Pulse 8.1

Deployment Guide

About Genesys

Genesys is the world's leading provider of customer service and contact center software—with more than 4,000 customers in 80 countries. Drawing on its more than 20 years of customer service innovation and experience, Genesys is uniquely positioned to help companies bring their people, insights and customer channels together to effectively drive today's customer conversation. Genesys software directs more than 100 million interactions every day, maximizing the value of customer engagement and differentiating the experience by driving personalization and multi-channel customer service—and extending customer service across the enterprise to optimize processes and the performance of customer-facing employees. Go to www.genesys.com for more information.

Each product has its own documentation for online viewing at the Genesys Documentation website or on the Documentation Library DVD, which is available from Genesys upon request. For more information, contact your sales representative.

Notice

Although reasonable effort is made to ensure that the information in this document is complete and accurate at the time of release, Genesys Telecommunications Laboratories, Inc., cannot assume responsibility for any existing errors. Changes and/or corrections to the information contained in this document may be incorporated in future versions.

Your Responsibility for Your System's Security

You are responsible for the security of your system. Product administration to prevent unauthorized use is your responsibility. Your system administrator should read all documents provided with this product to fully understand the features available that reduce your risk of incurring charges for unlicensed use of Genesys products.

Trademarks

Genesys and the Genesys logo are registered trademarks of Genesys Telecommunications Laboratories, Inc. All other company names and logos may be trademarks or registered trademarks of their respective holders.

The Crystal monospace font is used by permission of Software Renovation Corporation, www.SoftwareRenovation.com.

Technical Support from VARs

If you have purchased support from a value-added reseller (VAR), please contact the VAR for technical support.

Customer Care from Genesys

If you have purchased support directly from Genesys, please contact **Genesys Customer Care**. Before contacting Customer Care, please refer to the ***Genesys Care Program Guide*** for complete contact information and procedures.

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the ***Genesys Licensing Guide.***

Released by

Genesys Telecommunications Laboratories, Inc. www.genesys.com

Document Version: 81pulse_dep_04-2014_8.1.403.00

Table of Contents

List of Procedures

Preface

Welcome to the *Pulse 8.1 Deployment Guide*. This document introduces you to the concepts, terminology, and deployment procedures relevant to Pulse.

This document is valid only for the 8.1.*x* release of this product.

This preface contains the following sections:

For information about related resources and about the conventions that are used in this document, see the supplementary material beginning on page 67.

About Pulse

Pulse is a Genesys Administrator Extension (GAX) plug-in application that enables at-a-glance views of real-time contact center statistics within the GAX graphical user interface. Pulse uses widgets to display user-defined Donut, Key Performance Indicator (KPI), or List charts of statistics for objects.

Using Pulse you can:

- Create report views from predefined and user-defined templates for a fast and easy text or graphical presentation of selected or user-defined object statistics.
- Monitor the current state and activity of contact center objects to help make decisions about staffing, scheduling and call routing strategies.

Intended Audience

This document is primarily intended for system integrators, system administrators, contact center managers, and operations personnel. It has been written with the assumption that you have a basic understanding of:

- Computer-telephony integration concepts, processes, terminology, and applications
- Network design and operation
- Your own network configurations

You should also be familiar with:

- The User Interaction Layer of the Genesys Framework architecture
- The GAX user interface and its functions

Making Comments on This Document

If you especially like or dislike anything about this document, feel free to e-mail your comments to Techpubs.webadmin@genesys.com.

You can comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this document. Please limit your comments to the scope of this document only and to the way in which the information is presented. Contact your Genesys Account Representative or Genesys Customer Care if you have suggestions about the product itself.

When you send us comments, you grant Genesys a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Contacting Genesys Customer Care

f you have purchased support directly from Genesys, please contact Genesys Customer Care.

Before contacting Customer Care, please refer to the *Genesys Care Program Guide* for complete contact information and procedures.

Document Change History

This section lists content that is new or that has changed significantly since the first release of this document. The most recent changes appear first.

New in Document Version 8.1.403.00

- The order of steps to create a new Database Access Point was corrected in Procedure: Deploying Pulse Collector, on page 20.
- The `access_url`, `backend-file-ext`, `generate-layout-info-file`, and `heartbeat-period` configuration options were updated in "Pulse Configuration Options", beginning on page 27.
- The list of "Application Files", beginning on page 63 were updated.

New in Document Version 8.1.402.00

- EZPulse was renamed Pulse.
- Software releases for Stat Server and DB Server were updated in Chapter 2 on page 17.
- The following changes to configuration options were made in Chapter 3, "Pulse Configuration Options," beginning on page 27.
  - The new `metagroup-contents-recheck-delay` configuration option in the [`configuration-monitoring`] was added.
  - The valid values were changed to `3-3600` for the `heartbeat-period` option in the [`heartbeat`] section from the Collector application.
  - The `heartbeat_timeout` configuration option in the [`pulse`] section was removed from the Pulse application template, because Pulse now uses the `heartbeat-period` option in the [`heartbeat`] section from the Collector application.
  - The default value was changed to `15` for the `cleanup-file-modification-age-threshold` configuration option in the [`transport-*`] section.
  - The `collector.js` value was added to the default values for the `js-modules` configuration option in [`scripting`] section.
  - The `ezpulse-file-ext` configuration option in the [`transport-gpb-out`] section was renamed `backend-file-ext`.
- Many files named `EZPulse_Collector` were renamed `Collector`.
- The warning to restrict the number of agents in a widget to no more than 100 agents was moved to the online *Pulse Help*.

New in Document Version 8.1.401.00

- Support for the following items have been added or updated:
  - Relational database platforms: Microsoft SQL 2008, and PostgreSQL 9
  - Operating systems: Red Hat Enterprise AS 6 (64-bit system)
  - Web browsers: Google Chrome version 22 or higher, Mozilla Firefox version 17 or higher, Microsoft Internet Explorer version 10, and Apple Safari version 6

- "Product Overview", beginning on page 11 has been updated.
- Instruction to import the `pulse_statistics.cfg` file has been added to "Deploying Pulse Collector", beginning on page 20.
- Instruction to update the Pulse Collector application object settings on the Tenants tab have been added to "Deploying Pulse Collector", beginning on page 20.
- Privileges have been added to "Configuring objects for Pulse", beginning on page 22.
- "Pulse Configuration Options", beginning on page 27 has been revised.

New in Document Version 8.1.301.00 (restricted release)

- Chapters have been added to this document to improve documentation quality:
  - "Product Overview", beginning on page 11.
  - "Starting and Stopping Pulse", beginning on page 55.
  - "Application Files", beginning on page 63.
- The prerequisites have been updated to include the minimum release of Genesys components with which Pulse interoperates and a detailed listing of requisite third-party software. See pages 17–19.
- The `postgre_71_compatible` configuration option is introduced in this release for configuration of the Database Access Point serving the Pulse database. See page 23.
- The following additional options are introduced in this release to help you fine-tune Pulse Collector configuration:
  - `data-flow-check-interval`, see page 50.
  - `data-flow-timeout`, see page 51.
  - `max-output-interval`, see page 49.
  - `min-output-interval`, see page 49.
- The default value of the `output-transport` configuration option has changed from `gpb` to `gpb-out`. See page 29.
- PostgreSQL support is added in this release. Configuration and deployment instructions for this RDBMS is included throughout this document.

Other Changes

Other product changes, such as the enhanced Pulse user interface, are described in the online *Pulse Help,* the *Pulse 8.1 Release Notes,* and the *Pulse 8.1 Release Advisory.*

Chapter

1 Product Overview

Pulse is a widget-driven, graphical user application, which is accessible from a web browser as a Genesys Administrator Extension (GAX) plug-in application. Using a direct communication link to a real-time metrics engine, Stat Server, Pulse enables at-a-glance views of real-time contact center statistics within the GAX user interface.

Pulse uses widgets to display user-defined Donut, Key Performance Indicator (KPI), or List charts of statistics for objects. You can also view and select additional report details and options by maximizing any widget.

See the online *Pulse Online Help* for a details about the Pulse interface.

The following sections describe the architecture that enables this functionality:

Pulse Components

Major aspects of Pulse are shown in Figure 1.

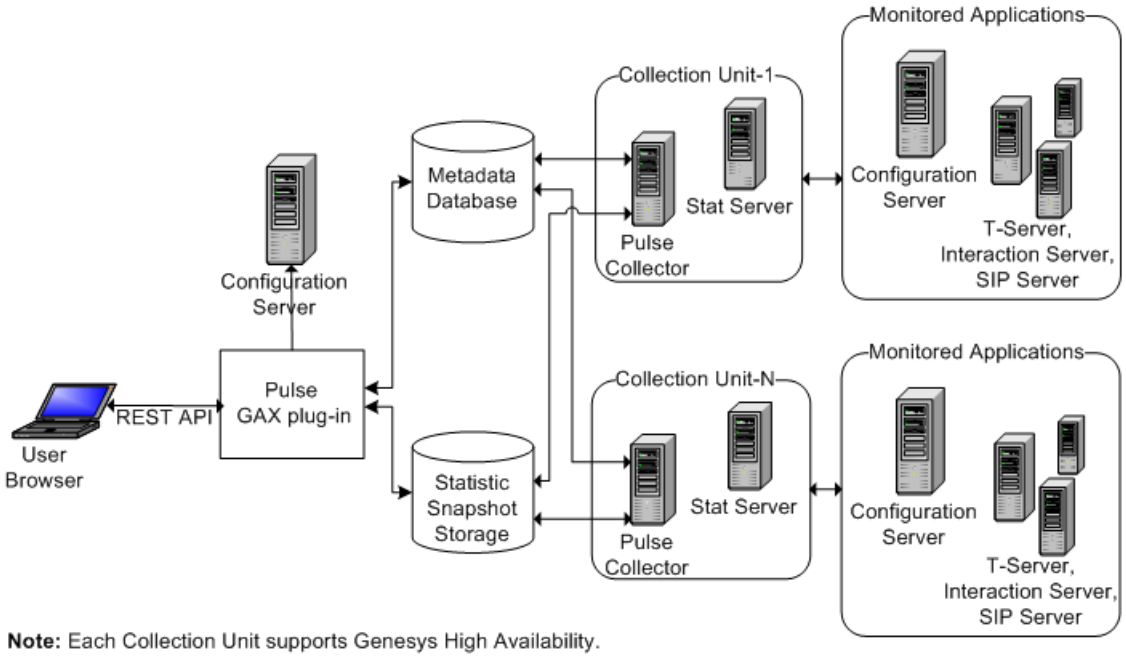

Figure 1: Pulse Architecture

Pulse Collector

Pulse Collector is a background near-realtime statistical data collection and processing engine. It performs the following activities:

- Reads the metadata from the Pulse database upon startup and whenever changes are made to report definitions in Pulse.
- Uses the report definitions stored in the Pulse database to determine which statistics and objects to include.
- Creates snapshots with current data from Stat Server and formula-based statistics calculated by Pulse Collector, on the specified file system for reference by Pulse.
- Maintains a constant connection with Configuration Server to retreive changes, additions, and deletions to configuration objects.

Pulse

Pulse is a GAX plug-in that performs the following activities:

- Handles user authentication and permissions validation.
- Filters and delivers report data according to the permissions and tenancy of the user who is requesting the data.

- Displays report content in widgets, such as the listing and content of reports.
- Saves the report definitions to the Pulse Database, which it shares with Pulse Collector.

You can have only one Pulse application for each GAX instance. For configuration, Pulse requires a `[pulse]` section and is added to the connections of the GAX `Application` object. Pulse might have its own database access point, but otherwise uses Management Framework and other connections from GAX.

Pulse uses the existing GAX client architecture, which is described in the *Genesys Administrator Extension Deployment Guide.*

Refer to *Pulse Online Help* to learn how to operate this user interface. This document is accessible only from within the software, when you click `help`.

Genesys Framework Components

Pulse interacts with several products within the Genesys Framework to provide real-time snapshots of contact center data, including:

- Configuration Server
- Stat Server
- DB Server

Configuration Server

Pulse Collector receives the following data from Configuration Server:

- Information about the existence of contact center objects (for example, tenants, agents, places, calling lists, or campaigns)
- Statistical parameters (for example, time ranges, time profiles, filters, and statistical types)
- Information about changes to contact center objects (for example, a deleted agent, a renamed queue, or a new Agent Group).

Pulse Collector uses this data to provide content for Pulse.

Both Pulse and Pulse Collector connect to Configuration Server in order to retrieve their own configuration.

Stat Server

Stat Server tracks information about customer interaction networks consisting of one or more contact centers in conjunction with one or more computer networks. Stat Server receives information from one or more T-Server or SIP Server applications and converts the accumulated data for directory numbers, agents, agent groups, and so on, into statistical information.

As a client of Stat Server, Pulse Collector requests statistics for objects belonging to particular reports. Stat Server supplies both information about the interactions that the objects that handle as well as noninteraction-related data about the objects themselves (for example, agent status). Pulse Collector uses only those statistical types defined in the Stat Server application that is listed among Collector's connections.

Refer to the *Framework Stat Server User's Guide* and *Framework Stat Server Deployment Guide* for information about Stat Server.

DB Server

DB Server is the Genesys component that handles database requests from multiple client processes. DB Server provides a single interface from the clients to a variety of database engines, including Microsoft SQL Server, Oracle and PostgreSQL. As a client of DB Server, Pulse Collector reads information about active widgets and updates the layout statuses, when layout status changes occur within the Pulse Collector.

Refer to the *Framework DB Server User's Guide* for information about DB Server.

New in this Release

New in Release 8.1.4

Some of the primary new features in release 8.1.4 are:

- Genesys EZPulse is renamed to Pulse.
- Pulse displays real-time data that is collected by Stat Server through widgets.
- Pulse provides additional predefined templates.
- Users can print charts and grids.
- With appropriate privileges, users can:
  - Create or edit Widget Templates to simplify widget creation and selecting widget content.
  - Customize statistic formulas.
  - Modify statistics.
  - Add previously removed statistics back into a widget.
- Pulse integrates with GAX 8.1.4.
- Pulse supports Red Hat Linux 6 (64-bit).
- Pulse supports Microsoft SQL Server 2008.
- Pulse supports Microsoft Internet Explorer 10.

- Pulse supports Statistics of categories `CurrentTargetState`, `CurrentStateReasons`, and `JavaCategory`.
- Pulse supports virtual agent groups.

New in Release 8.1.3

Some of the primary new features in release 8.1.3 are:

- Integration with GAX 8.1.3, which offers many user interface improvements including the ability to collapse panels
- Adapted to work with PostgreSQL
- Several panel and grid functionality improvements, including thumbnail report previews within the list of layouts
- Additional chart design options, including the ability to display multiple statistics within a single chart.

Chapter

2 Pulse Deployment

This chapter describes how to install and configure Pulse. It contains the following sections:

Prerequisites

You deploy Pulse as a Genesys Administrator Extension (GAX) plug-in on a web application server to be accessed using a web browser through the GAX user interface. No additional Genesys software is needed on the client machine.

The following are prerequisites for Pulse deployment:

- Genesys Administrator Extension release 8.1.4 or higher must be configured and installed. Refer to the *Framework 8.1 Genesys Administrator Extension Deployment Guide* for details.
- GAX 8.1.4 uses an embedded Jetty instance as the web application server; as a result, Tomcat is no longer a prerequisite to use GAX. For those who choose to use Tomcat instead of Jetty, refer to the *Genesys Administrator Extension 8.1 Deployment Guide* for additional information.
- Genesys DB Server must be release 8.1.300.05 or higher. Refer to the *Framework 8.0 DB Server User's Guide* for details.
- Your Relational Database Management System (RDBMS) must be up and running. The 8.1.4 release of Pulse supports the following relational database platforms:
  - Microsoft SQL Server 2008
  - Oracle 11g
  - PostgreSQL 9.0

- The computer on which you install Pulse must be running the following:
  - Windows Server 2008 R2 (64-bit system)

  Or,
  - Red Hat Enterprise Linux AS 5.5 or 6.4 (64-bit system), with `Updates from RHN` enabled

Note: If you use Red Hat Enterprise Linux AS 6, you must also install the following libraries to ensure backward compatibility for applications:

- compat-db
- compat-expat1
- compat-glibc
- compat-libf2c-34
- compat-libgcc-296
- compat-libgfortran-41
- compat-libstdc++-295
- compat-libstdc++-296
- compat-libstdc++-33
- compat-libtermcap
- compat-openldap
- openssl098e

For more information, see http://www.linuxtopia.org/online_books/rhel6/rhel_6_migration_guide/rhel_6_migration_ch01s02.html

- Although not required for deployment, you must have Stat Server release 8.1.000.45 or higher installed for basic operation.
- Although not required for deployment, you must have release 8.1.200.08 or higher for DB Server 8.1.2 or release 8.1.300.06 for DB Server 8.1.3 installed for basic operation.

Deployment Task Summary

The Task Summary: Deploying Pulse table below lists the objectives for deploying Pulse.

Task Summary: Deploying Pulse

Objective	Related procedures and actions
1. Ensure your system meets the deployment prerequisites.	See "Prerequisites" on page 17.
2. Prepare the Pulse database.	**Microsoft SQL Server** 1. Create a new empty Microsoft SQL Server database. 2. Create a new Microsoft SQL Server user account. 3. Set the new database as default database for the user. 4. Grant the new user sufficient privileges for the new database. User must be able to create database objects and select, insert, update, and delete data in tables. **Oracle** Create a new user/schema to be used by Pulse. The user must have `RESOURCE` and `CREATE VIEW` privileges. **PostgreSQL** 1. Create a new empty PostgreSQL database. 2. Create a new PostgreSQL user account. 3. Set the new database as default database for the user. 4. Grant the new user sufficient privileges for the new database. User must be able to create database objects and select, insert, update, and delete data in tables.
3. Deploy Pulse Collector.	Complete the following procedure: • "Deploying Pulse Collector" on page 20
4. Create applications within Genesys Administrator and configure them.	Complete the following procedure: • "Configuring objects for Pulse" on page 22
5. Deploy Pulse.	Complete the following procedure: • "Deploying the Pulse Web Application" on page 24

Deployment Procedures

This section provides detailed procedures for configuring and deploying Pulse.

Note: In these procedures, use the instructions provided in *Genesys Administrator Help* or the *Framework 8.x Deployment Guide*, and add the object-specific configuration requirements listed here.

Procedure: Deploying Pulse Collector

Purpose: To deploy Pulse Collector, which connects directly to Stat Server to collect statistics for contact center objects. Pulse Collector also connects to DB Server, through which it accesses the database where reporting layouts are stored.

Prerequisites

- For Windows only: Ensure you have installed Microsoft Visual C++ 2005 SP1 MFC Security Pack Redistributable.
- Ensure your DB Server version is 8.1.300.06 or higher.
- Ensure your Stat Server version is 8.1.000.45 or higher.
- Ensure that the `statserver` application object for both the primary and backup Stat Server have the option `accept-clients-in-backup-mode` and set its value to yes.

  Note: This option is required even if there is no backup application specified for the Stat Server application.

Start of procedure

1. In Genesys Administrator, import the application template for Pulse Collector by uploading the file `collector.apd` from the installation package.
2. Create an Pulse Collector `Application` object using the application template that you imported in Step 1.

   Note: Pulse Collector applications have an application type of `Data Sourcer` within Genesys Administrator.

3. Depending on the platform, do the following:

   Windows: In the Pulse Collector installation package, locate and run `setup.exe`.

   UNIX: In the Pulse Collector installation package, locate the shell script called `install.sh` and run this script from the command prompt.

   For example:

   `sh ./install.sh` or
   `./install.sh`

4. Follow the InstallShield Wizard prompts to install Pulse Collector.
5. In Genesys Administrator, open the Pulse Collector `Application` object and add or modify the values of configuration options described in "Pulse Collector Application Object" on page 29.
6. For a high-availability (HA) deployment, repeat Steps 2–5 to install a backup instance of Pulse Collector.
7. In Genesys Administrator, add the Pulse Collector `Application` object to the connections of your GAX server `Application` object. For a load-balanced environment configuration, go to Step 8.
8. In a load-balanced environment with two GAX applications and Pulse plug-ins, associate one GAX with a primary Pulse Collector and the second GAX with a backup Pulse Collector as follows:
   a. Add the primary Pulse Collector `Application` object to the connections of the first (or primary) GAX server `Application` object.
   b. Add the backup Pulse Collector `Application` object to the connections of the second (or backup) GAX server `Application` object.
   c. Add the backup Pulse Collector `Application` object to the `Server Info` tab of the primary Pulse Collector `Application` object as a backup server.

   Note: One GAX pair can be connected to multiple Pulse Collector HA pairs. In this case, you must associate all primary Pulse Collectors with one GAX, and all the backup Pulse Collectors with the other GAX.

9. Create a new Database Access Point, which is necessary for connectivity to the Pulse database through the DB Server:
   a. In the `DB Info` frame of the `Configuration` tab, specify the connectivity parameters to your RDBMS and select the `Default` connection type.
10. Add the `Database Access Point` object to be used by Pulse Collector to its connections in the Pulse Collector `Application` object. (See "Prerequisites" on page 20).
11. In the connections of the Pulse Collector `Application`, add the primary Stat Server application that is to be used by Pulse Collector (see page 18).

12. Add the Tenant objects to the Tenants tab for all Pulse Collectors that you plan to monitor in Pulse.

End of procedure

Procedure: Configuring objects for Pulse

Purpose: To configure the necessary objects required by Pulse using Genesys Administrator.

Note: In this procedure, use the instructions provided in *Genesys Administrator Help* or the *Framework 8.x Deployment Guide*, and add the object-specific configuration requirements listed here.

Start of procedure

1. Import the application template for Pulse:
   a. Upload the `Pulse.apd` file from the installation package.
   b. Import the metadata from the `Pulse.xml` file.
   c. Click `Save & Close` to save the new object.

   The Pulse `Application Template` is used to import role privileges.
2. Configure permissions for the GAX `Application` object:
   a. Open the GAX `Application` object.
   b. Click the `Permissions` tab.
   c. Select the `SYSTEM` account.
   d. Select `Read`, `Change`, and `Execute` permissions.
   e. Click `Save & Close` to save the object.
3. Optional: Configure options for Pulse in the GAX `Application` object:

   Note: During startup, GAX looks for all options that are required for Pulse operation and adds them to the GAX `Application` object if they are not explicitly configured. If a required option is either not configured or specifies an invalid option value, GAX uses the option's default value.

   a. Open the GAX `Application` object.
   b. Under the `Options` tab, create the `pulse` section options as described on page 28.

4. Create a new `Role` object to provide access to GAX functionality in for Pulse:
   a. Define the privileges granted by the Role on the `Role Privileges` tab in the `Genesys Administrator Extension - Pulse` section. Each subsequent user privilege requires granting the previous user privileges in the list along with all corresponding privileges, as follows:
      i. Pulse Read Layout—This role requires the following GAX permissions:
         — Stay on Dashboard Indefinitely
         — Move Widget
         — Edit Dashboards
      ii. Pulse Update Layout—This role requires the following GAX permissions:
         — Edit Widget
      iii. Pulse Write Layout—This role requires the following GAX permissions:
         — Add Widget
         — Clone Widget
         — Remove Widget

         This role must also have write access to their `person` object.
      iv. Pulse Write Template—This role requires the following GAX permissions:
         — Edit Default Dashboard

         This role also requires write access to their `tenant` object.
   b. Assign the `Role` to Users and Access Groups in the `Members` section as required.
5. Create a new Database Access Point, which is necessary for connectivity to the Pulse database:
   a. In the `DB Info` frame of the `Configuration` tab, specify the connectivity parameters to your RDBMS and select the `JDBC` connection type and enter the following field values:
      - Role = Main
      - Debug = false
      - JDBC Query Timeout: 15
      - Case Conversion: any
   b. In the `Server info` frame of the `Configuration` tab, specify the RDBMS host and the connection port.
   c. Under the `Options` tab, add the `role` configuration option with its value set to `pulse` in the `GAX` section to identify this Database Access Point as the database schema created for Pulse.
   d. For a PostgreSQL database, add the `postgre_71_compatible` configuration option with its value set to `false` in the `GAX` section.

Refer to the *Framework 8.0 DB Server User's Guide* for additional information about configuring this application.

6. Add the Database Access Point to the connections of your GAX server `Application` object.
7. For a high-availability deployment, complete Steps 2–3, 5, and 6, for each GAX server `Application` object to be used with the Pulse plug-in.

End of procedure

Procedure: Deploying the Pulse Web Application

Purpose: To deploy the Pulse plug-in.

Prerequisites

- This procedure must be performed on the computer where the Genesys Administrator Extension is installed.

Start of procedure

1. Depending on your platform, do the following:
   - Microsoft Windows: in the Pulse installation package, locate and run `setup.exe`.
   - UNIX: in the Pulse installation package, locate the shell script called `install.sh` and run this script from the command prompt by typing the `sh` command and the file name.

     For example:

     `sh ./install.sh` or
     `./install.sh`
2. Follow the InstallShield Wizard prompts to install Pulse. Refer to the *Genesys Administrator Extension 8.1 Deployment Guide* for additional information.

   Note: When prompted to specify a path to the installation folder, specify any temporary folder that is used for storing unzipped files. It must not be a GAX installation folder.

3. Within both primary and backup Stat Server applications, import the `pulse_statistics.cfg` file from the Pulse installation files to create the statistics that Pulse should monitor.

   Refer to the *Framework 8.1 Stat Server User's Guide* for guidance.

4. From the user account created in Step 2 on page 19, execute the SQL statements in the appropriate initialization script deployed during installation—either:
   - `pulse_init_mssql.sql`
   - `pulse_init_oracle.sql`
   - `pulse_init_postgres.sql`

End of procedure

Genesys

Chapter

3 Pulse Configuration Options

This chapter describes the options to configure Pulse and Pulse Collector. The configuration options are presented in alphabetical order. This chapter contains the following sections:

Notes:
- The application templates for Pulse might contain other configuration options that are not described in this chapter. These options must remain set to the default values that are provided in the application templates. Changing these values might cause unexpected behavior.
- You are not required to configure any options to start Pulse. Pulse supplies default values for all options that it requires to function.

GAX Application Object

Set the configuration options in the corresponding sections on the Options tab of the GAX `Application` object.

[pulse] Section

access_url

Valid Value: `http://<gax_host>:<gax_port>`
where:
`gax_host`—The name or IP address of the GAX host computer
`gax_port`—The port of the GAX server
For example: `http://pulsedev01:8280/gax`
Default Value: `http://host:port`
Changes Take Effect: After GAX restart

Specifies the URL of the Genesys Administrator Extension (GAX) in which Pulse has been configured.

health_expire_timeout

Valid Values: zero or any positive number
Default Value: 30
Changes Take Effect: After GAX restart

Specifies how long, in seconds, Pulse stores result of previous health check, which includes the heartbeat, DB connection, and Configuration Server connection.

pause_timeout

Valid Values: Any valid integer
Default Value: `0`
Changes Take Effect: After GAX restart

Specifies the time interval, in minutes, that Pulse waits before placing a layout in the `Paused` state (Pulse Collector does not collect data for paused layouts), which occurs if no statistical data pertaining to a layout is requested while the layout remains active. A value of zero (`0`) disables pausing altogether.

Note: This option is not applied if GAX is configured in High Availability.

Pulse Collector Application Object

Set the configuration options in the corresponding sections on the `Options` tab of the Pulse Collector `Application` object.

[collector] Section

application-id

Valid Values: `0-65535`
Default value: `0`

Note: A zero (0) value instructs Pulse Collector to attempt to determine the application ID automatically.

The Pulse Collector application ID, as registered in the Pulse database.

hostname

Valid Values: Valid host name
Default Value: Empty value
Changes Take Effect: Upon restart

Specifies the Simple Network Management Protocol (SNMP) host name.

management-port

Valid Values: Positive integers
Default Value: No default value

Warning! No other application should use this port.

Changes Take Effect: Upon restart

Specifies the TCP/IP port that Pulse Collector reserves for SNMP Option Management Client connections. If this option is absent or null, a server for Management Client is not created.

Warning! You must specify a value for this option if you are using an SNMP connection. Do not change the value for this option while Pulse Collector is running.

output-transport

Default Value: No default value (but the application template supplies the preconfigured transport name of `gpb-out`)
Valid Values: the names of all transport protocols that you have defined in the `transport-*` section.

Changes Take Effect: After restart

Specifies the output transport to be used to produce outputs.

[configuration-monitoring] Section

auto-detect-dynamic-metagroups

Valid Values: `yes,no`
Default Value: `yes`
Changes Take Effect: After restart

Determines whether to force Pulse Collector to attempt detecting dynamic metagroup automatically, if layout definitions do not explicitly mark metagroup as `dynamic` or `non-dynamic`. If option is set to `no`, Pulse Collector relies on the dynamic metagroup mark in the layout definition for the purpose of determining dynamic metagroups.

check-layout-presence-timeout

Valid Values: `0-3600`
Default Value: `900`
Changes Take Effect: After restart

Specifies how often, in seconds, Pulse Collector checks for the deleted layouts. A zero (0) value completely disables the check.

Note: This defines the minimum timeout between two checks. The actual timeout depends on the database polling cycle, because the check is conducted after finishing subsequent database polling cycle.

db-poll-period

Valid Values: `3-3600`
Default Value: `30`
Changes Take Effect: After restart

Specifies how often, in seconds, Pulse Collector obtains updates from the Pulse database.

Note: Genesys recommends that you set this option to no less than 15 seconds.

enable-differential-layout-recheck

Valid Values: `yes,no`
Default Value: `yes`
Changes Take Effect: After restart

Determines if the differential or full layout definition recheck procedure is used.

metagroup-contents-recheck-delay

Valid Values: `0...3600`
Default Value: `60`
Changes Take Effect: Immediately

Specifies the delay in seconds between when Pulse Collector verifies metagroup contents (such as Agent Group, Place Group, and DN Group) after notification from Configuration Server notifies Pulse Collector about changes in the contents of the metagroup object. Zero value of this option eliminates timeout and metagroup change is processed immediately.

Note: This configuration option impacts ANY changes in the metagroup (for example, both adding and deleting objects), so new objects added to the metagroup appear in the layout after the specified delay.

new-object-delay

Valid Values: `0-86400`
Default Value: `0`
Changes Take Effect: Immediately

Specifies the delay, in seconds, between when Pulse Collector receives notification of a new object in Configuration Server, and when it starts to process this notification. Setting this option to `0` enables Pulse Collector to process new objects without delays.

ods-wait-timeout

Valid Values: `10–3600`
Default Value: `300`
Changes Take Effect: After restart

Specifies the time, in seconds, that Pulse Collector waits before re-checking the Pulse database for proper initialization.

recheck-active-layout-definition

Valid Values: `yes,no`
Default value: `yes`
Changes Take Effect: After restart

Determines whether Pulse Collector conducts configuration recheck process for the layout, when that layout definition is changed.

strict-tenant-security

Valid Values: `yes,no`
Default value: `no`
Changes Take Effect: After restart

Determines whether Pulse Collector prevents individual objects belonging to different tenants into the layout or objects from different tenants included in the group.

[heartbeat] Section

heartbeat-folder

Valid Values: Valid folder path
Default Value: `./output/heartbeat` (as provided by template file)
Changes Take Effect: After restart

Specifies the path in which Pulse Collector writes the heartbeat file.

heartbeat-period

Valid Values: `3-3600`
Default Value: `30`
Changes Take Effect: After restart

Specifies the period of a heartbeat update, in seconds.

heartbeat-success-condition

Valid Values:
One or any combination of the following conditions:

- `statserver`—Stat Server connection should be available.
- `snapshot-writer`—The snapshot writer should be producing outputs without error.
- `collector-db`—The main DB connection should be available.

Default Value: `statserver, snapshot-writer`
Changes Take Effect: After restart

Specifies a comma-separated list of conditions to be checked for criteria of heartbeat success, which means the heartbeat is updated only if this criteria is met.

[layout-validation] Section

enable-layout-validation

Valid Values: `yes,no`
Default Value: `yes`
Changes Take Effect: After restart

Enables or disables layout validation in Pulse Collector.

Note: Pulse Collector always checks for consistency of `LayoutID` and `TenantID`. These validations apply even if options are set to a value `no`.

enable-strict-tenant-security-validation-rule

Valid Values: `yes,no`
Default Value: `yes`
Changes Take Effect: After restart

Enables Pulse Collector to fail the layout validation when the `option strict-tenant-security` in the `configuration-monitoring` section is set to `yes` and Pulse encounters an individual object that belongs to a tenant that is different than layout's tenant.

[limits] Section

max-formulas-per-layout

Valid Values: `1-100000`
Default Value: `50`
Changes Take Effect: After restart

Specifies the maximum number of formula-based statistics for each layout.

max-metagroups-per-layout

Valid Values: `1-100000`
Default Value: `50`
Changes Take Effect: After restart

Specifies the maximum number of metagroups for each layout.

max-objects-per-layout

Valid Values: `1-100000`
Default Value: `400`
Changes Take Effect: After restart

Specifies the maximum number of objects for each layout.

max-statistics-per-layout

Valid Values: `1-100000`
Default Value: `100`
Changes Take Effect: After restart

Specifies the maximum number of statistics for each layout.

[localization] Section

default-snapshot-message-language-code

Valid Values: Language code (non-empty string, typically 2 letters)
Default Value: `en`
Changes Take Effect: After restart

Specifies the default language code to use for layout snapshot messages, if the layout snapshot does not override this value.

language-pack-folder

Valid Values: Folder path
Default Value: `./messages`
Changes Take Effect: After restart

Specifies the folder from which Pulse Collector reads localized message files to show layout errors in the localized form.

[Log] Section

all

Valid Values:

- `stdout`—Log events are sent to the Standard output (`stdout`).
- `stderr`—Log events are sent to the Standard error output (`stderr`).
- `network`—Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores log events in the Log Database.

  Setting the all log-level option to network enables Data Sourcer to send log events of Standard, Interaction, and Trace levels to Message Server. Log events of Debug level are neither sent to Message Server nor stored in the Log Database.
- `memory`—Log events are sent to the memory output on the local disk. This output is the safest in terms of the application performance.
- `[filename]`—Log events are stored in a file with the specified name. If you do not specify a path, the file is created in the application's working directory.

Default Value: `stdout`

Changes Take Effect: Immediately

Specifies the outputs to which Data Sourcer sends all log events. You must separate log-output types with commas when you configure more than one output type. For example, `all = stdout, logfile`

Note: To ease the troubleshooting process, consider using unique names for log files that different applications generate.

buffering

Default Value: `false`

Valid Values: `true,false`

Changes Take Effect: Immediately

Specifies whether the operating system file buffering is on or off. This option applies only to `stderr` and `stdout` output. Setting this option to true increases output performance.

Note: When you enable buffering, log messages might appear in the log after a delay.

segment

Valid Values:

- `false`—No segmentation allowed.
- `<number> KB` or `<number>`—Sets the maximum segment size in kilobytes. The minimum segment size is 100 KB.
- `<number> MB`—Sets the maximum segment size, in megabytes.
- `<number> hr`—Sets the number of hours for which the segment stays open. The minimum number is 1 hour.

Default Value: `10 MB`

Changes Take Effect: Immediately

Specifies if there is a segmentation limit for a log file. If there is, this option sets the unit of measurement along with the maximum size. If the current log segment exceeds the size set by this option, the current file is closed and a new file is created.

verbose

Valid Values:

- `all`—All log events (that is, log events of Standard, Trace, Interaction, and Debug levels) are generated if you set the debug-level option in the statserver section to all.
- `debug`—The same as all.
- `trace`—Log events of the Trace and higher levels (that is, log events of Standard, Interaction, and Trace levels) are generated, while log events of the Debug level are not generated.
- `interaction`—Log events of the Interaction and higher levels (that is, log events of Standard and Interaction levels) are generated, while log events of the Trace and Debug levels are not generated.
- `standard`—Log events of the Standard level are generated, while log events of the Interaction, Trace, and Debug levels are not generated.
- `none`—Produces no output.

Default Value: `all`

Changes Take Effect: Immediately

Determines whether a log output is created. If it is, this option specifies the minimum level of log events that are generated. The log-event levels, starting with the highest-priority level, are Standard, Interaction, Trace, and Debug.

Refer to the Framework Deployment Guide or Framework Solution Control Interface Help for more information on the Standard, Trace, Interaction, and Debug log levels

[object-name-format] Section

You can use a custom format for an object name, which can include a mix of predefined text and additions to the object properties within their actual values with optional width and trimming rules.

For details, see:

- "Valid object name format string" on page 45
- "Object Information" on page 47

AccessResource

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Access Resource`. For details, see "Object Properties" on page 46.

ACDPosition

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `ACD Position`. For details, see "Object Properties" on page 46.

ACDQueue

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for the object of the of type DN with subtype `ACD Queue`. For details, see "Object Properties" on page 46.

Agent

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for the object of the of type `Agent`. For details, see "Object Properties" on page 46.

AgentGroup

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for the object of the of type `Agent Group`. For details, see "Object Properties" on page 46.

CallingList

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Calling List`. For details, see "Object Properties" on page 46.

Campaign

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Campaign`. For details, see "Object Properties" on page 46.

CampaignCallingList

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Campaign Calling List`. For details, see "Object Properties" on page 46.

CampaignGroup

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Campaign Group`. For details, see "Object Properties" on page 46.

Cellular

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Cellular`. For details, see "Object Properties" on page 46.

Chat

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Chat`. For details, see "Object Properties" on page 46.

CoBrowse

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `CoBrowse`. For details, see "Object Properties" on page 46.

CommDN

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Comm DN`. For details, see "Object Properties" on page 46.

CP

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `CP`. For details, see "Object Properties" on page 46.

Data

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Data`. For details, see "Object Properties" on page 46.

DestinationLabel

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Destination Label`. For details, see "Object Properties" on page 46.

DN

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN`. For details, see "Object Properties" on page 46.

DNGroup

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN Group`. For details, see "Object Properties" on page 46.

EAPort

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `EA Port`. For details, see "Object Properties" on page 46.

Email

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Email`. For details, see "Object Properties" on page 46.

Extension

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Extension`. For details, see "Object Properties" on page 46.

ExtRoutingPoint

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Ext Routing Point`. For details, see "Object Properties" on page 46.

FAX

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Fax`. For details, see "Object Properties" on page 46.

GVPDID

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `GVPDID`. For details, see "Object Properties" on page 46.

Mixed

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Mixed`. For details, see "Object Properties" on page 46.

Music

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Music`. For details, see "Object Properties" on page 46.

Place

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Place`. For details, see "Object Properties" on page 46.

PlaceGroup

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Place Group`. For details, see "Object Properties" on page 46.

RoutingPoint

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Routing Point`. For details, see "Object Properties" on page 46.

RoutingQueue

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Routing Queue`. For details, see "Object Properties" on page 46.

RoutingStrategy

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Routing Strategy`. For details, see "Object Properties" on page 46.

Script

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Script`. For details, see "Object Properties" on page 46.

ServiceNumber

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Service Number`. For details, see "Object Properties" on page 46.

StagingArea

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Script` with subtype `Staging Area`. For details, see "Object Properties" on page 46.

Switch

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Switch`. For details, see "Object Properties" on page 46.

Tenant

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Tenant`. For details, see "Object Properties" on page 46.

TieLine

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Tie Line`. For details, see "Object Properties" on page 46.

TieLineGroup

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Tie Line Group`. For details, see "Object Properties" on page 46.

Trunk

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Trunk`. For details, see "Object Properties" on page 46.

TrunkGroup

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `TrunkGroup`. For details, see "Object Properties" on page 46.

Video

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Video`. For details, see "Object Properties" on page 46.

VirtACDQueue

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Virt ACD Queue`. For details, see "Object Properties" on page 46.

VirtRoutingPoint

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Virt Routing Point`. For details, see "Object Properties" on page 46.

VoiceMail

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Voicemail`. For details, see "Object Properties" on page 46.

VoIP

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `VoIP`.

VoIPService

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `VoIp Service`. For details, see "Object Properties" on page 46.

Workbin

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `Script` with subtype `Workbin`. For details, see "Object Properties" on page 46.

Workflow

Valid Values: Valid object name format string. For details, see page 45.
Default Value: `%ObjectName%`
Changes Take Effect: After restart

Specifies the object name formatting rule for an object of type `DN` with subtype `Workflow` For details, see “Object Properties” on page 46.

Valid object name format string

The valid object name format string is free text string, which allows values of object properties:
`%<PropertyName>:<side><padding><length>%`

- PropertyName—Specifies the property name.

  Note: Property names are case sensitive.

- side—Specifies the side, `L` (left) or `R` (right) , from where the length must be counted. If you do not specify a side, Pulse uses `L` by default.

  `L` is commonly used for string or text properties.

  `R` is commonly used for numbers.
- padding—Specifies the padding when the property value length is less than the specified custom length
  - . (dot) to pad with space characters
  - `0` (zero) to pad with zero characters.
- length—Specifies the maximum number of characters for the property name.

Table 1: Examples

Valid Object Name Format String	Description
• %EmployeeID:10% • %EmployeeID:L10%	Both specify 10 characters of property EmployeeID from the left side.
• %EmployeeID:.10% • %EmployeeID:L.10%	Specifies 10 characters of property EmployeeID from left, but if length was less than 4 symbols, pad it with spaces.
• %DBID:R4%	Specifies 4 characters of property DBID from the right side.
• %DBID:R04%	Specifies 4 characters of property DBID from the right side, but if the length is less than 4 characters, pad it with zeros.

Table 2: Examples of Results

Valid Object Name Format String	Result
%Number%@%SwitchID%	1001@Switch1
[%Alias%] - %Number%@%SwitchID%	[Main IVR Entry Point] - 5001@SIP_Switch
%LastName%, %FirstName%	Brown, James
%LastName%, %FirstName% [%EmployeeID%]	Brown, James [jbrown]

Object Properties

Table 3 on page 47 lists the object properties available for use in the format strings.

Table 3: Object Information

Object Type	Property	Description
All Object Types	DBID	Specifies the ID of the object. For example, `Campaign Group` is in group DBID, `Campaign Calling List` is in CallingList ID.
All Object Types	ObjectID	Specifies the ID number of the object in Pulse Collector. **Notes:** • This is the typically the configuration layer DBID, but for some types of objects (for example, `Campaign Group`, `Campaign Calling List`) this is a composite 64-bit ID. • The composite 64-bit ID is an unsigned 64-bit number with the following composition: ◦ higher 32 bits: DBID of Campaign ◦ lower 32 bits: DBID of Calling List or Agent/Place group
All Object Types	ObjectName	Specifies the name of the object, which is written to the snapshot file.
All Object Types	ObjectType	Specifies the type of the object.
All Object Types	TenantID	Specifies the Tenant ID of the object.
All Object Types	type	Specifies the type of the object.
Agent	EmailAddress	Specifies the email address of the agent (person).
Agent	EmployeeID	Specifies the employee ID of the agent (person).
Agent	ExternalID	Specifies the external ID of the agent (person).
Agent	FirstName	Specifies the first name of the agent (person).
Agent	LastName	Specifies the last name of the agent (person).
Agent	UserName	Specifies the user name of the agent (person).
Calling List	Description	Describes the calling list.
Campaign	Description	Describes the campaign.
Campaign Calling List	CallingListDescription	Describes the underlying Calling List object.

Table 3: Object Information (Continued)

Object Type	Property	Description
Campaign Calling List	CallingListName	Specifies the DBID of the Calling List object.
Campaign Calling List	CampaignDBID	Specifies the DBID of the Campaign object.
Campaign Group	CampaignDBID	Specifies the DBID of the Campaign object.
Campaign Group	GroupDBID	Specifies the DBID of the group object.
Campaign Group	GroupType	Specifies the numeric Type of the group object (`CFGAgentGroup` or `CFGPlaceGroup`).
DN, Routing Queue, Routing Point	Alias	Specifies the DN Alias.
DN, Routing Queue, Routing Point	Number	Specifies the DN number.
DN, Routing Queue, Routing Point	SwitchDBID	Specifies the switch DBID.
DN, Routing Queue, Routing Point	SwitchID	Specifies the switch name.
Routing Strategy, Staging Area, Workbin	ScriptType	Specifies the script type ID.
Switch	DNRange	Specifies the switch DN Range.
Switch	SwitchType	Specifies the switch type.

[output] Section

cleanup-output

Valid Values: `yes,no`

Default Value: `yes`

Changes Take Effect: After restart

Determines whether Pulse Collector conducts output directory cleanup cycles, which remove outdated output files belonging to layouts that are no longer present in the Pulse database.

max-output-interval

Default Value: `3600`
Valid Values: `3–3600`
Changes Take Effect: After restart

The maximum allowed output interval for all report layouts. Users can independently set output frequencies *by layout* within the Pulse user interface. If the set frequency, however, is greater than the value of this option, Pulse uses the value of this option instead.

Note: The value of this option must be greater than the value of the `min-output-interval` option or Pulse Collector logs an appropriate error.

min-output-interval

Default Value: `3`
Valid Values: `3–3600`
Changes Take Effect: After restart

The minimum allowed output interval for all report layouts. Users can independently set output frequencies *by layout* within the Pulse user interface. If the set frequency, however, is less than the value of this option, Pulse uses the value of this option instead.

[scripting] Section

definition-script-execution-timeout

Valid Values: `1-900`
Default Value: `45`
Changes Take Effect: After restart

Time in seconds allowed for definition script execution.

formula-script-execution-timeout

Valid Values: `1-900`
Default Value: `5`
Changes Take Effect: After restart

Time in seconds allowed for single formula evaluation script execution.

init-script-execution-timeout

Valid Values: `1-900`
Default Value: `60`
Changes Take Effect: After restart

Time in seconds allowed for initialization script execution.

js-lib-path

Valid Values: Valid folder paths
Default Value: `./jslib/standard`
Changes Take Effect: After restart

Comma-separated list of locations of the directories that contain additional JavaScript libraries to be used within the formula scripting engine.

js-modules

Valid Values: Comma-separated list of JavaScript files
Default Value: `collector.js,cfglib.js,statlib.js,gts.js`
Changes Take Effect: After restart

Comma-separated list of modules to preload into the scripting engine.

[statistic-request-handling] Section

always-use-statserver-newapi

Valid Values: yes, no
Default Value: no
Changes Take Effect: After restart

Determines whether to force Pulse Collector to request all statistics through the StatServer New API that uses the proper parameter set.

data-flow-check-interval

Default Value: `360`
Valid Values: `5–3600`
Changes Take Effect: After restart

Recommended Values: Positive integers that are less than the maximum notification interval value, either with reset-based notification using the time profile or time-based notification using the notification interval parameter.

Note: Standard Pulse layouts use a 60 second notification interval.

Changes Take Effect: Immediately

How often, in seconds, Pulse Collector verifies that there is data flow from Stat Server (primary and backup), provided that Pulse Collector contains active report layouts and has a live connection to Stat Server.

data-flow-timeout

Default Value: `120`
Valid Values: `5–86400`
Changes Take Effect: After restart
Recommended Values: Positive integers that are greater than the value that you specify for `data-flow-check-interval` and less than two times of the maximum notification interval value, either with reset-based notification using time profile or time-based notification using notification interval parameter.

Note: Standard Pulse layouts use a 60 second notification interval.

Changes Take Effect: Immediately

Time, in seconds, that is acceptable to suspend the transfer of data from Stat Server (primary or backup), provided that Pulse Collector contains active report layouts and has a live connection to Stat Server. The timer starts upon the last successful data transfer.

data-source-choice-strategy

Valid Values: `PrimaryRunMode, LastGood, FirstAvailable, MostUpToDate, PrimaryInCME`

Note: All values are case-insensitive.

Default Value: `PrimaryRunMode`
Changes Take Effect: After restart

Specifies which StatServer configured in the configuration layer that Pulse Collector uses as a data source for a snapshot:

- `PrimaryRunMode`—Pulse Collector uses the StatServer running in the primary mode. If both Stat Servers appear to be backup, Pulse Collector attempts the next strategy.
- `LastGood`—Pulse Collector uses the last good StatServer if available. Otherwise, Pulse Collector attempts the next strategy.
- `FirstAvailable`—Pulse Collector uses the Primary StatServer if available. If uses the Primary StatServer is unavailable, Pulse Collector uses the Backup StatServer. Otherwise, Pulse Collector attempts the next strategy.
- `MostUpToDate`—Pulse Collector uses StatServer that sent statistic data with most recent Server Time. Otherwise, Pulse Collector attempts the next strategy.
- `PrimaryInCME`—Pulse Collector always uses the Primary StatServer.

statserver-batch-size

Valid Values: `1-10000`
Default Value: `500`
Changes Take Effect: Immediately

Specifies the number of statistic requests sent to Stat Server in a single packet. The recommended value depends on the number of statistic requests you plan to run, network bandwidth, and processing capabilities of the Stat Server and Pulse Collector servers. If Stat Server disconnects Pulse Collector when it is opening statistics with error message `Client too slow`, decrease value of this option.

statserver-profiles-timeout

Valid Values: `1-86400`
Default Value: `600`
Changes Take Effect: Immediately

Specifies the timeout, in seconds, to receive and process server profiles from Stat Server. If profiles are not received and processed within the given timeout, Pulse Collector closes the current connection to StatServer and attempts to reconnect.

suspend-statistic-notifications-for-paused-layouts

Valid Values: `yes, no`
Default Value: `no`
Changes Take Effect: After restart

Determines whether to suspend the statistic notifications for paused layouts.

Note: You must have Stat Server version 8.1.200.17 or higher for this functionality to work correctly, if you set the value of `suspend-statistic-notifications-for-paused-layouts` to `yes`.

verbose-request-statistics

Valid Values: `true, false`
Default Value: `false`
Changes Take Effect: Immediately

Determines whether to enable Pulse Collector to log verbose messages about the objects for which it requests statistics.

[transport-*] Section

These sections must be called `transport-*`, where `*` is the transport name (for example, the [`transport-gpb-out`] or [`transport-gpb-history`] section).

backend-file-ext

Valid Values: Valid file extension
Default Value: `be`
Changes Take Effect: After restart

Specifies which file extension Pulse uses to consume Pulse Collector's output files. This option is for use only in the `[transport-gpb-out]` section.

cleanup-file-modification-age-threshold

Valid Values: `0-365`
Default Value: `15`
Changes Take Effect: After restart

Specifies how many days Pulse Collector keeps output files for deleted layouts. Time is counted since the last modification timestamp of the file.

generate-layout-info-file

Valid Values: `yes,no`
Default Value: `no`
Changes Take Effect: After restart

Determines whether to generate output to a layout information file and snapshot file. This option is for use only in the `[transport-gpb-out]` section.

Note: If transport is used for writing history, a layout information file is never generated, regardless of the value of this option.

output-file-ext

Valid Values: Valid file extensions for your operating system
Default Value: `gpb`
Changes Take Effect: After restart

Specifies the file extension for the full output file format.

output-folder

Valid Values: Valid folder paths
Default Value: No default (the `collector.apd` file supplies the value of a sample output directory)
Changes Take Effect: After restart

Specifies the path in which Pulse Collector writes output files. If you specify a folder that does not exist, Pulse Collector creates it for you.

output-format

Default Value: `binary`
Valid Values:

- `binary`—Outputs files in binary format. You must set the value of this option to `binary` if, you want to use the Pulse Plug-in for GAX.
- `text`—Reserved for Genesys Engineering

Changes Take Effect: After restart

Specifies the output format.

transport-type

Default Value: `gpb-file`
Valid Values: `gpb-file`
Changes Take Effect: After restart

Specifies the transport output type.

Common Log Options

Pulse Collector also supports configuration of options common to most Genesys servers. Refer to the "Common Configuration Options" chapter of the *Framework 8.1 Configuration Options Reference Manual* for descriptions of options that control the logging of events. Specifically, refer to the following sections:

- `log`
- `log-extended`
- `log-filter`
- `log-filter-data`
- `security`
- `security-authentication-rules`
- `sml`

For the Pulse Plug-in for GAX, log activity is sent to GAX logs.

Recommended Settings for the log Section

```
[log]
segment=10 MB
verbose=all
all=stdout,pathToLogFile
```

Chapter

4 Starting and Stopping Pulse

This chapter describes how to start and stop Pulse components. Start procedures assume that you have properly configured and installed application components. If not, refer to Chapters 2 and 3.

This chapter contains the following sections:

Starting and Stopping Pulse Plug-in for GAX

This section lists the prerequisites for starting the Pulse Plug-in for GAX as well as how to invoke and stop it.

What Must Be Running Prior to Start

- Your web browser must be one of the following:
  - Google Chrome version 22 or higher
  - Mozilla Firefox version 17 or higher
  - Microsoft Internet Explorer version 10
  - Apple Safari version 6
- Your monitor must be set to a resolution of no less than 1024x768.
- The DB Server for Pulse Collector must be running.
- The RDBMS for the Pulse database must be running.
- Configuration Server and the DB Server for Configuration Server must be running.

- The user's login must have at least Read permission to their own `person` object and Read and Execute permissions on the Genesys Administrator Extension client object. Refer to the *Genesys 8.x Security Deployment Guide* for information about permissions. Genesys Administrator Extension uses the permissions that are set for Environments and Tenants. For example, if a user does not have Write permission for the Default dashboard, they can view changes they create, but next time the page is loaded, it reverts to the Default dashboard. See the steps about defining privileges in Procedure: Configuring objects for Pulse, on page 22.

Starting Pulse Plug-in for GAX

You start Pulse by logging into Genesys Administrator Extension (GAX). Refer to the *Framework 8.1 Genesys Administrator Extension Deployment Guide* for information about how to start this application. Once you are logged in, you access Pulse by clicking the `Reports` icon along the top menu bar. Refer to *Genesys 8.1 Pulse Help* for information about how to use this plug-in.

Stopping Pulse Plug-in for GAX

Stop Pulse by logging out of GAX. Click `log out` along the top menu bar.

Starting Pulse Collector

You can start Pulse Collector on a Windows or UNIX platform. Invoking Pulse Collector starts a series of internal checks for proper configuration. The value of the `max-output-interval` option, for example must be greater than the value of the `min-output-interval` option or Pulse Collector exits. Verify your Collector `Application` object for proper configuration before starting Pulse Collector.

What Must Be Running Prior to Start

Stat Server Before starting Pulse Collector for the first time, ensure you have properly configured your Stat Server `Application` object. This is especially important if you configured this object manually.

Warning! You must set the configuration option `accept-clients-in-backup-mode` to `yes` for each Stat Server application that is used with Pulse Collector, even when Stat Server is not in an HA configuration (where you have only one Stat Server). If the option is missing, misleading data unavailability errors might occur.

Backup Configuration Server

To restart the Pulse Collector application when the backup Configuration Server switches to the primary mode, you must specify the backup Configuration Server parameters when starting Pulse Collector.

On the command line, specify these parameters using the following two arguments:

`-backup_host` *hostname*
`-backup_port` *port-number*

For more information on starting Pulse Collector from a console window, see "From a Console Window" on page 58 of this guide.

National Characters

To use national characters in the string properties correctly, Pulse Collector determines which multibyte encoding is used by Configuration Server. You can allow Pulse Collector to use the default setting or specify the characters by editing the following configuration options:

Windows

- You can specify Configuration Server multibyte encoding through the command-line parameter `-cs_codepage` following the Windows code page number (for example, `1251`). For more information about Windows code pages and their numbers, see http://msdn.microsoft.com/en-us/library/windows/desktop/dd317756(v=vs.85).aspx
  If the parameter is not specified, Pulse Collector assumes Configuration Server uses the code page that corresponds to the locale of the Windows operating system where Pulse Collector is running.

Or,

- You can specify Configuration Server multibyte encoding through the command-line parameter `-cs_encoding`. It accepts Microsoft-compatible symbolic name of locale and results choice of default code page for that locale. For more information about Microsoft-compatible locale names, see http://msdn.microsoft.com/en-us/library/dd373814.aspx

Notes:
- These parameters work only on Windows Server version 2008 or higher. If you need this functionality on the downlevel systems (Windows 2003 Server), you must install the downlevel locale mapping DLL.
- For more information about locale mapping on the downlevel systems, see http://msdn.microsoft.com/en-us/library/ms776343.aspx.
- To download a downlevel locale mapping compatibility DLL redistributable package, proceed here http://www.microsoft.com/en-us/download/confirmation.aspx?id=25241.

Linux

You can specify Configuration Server multibyte encoding through the command-line parameter `-cs_encoding` following the iconv-compatible symbolic name of encoding (for example, UTF-8).

To display the list of encodings supported on the your system, enter the following command in the Linux console:

```
iconv --list
```

Starting Pulse Collector from SCI

You can start Pulse Collector on any of the supported platforms. To do so from the Solution Control Interface (SCI):

1. From the `Applications` view, select your Pulse Collector `Application` object in the list pane.
2. Click the `Start` button on the toolbar, or select `Start` either from the `Action` menu or the context menu. (Right-clicking your `Application` object displays the context menu.)
3. Click `Yes` in the confirmation box that appears. Your Pulse Collector application starts.

For information about how to use SCI, refer to *Framework 8.0 Solution Control Interface Help*.

Starting Pulse Collector from Windows Platforms

On a Windows platform, you can start Pulse Collector:

- Manually from the `Programs` menu as an application.
- Manually from a console window as an application.
- Automatically or manually as a Windows service.

From the Programs Menu

To start Pulse Collector from the `Programs` menu as an application, select `Start Pulse Collector` from the program group created during installation. The application opens a console window and automatically issues the parameters specified during configuration to start Pulse Collector. The Pulse Collector application name, version, and connectivity parameters appear in the title bar.

From a Console Window

To start Pulse Collector as an application from a console window:

1. At the command-line prompt, go to the directory where Pulse Collector has been installed.

2. collectorType the name of the Pulse Collector executable followed by the appropriate command-line parameters using the following syntax:

   `collector.exe -host` *hostname* `-port` *portno* `-app` *application*

   where:

   - *hostname* refers to the name of the computer on which Configuration Server is running.
   - *portno* refers to the communication port on which Configuration Server is running.
   - *application* refers to the name of the Pulse Collector `Application` object as defined in Genesys Administrator.

   Note: If the host or application name contains spaces or hyphens (–), enclose it in double quotation marks.

   For example, to start Pulse Collector with parameters specifying the host as `cs-host`, port as `2020`, and name as `Pulse Coll`, type:

   ```
   collector.exe -host "cs-host" -port 2020 -app "Pulse Coll"
   ```

   Note: If needed, specify the optional parameters `-backup_host`, `-backup_port`, `-cs_codepage`, and `-cs_encoding`.

As a Windows Service

1. From the task bar, choose `Start - Administrative Tools > Computer Management`.
2. Open `Services and Applications > Services`.
3. Right-click your Pulse Collector service from the list and select `Start`.

Note: Since the Local Control Agent (LCA) can be installed as a Windows service with the user interface disabled, all servers started through SCI, in this case, are started without a console unless you specifically select the `Allow Service to Interact with Desktop` check box for both LCA and Pulse Collector.

Manually Starting Pulse Collector from UNIX Platforms

1. Go to the directory where Pulse Collector has been installed.

   Note: You can invoke Pulse Collector only from the directory in which it was installed.

2. Type the name of the Pulse Collector executable followed by the appropriate command-line parameters using the following syntax:

   `./collector -host` *hostname* `-port` *portno* `-app` *application*

where:

- *hostname* refers to the name of the computer on which Configuration Server is running.
- *portno* refers to the communication port on which Configuration Server is running.
- *application* refers to the name of the Pulse Collector Application object as defined within Genesys Administrator.

Note: If the host or application name contains spaces or hyphens (–), enclose it in double quotation marks.

For example, to start Pulse Collector with parameters specifying the host as cs-host, port as 2020, and name as Pulse Coll, type:

```
./collector -host "cs-host" -port 2020 -app "Pulse Coll"
```

Note: If needed, specify the optional parameters -backup_host, -backup_port, and -cs_encoding.

Configure the stdout option in the log section of collector options to write to a log file, so that you can check for errors in its configuration if Pulse Collector fails to start. If you cannot resolve a problem, contact Genesys Customer Care and provide the entire content of the log.

You can also type the name of the Pulse Collector executable and its command-line parameters into a shell script and execute the script using the following command:

./run.sh *[Name of script]*

To redirect Pulse Collector output (on most UNIX shells), use the following syntax:

```
./collector -host hostname -port portno -app appl >
    log_file.log
```

To have both log file and console, within Genesys Administrator add the following to Pulse Collector's application properties:

- Section: Log
- Option: all with the following value:

  stdout,<log_file_name.log>,network

  Separate values with commas. Instead of stdout, you can also specify stderr.

  Pulse Collector writes messages to <log_file_name.log> in the same directory where Pulse Collector is installed. To have Pulse Collector write to a different location, specify the full path for this parameter.

Stopping Pulse Collector

Stopping Pulse Collector on Windows Platforms

If Pulse Collector is running as an application, switch to its console window and press Control-C (^C) to stop it. If you are running Pulse Collector as a Windows service, you should stop it only from the Microsoft Management Console.

To stop Pulse Collector running as a Windows service:

1. From the task bar, choose Start - Administrative Tools > Computer Management.
2. Open Services and Applications > Services.
3. Right-click your Pulse Collector service from the list and select Stop.

Note: Be sure that the autorestart property is cleared for the Pulse Collector Application object in Genesys Administrator.

If you use LCA and Solution Control Server (SCS), you can stop Pulse Collector from SCI. To do so:

1. From the Applications view, select your Pulse Collector Application object in the list pane.
2. Click Stop on the toolbar, or select Stop either from the Action menu or the context menu.
3. Click Yes in the confirmation box that appears.

Stopping Pulse Collector on UNIX Platforms

Note: Be sure that the autorestart property is cleared for the Pulse Collector Application object in Genesys Administrator.

Stop Pulse Collector on UNIX using either of the following methods:

- On the command line, type kill -9 *processid,* where *processid* is Pulse Collector's UNIX process ID.
- Press ^C from the active Pulse Collector console.
- If LCA and SCS are used, you can stop Pulse Collector from SCI.

Chapter

5 Application Files

The Pulse installation routine creates a root application folder with a `scripts` subfolders.

This chapter briefly describes the files that are deployed by the installation routines of the following components.

Pulse Plug-in for GAX

Table 4 describes the files in the Pulse root folder. Table 5 describes the files in the `scripts` subfolder.

Table 4: Contents of the Pulse Root Folder

File Name	Description
pulse.jar	Executable Jar file.
pulse-proto.jar	Executable Jar file.
ip_description.xml	File storing installation package content.
protobuf-java-2.5.0.jar	Jar file that contains the Google Protocol Buffers library.
read_me.html	File that contains general information about the installation package.
dbtool	Folder that contains dbtool files.
thirdparty	Folder that contains third-party licenses and additional files as required.

Table 4: Contents of the Pulse Root Folder (Continued)

File Name	Description
scripts	Folder that contains: • Folders that hold the database initialization SQL scripts for each of the following RDBMS types: ◦ Microsoft SQL ◦ Oracle ◦ PostgreSQL • pulse_statistics.cfg file—File that specifies the options needed for Stat Server to process Pulse preconfigured templates. See Table 5 for the contents of each subfolder.

Table 5: Contents of the scripts\[dbtype] Subfolder

File Name	Description
mssql\pulse_init_mssql.sql	Database initialization script for a Microsoft SQL RDBMS.
oracle\pulse_init_oracle.sql	Database initialization script for an Oracle RDBMS.
postgres\pulse_init_postgres.sql	Database initialization script for a PostgreSQL RDBMS.

Pulse Collector

Table 6 describes the files that the Pulse Collector installation routine deploys to the Collector root folder.

Table 6: Contents of the Pulse Collector Root Folder

File Name	Description
common.lms	File that stores common log messages.
collector collector.exe	Application executables for Unix and Microsoft Windows platforms.
collector.debug	File that stores debugging information for collector executable.
collector.lms	File that stores Pulse Collector–specific log messages.
extras	Folder that stores file extra configuration options.
ip_description.xml	File that stores installation package content.

Table 6: Contents of the Pulse Collector Root Folder (Continued)

File Name	Description
messages	Folder that stores localized error messages.
ospatchlist.txt	File that contains a list of OS patches.
run.sh	On UNIX platforms, shell script that launches Pulse Collector.
startServer.bat	On Windows platforms, batch file that starts Pulse Collector as a regular application, not as a Windows Service.
thirdparty	Folder that contains third party licenses and additional files as required.
jslib	Folder that contains two subfolders, `extras` and `standard`, which contain the JavaScript library files to be used by Pulse Collector.

Table 7: Contents of the Pulse Collector extras Folder

File Name	Description
finetuning.cfg	File that provides fine tuning configuration options.

Table 8: Contents of the Pulse Collector jslib/standard Folder

File Name	Description
cfglib.js	File that provides JavaScript definitions for Configuration Server.
collector.js	File that provides JavaScript definitions for Pulse Collector.
gts.js	File that provides additional JavaScript functions used in the standard Pulse layout templates.
statlib.js	File that provides JavaScript definitions for various constants for Stat Server.

Table 9: Contents of the Pulse Collector messages Folder

File Name	Description
layout-en.msg	Localized message file.

Table 9: Contents of the Pulse Collector messages Folder (Continued)

File Name	Description
layout-ru.msg	Localized message file.
layout-uk.msg	Localized message file.

Related Documentation Resources

The following resources provide additional information that is relevant to this software. Consult these additional resources as necessary.

Pulse

- The *Pulse 8.1 Help* file, available by clicking help within the Pulse interface.
- Release Notes and Product Advisories for this product, which are available on the Genesys Documentation website at http://docs.genesys.com/.

Framework

- The *Framework 8.1 Genesys Administrator Extension Deployment Guide,* which helps you deploy Genesys Administrator Extension.
- *Framework 8.1 Genesys Administrator Help,* which helps you use Genesys Administrator.
- The *Framework 8.1 Stat Server User's Guide*, which helps you understand statistics configuration.
- *Framework 8.1 Configuration Options Reference Manual,* which describes the log options that are common to most Genesys servers.

Microsoft Visual C++ 2005 SP1 MFC Security Pack Redistributable

Download the installer from the following URL:
http://www.microsoft.com/download/en/details.aspx?id=26347

Genesys

- The *Genesys 8.1 Security Deployment Guide,* which describes the security features provided by Genesys software and provides detailed instructions on deploying the features.
- *Genesys Technical Publications Glossary,* which ships on the Genesys Documentation Library DVD, provides a comprehensive list of the Genesys and computer-telephony integration (CTI) terminology and acronyms used in this document.
- *Genesys Migration Guide*, which ships on the Genesys Documentation Library DVD, provides documented migration strategies for Genesys product releases. Contact Genesys Customer Care for more information.
- Release Notes and Product Advisories for this product, which are available on the Genesys Customer Care website at http://www.genesys.com/customer-care.

For additional system-wide planning tools and information, see the release-specific listings of System-Level Documents on the Genesys Documentation website (docs.genesys.com).

Genesys product documentation is available on the:

- Genesys Customer Care website at http://www.genesys.com/customer-care
- Genesys Documentation Library DVD, which you can order by e-mail from Genesys Order Management at orderman@genesys.com.

Document Conventions

This document uses certain stylistic and typographical conventions that serve as shorthands for particular kinds of information.

Document Version Number

A version number appears at the bottom of the inside front cover of this document. Version numbers change as new information is added to this document. Here is a sample version number:

80fr_ref_06-2011_v8.0.001.00

You will need this number when you are talking with Genesys Customer Care about this product.

Screen Captures Used in This Document

Screen captures from the product graphical user interface (GUI), as used in this document, may sometimes contain minor spelling, capitalization, or grammatical errors. The text accompanying and explaining the screen captures corrects such errors *except* when such a correction would prevent you from installing, configuring, or successfully using the product. For example, if the name of an option contains a usage error, the name would be presented exactly as it appears in the product GUI; the error would not be corrected in any accompanying text.

Type Styles

Table 10 describes and illustrates the type conventions that are used in this document.

Table 10: Type Styles

Type Style	Used For	Examples
Italic	• Document titles • Emphasis • Definitions of (or first references to) unfamiliar terms • Mathematical variables Also used to indicate placeholder text within code samples or commands, in the special case where angle brackets are a required part of the syntax (see the note about angle brackets on page 70).	Please consult the *Genesys Migration Guide* for more information. Do *not* use this value for this option. A *customary and usual* practice is one that is widely accepted and used within a particular industry or profession. The formula, $x + 1 = 7$ where x stands for . . .

Table 10: Type Styles (Continued)

Type Style	Used For	Examples
Monospace font (Looks like `teletype` or `typewriter text`)	All programming identifiers and GUI elements. This convention includes: • The *names* of directories, files, folders, configuration objects, paths, scripts, dialog boxes, options, fields, text and list boxes, operational modes, all buttons (including radio buttons), check boxes, commands, tabs, CTI events, and error messages. • The values of options. • Logical arguments and command syntax. • Code samples. Also used for any text that users must manually enter during a configuration or installation procedure, or on a command line.	Select the `Show variables on screen` check box. In the `Operand` text box, enter your formula. Click `OK` to exit the `Properties` dialog box. T-Server distributes the error messages in `EventError` events. If you select `true` for the `inbound-bsns-calls` option, all established inbound calls on a local agent are considered business calls. Enter `exit` on the command line.
Square brackets ([ ])	A particular parameter or value that is optional within a logical argument, a command, or some programming syntax. That is, the presence of the parameter or value is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information.	`smcp_server -host [/flags]`
Angle brackets (< >)	A placeholder for a value that the user must specify. This might be a DN or a port number specific to your enterprise. **Note:** In some cases, angle brackets are required characters in code syntax (for example, in XML schemas). In these cases, italic text is used for placeholder values.	`smcp_server -host <confighost>`

Index

Symbols

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W