback,  9-4, 9-9, 9-24, 9-30  
   from transaction failure,  9-23  
   function of Oracle integrating server,  9-5  
in bug no. 689304,  C-12  
ROLLBACK FORCE,  9-27  
RPC  
   definition,  2-5  
   function  
      PGATERM,  9-5  
   function  
      PGAINIT,  2-4, 2-10, 9-5  
      PGATERM,  2-4  
      PGAXFER,  2-4, 2-10, 9-5  
      within the gateway,  9-5  
      functions within the gateway,  2-2, 2-10  
processing,  2-1  
RRM,  9-3  
   cold start,  9-20  
   configuring initialization parameters for two-phase commit,  9-14  
   description for two-phase commit,  9-4  
   error messages,  B-4  
   in two-phase commit,  9-11  
   interaction with Oracle logging server in two-phase commit,  9-5  
   issued messages  
      PGA-21215,  9-21  
      PGA-21218,  9-21  
   parameter list,  9-14  
   recovery,  9-20  
   required component of two-phase commit,  9-3  
   startup and shutdown procedures in two-phase commit,  9-18  
   startup with AIX user ID,  9-19  
   TP_PROFILE,  9-15  
   RRMSIDE_PROFILE,  9-15  
   rrmstart script,  9-20  
   customizing for two-phase commit,  9-18  
   rrmstop script,  9-18  
RTP  
   and Side Information Profiles,  7-4  
   definition,  2-5  
   executing,  2-4  
   function in the gateway,  2-2  
   PGA_CAPABILITY settings for read-only RTPs,  A-9, A-12  
S  
sample  
   initPGA.ora file,  A-18  
sample applications  
   included,  4-37  
   installing,  4-37  
script  
   gpglocal.pkb,  4-19  
   gpglocal.pkh,  4-19  
   pga2pcau.sql,  9-12, 9-13  
   pga2pclg.sql,  9-12, 9-13  
   pga2pclu.sql,  9-12, 9-13, 9-20
Index-16

SDLC
connection defined by TRLINE or SDCLINE directives, 8-7
security and database links, 5-4
and SNA validation, 6-2, 7-2, 8-2
authenticating application logons, 5-3
authentication mechanisms, 6-3, 7-3, 8-3
link accessibility for public and private databases, 5-4
links and CONNECT clauses, 5-4
overview of gateway security requirements, 5-2
processing inbound connections, 6-4, 7-4, 8-4
requirements, 5-2
specifying SNA conversation security, 6-2, 7-2, 8-2
SECURITY=None SNA security option, 6-2, 7-2, 8-2
SECURITY=PROGRAM, 6-3, 7-3, 8-3
SECURITY=SAME SNA security option, 6-3, 7-3, 8-3
SESS_ID column
PGA_2PC_PENDING table, 9-9
SESS_LMT parameter, 8-7
SET gateway initialization parameter, 4-24
SET command, 4-24
shells see Bourne, Korn, and C shells, 4-1
shutdown script, 9-20
SID column
PGA_2PC_PENDING table, 9-9
side information and Sym Dest Name option, 7-6
side information files, 8-8
identified by Symbolic Destination Name, 8-9
Side Information Profile, 8-5
side information profile, 6-7
defined in two-phase commit, 9-17
function, 6-9
Side Information Profiles, 7-5, 7-9
description, 7-4
side profile, 9-15
side profile name, 4-34
SIDE_NAME column
PGA_2PC_PENDING table, 9-9
SIDE_PROFILE, 9-15
SINGLE_SITE and PGA_CAPABILITY parameter, A-12
smit "Verify Configuration Profiles" option, 6-12
creating SNA profiles for the gateway, 6-5
System Management Interface Tool for AIX, 6-5
SNA parameters, 2-10
profiles, 6-2, 7-2, 8-2
security validation, 6-2, 7-2, 8-2
SNA APPC also see APPC function in the gateway, 2-2
SNA LU, 6-8
SNA node and Link Station, 7-8
SNA node profile, 6-6
SNA profiles See SNA server profiles
SNA security options SECURITY=None, 6-2, 7-2, 8-2
SECURITY=PROGRAM, 6-2, 7-2, 8-2
SECURITY=SAME, 6-3, 7-3, 8-3
SNA server, 6-1, 6-10
configuring
Index-17

for two-phase commit, 9-16
dependent LUs, 6-4
error messages, B-2, B-3
gateway error PGA-20995, B-3
independent LUs, 6-4
profiles, 6-1, 6-5, 6-10
activating and verifying, 6-12
creating, 6-5
definition, 6-1
for non APPN-capable nodes, 6-8
link station, 6-6
local LU profile, 6-7, 9-16
mode profile, 6-6, 9-16
partner LU, 6-8
partner LU location profile, 6-8
prior to creating, 6-5
side information profile, 6-9
SNA node profile, 6-6
transaction program name, 6-9
types, 6-5
RRM error message PGA-21451, B-5
verify and activate profiles, 6-12
SNA/APPN
services, 8-4
SNAplus2, 4-29, 4-34, 4-35, 4-36, 6-3, 7-3, 7-6, 8-3
and LUs, 7-5
and SNA security option, 6-3, 7-3, 8-3
and SNA security validation, 6-2, 7-2, 8-2
APPN menu
Sym Dest Names option, 7-6
configuring, 7-7
for HP-UX on the gateway, 4-23
creating definitions, 7-6
creating profiles for the gateway, 7-4
function, 7-1
testing the connection, 7-10
using xsnadmin to configure, 7-4
SNASVCMG mode entry, 4-29
software requirements, 3-3
for AIX-Based Systems, 3-3
SQL*Plus
sample, used when gateway and Oracle
integrating server share machine, 4-17
to configure Oracle integrating server, 4-15
using to connect to Oracle integrating
server, 4-17
stand-alone
installation, 4-26
location of PGAU execution, 4-5
installing or upgrading on, 4-14
starting
the gateway, 2-9
startup shell script
also see pg4hoal.sh
file
new parameter added, C-2
location of new initialization parameters since V4
gateway, 10-3
STATE column
PGA_2PC_PENDING table, 9-8
statement
CDMSLIB DD, 4-30
describe, 4-16
DFHRPL DD, 4-28
storage measurements, xviii
Sun Solaris
APPc support provided by SunLink SNA Peer-to-Peer, 8-1
SunLink SNA P2P LU6.2
setting up server name, 8-5
SunLink SNA Peer-to-Peer, 8-5, 8-6
and environmental variables, 8-6
and LUs, 8-5
configuration overview, 8-4
configuring, 8-4
CP directive, 8-7
DLC directive, 8-7
LU directive, 8-7
MODE directive, 8-8
PTNR_LU directive, 8-8
sample side information file, 8-10
setting up a configuration file, 8-6
setting up a server name, 8-5
side information files, 8-8
TRLINE or SDLCLINE directive, 8-7
dependent LUs, 8-5
function in gateway communication, 8-1
independent and dependent LUs, 8-5
main components
SunLink SNA PU2.1 server and SunLink P2P
LU6.2 server, 8-4
provides support for Sun Solaris APPC, 8-1
starting the PU2.1 server, 8-10
SunLink SNA PU2.1 server, 8-4
setting up configuration file, 8-6
starting the server, 8-10
sunpu2.config file, 8-6
Sym Dest Name
symbolic destination name, 7-6
Symbolic Destination Name, 8-9
system identifier
choosing, 4-4
system requirements, 3-1

T

table
DBA_2PC_PENDING, 9-7
GLOBAL_TRAN_ID, 9-8
PGA_2PC_LUS, 9-3
columns, 9-8
maintained by RRM, 9-7
modified by logging server, 9-5
with two-phase commit, 9-4
PGA_2PC_PENDING, 9-3
columns, 9-8
created, 9-12
maintained by gateway server, 9-7
modified by logging server, 9-5
with two-phase commit, 9-4, 9-6
TCP protocol, 4-25
TCP/IP protocol adapter, 4-25
TERMD parameter, 8-7
terms, gateway terms defined, 2-4
testing the SNAplus2 connection, 7-10
TIP
Also called PL/SQL package
body output files, 4-22
considerations, with stand-alone
installation, 4-5
conversions, 2-3
converting PL/SQL datatypes to RAW, 2-10
definition, 2-5
developer access to PL/SQL packages, 4-20
developer authorization on gpglocal, 4-19
function in Oracle integrating server, 2-10
functions, 2-9
invalidated if package specifications
reinstalled, 4-18
override, 6-2, 7-2, 8-2
recompile
after reinstalling package specifications, 4-18
on upgrade from Release 4.0.1, 4-3
regenerate to upgrade function and
maintenance, 4-21
remote transaction initiation (PGAINIT), 2-10
specification output files, 4-22
specifications, 4-19, 4-20, 4-21
generated by PGAU, 2-9, C-3
PL/SQL v3 compatibility, C-3
use UTL_PG, 4-16
use UTL_RAW, 4-15
specifying LUs, 6-4, 7-5, 8-5	race access PL/SQL routines, 4-19
upgrade considerations from previous
versions, 4-3
TNS listener, 4-5, 4-24
TNS_ADMIN environment variables, 9-13
tnsnames.ora
and Oracle Net considerations, 10-6
tnsnames.ora file, 4-7, 4-17, 4-24, 4-25, 4-26, 4-40, 9-13, 9-14, A-11, A-13, A-14
Token Ring
connection defined by TRLINE or SDCLINE
directives, 8-7
parameters used for DLC directive, 8-7
TP name, 8-9, A-4, A-8
TP profile, 9-15
TP_NAME column
PGA_2PC_PENDING table, 9-10
TP_PROFILE, 9-15
trace access, 4-19
TRACE_DIRECTORY RRM parameter, 9-15, 9-16
TRACE_FILE RRM parameter, 9-16
TRACE_LEVEL parameter, 10-3, A-5, A-9
moved to initsid.ora, C-2
TRACE_LEVEL RRM parameter, 9-16
TRAN_COMMENT column
PGA_2PC_PENDING table, 9-9
trandef.doc file, 4-29

Index-18
Transaction Interface Package
See TIP
transaction program name profile
and two-phase commit, 6-9, 9-17
transaction types
one-shot, persistent and
multi-conversational, 2-11
transferring
initsid.ora gateway initialization file
parameters, 10-2
transparency
(application), 2-2
(location), 2-2
TRLINE or SDLCLINE directives, 8-7
two-phase commit, A-3
and Oracle logging server, 9-3
and PGA_CAPABILITY parameter, A-9
and Recovery Resource Manager, 9-3
application design requirements, 9-24
architecture, 9-2
components, 9-4
interactions, 9-5
configuring, 9-11
CICS/ESA, 9-2
Gateway, 9-12
Oracle logging server, 9-12
RRM initialization parameters, 9-14
the SNA server, 9-16
customizing rrmstart script, 9-18
flow of control, 9-6
function, 9-1
gateway server and RRM LU6.2 log tables, 9-7
implementing, 9-2
logging from CICS applications, 9-29
manual recovery of in-doubt transactions, 9-26
needs transaction program name profile, 6-9
not supported by IBM IMS/TM and
APPC/MVS, 9-2
periodic log cleanup, 9-33
PGALUWID subroutine, 9-31
post-configuration tasks, 9-18
pre-configuration tasks, 9-10
required components, 9-3
RRM recovery, 9-20
RRM startup and shutdown, 9-18
setting up local LUs, 6-7
setting up node profile, 6-6
supported by gateway server, 9-3
supported OLTPs, 9-2
supported with CICS/ESA, 9-2
the four main components, 9-2
with CICS/ESA, 9-23
two-phase commit flow, step by step, 9-6

U
uname.rrm file, 9-18
upgrading
considerations, 4-3
existing TIP specifications, 4-21
the gateway
renaming links or directories, 4-14
user ID
as security authentication, 5-3
user ID mapping
OLTP, 5-4
USING parameter, 4-24
UTL_PG, 4-20
installing, 4-16
package, 4-20, 4-38
definition, 2-6
invalidated or deinstalled, 4-18
UTL_RAW, 4-15, 4-20
interface
PL/SQL package, 4-20
package, 4-38
invalidated or deinstalled, 4-18
PL/SQL package
definition, 2-6
utlpg.sql
file, 4-18
script, 4-16, 4-18
utlraw.sql
file, 4-18
script, 4-15, 4-18

V
VTAM, 4-30, 4-31
and Side Information Profiles, 7-4
configuring for connection to the gateway, 4-27, 4-28
setting partner LU, 6-8
VTAM logmode table, 4-27, 4-28, 4-30, 4-31

X

xsnapadmin, 7-7
invoking, 7-7
to configure SNAplus2, 7-4, 7-7
to create SNA definitions, 7-6
X-Windows
see xsnapadmin, 7-4