GENESYS®
AN ALCATEL-LUCENT COMPANY

Reporting 7.5

Reference Manual

About Genesys

Genesys Telecommunications Laboratories, Inc., a subsidiary of Alcatel-Lucent, is 100% focused on software for call centers. Genesys recognizes that better interactions drive better business and build company reputations. Customer service solutions from Genesys deliver on this promise for Global 2000 enterprises, government organizations, and telecommunications service providers across 80 countries, directing more than 100 million customer interactions every day. Sophisticated routing and reporting across voice, e-mail, and Web channels ensure that customers are quickly connected to the best available resource—the first time. Genesys offers solutions for customer service, help desks, order desks, collections, outbound telesales and service, and workforce management. Visit www.genesyslab.com for more information.

Each product has its own documentation for online viewing at the Genesys Technical Support 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, the Genesys logo, and T-Server are registered trademarks of Genesys Telecommunications Laboratories, Inc. All other trademarks and trade names referred to in this document are the property of other companies. 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.

Technical Support from Genesys

If you have purchased support directly from Genesys, please contact Genesys Technical Support at the following regional numbers:

Region	Telephone	E-Mail
North and Latin America	+888-369-5555 or +506-674-6767	support@genesyslab.com
Europe, Middle East, and Africa	+44-(0)-1276-45-7002	support@genesyslab.co.uk
Asia Pacific	+61-7-3368-6868	support@genesyslab.com.au
Japan	+81-3-6361-8950	support@genesyslab.co.jp

Prior to contacting technical support, please refer to the *Genesys Technical Support 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 7 Licensing Guide*.**

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 75rt_ref_07-2008_v7.5.101.00

Table of Contents

Preface

Welcome to the *Reporting 7.5 Reference Manual.* This manual provides general information about performance measurements, how Reporting behaves during time shifts, and how to set up custom reports for skills-based and partial-period reporting.

This manual is valid only for the 7.5 release of this product.

Note: For releases of this document created for other releases of this product, please visit the Genesys Technical Support website, or request the Documentation Library DVD, which you can order by e-mail from Genesys Order Management at `orderman@genesyslab.com`.

This preface provides an overview of this manual, identifies the primary audience, introduces document conventions, and lists related reference information:

- Intended Audience, page 6
- Chapter Summaries, page 6
- Document Conventions, page 7
- Related Resources, page 8
- Making Comments on This Document, page 9

Reporting 7.5 consists of two applications, CCPulse+ 7.5 and Contact Center Analyzer (CC Analyzer) 7.2. CCPulse+ is a desktop-level reporting application that provides both real-time and historical views of contact center operations via a Win32 graphical user interface. Designed to facilitate daily resource management tasks, CCPulse+ is an ideal tool for at-a-glance evaluation of present and past contact center performance and efficient operational decision-making.

CC Analyzer is an enterprise-class, historical reporting solution featuring powerful report development tools, on-demand and scheduled report generation functions, and means of report delivery across the enterprise intranet. A set of business intelligence applications, developed by Hyperion Corporate Solutions, (formerly Brio Technology, Inc.), enable these advanced reporting capabilities.

Intended Audience

This manual, primarily intended for system administrators, assumes that you have a basic understanding of:

- Computer-telephony integration (CTI) concepts, processes, terminology, and applications.
- Your own Reporting configurations.
- The Genesys Framework.

You should also have a solid understanding of CC Analyzer and CCPulse+.

Chapter Summaries

In addition to this opening chapter, this manual contains these chapters:

- Chapter 1, "Skills-Based Reporting," on page 11, provides a theoretical approach on how to set up reporting based on agents skill sets.
- Chapter 2, "Partial-Period Reporting," on page 17, provides tips for tuning your Reporting environment to generate near-accurate reports when only part of a period has elapsed. For example, how do you get the best numbers from the agent daily report when the day hasn't yet completed?
- Chapter 3, "Performance Measurements," on page 21, provides test results from the Genesys lab on the performance of Reporting components common to both CC Analyzer and CCPulse+.
- Chapter 4, "Daylight Saving Time and Other Time Shifts," on page 35, describes the effect of time shift on the machines where your servers reside focusing specifically on observance of daylight saving time. This chapter also describes reporting over multiple time zones.
- Chapter 5, "Database Backups," on page 43, provides references where you can obtain more information about data recovery and backup applications and procedures.

After reading through the scenarios described in this manual, you will see that with a little creativity, you can customize the Reporting components and your environment to generate many different types of reports beyond those generated by the provided report templates. For example, you may decide to design profitability reports or service-level reports.

You can even tailor the definitions of the underlying statistics collected (such as `ServiceFactor`) to meet your business needs.

Document Conventions

This document uses certain stylistic and typographical conventions—introduced here—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:

75rt_ref_04-2007_v7.5.001.02

You will need this number when you are talking with Genesys Technical Support about this product.

Type Styles

Italic

In this document, italic is used for emphasis, for documents' titles, for definitions of (or first references to) unfamiliar terms, and for mathematical variables.

Examples:

- Please consult the *Genesys 7.2 Migration Guide* for more information.
- *A customary and usual practice* is one that is widely accepted and used within a particular industry or profession.
- Do *not* use this value for this option.
- The formula, x +1 = 7 where x stands for . . .

Monospace Font

A monospace font, which looks like `teletype or typewriter text`, is used for 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; and code samples.

Examples:

- Select the `Show variables on screen` check box.
- Click the `Summation` button.
- In the `Properties` dialog box, enter the value for the host server in your environment.
- In the `Operand` text box, enter your formula.
- Click `OK` to exit the `Properties` dialog box.

- The following table presents the complete set of error messages T-Server® distributes 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.

Monospace is also used for any text that users must manually enter during a configuration or installation procedure, or on a command line:

Example:
- Enter `exit` on the command line.

Screen Captures Used in This Document

Screen captures from the product GUI (graphical user interface), as used in this document, may sometimes contain a minor spelling, capitalization, or grammatical error. 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.

Square Brackets

Square brackets indicate that a particular parameter or value is optional within a logical argument, a command, or some programming syntax. That is, the parameter's or value's presence is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information. Here is a sample:

```
smcp_server -host [/flags]
```

Angle Brackets

Angle brackets indicate a placeholder for a value that the user must specify. This might be a DN or port number specific to your enterprise. Here is a sample:

```
smcp_server -host <confighost>
```

Related Resources

Consult these additional resources as necessary:

- The Framework documentation set for an explanation of the Genesys Configuration Layer.
- The documentation set for the Enterprise Routing Solution for help defining and implementing routing strategies.

- The *Framework 7.x Stat Server User's Guide,* for information about reset-based and time-based notification modes.
- The *Reporting 7.2 Data Sourcer User's Guide,* which shows you how to implement database binding functionality on Oracle RDBMSs.
- *Reporting 7.2 Data Modeling Assistant Help,* which includes discussions on activating and deactivating report layouts and assigning time profiles to them.
- The *Reporting 7.2 ETL Runtime User's Guide,* which provides information about the Transformation and Aggregation modules as well as ETL Runtime's parameters.
- *Reporting 7.2 ETL Assistant Help,* which explains how report folders and report views are organized within the Data Mart.
- The *Reporting 7.5 Master Index,* which provides a handy reference where other related topics are documented.
- The *Genesys Technical Publications Glossary,* which ships on the Genesys Documentation Library DVD and which provides a comprehensive list of the Genesys and CTI terminology and acronyms used in this document.
- The *Genesys Migration Guide*, also on the Genesys Documentation Library DVD, which contains a documented migration strategy for Genesys product releases 6.x and later. Contact Genesys Technical Support for additional information.
- The Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at `http://genesyslab.com/support`.

Information on supported hardware and third-party software is available on the Genesys Technical Support website in the following documents:

- *Genesys 7 Supported Operating Systems and Databases*
- *Genesys 7 Supported Media Interfaces*

Genesys product documentation is available on the:

- Genesys Technical Support website at `http://genesyslab.com/support`.
- Genesys Documentation Library DVD, which you can order by e-mail from Genesys Order Management at `orderman@genesyslab.com`.

Making Comments on This Document

If you especially like or dislike anything about this document, please feel free to e-mail your comments to `Techpubs.webadmin@genesyslab.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 information in this document only and to the way in which the information is presented. Speak to Genesys Technical Support 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.

Chapter

1 Skills-Based Reporting

In the simplest (and probably most common) form of skills-based routing (SBR), agents are assigned one or more skills that reflect the types of calls they are permitted to handle. When a call arrives, a combination of ANI, DNIS, and/or IVR prompts is used to determine the primary skill needed. The call moves to a routing point, where a Genesys strategy routes the call using a target list that includes all agents with the required skill.

Typically, you use the Agent Skills feature within Framework's Configuration Layer to accomplish this. Alternatively, you can arrange agents into Agent Groups in the Configuration Layer that are defined by the skill required for group membership. An agent with three skills, for example, would be placed into three groups.

This setup is straightforward and CC Analyzer can capture both the routing statistics for each skill (calls offered, calls distributed, calls abandoned, and so forth) and agent-type statistics for each skill (talk time, calls put on hold, calls transferred, and so forth). But how do you use CC Analyzer to report by skill type when agents possess more than one skill?

This chapter explains how in the following sections:

Capturing Routing Performance by Skill Type

CC Analyzer can capture statistics on routing performance for the following object types:

- Route Point—Only those calls for a specific skill are routed through a given routing point; otherwise, the statistics for multiple skills would be combined and there would be no way to separate them.
- Virtual Queue—You can define a virtual queue in the Configuration Layer for each skill (for example, VQBilling, or VQSpanish), and then add the queue at selected points within one or more routing strategies. This method allows more than one routing point to reference a given virtual queue, and Stat Server will combine the results. As with the routing point restriction noted above, calls for multiple skills are not routed through a given virtual queue.

One way to implement skills-based reporting is to create agent skill groups in the Configuration Layer and then use the groups as routing targets whenever a certain skill is required. The statistics reported by CCPulse+ and CC Analyzer for these groups will reflect the calls that required those skills.

However, because statistics reflect the calls that went to the agents, regardless of the required skills used to route the call, statistical double counting across the groups may occur for agents with many skills. That is, if you add up the number of calls answered by each skill group, the sum may exceed the total number of calls answered by the contact center. Another way to look at this is that totals for the Billing group, for example, may include calls that were routed to agents in that group not because they handled billing questions, but because they also spoke Spanish. In effect, statistics for the Billing group reflect some of the Spanish calls! Perhaps sales and service calls, too.

Segregating Statistics of Multiskilled Agents

To avoid double counting, you must use attached data, define filters in the Configuration Layer, and create filtered statistics in CC Analyzer (or CCPulse+). For example, you could take the following approach:

1. Attach a KVP (key-value pair) named “`Skill`” in the routing strategies at the point where the required skill has been identified.
2. Create filters in the `Stat Server Application` object in the Configuration Manager for each of the possible skills. These filters might look like this:
   ```
   SpanishFilter = PairExists("Skill","Spanish")
   BillingFilter = PairExists("Skill","Billing")
   TechSupportFilter = PairExists("Skill","TechSupport")
   ```

3. Create a custom report layout containing all required filtered statistics for agent type in CC Analyzer such as:

```
SpanishInboundCallsAnswered, SpanishInboundTalkTime,
SpanishOutboundCallsDialed, SpanishOutboundTalkTime,
SpanishCallsHeld, SpanishHoldTime, SpanishACWTime,
SpanishCallsTransferred, SpanishCallsConferenced

BillingInboundCallsAnswered, BillingInboundTalkTime,
BillingOutboundCallsDialed, BillingOutboundTalkTime,
BillingCallsHeld, BillingHoldTime, BillingACWTime,
BillingCallsTransferred, BillingCallsConferenced

TechInboundCallsAnswered, TechInboundTalkTime,
TechOutboundCallsDialed, TechOutboundTalkTime,
TechCallsHeld, TechHoldTime, TechACWTime,
TechCallsTransferred, TechCallsConferenced
```

The statistics for agents with many skills are now accurately reflected for both Agent-level and Agent Group-level reporting. However, because the statistics are attached to a custom layout in the Data Mart, the CC Analyzer report templates can't use them. You must now define a structure on which to base a custom report.

Normalizing Statistics in the Data Mart

A statistics table for a custom layout in the Data Mart might have the irregular structure shown in Table 1.

Table 1: Sample of Irregular Table Structure

Time Key	Object	SpanishInbound CallsAnswered	BillingInbound CallsAnswered	TechInbound CallsAnswered	Etc.
Jan 3	BlueGrp	347	1310	562	
Jan 3	RedGrp	566	1799	822	

For tools such as Hyperion Query Designer to deal more effectively with this data, it must be normalized; that is, presented in a structure the program recognizes. While you can't change the Data Mart table structure, you can build views over those tables to present the data in a normalized format such as that shown in Table 2.

Table 2: Sample of Normalized Format

Time Key	Group	Skill	InboundCalls Answered	Inbound TalkTime	Outbound CallsDialed	Outbound TalkTime	Etc.
Jan 3	BlueGrp	Spanish	347	67405	127	41973	
Jan 3	BlueGrp	Billing	1310	129855	55	19747	
Jan 3	BlueGrp	Tech	562	289933	477	162101	
Jan 3	RedGrp	Spanish	566	104377	188	59200	
Jan 3	RedGrp	Billing	1799	188453	90	36109	
Jan 3	RedGrp	Tech	822	400944	700	196404	

Thus, the final step before generating reports is to create views that provide the report developer with a normalized structure, such as the following code developed using Microsoft SQL:

```
create view AgentSpanishDailyStats as
              select TimeKey TimeKey,
              Object GroupName,
              'Spanish' Skill,
              SpanishInboundCallsAnswered InboundCallsAnswered,
              SpanishInboundTalkTime InboundTalkTime,
              SpanishOutboundCallsDialed OutboundCallsDialed,
              SpanishOutboundTalkTime OutboundTalkTime,
              SpanishCallsHeld CallsHeld,
              SpanishHoldTime HoldTime,
              SpanishACWTime ACWTime,
              SpanishCallsTransferred CallsTransferred,
              SpanishCallsConferenced CallsConferenced
              from R_[template name]_[DAY|WEEK|MONTH|
              QUARTER|YEAR];

create view AgentBillingDailyStats as
              select TimeKey TimeKey,
              Object GroupName,
              'Billing' Skill,
              BillingInboundCallsAnswered InboundCallsAnswered,
              BillingInboundTalkTime InboundTalkTime,
              BillingOutboundCallsDialed OutboundCallsDialed,
              BillingOutboundTalkTime OutboundTalkTime,
              BillingCallsHeld CallsHeld,
              BillingHoldTime HoldTime,
              BillingACWTime ACWTime,
              BillingCallsTransferred CallsTransferred,
              BillingCallsConferenced CallsConferenced
              from R_[template name]_DAY;

create view AgentTechDailyStats as
              select TimeKey TimeKey,
              Object GroupName,
              'Tech' Skill,
              TechInboundCallsAnswered InboundCallsAnswered,
              TechInboundTalkTime InboundTalkTime,
              TechOutboundCallsDialed OutboundCallsDialed,
              TechOutboundTalkTime OutboundTalkTime,
              TechCallsHeld CallsHeld,
              TechHoldTime HoldTime,
              TechACWTime ACWTime,
              TechCallsTransferred CallsTransferred,
              TechCallsConferenced CallsConferenced
              from R_[template name]_DAY;
```

Finally, merge the results from the three skill-specific views into one:

```
create view AgentDailyStats as
              select * from AgentSpanishDailyStats
              union select * from AgentBillingDailyStats
              union select * from AgentTechDailyStats;
```

Now you can construct a custom report that supports all of the slice-and-dice capabilities of a product like Hyperion Query Designer, including pivot reports and drill-downs. Creating pie charts and stacked bar charts now requires a drag-and-drop operation.

Chapter

2 Partial-Period Reporting

When properly installed and using all default settings, the Data Collection and Data Mart Services for both CC Analyzer and CCPulse+ generate historical reports that are accurate to 48 hours. That is, the daily, weekly, and other reports that you produce today provide a true picture of contact center activity as of two days ago.

If you attempt to run these historical reports inside the two-day window using the default settings, you may end up with misleading and inaccurate results, particularly in the daily reports. How, then, do you configure the Data Collection and Data Mart Services to generate accurate historical reports within this two-day window?

Note: This chapter specifically addresses Historical Reporting functionality present in CC Analyzer and CCPulse+. It does not address real-time reporting functionality present in CCPulse+. Refer to *Reporting 7.5 CCPulse+ Help* or the *Reporting 7.5 CCPulse+ Administrator's Guide* for this information.

To achieve this end, you must understand how Historical Reporting components function and how you can affect the timing of their interaction by setting the appropriate runtime parameters. This chapter discusses these issues and provides examples in the following sections:

- Summary of Historical Reporting Functionality, page 18
- Examples, page 19

When this chapter talks about a Data Sourcer connected to a Stat Server application, it refers to a standard Data Sourcer (as opposed to Data Sourcer connected to the Internet Contact Server Database).

Summary of Historical Reporting Functionality

Data Sourcer gathers information from the Stat Server dedicated for Historical Reporting and writes to the Operational Data Storage (ODS) following each 15-minute interval (the default interval). You can shorten this interval by assigning a different time profile to your report layout or by redefining the `CollectorDefault` time profile, but you should analyze impact on performance first.

ETL Runtime checks ODS every five minutes. This, too, is a default interval. You can alter this figure using the `srcRefreshTimeout` runtime parameter located in the properties files used by the transformation module: `trans_only.properties` or `trans_and_agg.properties`. So, if you want ETL Runtime to pull information from ODS more frequently (say, every minute), set this parameter to `60` seconds, and then restart ETL Runtime.

After ETL Runtime detects new data chunks in ODS, using the Transformation mode, ETL Runtime writes them into corresponding minute-level tables of the Data Mart. You can configure several ETL Runtime parameters in the `.properties` files to tune performance and speed up data transference, adjusting the maximum number of open connections to the Data Mart (`maxNumberOfConnToDist` runtime parameter), the number of writers per folder/ view (`numberOfWriters` runtime parameter), and so forth, *if your RDBMS has enough resources and is tuned*.

Once data is written to the Data Mart, data is immediately available for reporting, but note that this data is *not* aggregated. You can generate a report that displays all available 15-minute intervals from the beginning of the day. And you can aggregate information at the report level or use the views defined on minute-level tables such as `AGG_BY_HOUR` and `AGG_BY_DAY`. This approach, however, may significantly slow down report generation.

Another approach is to run ETL Runtime's `Aggregation-only` (`agg_only`) mode more frequently (such as every half hour) than the default once an hour to create the seven default aggregation levels for each report. The reports you generate will accurately reflect contact center activity to the last time aggregation was performed. This mode will save most of your computer resources including CPU and disk space.

Still another approach is to run ETL Runtime's Transformation and Aggregation (`trans_and_agg`) mode, which aggregates data immediately following data transfer. Note, though, that this method consumes significant resources especially if the six `numOfIntervalsToWaitFor` runtime parameters in your `.properties` file are set inappropriately.

Note: Refer to the "Aggregation Levels" chapter in the *Reporting 7.2 ETL Runtime User's Guide* to learn how to set the `numOfIntervalsToWaitFor` parameters. Refer to the "Runtime Parameters" chapter in the same document for additional information about these and other ETL Runtime parameters.

Examples

Now consider the following three examples, which demonstrate how to generate partial-period reports:

- Setting a five-minute profile
- Obtaining partial-year aggregations
- Changing daily report output to 30-minute aggregations

Example 1 Let's say you have configured Data Sourcer to pull data from Stat Server using a five-minute time profile, and ETL Runtime pulls new data chunks from ODS every minute. Depending on the data you wish to capture, you could change the default setting of the `numofIntervalsToWaitFor_AGG_BY_HOUR` parameter from `5` [that's four 15-minute intervals plus one] in any of the following ways:

- To a value less than `13` [12 five-minute intervals plus 1] (a significant drain on resources as hour-level aggregation is performed on partial-hour data)
- To `13` so that hourly aggregation is performed when complete hourly data is obtained
- To a value greater than `13` where hour-level aggregation is performed after several hours have elapsed

Likewise, you should analyze the settings for the remaining five `numofIntervalsToWaitFor_` parameters to reflect report data you want when you want it.

Example 2 Let's say you don't want to wait for the current year to elapse to view the report data based on year-level aggregations. You might consider setting `numberOfIntervalsToWaitFor_AGG_BY_YEAR` to `1` to direct ETL Runtime to maintain aggregated, partial-year data for the months of the current year.

Report accuracy depends on how and when aggregation is performed.

Example 3 Perhaps you want to aggregate daily report output every 30 minutes. Before making that change in CC Analyzer, ensure that Stat Server, Data Sourcer, and ETL Runtime are *not* running. Now follow these steps:

1. Before running Data Sourcer for the first time, change the `CollectorDefault` time profile within Stat Server to `0:00+0:30`.
2. Start CC Analyzer's dedicated Stat Server.
3. Start Data Sourcer for the first time to initialize ODS and populate it with all necessary default tables, layout templates, and report layouts.

4. Open the Agent Daily report in the Genesys Report Designer and log in.
5. From the View menu, select Section/Catalog.
6. Locate Detail Level Query in the Section window and select it.
7. Right-click the T DETAILS window title bar and select Properties from the shortcut menu.
8. Change the Physical Name property from T_AGENT_HOUR to T_AGENT_NO_AGG, and then click OK.
9. Right-click the V DETAILS window title bar and select Properties from the shortcut menu.
10. Change the Physical Name property from V_AGENT_HOUR to V_AGENT_NO_AGG.
11. Select Agent Daily Report at the top of the Section window.
12. In the Report View window, scroll to the report's Details section and select the data in the first column of the table view. (Do not select the column header.)
13. With the entire column selected, right-click and select Number from the shortcut menu.
14. In the Category window, click time, and then select any time format that shows hours and minutes (for example, HH:MM or HH:mm am).
15. Repeat Steps 12 through 14 for any additional data columns.
16. Increase column width as appropriate for any data that displays as a series of number signs (#).
17. Save the report.

Chapter

3 Performance Measurements

In most circumstances, Stat Server is the defining component affecting both CC Analyzer and CCPulse+ performance. If your hardware has adequate disk space and memory, both products perform as documented. In the following sections, this chapter describes the performance expectations given a properly tuned Reporting environment:

Some performance parameters are also provided.

Stat Server Performance

You must install Stat Server for Data Sourcer on a computer with sufficient physical memory to avoid swapping. The physical memory, in megabytes, required for Stat Server to perform is estimated to be:

$$\mathit{MemoryReqd} = 100 + (\mathit{NStatistics} \text{ x } 0.0012)$$

where *NStatistics* represents the number of open statistics and 0.0012 refers to approximately 1.2 KB of memory per statistic.

For example, Stat Server on a box with 1.5 GB of memory should be more than ample to handle the requests of 30,000 active Agent or Place objects.

$$\begin{aligned} \mathit{NStatistics} &= 28 \text{ statistics/report layout x } 30{,}000 \text{ objects} \\ &= 840{,}000 \text{ statistics} \\ \mathit{MemoryReqd} &= 100 + (840{,}000 \text{ x } 0.0012) \\ &= 1{,}108 \text{ MB} \end{aligned}$$

For large Reporting environments, Genesys recommends that you install Stat Server and your RDBMS on different machines, so that the components do not compete for the same resources.

Data Sourcer Performance

- The elements defining Data Sourcer performance are:
- The number of statistics Data Sourcer requests from Stat Server.
- Whether the `use-prepared-statement` configuration option is turned on for Reporting environments using Oracle or DB2 ODSs.
- RDBMS performance.
- Network speed.

For Microsoft SQL, Sybase, DB2, and Oracle ODSs (where binding is turned off), the time, in seconds, required for Data Sourcer to write one block of data to ODS is roughly estimated at 375 seconds:

TimeReqd = 0.0025 seconds/statistic x *NStatistics*

A *block*, defined within the realm of Historical Reporting, is the data that Data Sourcer collects from Stat Server during a specific time interval. CC Analyzer uses a default time profile of 15 minutes; so a block represents the data collected every 15 minutes. Blocks vary in size depending on the number of active report layouts (and time profile). For purposes of describing Data Sourcer performance, Genesys has arbitrarily set one block equivalent to 150,000 statistics—the amount of data collected by 2,500 active agent (or place) report layouts; 2,500 queue report layouts; and 2,500 route point report layouts. [(2,500 x 28) + (2,500 x 11) + (2,500 x 11)]

The above assumes a conservative Oracle database insert rate of 300 inserts/second without binding. Insert rates differ between RDBMS types and hardware used and can vary with time, especially if there are other RDBMS users. For example, a Microsoft SQL RDBMS regularly writes at a rate of 700 inserts per second.

For an Oracle DBMS with binding enabled, performance significantly improves as indicated by the following formula:

TimeReqd = 0.0004 seconds/statistic x *NStatistics*

With binding, Data Sourcer can process one block of data in just about 60 seconds, nearly a fivefold improvement in performance.

Data Sourcer memory requirements depend on the size of its environment. For 32-bit platforms, use the following formula to calculate the minimum required memory, in megabytes:

19 + (*NRequests* x 22.4 x 10*e*-5) + 0.004 x *NLayouts*

where *NLayouts* is the number of active layouts and *NRequests* is the number of successful statistic requests from Data Sourcer to Stat Server. This is for normal work mode, which assumes fast and stable connections to two Stat Servers and DB Servers. Memory usage may increase slightly when Data Sourcer writes or reads data from the database. It may increase significantly if Data Sourcer is unable to write chunks to the database for a long time. In the latter case, memory usage will continuously increase by at least 0.001 x

NLayouts + *NRequests* x 22.4 x 10*e*-5 MB per *NLayouts* chunks. After the number of unsaved chunks reaches *NLayouts* x 3, memory usage will increase by approximately 0.001 MB per nonwritten chunk.

The preceding formulas are for 32-bit platforms. 64-bit platforms may require one-and-a-half to two times more memory.

ETL Runtime Performance

The elements affecting performance for ETL Runtime's Transformation module are as follows:

- The number of data sources defined to ETL Runtime
- The number of statistics
- Number of active report layouts
- Whether ODS and Data Mart reside on the same machine
- Network speed

Although deriving one simple formula to reflect ETL performance under differing circumstances is difficult, a linear relationship between the number of statistics and the time required to transform and/or aggregate them does exist. For example, *AverTimeReqd* = *R* x *NODSStatistics* where *NODSStatistics* = $\Sigma\, NStatistics_i$ for all ODSs (*i*) defined to ETL Runtime and *R* represents some constant factor determined by your specific environment that measures seconds per statistic.

Also, performance improves when ODS and Data Mart are installed on different disks or machines. Figure 1, based on tests from the Genesys lab, shows that ETL Runtime generally processes more statistics in less time when ODS and Data Mart are on different hosts.

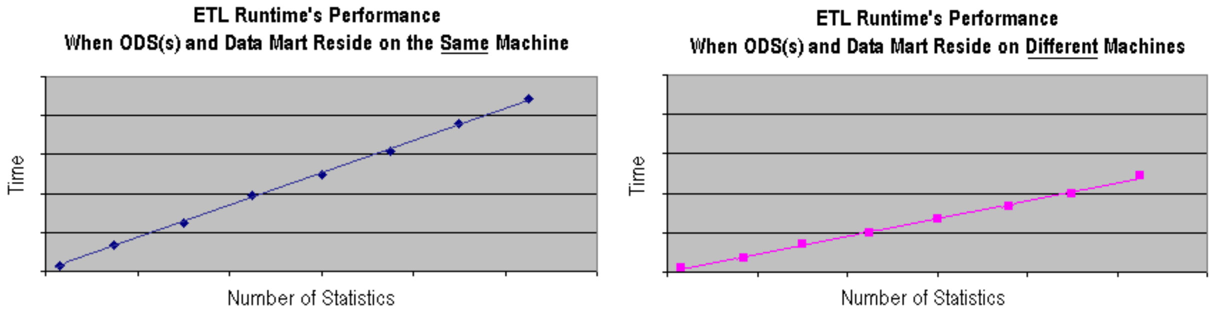


Figure 1: ETL Runtime Performance

Starting with the 7.0.2 release, the Genesys lab also verified a many-fold reduction in memory requirements for initializing the Data Mart, starting ETL Runtime, transforming chunks from ODS, and aggregating them within the Data Mart. Likewise, ETL Runtime performance improves this with release on a similar scale. In your environment, you should expect to notice a significant reduction, specifically, in the amount of time ETL Runtime expends to transform and aggregate data.

And, for Oracle RDBMSs, ETL Runtime 7.0.2 and forward releases require fewer open cursors than it previously required.

CCPulse+ Performance

The general factors affecting CCPulse+ performance are:

- Speed of your computer
- Number of agent and place objects being monitored
- Number of active views per workspace
- Aggregation level requested
- RDBMS speed
- Reporting interval (From/To, Last N)
- Number of objects
- Other processes running on the same box as CCPulse+
- Certain CCPulse+ configuration options

The act of invoking CCPulse+ itself merits special discussion as upon invocation (with the `CurrentStateEnable` configuration option turned on), CCPulse+ immediately sends requests to Stat Server to obtain the current status of all monitored Agent and Place objects within the workspace. Genesys recommends that you keep the number of statistical requests below the maximum number tested in the Genesys lab (see Figures 2 through 5).

To improve CCPulse+ performance, you can either manipulate the options introduced in the 7.0 and 7.2 releases (in particular, the `CurrentStateEnable` configuration option) or create several views in one workspace, instead of one view for the entire contact center, and activate only those views that are required.

Note: *Reporting 7.5 CCPulse+ Help* and the *Reporting 7.5 CCPulse+ Administrator's Guide* describe these features.

Factors Affecting Real-Time Performance

The number of views CCPulse+ must open in your workspace also plays a significant role in CCPulse+ performance. There is a GUI limitation on the number of views that can be opened within a workspace. Therefore, Genesys recommends that you create no more than 70 views in any given workspace.

In addition, with each workspace, Genesys has determined that there is a limit to the number of statistical requests CCPulse+ can make. Restrict your workspace to fewer real-time statistical requests than the maximum number shown in each of the real-time performance figures. Otherwise, above this maximum number, different facets of CCPulse+ start competing for Stat Server responses. CCPulse+ will still be in the process of opening the remaining views when, at the same time, it receives responses from Stat Server and refreshes the data (every 10 seconds, for most requests) according to the notification model and frequency specified.

Other factors that hamper performance if set inappropriately include:

- Notification frequency for statistics
- Whether statistical requests are time or change based
- The number of statistics per view
- The number of applied thresholds per view and entire workspace
- The number of open statistics

The tables and figures in the following sections depict the various dependencies affecting CCPulse+ real-time performance—particularly, the time required to load or save a workspace. The graphs illustrate linear and sometimes exponential relationships between a dependency and time.

All measurements were taken with the CCPulse+ application running on an Intel Pentium D, 3.0 GHz/2MB, 4 GB, Gigabit Ethernet, using Microsoft Windows. No performance differences were observed for other Windows operating systems and releases.

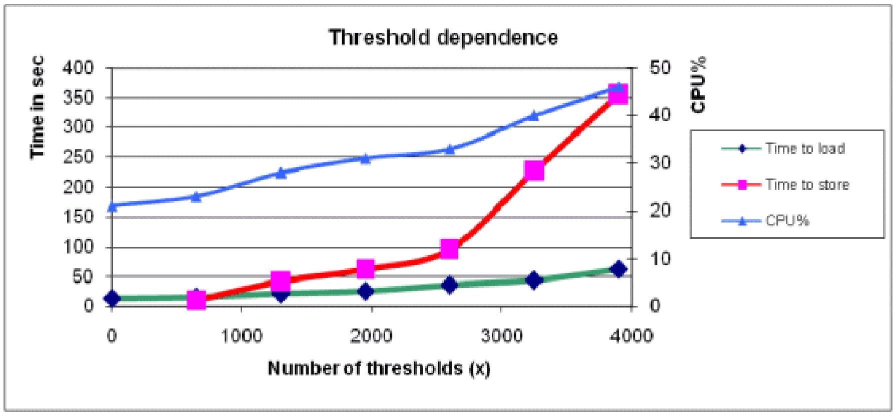
Number of Thresholds Dependency

With the following parameters fixed, Figure 2 shows a close to linear relationship between the time it takes to load a workspace and the number of thresholds:

- 4,551 statistics
- Workspace includes 5 views, 650 agents, and 6 statistics per agent
- 15-second notification frequency
- `CurrentStateEnable = True`
- `ExtendedCurrentStatus = True`

To improve the performance of CCPulse+ when thresholds are set, enable the `SafeScriptingMode` option.

Note: A large number of thresholds can significantly increase the time to save a workspace and cause the Stat Server to disconnect.



Number of Thresholds	Time to Load Workspace	Time to Save Workspace	CPU%
0	14		21
650	16	10	23
1300	22	42	28
1950	26	62	31
2600	36	96	33
3250	44	228	40

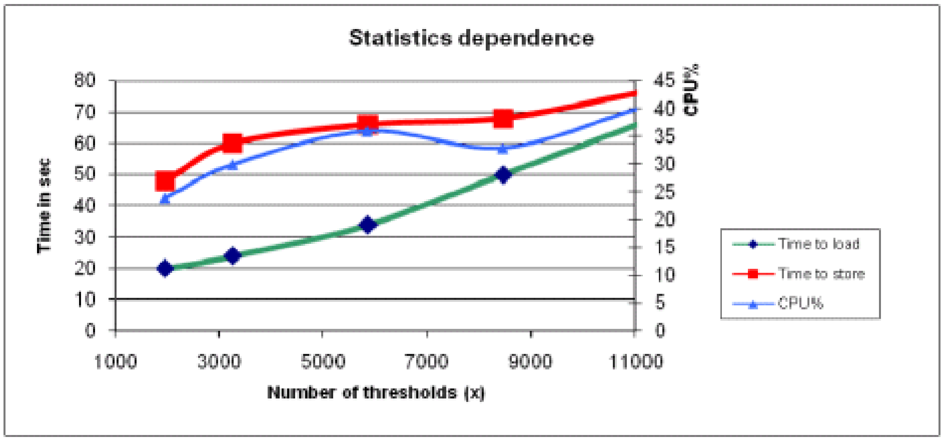
Figure 2: Real-time CCPulse+ performance has a threshold dependency

Number of Statistics Dependency

This test determined the number of statistics CCPulse+ can process before performance is compromised. The following parameters were chosen to realistically simulate a customer's environment:

- 1,950 thresholds
- 15-second notification frequency
- Workspace includes 5 views, 650 agents, and 3 thresholds per agent
- `CurrentStateEnable = True`
- `ExtendedCurrentStatus = True`
- Sorting is `OFF`

Figure 3 shows the relationship between the time it takes to load a workspace and the number of statistics. The figure also shows a linear relationship between the time it takes CCPulse+ to save a workspace and the number of statistics.



Number of Statistics	Time to Load Workspace	Time to Save Workspace	CPU%
1951	20	48	24
3251	24	60	30
5851	34	66	36
8451	50	68	33
11051	66	76	40

Figure 3: Real-time CCPulse+ performance has a statistic dependency

Notification Frequency Dependency

With the following parameters fixed in Figure 4, CCPulse+ shows a dependency between the amount of time required to load or save a workspace, and how often statistics are refreshed:

- 8,451 statistics
- 650 thresholds
- Workspace includes 650 objects, and 1 view

To improve CCPulse+ performance, set a different notification frequency for each template statistic. This will schedule Stat Server to update smaller blocks of statistics in staggered intervals, instead of updating all the statistics at the same time.

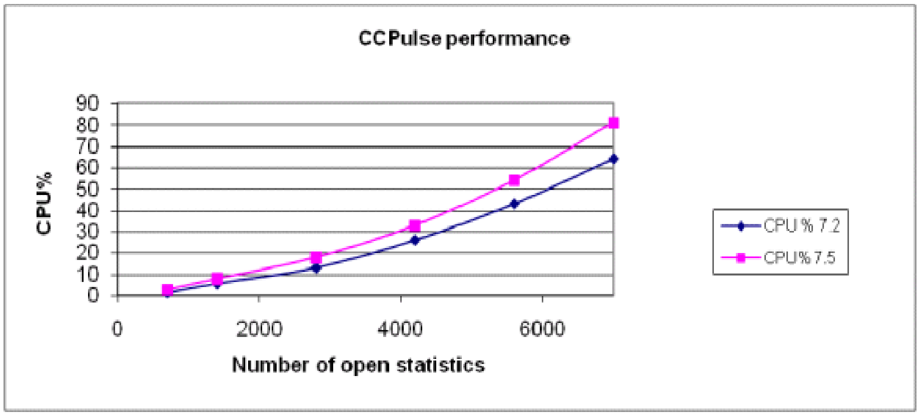
Notification Frequency (s)	Time to Load Workspace	Time to Save Workspace	CPU%
10	256	76	83
20	130	32	49
40	100	32	32
60	92	32	20

Figure 4: Real-Time CCPulse+ Performance–Dependent on Notification Frequency

Number of Objects In View Dependency

With the following parameters fixed in Figure 5, CCPulse+ shows a linear relationship between the time it takes to save a work-space and the number of objects in view:

- 1 current state statistic and 5 statistics per object
- 15-second notification frequency
- Workspace includes 1 view



Number of Objects	Number of Statistics	Time to Save Workspace	CPU%
100	700	4	3
200	1400	6	8
400	2800	10	18
600	4200	23	33
800	5600	42	54
1000	7000	90	81

Figure 5: Real-Time CCPulse+ Performance–Dependent on Number of Objects

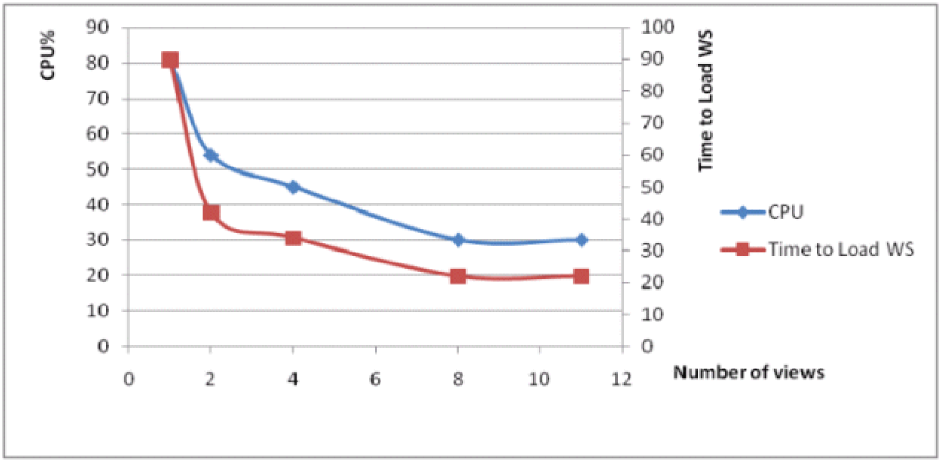
Note: CCPulse+ performance depends on the number of current state statistics requested in the views, and the value of the option, `CurrentStateRefreshInterval`. Increasing the value of this option may improve CCPulse+ performance.

Number of Views Dependency

With the following parameters fixed in Figure 6:

- 1 current state statistic and 5 statistics per object
- 15-second notification frequency
- Workspace includes 1 view
- 1000 agents

CCPulse+ shows a nonlinear relationship between the time it takes to save a work-space and the number of views.



Number of Views	Number of Statistics	Time to Save Workspace	CPU% f
1	7000	90	81
2	7000	42	54
4	7000	34	45
8	7000	22	30
11	7000	22	30

Figure 6: Real-Time CCPulse+ Performance–Dependent on Number of Views

Factors Affecting Historical Performance

You must specify three options to create an historical view:

- `Last N` intervals
- `From` constraint
- `To` constraint

This release optimizes the `Last N` option for performance. The challenge here was to be able to extract the last *N* records from the Data Mart without having the DBMS process the entire table(s), sort the data, and only then return the last *N* records. Of course, this issue could be resolved with a change to Data Mart schema but Genesys commitment to backward compatibility prevents such a change. Therefore, to minimize the response time for retrieving historical information, the CCPulse+ query to the Data Mart was optimized to extract just that portion of data falling within a time interval roughly equivalent to *N**2.

As a result, system response time meets expectations, but it comes at a cost. CCPulse+ reports no data in some cases where there actually is data in the Data Mart, but that data is older than *N**2 intervals.

Figures 7 and 8 provide measurements depicting the various dependencies affecting CCPulse+ historical performance. As with the real-time performance measurements, the graphs illustrate linear and sometimes exponential relationships between the dependency and time. All measurements were taken with:

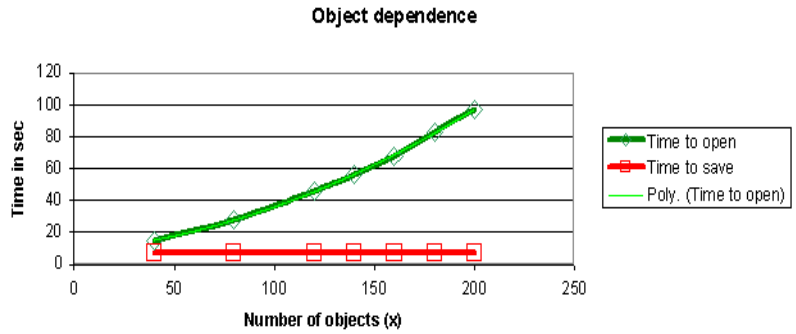
- The CCPulse+ application running on an Intel Pentium D, 3.0 GHz/2MB, 4 GB, Gigabit Ethernet, using Microsoft Windows. There were no performance differences observed for other Windows operating systems and releases.
- DB Server running on a Sun Enterprise 250 2 x UltraSPARC II, 400 MHz workstation with 1 GB memory using the SunOS 5.8 operating system platform.
- Data Mart using the Oracle 9.2 RDBMS on the same machine as DB Server.

Dependency on Number of Objects

With the following parameters in Figure 7 fixed, CCPulse+ reveals an exponential relationship between the time to open views and the number of objects in each view:

- `CurrentStateEnable = OFF.`
- 10 thresholds per view.
- Hourly aggregation over a 24-hour period.
- 13 statistics from the Agent template.

The figure also shows that CCPulse+ has a linear relationship between the time it takes to save views and the number of objects in each view.



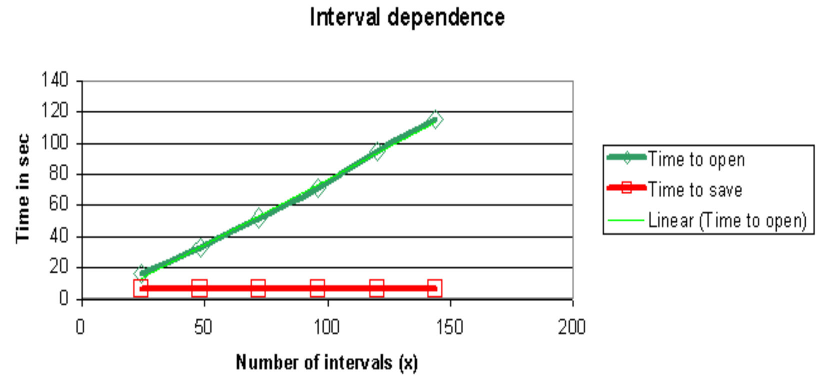
Number of Objects	Number of Statistics	Time to Open Views (s)	Time to Save Views (s)
40	520	15	7
80	1040	28	7
120	1560	46	7
140	1820	56	7
160	2080	68	7
180	2340	83	7
200	2600	97	7

Figure 7: Historical CCPulse+ Performance—Dependent on Number of Objects

Number of Time Intervals Dependency

With the following parameters fixed in Figure 8, CCPulse+ shows a linear relationship between the time to open/save views and the number of time intervals represented in those views:

- 10 thresholds per view
- Sorting turned off
- Hourly aggregation
- Genesys-provided Agent report using 50 agents
- The database contains over three months of data on 1,000 objects.



Number of Intervals	Time to Open Views (s)	Time to Save Views (s)
24	16	7
48	33	7
72	52	7
96	71	7
120	95	7
144	115	7

Figure 8: Historical CCPulse+ Performance–Dependent on Number of Intervals

Interval Granularity Dependency

CCPulse+ historical reporting performance depends on the granularity of the interval chosen for reporting. If you choose hourly report intervals rather than monthly, for example, you typically produce a substantial increase in the amount of data to be processed. Therefore, time to open and save views increases correspondingly.

Chapter

4 Daylight Saving Time and Other Time Shifts

There are well over 300 time zone variations across the globe, and they are continually changing to observe, among other things, daylight saving time (DST). These variations are stored in the `TIME_ZONE` table of the Data Mart. Given the erratic implementation of DST observances around the world, you should know that CC Analyzer and CCPulse+ perform well regardless. Their performance is not affected by:

- The time zone in which your agents are stationed.
- The time zone in which your server is located.
- Whether your time zone is subject to DST.
- The interval of time that is offset by DST. For example, some countries offset 15 minutes, some 45 minutes. Most offset one hour.
- The time of the DST offset (for example, 02:00:00 AM in the United States, 1:00:00 AM for most countries in western Europe).
- The date of the DST offset (usually based on a formula such as the second Sunday of March or the last Sunday in October).
- The offset frequency (for example, twice a year or not at all).

Your operating system maintains Greenwich Mean Time (GMT) and can convert it to a specified local time. The following Reporting components rely on your machine’s GMT key and its converted local time value:

- Stat Server
- Data Sourcer
- IS Data Sourcer
- Data Modeling Assistant (DMA)
- ETL Runtime

Report Generation Assistant and ETL Assistant also display time information. The values displayed, however, merely reflect the time zone noted in the

TIME_ZONE table of the Data Mart—values put there by ETL Runtime—so these components are not discussed further in this chapter. The time information reported in Genesys-provided reports stems from ETL Runtime, which uses the Java Runtime Environment to perform time conversions.

The Reporting products perform the necessary adjustments to recognize data and report on it when passing or manipulating data from one component to the next. In the following sections, this chapter describes how these four components behave during time shifts:

- How Stat Server Handles Time, page 36
- How Data Sourcer Handles Time, page 38
- How DMA Handles Time, page 39
- How ETL Runtime Handles Time, page 40
- Reporting Across Multiple Time Zones, page 41

The last section, "Reporting Across Multiple Time Zones" on page 41, presents four methods for generating consolidated reports in Reporting environments distributed over multiple sites.

Note: It is assumed that your system's GMT setting is accurate and synchronized among the servers in your environment.

How Stat Server Handles Time

For a reset-based notification mode, Stat Server references your system's local time setting when notifying clients based on a growing time profile (such as the Data Collection Services for CC Analyzer and CCPulse+). For example, Data Sourcer's default 15-minute time profile is interpreted as a set of fixed points (or moments) of local time (such as 0:00:00, 0:15:00, ... 23:45:00), upon which Stat Server sends notifications to Data Sourcer. On a regular day in the year where time does not shift to observe DST, Stat Server generates 96 points (24 x 4) in local time and sends 96 notifications to Data Sourcer (assuming the default time profile is used).

To correctly process DST changes, your operating system settings reflecting time zone and DST should be set correctly. Other than this, Stat Server does not require reconfiguration to handle DST changes.

How Stat Server Handles Clock Changes

Stat Server calculates the appearances of T-Server events (TEvents) and their duration independent of your system's time. So, statistical values are accurate and reflect the true sequence and durations of TEvents regardless of global or local time changes. However, client notification for reset-based statistics is dependent on local time. The following scenarios describe what happens when your system clock changes, for instance, to observe DST changes.

Scenario 1 In the scenario where your system's clock is moved forward, Stat Server searches for the nearest point in time in the future that satisfies Data Sourcer's time profile. Stat Server schedules its next notification at that point. If system clock changes are made outside the boundaries of the current 15-minute interval, Stat Server determines the interval's start time point to be the nearest time point in the past that satisfies the time profile.

For example, if the current system time is 10:07:00 and the system clock is moved forward by three hours and three minutes, the resulting time is 13:10:00. As soon as Stat Server detects this change, Stat Server sends notification to Data Sourcer based on 13:00:00 time point (occurring in the past) and schedules its next update at 13:15:00. The true duration between this notification and the last is seven minutes for the first notification (from 10:00:00 to 10:07:00) plus five minutes for the future notification (from 13:10:00 to 13:15:00).

Scenario 2 In the scenario where your system clock is moved backward, Stat Server tries to simulate monotonic time changes for Data Sourcer. That is, Stat Server schedules a notification for a point in time calculated before the time change (to prevent duplicated data chunks from being sent to Data Sourcer).

For example, if the current system time is 10:07:00 and the system clock is moved back 20 minutes, the resulting local time is 9:47:00. The next update to Data Sourcer, however, is scheduled for 10:15:00—not 10:00:00. The true duration between this notification and the last is 35 minutes—20 minutes for the first notification (just before the time changed) plus 15 minutes for the second notification (following the time change).

Genesys does not recommend changing the system time while Genesys software is running. Doing so could lead to results that are difficult to interpret (though they are accurate). But if system time-synchronization software does change the system clock periodically by some small amount (such as a few seconds), as of release 6.5, Stat Server accepts the change.

How Stat Server Handles DST Changes

In case of DST changes, your system's time does not change. (Remember, system time is recorded in GMT.) Rather, it is your system's converted local time that changes. And, because your system time does not change, notifications to Data Sourcer do not change either. Nonetheless, if today were:

- March 11, 2007—a day consisting of only 23 hours throughout most of the United States—Stat Server generates 92 time points (excluding the 4 time points in the omitted hour) and sends 92 notifications to Data Sourcer.
- November 4, 2007—a day consisting of 25 hours throughout most of the United States—Stat Server generates 100 time points (including 4 time points from the repeated hour) and sends 100 notifications to Data Sourcer.

In both cases, the duration of all data chunks sent to Data Sourcer is 15 minutes.

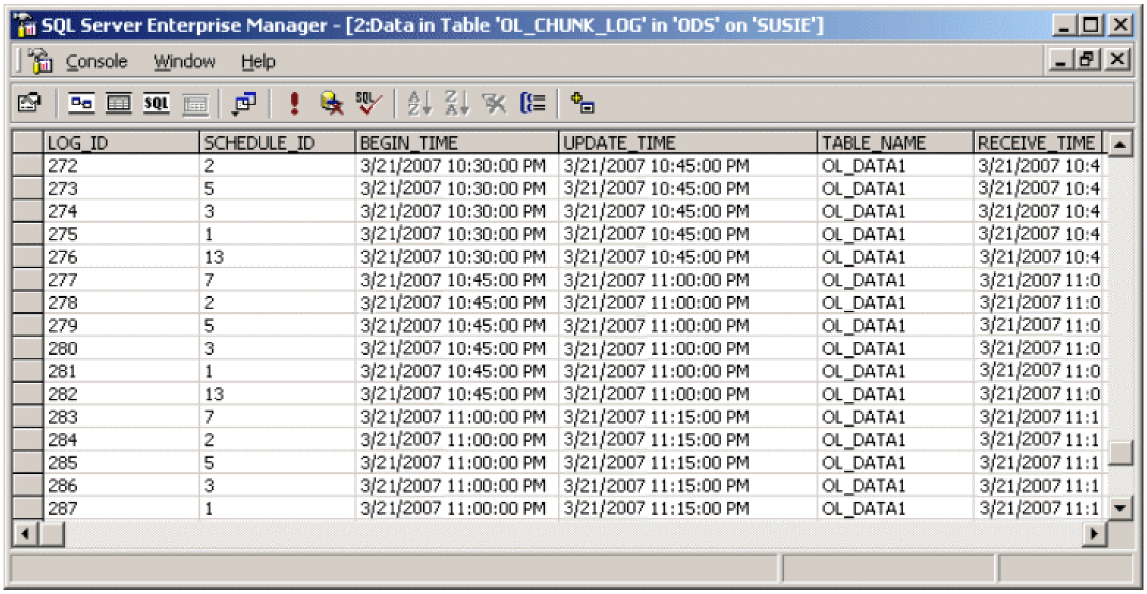
How Data Sourcer Handles Time

All time calculations involving Data Sourcer (and IS Data Sourcer) are based on GMT, including:

- The time values sent from Stat Server to ODS.
- The time values sent from the Contact Server Database (for the Internet Contact Solution) to ODS.
- The time values sent to the Data Mart from ODS.
- Any calculations that Data Sourcer performs on ODS data requiring a timestamp.

When you upgrade your version of Data Sourcer, the timestamp is recorded using GMT. When you migrate ODS data, the timestamp again is GMT. Inputs, outputs, and throughputs in Data Sourcer are all GMT based.

Figure 9 shows timing information in GMT for records Data Sourcer receives from Stat Server and writes to ODS.


SQL Server Enterprise Manager - [2:Data in Table 'OL_CHUNK_LOG' in 'ODS' on 'SUSIE']
Console
Window
Help

LOG_ID	SCHEDULE_ID	BEGIN_TIME	UPDATE_TIME	TABLE_NAME	RECEIVE_TIME
272	2	3/21/2007 10:30:00 PM	3/21/2007 10:45:00 PM	OL_DATA1	3/21/2007 10:4
273	5	3/21/2007 10:30:00 PM	3/21/2007 10:45:00 PM	OL_DATA1	3/21/2007 10:4
274	3	3/21/2007 10:30:00 PM	3/21/2007 10:45:00 PM	OL_DATA1	3/21/2007 10:4
275	1	3/21/2007 10:30:00 PM	3/21/2007 10:45:00 PM	OL_DATA1	3/21/2007 10:4
276	13	3/21/2007 10:30:00 PM	3/21/2007 10:45:00 PM	OL_DATA1	3/21/2007 10:4
277	7	3/21/2007 10:45:00 PM	3/21/2007 11:00:00 PM	OL_DATA1	3/21/2007 11:0
278	2	3/21/2007 10:45:00 PM	3/21/2007 11:00:00 PM	OL_DATA1	3/21/2007 11:0
279	5	3/21/2007 10:45:00 PM	3/21/2007 11:00:00 PM	OL_DATA1	3/21/2007 11:0
280	3	3/21/2007 10:45:00 PM	3/21/2007 11:00:00 PM	OL_DATA1	3/21/2007 11:0
281	1	3/21/2007 10:45:00 PM	3/21/2007 11:00:00 PM	OL_DATA1	3/21/2007 11:0
282	13	3/21/2007 10:45:00 PM	3/21/2007 11:00:00 PM	OL_DATA1	3/21/2007 11:0
283	7	3/21/2007 11:00:00 PM	3/21/2007 11:15:00 PM	OL_DATA1	3/21/2007 11:1
284	2	3/21/2007 11:00:00 PM	3/21/2007 11:15:00 PM	OL_DATA1	3/21/2007 11:1
285	5	3/21/2007 11:00:00 PM	3/21/2007 11:15:00 PM	OL_DATA1	3/21/2007 11:1
286	3	3/21/2007 11:00:00 PM	3/21/2007 11:15:00 PM	OL_DATA1	3/21/2007 11:1
287	1	3/21/2007 11:00:00 PM	3/21/2007 11:15:00 PM	OL_DATA1	3/21/2007 11:1

Figure 9: Records in the OL_CHUNK_LOG ODS Table in GMT

Because Data Sourcer performs all collections and manipulations in GMT, shifts in time have no affect on Data Sourcer.

These are some of the ODS tables that record time:

- OL_BREAKDOWN_CODE
- OL_CHUNK_ID
- OL_CHUNK_LOG
- OL_DB_PROPERTIES
- OL_OBJ_TO_REPORT
- OL_OBJECT
- OL_REPORT_LAYOUT
- OL_SCHEDULE
- OL_SOLUTION_TYPE
- OL_STATISTIC
- OL_TEMPLATE
- OL_TENANT

Note: Refer to the *Standard PDM Report* for your RDBMS-specific ODS for more information about these tables and their column descriptions. You can find this report in the `database` subfolder where Data Sourcer is installed. (File names are `ods_sybase.rtf`, `ods_mssql.rtf`, and so forth.)

How DMA Handles Time

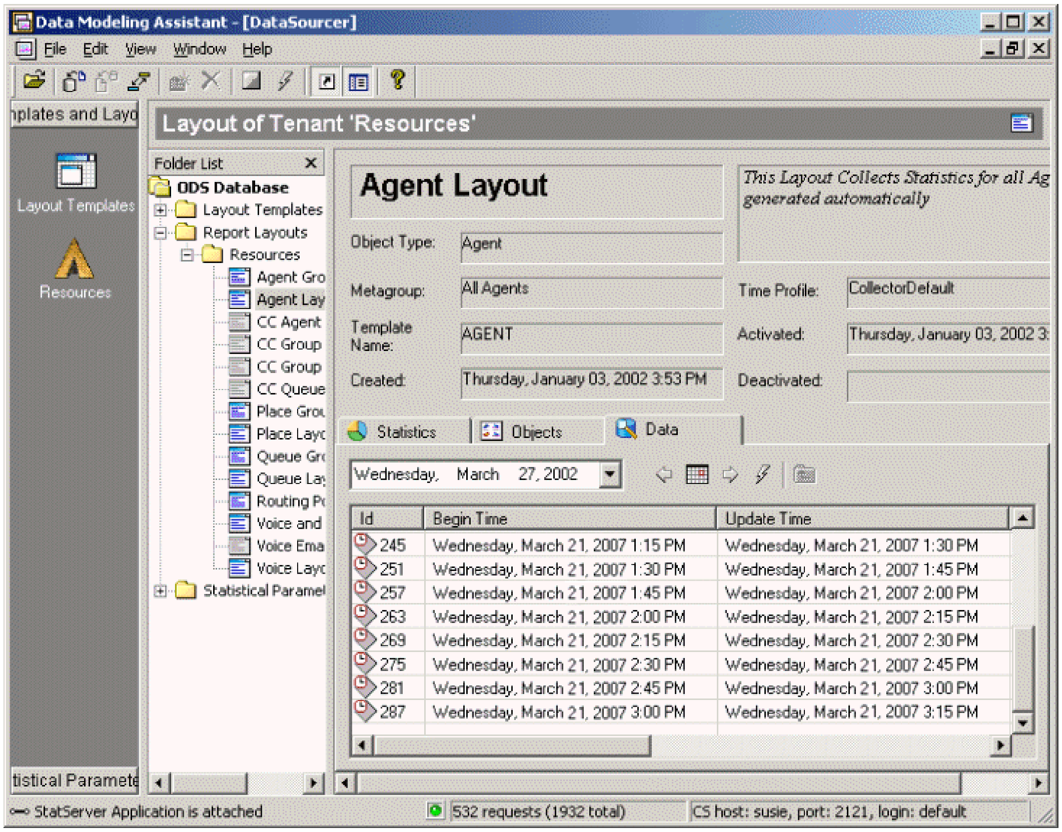
DMA translates GMT information stored in ODS to the local time of the environment from which DMA is started for *display purposes only*. Local time information is not written to ODS. DMA time-based windows show:

- When layout templates and report layouts are created.
- When report layouts are activated or deactivated.
- When layout templates and report layouts are marked for deletion.
- When Stat Server sends data chunks to ODS.

As DMA also connects to the Configuration Server, DMA displays Stat Server's time profile information, which is both configured and displayed using time relative to local time.

Figure 10 shows DMA displaying some of the same records depicted in Figure 9 on page 38. Compare IDs 275, 281, and 287 in both figures. Notice that Figure 10 shows begin times of 2:30 PM, 2:45 PM, and 3:00 PM, respectively for these IDs with end times 15 minutes later. These time values represent the local-equivalent time when Data Sourcer transferred data from Stat Server to ODS. Figure 9 shows the begin times for these IDs actually stored in ODS; namely, 10:45:00 PM, 11:00:00 PM, and 11:15:00 PM, respectively. These time values are expressed in GMT.

DMA does not display the affiliated time zone with local times, so during repeated time periods, as when the time is set back one hour, DMA displays two time periods' worth of data. And for the scenario where the clock is moved forward, DMA displays no activity at all for the skipped time period.



Layout of Tenant 'Resources'

Agent Layout

This Layout Collects Statistics for all Ag generated automatically

Object Type: Agent

Metagroup: All Agents

Time Profile: CollectorDefault

Template Name: AGENT

Activated: Thursday, January 03, 2002 3:

Created: Thursday, January 03, 2002 3:53 PM

Deactivated:

Wednesday, March 27, 2002

Id	Begin Time	Update Time
245	Wednesday, March 21, 2007 1:15 PM	Wednesday, March 21, 2007 1:30 PM
251	Wednesday, March 21, 2007 1:30 PM	Wednesday, March 21, 2007 1:45 PM
257	Wednesday, March 21, 2007 1:45 PM	Wednesday, March 21, 2007 2:00 PM
263	Wednesday, March 21, 2007 2:00 PM	Wednesday, March 21, 2007 2:15 PM
269	Wednesday, March 21, 2007 2:15 PM	Wednesday, March 21, 2007 2:30 PM
275	Wednesday, March 21, 2007 2:30 PM	Wednesday, March 21, 2007 2:45 PM
281	Wednesday, March 21, 2007 2:45 PM	Wednesday, March 21, 2007 3:00 PM
287	Wednesday, March 21, 2007 3:00 PM	Wednesday, March 21, 2007 3:15 PM

Figure 10: DMA Displays Local Time

How ETL Runtime Handles Time

ETL Runtime converts GMT information sent from Data Sourcer back to the local-equivalent time in the specified time zone for each report layout. This converted time information (as well as the original GMT values) is actually written to the Data Mart in various tables. Canned and custom reports, then, show activity that is not GMT based.

Local time is comprised of the local time and its time zone, so if there is ETL Runtime activity during the shift in DST, two values recorded at the same local time would hold different values in the Data Mart. For example, assume that two activities are recorded against a particular object in a Data Mart based in San Francisco, California—one exactly at 01:15:00 AM Pacific standard time (PST) on November 4, 2007, the other at 01:15:00 AM Pacific daylight time (PDT) on November 4, 2007. Data Mart recognizes the shift in DST, which, at 2:00:00 AM on November 4 sets the clock back one hour (hence the 1 o'clock to 2 o'clock hour repeats). The Data Mart would show the following time values in the appropriate fields:

- 1:15:00 AM, PDT
- 1:15:00 AM, PST

With regard to database maintenance, local time is used to determine when purge rules are activated and when purging actually started/ended; however, GMT time is used in the actual purge algorithm.

These are some of the Data Mart tables that record time:

- CHUNK_LOAD_ERR_LOG
- CHUNK_LOG
- FOLD_TO_COMP_STAT
- OBJ_TO_LAYOUT
- OBJ_TO_OBJ
- OBJECT
- PURGING_LOG
- PURGING_RULES
- REPORT_FOLDER
- REPORT_LAYOUT
- REPORT_TABLE
- REPORT_VIEW
- REP_REBUILD_LOG
- SOURCE

The names of the various time-related fields in these tables reveal whether they are GMT based; for example, `ADD_TIME_GMT`, `BEGIN_TIME_GMT`, and `DELETE_TIME_GMT`. `ADD_TIME`, `BEGIN_TIME`, and `DELETE_TIME`, on the other hand, store time data derived from local time.

Note: Refer to the *Reporting 7.2 Physical Data Model* document for your RDBMS-specific Data Mart for more information about these tables and their column descriptions.

Reporting Across Multiple Time Zones

The challenge in generating reports in multisite environments spanning more than one time zone is to present meaningful information in both local and consolidated context in a cost-effective and timely manner. Managers at local sites want the flexibility of generating reports presented in their own local time. Headquarter officers generally want comparative reports over the same time periods presented in one common time-zone format. There are more than a few ways to accomplish reporting across multiple time zones, including:

- Operating one central Data Mart.
- Accepting multiple zones in your reports.

You should evaluate the pros and cons of each before choosing which method to use.

One ODS/One Central Data Mart

The least resource-intensive solution involves customizing the Genesys-provided reports. One centrally located Data Mart has its data pulled from ODSs located at all of the different sites. Only one ODS per time zone is required in this scenario, although you may prefer to operate one ODS per site, which is fine.

Warning! Be sure to assign the correct time zone when defining ODSs to the Data Mart (using ETL Assistant). You cannot change this assignment after initial entry.

Thereafter, any report that any site runs on any object across any time zone will be displayed in that site's time zone.

To generate reports presented in a particular time zone (other than your own)—such as in GMT time—you must customize the selection criteria derived from time fields that are not GMT-based, in each of the desired canned reports, to hold the equivalent values of the time zone on which you wish to generate reports. That is, wherever search criteria is used in each of these reports (such as specifying the date range on which to generate the report), each non-GMT-based time field used in that search criteria would have to be converted to the appropriate time zone value. For instance, where search criteria are derived from `BEGIN_TIME`, you might substitute `NEW_TIME(BEGIN_TIME_GMT,'GMT','CST')` using the Oracle `NEW_TIME` function to convert `BEGIN_TIME` to its Central Standard Time equivalent. Criteria may involve any time-related fields for which GMT, generally, does not appear in the column name, such as `ADD_TIME`, `BEGIN_TIME`, `DELETE_TIME`, `END_TIME`, `OBJ_ADD_TIME`, and `OBJ_DEL_TIME`.

Multiple Time Zones in One Report

Lastly, if the consolidation you require is not at the time zone level, you could use the Genesys-provided report templates to generate reports showing data for all of your sites. One Agent report, for example, would yield results regarding all agents from all sites. You might want to customize these reports, though, to provide logical groupings based on the site object, so that you are not comparing apples to oranges. Search criteria that are based on local time zones would yield data specific to the time zone from which the report is run, and you would have to understand this in the reports you analyze. To use this solution, each site would only have to maintain its own and only one ODS. ETL Runtime would populate a centrally located Data Mart with data from each site ODS, and a site-based Data Mart would provide meaningful reports specific to each site.

Chapter

5 Database Backups

Database backups offset the financial and business ramifications that could result from hardware or system failure, human error, software failure, computer viruses, natural disasters, or theft.

This chapter describes the following:

Performing Backups

Genesys recommends that you have a database backup/recovery policy, which includes performing regular database backups. The following links contain useful information about database recovery, backup, and disaster recovery procedures and applications for the different database types that Genesys supports for Solution Reporting:

- IBM DB2— "Recovery Management Tools"
  `http://www-306.ibm.com/software/data/db2imstools/recovrep.html`
- Sybase—"What Backup, Recovery, and Disaster Recovery Mean to Your Adaptive Server Anywhere Database"
  `http://my.sybase.com/detail?id=47877`
- Oracle—"Oracle Backup and Recovery FAQ"
  `http://www.orafaq.com/faqdbabr.htm`
- Microsoft SQL—"Backing Up and Restoring Databases"
  `http://msdn2.microsoft.com/en-us/library/aa196685(SQL.80).aspx`

Genesys recommends that you run database backups regularly, but as a matter of policy, Genesys can neither endorse a specific commercial solution for this purpose nor recommend a third-party application. Genesys does recommend the following:

- Always having the database available for data transaction.

- Avoid stopping the database in the middle of data transactions to make a database backup.
- Using hot-backups or other tools to make database backups without affecting the availability of data.

Recovering from Backup

The best conditions for restoring data from backup exist when:

- You stop both of the affected Data Sourcer and ETL Runtime applications.
- You understand the structure of the Operational Data Storage (ODS) and Data Mart tables.
- The backup of the affected Data Mart and/or ODS is consistent.
- ODS and Data Mart are restored up to the same point in time.

Stopping Data Collection and Transformation

Regardless of which database (ODS and/or Data Mart) crashed, Genesys recommends that you stop data collection *and* data transformation during database recovery. To do this, either stop Data Sourcer, or stop the supporting Stat Server along with ETL Runtime and any of its scheduled processes that might start during recovery. Failure to stop these applications might contribute to loss of data and database inconsistencies.

Reviewing Database Schema

Some recovery procedures require that you understand ODS and/or Data Mart structure. Refer to the physical data models (PDMs) published for your relational database management system (RDBMS). ODS PDMs are deployed to your Data Sourcer installation directory in the `database` subfolder. The files follow the `ods_`*`RDBMS`*`.rtf` naming convention. Data Mart PDMs can be found on the Documentation Library DVD or on the Technical Support website. These files are named *Reporting 7.2 Physical Data Model for an Oracle Database,* and so forth. Also, consider contacting Genesys Technical Support for assistance.

Maintaining Database Consistency

An ODS backup should reflect a snapshot of all its tables at exactly the same moment; likewise, a Data Mart backup should reflect a snapshot of its tables at exactly the same moment. Ideally, both snapshots should be captured at, or very close to, the same time. The strategy for achieving this consistency is dependent on the RDBMS. It is the responsibility of the database administrator to work out a strategy for maintaining database consistency.

Handling Different Restoration Points

Hardware or software failures might prompt you to restore ODS and/or Data Mart from backup. Although it is unlikely that your ODS and Data Mart would both crash, and that they would crash at the same time, recovery is best achieved when *both* databases are restored from snapshots that were taken at the same in the past.

A more likely scenario occurs when one database crashes, but the other does not. In this situation, you can still effectively deploy a restored backup in your Reporting environment with some manual effort. The process centers around knowing both of the following:

- What is the largest chunk ID in each database:
  - In Data Mart, you will find the largest chunk ID in the `CHUNK_ID` column of the `CHUNK_LOG` table.
  - In ODS, `OL_CHUNK_LOG.LOG_ID` holds this value.
- What is the comparative relation of these largest chunk IDs to one other. This comparative relation can only be one of the following:
  - The largest chunk ID in ODS is the same value as the largest chunk ID in Data Mart.
  - The largest chunk ID in ODS is greater than the largest chunk ID in Data Mart.
  - The largest chunk ID in ODS is less than the largest chunk ID in Data Mart

The following sections discuss each of these relations in turn.

CHUNK_ID[ODS] = CHUNK_ID[DM]

If the chunk IDs in both databases are equivalent, no extra effort is required on your part. Restore the database that crashed by using normal recovery procedures for your RDBMS. After data recovery, restart Data Sourcer and ETL Runtime and verify that both applications operate correctly by checking for errors in the logs.

CHUNK_ID[ODS] > CHUNK_ID[DM]

The likely scenario, if the largest chunk ID in ODS is greater than the largest chunk ID in Data Mart, is that Data Mart crashed, and ODS did not. You can use the normal recovery procedures for your RDBMS to restore the downed database.

However, before you start either application (Data Sourcer or ETL Runtime), you should check ODS to determine whether records were marked as transferred during the “limbo” period (the time between backup and crash). These records will need to be transferred to the Data Mart again. Focus on those records in the `OL_CHUNK_LOG` table that have chunk IDs larger than the largest Data Mart chunk ID, and that have transferred times later than the time of the last backup; then, manually change the record to indicate that it has not been transferred. For example:

```
UPDATE OL_CHUNK_LOG (...)
     SET TRANSFERRED_TIME = NULL WHERE LOG_ID = 342;
```

In the unfortunate event these records were already purged from ODS (such as when the `dropTransferredTables` configuration option has been set to `true`), you will not be able to recover such data.

Upon restart, ETL Runtime re-transfers from ODS the data that it transferred immediately before the crash. Given this scenario and a consistent database, you should lose no data.

CHUNK_ID[ODS] < CHUNK_ID[DM]

The likely scenario if the largest chunk ID in ODS is less than the largest chunk ID in Data Mart is that ODS crashed and Data Mart did not. You must manually correct this disparity in chunk IDs to avoid problems.

Presumably, Data Mart has already processed ODS data that was collected after the point of restoration and before database crash. However, as Data Sourcer collects new data after successful recovery and restart, it will re-use the same chunk IDs that it assigned during that limbo period. ETL Runtime, having already processed records with those chunk IDs, will discard the records. Furthermore, if report layouts were defined in ODS since the last backup, which are already present in Data Mart, inconsistency will result.

To avoid this loss of data during ODS “catch-up,” before you start Data Sourcer, roll ODS chunk IDs forward to a number that is larger than the largest chunk ID in Data Mart. To do this, insert one fake record into the `OL_CHUNK_LOG table` in ODS—for example:

```
INSERT INTO OL_CHUNK_LOG (LOG_ID, ..., TABLE_NAME, TRANSFERRED_TIME, ...)
      VALUES (342, ..., 'OL_DATA35', getdate(), ...)
```

where:

- `LOG_ID` should be greater than that in the Data Mart.
- `TABLE_NAME` should match the latest `OL_DATA` table in ODS.
- `TRANSFERRED_TIME` should not be `NULL`.

Upon Data Sourcer restart, collection will resume, and numbering of data chunks will resume from the fake ID that you inserted.

Using Emergency Save Files

You should decided on a case-by-base basis whether to use Data Sourcer’s emergency save file(s). The set of configuration objects, as well as the set of statistics, might have changed after ODS crash. Therefore, using the emergency save file(s) might introduce inconsistencies in ODS.

Index