GENESYS
AN ALCATEL-LUCENT COMPANY

Framework 8.0

Stat Server

User's Guide

About Genesys

Genesys Telecommunications Laboratories, Inc., a subsidiary of Alcatel-Lucent, is 100% focused on software for contact 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 regional numbers provided on page 9. For complete contact information and procedures, refer to the ***Genesys Technical Support Guide***.

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.genesyslab.com

Document Version: 80fr_us-statserver_11-2009_v8.0.001.00

Table of Contents

Preface

Welcome to the *Framework 8.0 Stat Server User's Guide*. This document introduces you to the concepts, terminology, and procedures relevant to this Genesys server and is valid only for the 8.0 release(s) of Stat Server.

Note: For versions 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 guide, identifies the primary audience, introduces document conventions, and lists related reference information:

- Intended Audience, page 7
- Chapter Summaries, page 8
- Making Comments on This Document, page 8
- Contacting Genesys Technical Support, page 9

Intended Audience

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

- Computer-telephony integration (CTI) concepts, processes, terminology, and applications.
- Network design and operation.
- Your own network configurations.
- Basic Microsoft Windows and/or UNIX concepts.

You should also be familiar with:

- Genesys Framework architecture and functions.
- Tasks and functions of the Genesys solution with which you are using Stat Server.

- The Genesys T-Server model, the Genesys Interaction Model, and the specific T-Server/Interaction Server that you use in your environment.

Chapter Summaries

In addition to this preface, this guide contains the following chapters and two appendixes:

- Chapter 1, “Introduction,” on page 11, explains the main Stat Server functions and describes Stat Server architecture.
- Chapter 2, “Statistic Configuration Options,” on page 19, contains descriptions of configuration options for statistics.
- Chapter 3, “Stat Server Object Types,” on page 49, describes the Configuration Server objects that Stat Server monitors.
- Chapter 4, “Stat Server Actions,” on page 57, explains what an action is in Stat Server terms, and how actions are classified and defined.
- Chapter 5, “Object Statuses,” on page 125, explains what a status is in Stat Server terms, and how statuses are classified, defined, and determined.
- Chapter 6, “Statistical Categories,” on page 137, introduces statistical categories and explains how statistics in each category are calculated.
- Chapter 7, “Statistical Subjects,” on page 165, describes the perspectives from which Stat Server can view an interaction.
- Chapter 8, “Stat Server Timestamps,” on page 169, describes the algorithm Stat Server uses to compute the durations of actions using timestamps provided by an event-supplying server.
- Chapter 9, “Campaign Statistics,” on page 175, introduces statistics that can be calculated for the Outbound Contact Solution.
- Chapter 10, “Custom Formulas,” on page 181, explains how custom-value statistics are defined and calculated.
- Chapter 11, “Virtual Agent Groups,” on page 185, describes the types of virtual agent group definitions that Stat Server supports and how to configure them in Configuration Server.
- Appendix, “Predefined Statistical Types” on page 189, lists the predefined statistical types in the Stat Server configuration.

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@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 scope of this document only and to the way in which the information is presented. Contact your Genesys Account Representative or 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.

Contacting Genesys Technical Support

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

Region	Telephone	E-Mail
North America and Latin America	+888-369-5555 (toll-free) +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
Malaysia	1-800-814-472 (toll-free) +61-7-3368-6868	support@genesyslab.com.au
India	1-800-407-436379 (toll-free) +91-(022)-3918-0537	support@genesyslab.com.au
Japan	+81-3-6361-8950	support@genesyslab.co.jp
Before contacting technical support, refer to the *Genesys Technical Support Guide* for complete contact information and procedures.		

Chapter

1 Introduction

In the following sections, this chapter explains Stat Server's primary functions, describes its architecture, and highlights the features introduced in the 8.0 release.

What Is Stat Server?

Stat Server tracks information about customer interaction networks (contact center, enterprise-wide, or multi-enterprise telephony and computer networks). Stat Server also converts the data accumulated for directory numbers (DNs), agents, agent groups, and non-telephony-specific object types, such as e-mail and chat sessions, into statistically useful information, and passes these calculations to other software applications that request data. For example, Stat Server sends data to Universal Routing Server (URS) to inform the URS about agent availability. You can also use Stat Server's numerical statistical values as routing criteria.

Stat Server provides contact center managers with a wide range of information, allowing organizations to maximize the efficiency and flexibility of customer interaction networks.

Monitoring Contact Centers

Stat Server tracks what is happening at any DN—whether it belongs to an agent station, an individual agent who moves between stations, an interactive voice response (IVR), or a point in a private branch exchange (PBX) used for queuing or routing.

For example, for each DN, Stat Server tracks DN activity, call activity on a DN, and other relevant derived states, such as how long a phone is not in use, how long a call is on hold, how long it takes to dial a call, how long a DN is busy with an inbound call, and so forth.

From the gathered information, Stat Server performs a variety of statistical calculations to provide its clients:

- Duration in current state
- Total number of times each state occurs
- Cumulative time for each state
- Percentage of time for each state
- Average time for each state
- Maximum and minimum times for each state

For a queue or routing point, Stat Server can track the following data:

- Number of currently waiting calls
- Cumulative waiting time of queued calls
- Average waiting time of queued calls
- Maximum and minimum waiting time of queued calls
- Information on the outcome of calls after they have been distributed from the queue

Multimedia Support

To support the distribution strategies provided in the Genesys Multimedia Solution (previously called Multi-Channel Routing), in the 7.0 release, Stat Server architecture began to utilize two new statistical object types, `StagingArea` and `Strategy`. (See page 49 for a description of these object types.) Another feature introduced during the 7.*x* release was the *Genesys Resource Capacity Model* which reflects an agent's ability to handle multiple, simultaneous interactions of differing media types on both single-media and multimedia DNs. You can configure agent ability in Configuration Manager using the Resource Capacity Wizard to create capacity rule scripts. The *Genesys 8.0 Resource Capacity Planning Guide* describes this model and how to use it.

Stat Server Features

Dynamic Agent Tracking

Stat Server dynamically tracks customer service representatives as they occupy different places in a business environment. Each agent is identified by an ID, and regardless of the agent's location, Stat Server can track that agent's activity based on this ID.

Multi-Site Monitoring

Stat Server can monitor more than one T-Server and, therefore, more than one PBX switch. Even if you use different kinds of switches, Stat Server tracks what happens with all calls delivered to these switches, providing statistical information for different sites simultaneously.

Java Functionality

Starting with release 7.0, Stat Server architecture has been extended to include support for pluggable statistical modules written in Java. This added flexibility enables you to dynamically extend Stat Server functionality with new statistical types (residing in Stat Server’s Java Extensions [SSJE]) and to have Stat Server supply them to Genesys applications. The *Framework 8.0 Stat Server Deployment Guide* describes how to enable Java functionality in your Stat Server applications.

Stuck Call Recognition

Stat Server distinguishes stuck calls from those calls that are abandoned for reasons not related to the synchronization of Genesys software. A *stuck* call within the Genesys realm always involves a missynchronization between two or more interdependent contact center components (such as T-Server and the switch, Stat Server and T-Server, or the Genesys Router and Stat Server).

Many improvements were made within the 7.*x* releases of T-Server for better detection and clearing of stuck connection IDs. As a result:

- For regular queues, T-Server now distributes an abandoned or released TEvent, coupled with an `AttributeReliability` attribute other than `TReliabilityOk`, to its clients upon detecting a stuck call. When determining object actions and statuses, Stat Server considers such events (`EventAbandoned`/`EventReleased` with `AttributeReliability != TReliabilityOk`) for the termination of all call-related, durable actions.
- For virtual queues, starting with the URS 7.5 release, T-Server now distributes the `EventReserved_2` TEvent, which is generated by Universal Routing Server on behalf of virtual queue objects and received by Stat Server as confirmation that a call still resides at the virtual queue. Stat Server detects and removes stuck calls at the virtual queue when Stat Server does not receive the expected `EventReserved_2` event during the time frame indicated by the `call_kpl_time` Universal Routing configuration option. Stat Server interprets not receiving this event within the specified interval as *the call is no longer at the virtual queue and should be deleted from Stat Server memory.* To learn more about this functionality, refer to the description of the `call_kpl_time` configuration

option in the *Universal Routing 8.0 Reference Manual* and the check-vq-stuck-calls-frequency configuration option in the *Stat Server Deployment Guide*.

Tracking Virtual Queue Interactions in Multi-Site Scenarios

Improvements in the 7.*x* releases of Universal Routing Server (URS) enable Stat Server to more accurately track interactions that are distributed by virtual queue objects across different sites and calculate call-related statistics for them. Stat Server reads the TransferConnid attribute of attached data, which URS 7.6 attaches to the TEvent of the original call, and Stat Server uses this information to match the transferred or conferenced call to the original call. CallAnswered, CallMissed, CallReleased, and other retrospective, interaction-related actions that reflect regular DNs (page 104) now more accurately account for count and duration metrics in multi-site scenarios.

In addition, Stat Server now considers and relies on the value of the ThirdPartyDN attribute in EventDiverted TEvents from URS to determine the location to which calls were diverted from a virtual queue (page 105).

Network Attended Transfers

Stat Server 7.1 and later releases support network-attended transfers and conferences in much the same way as it supports two-step transfers and conferences handled by premise T-Server applications. Stat Server now monitors call operations (alternate, reconnect, network attended transfer, network attended conference) and generates corresponding call-related actions and statuses for Regular DN objects. Stat Server does not support monitoring of Mediation DN objects (such as ACD queues) in network-attended call scenarios.

New in This Release

Stat Server release 8.0 introduces a set of new features, highlighted in this section. Refer to the *Framework 8.0 Stat Server Deployment Guide* for a discussion of the new configuration options that are available in the [statserver], [common], and [log...] configuration sections.

Filter Improvements

In the 8.0 release, Stat Server's filtering capabilities have been improved:

- To enable filtering of attached data based on an interaction's Extensions attribute only. This feature is described on page 31.

- To recognize additional operators in values specified for filters using the Reason property. Previously, Stat Server recognized only the = and != operators for this property. This feature is described on page 33.

Better Synchronization with Historical Reporting Metrics

In the 8.0 release, Stat Server uses the time that Genesys servers transmit events rather than the time that Stat Server receives the events to calculate the durations of qualifying metrics. Network delays no longer cause CC Analyzer (which provides historical snapshots of real-time data furnished by Stat Server) to provide different results for comparable measures that are produced by Interaction Concentrator (and downstream Genesys applications), which already use source timestamps. Refer to the description of the UseSourceTimeStamps configuration option on page 41 to learn how to update the stat type definitions of your metrics in order to implement this feature.

Stat Server Recognizes Actions Generated for Outbound Contact Centers

Stat Server now recognizes a new event flow for generating ASM_Engaged and ASM_Outbound actions on agent DNs (as well as on communication port DNs) to support interactions in an Outbound Contact environment where the Outbound Contact Server attempts to engage agents to participate in interactions. The actions are described beginning on page 78.

Other New Features

- Stat Server now considers the value of the ThirdPartyDN attribute of the EventDiverted TEvent before generating CallAnswered actions on virtual queue objects under certain circumstances. See page 107 for details.
- Stat Server now generates CallReleased actions for virtual queue objects that were configured on a Multimedia switch. This feature enables Stat Server to provide handle-related metrics, such as [Chat Handle Time], of interactions that are processed by Multimedia’s Interaction Server. Refer to the description of the CallReleased action on page 106.
- Stat Server now generates the DialWrongNumber and LeadWrongNumber actions for Outbound campaigns and calling lists when notified of failed processing due to a wrong number. Refer to the complete list of campaign-related actions beginning on page 177.

Stat Server Architecture

Stat Server supplies statistical information to client applications such as Universal Routing Server (URS), Data Sourcer, and CCPulse+ (formerly known as Call Center Pulse or CC Pulse), upon request. URS, for example, can request information through user-designed strategies.

Stat Server is also a client of T-Server, which is essentially a translator or interpreter that mediates between a PBX switch and other Genesys software products. T-Server sends Stat Server information that is received from the PBX about what happens to each call or telephony object in the enterprise’s telephone network. Stat Server then acts as a server by interpreting T-Server’s information and providing it to other Genesys products.

Starting with release 7.0, Stat Server is also a client of Interaction Server, which is a component of Genesys Multi-Channel Routing. Figure 1 shows how Stat Server performs in an actual environment. The dashed entities are optional.

When a call comes into the PBX, it may be sent to any of the following:

- An internal PBX point for queuing or routing
- An Interactive Voice Response (IVR)
- An agent’s DN

This distribution decision may result from the PBX’s own algorithms, which distribute calls based on agent availability.

Regarding Stat Server 7.0.2 and forward releases, when an Internet interaction enters an e-mail, chat, or Web callback server, the interaction is routed through Interaction Server for processing, before it is sent on to Stat Server. Stat Server monitors these various interactions—tracking their status at any given moment, as well as historically over time.

Regarding Stat Server 7.0.1 and prior releases, when an Internet interaction enters an e-mail, chat, or Web callback server, MS T-Server emulates telephony events, so that Stat Server behavior can be described in PBX terms.

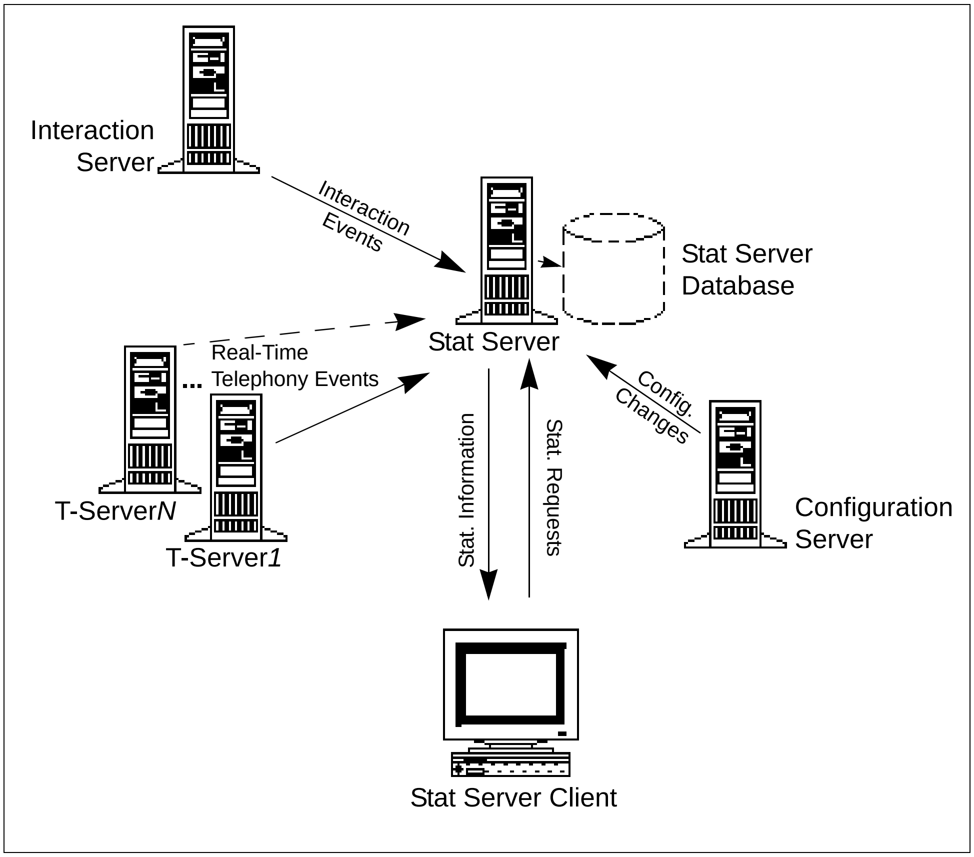


Figure 1: Stat Server Architecture

For example, a small contact center may create two different types of agent groups. One group handles calls in the main office and is defined by that location. Another group provides technical support; agents of this group work in different locations. In addition, a free-floating agent works at different stations; s/he can receive calls at any place that s/he logs in to.

In this example, Stat Server simultaneously monitors each place and prepares statistical information for configured agent groups and individual agents identified by a unique employee ID no matter where they are located. Stat Server re-creates what an individual agent is doing, based on the state of his or her media devices, and factors that agent’s work into the overall calculation for the entire group.

When the PBX routes a call to any of these agents, it also sends T-Server a message identifying the DN to which the call has been delivered. T-Server conveys that information to Stat Server and informs Stat Server whenever a change occurs in the condition of that DN. Stat Server updates information about each agent group, each agent in a group, and all individual agents.

`Note:` Statistical tools that switch vendors provide may take other approaches to statistical calculations than the approach implemented in Stat Server. Those tools may use different object types, different call definitions, and so forth. As a result, statistics that these tools generate may differ significantly from Stat Server statistics.

You use the Framework Configuration Layer to manage all configuration changes to the enterprise and its agents, groups, applications, and so on, and to notify all applications of the current contact center environment.

Persistent Statistics

The term *persistence* means that a statistic, once requested by a client, continues to be calculated even after the client closes. Stat Server treats all requested statistics as persistent. When a statistic that is not already available on the server side is requested, it is automatically added to the `Persistent Statistic Pool`. Stat Server continues to calculate the statistic even after the requesting client closes it. When the client reopens the request for this statistic, the `Persistent Statistic Pool` resends it with accurate values to the client. By default, a statistic becomes obsolete and is removed from this pool three days after a client last requested it.

You can configure Stat Server to save statistics periodically to disk using the `auto-backup-interval` configuration option. If Stat Server terminates for any reason, it initializes statistics in the `Persistent Statistic Pool` when restarted, but it does *not* restore their previous values.

Chapter

2 Statistic Configuration Options

In the Genesys Statistical Model, you define the metrics that Stat Server should collect for its clients within the Stat Server application itself. This chapter describes the options you can use to configure these metrics. To learn about the options that you can use to configure other aspects of the Stat Server application (in the `[statserver]`, `[log]`, `[common]`, and `[java...]` sections), refer to the "Fine-Tuning Stat Server Configuration" chapter in the *Framework 8.0 Stat Server Deployment Guide*.

A metric is defined by the values of configuration options, described in the following sections:

The Genesys Statistical Model is described in the *Reporting Technical Reference Guide*.

TimeProfiles Section

The `TimeProfiles` section defines the time intervals that Stat Server references for calculating historical, aggregate values for statistics. This section must be named `TimeProfiles` within the Stat Server `Application` object. Stat Server clients, such as CCPulse+, specify which defined time profile to use when they request statistics. The following information describes how to define time profiles. Table 1 lists the one option that is applicable to this section.

Table 1: Configuration Option for the TimeProfiles Section

Option	Description
<TimeProfileName>,<Type>	Defines the time interval over which a historical aggregate value is calculated. The option name must consist of two entries separated by a comma: `<TimeProfileName>` represents any string that names the time profile, and `<Type>` represents the time interval type, which includes one of the following: • `Sliding` • `Growing` • `Selection` • `SinceLogin` With the exception of `SinceLogin,` you must specify values for each interval type. The following subsections describe permissible values. Stat Server uses a special time profile, called `Default,` when Stat Server's client does not specify a time profile when requesting statistics. `Default` uses a `Growing` interval type and resets statistics to zero (`0`) every night at midnight. To override the reset time of this inherent time profile, you must add a `Default` time profile to the `TimeProfiles` section and redefine it as desired. See an example of this on page 22. Default Value: No default value Valid Values: Dependent on interval type. (See the following subsections.) Changes Take Effect: When Stat Server restarts

Stat Server projects actions and statuses onto time intervals (except for the `TotalAdjustedTime` and `TotalAdjustedNumber` statistical categories [described in Chapter 6]) on any time profile, as follows:

- Status duration time for a status in progress is included in a statistic, even if the status is not completed.
- Action duration time for an action in progress is not included in a statistic until the action is completed.

Values for Sliding Interval Type

Values for the `Sliding` interval type use the following format:

`interval`:*`sampling`*

where

`interval` specifies the duration, in seconds, of the reporting interval.

`sampling` (optional) specifies the duration, in seconds of the sampling. If sampling value is not specified, Stat Server uses its default of `10` seconds.

Example Suppose that you want to set up a time profile (`Last10`) that always tracks the last 600 seconds of activity, with a sampling taken every 2 seconds, as shown in Figure 2.

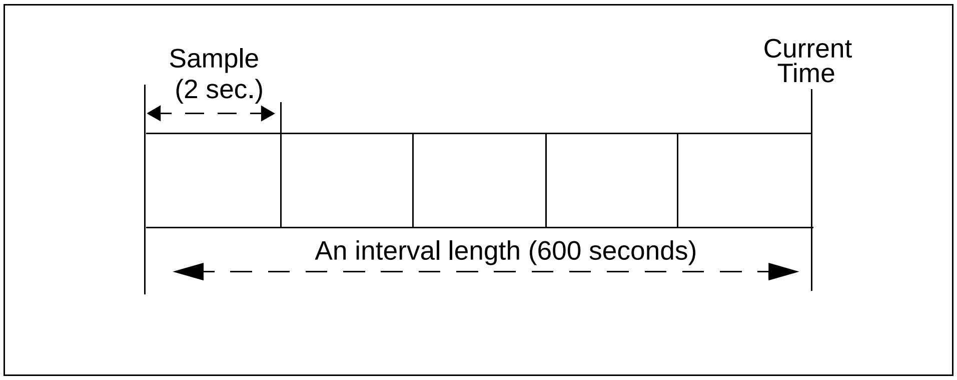
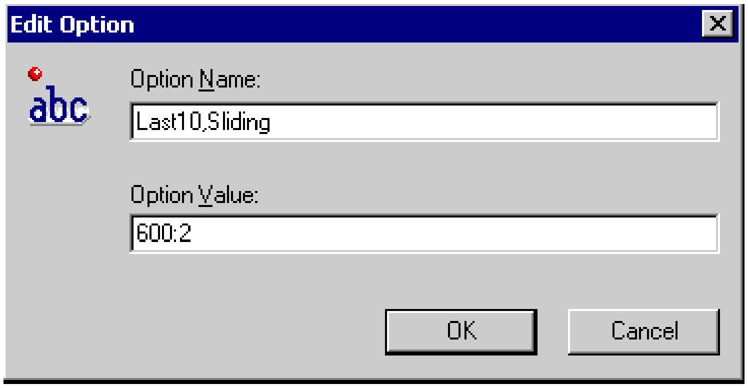


Figure 2: Example of Sliding Interval Type

To create this time profile, in Configuration Manager under the `[TimeProfiles]` section of your Stat Server application, enter `Last10,Sliding` in the `Option Name` field and `600:2` in the `Option Value` field, as shown in Figure 3.



Option Name: Last10,Sliding

Option Value: 600:2

Figure 3: Last10 Sliding Time Profile

Note: Stat Server release 7.0 and subsequent releases use the specified notification frequency instead of the sampling length configured for the `Sliding` time profile when a statistic is requested with the `TimeBased` notification mode. As a result, the statistic is calculated using the value of the notification frequency as a value of the sampling length. The `TimeBased` and other notification modes are described on page 24.

Values for Growing Interval Type

Values for the `Growing` interval type consist of:

Time to reset statistics to zero

The time to reset statistics is in the 24-hour clock format.

For example, `00:00` is midnight, `13:00` is 1:00 PM, and so on.

(Optional) Increment at which to reset statistics

The optional increment is also in the 24-hour clock format and is relative to the time to reset statistics to zero.

If no time profile is specified for a statistic requested by any client, Stat Server calculates statistics using the `Growing` interval type, which re-sets statistics to zero at `00:00` (midnight) unless a time profile named `Default` in the `TimeProfiles` section specifies a different initialization time. For example, to set `Default` to reset at 1 AM instead of midnight, enter `Default,Growing` in the `Name` field and `01:00` in the `Value` field.

Note: To specify more than one set of values, separate the sets with commas.

Example Suppose that you want to set up a time profile (named `Shifts`) that resets statistics to zero when shifts change at 3:00 AM, 7:00 AM, 11:00 AM, 1:00 PM, 7:00 PM, and 1:00 AM. To do so, enter `Shifts,Growing` in the `Name` field and `3:00 +4:00, 13:00 +6:00` in the `Value` field.

In this example, `3:00 +4:00` is translated as reset to zero at 3:00 AM, reset to zero at 3:00 AM plus 4 hours (7:00 AM), and then reset to zero again at 7:00 AM plus 4 hours (11:00 AM). The setting `13:00 +6:00` is translated as reset to zero at 1:00 PM (or 13:00 on the 24-hour clock), reset to zero at 1:00 PM plus 6 hours (7:00 PM, or 19:00 on the 24-hour clock), and then reset to zero again at 7:00 PM plus 6 hours (1:00 AM).

Figure 4 illustrates this example.

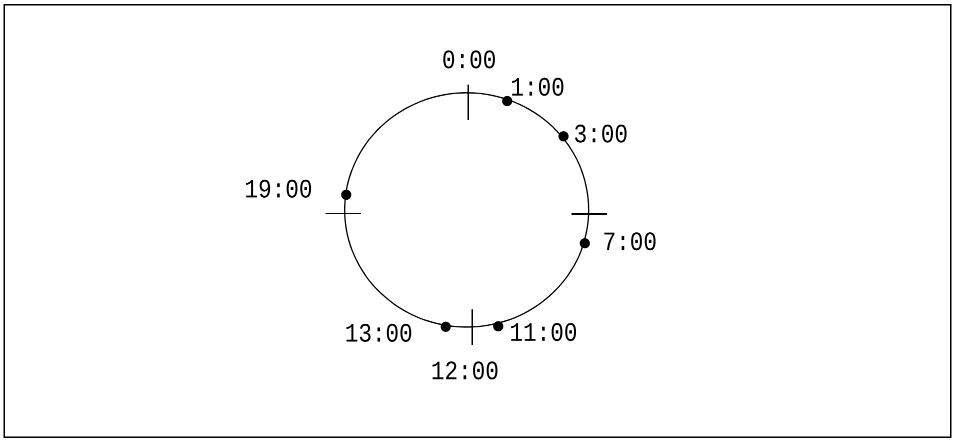


Figure 4: Example of Growing Interval Type

Values for Selection Interval Type

The `Selection` interval type calculates a time interval defined by the end or occurrence of the specified number of actions or statuses. A `Selection` interval lasts until the current time, or until the last action or status out of the specified number of actions or statuses has occurred (for instantaneous actions) or ended (for durable actions and statuses). The first time interval starts when Stat Server starts calculating a particular statistic. At a given moment, no more than the specified number of actions or statuses can occur during one `Selection` interval.

The actions or statuses taken into account are those listed either in the relative mask of the statistical type on which a statistic is based, or in the main mask if no relative mask is specified for the statistical type (see also “Statistical Type Sections” on page 37). The time interval varies depending on the amount of time it takes for the specified actions or statuses to occur.

The value for the `Selection` interval type must be an integer.

Note: You can specify a relative mask in a statistical type for the purpose of `Selection` intervals, even if the statistical category on which the type is based does not require a relative mask.

Example Suppose that you want to set up a time profile (named `Last5Calls`) that tracks the last five calls. To do so, enter `Last5Calls` with an interval type of `Selection`, and `5` in the `Value` field.

Figure 5 illustrates this example. In it, Total Interval 5 is calculated from the end of Action 4 until Current Time. Because no action is in progress at Current Time, the interval only includes durations of four actions (`5` through `8`).

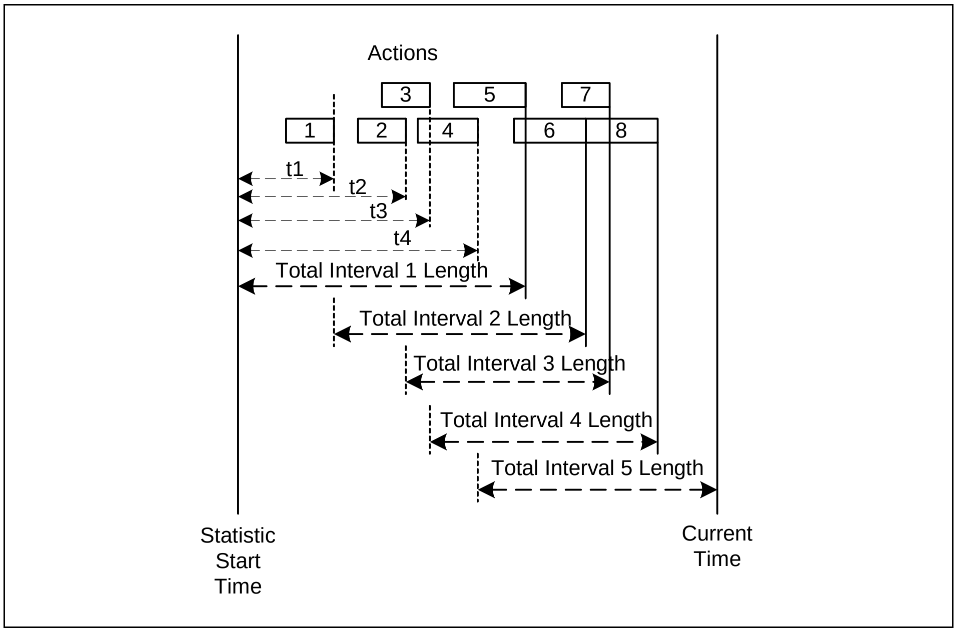


Figure 5: Example of Selection Interval Type

Values for SinceLogin Type

The `SinceLogin` interval type aggregates statistical data only for agent-object statistics—that is, statistics based on stat types with object type defined as `Agent.` Stat Server resets such statistics to zero (`0`) at the moment of agent login (that is, agent association to the place). Statistics continue to accumulate so long as an agent is associated with a specific place. The `SinceLogin` interval enables statistic requests *by agent*. This means you can now identify the least-occupied agent, for example, by requesting every agent's total handling time with `SinceLogin` interval).

No other parameters are passed with this interval.

Notification Modes

When requesting statistics, clients also specify how often they expect updates on the statistical values. Stat Server sends updates using one of the following *notification modes:*

- ChangesBased—Stat Server reports the current value whenever a statistical value changes. For time-related statistics, Stat Server reports the current value whenever a statistical value changes and with the specified notification frequency.
- TimeBased—Stat Server reports the current value at the specified notification frequency (for example, every two seconds).
- ResetBased—Stat Server reports the current value right before setting the statistical value to zero (`0`).

Table 2 demonstrates whether a particular notification mode is supported for a particular time profile.

Table 2: Compatible Notification Modes for Each Time Profile

Time Profile	Notification Mode		
	ChangesBased	TimeBased	ResetBased
Growing	Yes	Yes	Yes
Selection	Yes	Yes	No
Sliding	Yes	Yes	No
SinceLogin	Yes	Yes	No

Note that you can also request `Current` statistics with any of the three notification modes. These statistics do not require any time profile unless requested with ResetBased notification mode, in which case, you must use the `Growing` time profile. `CurrentState` statistics cannot be requested with ResetBased notification mode.

Insensitivity

Some Stat Server client applications, such as CCPulse+, specify an insensitivity value to further control the network “chatter” between agent PCs and Stat Server. *Insensitivity* describes a condition for Stat Server to send updates of statistical values to its clients. An increase in the value of this parameter usually decreases network traffic, but it also reduces reporting accuracy, because values are not updated as frequently. This setting is not visible in Stat Server configuration, but rather, clients pass its value to Stat Server along with each statistic request.

Insensitivity plays no role for reset-based statistics. For time-based or change-based notification mode, Stat Server only reports the recalculated value if the absolute value of the difference between the previous value and the recalculated value or its percentage ratio to recalculated value is at least equal to the number specified by insensitivity.

In addition, Stat Server uses a different algorithm of comparison with insensitivity depending on the data type of the result Stat Server calculates.

- If the result is a floating-point decimal—as is the case for statistics providing custom values, ratios, or averages—Stat Server uses percentages as the measure of comparison of insensitivity between a previous and a recalculated value. Given an `Insensitivity` setting of `5` for a floating-point statistic, for instance, Stat Server sends the recalculated result to its client only when the absolute value of the difference between the new and the old result is more than 4 percent of the absolute value previously sent. In the same scenario, but with an `Insensitivity` setting of `1,` Stat Server sends the recalculated result when it differs, by any amount, from the value previously sent.
- If the result has a long integer data type—as is the case for statistics measuring time—Stat Server uses the absolute difference in values for comparison. Given an `Insensitivity` setting of `5` in this case, Stat Server sends the recalculated result to its client when the absolute value of the difference between the new and old result is at least 5 (seconds, usually).

`Note:` This algorithm has changed throughout the releases. In 6.1 and prior releases, Stat Server did not use percentages to measure insensitivity.

Filters Section

The `Filters` section of the Stat Server application defines conditions for excluding call- and non-call-related activity based on certain criteria specified in a logical condition. If used, this section must be named `Filters.` Filters allow you to restrict Stat Server actions taken into account during the computation of aggregate values. In a filtered statistic, Stat Server only considers those actions that satisfy a filter condition on certain attributes of

TEvents, such as DNIS, ANI, CustomerID (or TenantID), MediaType, ThisQueue, TreatmentType, UserData, Reasons, and ExtensionReasonCode. Stat Server also allows filtering by Interaction Server-driven events via the UserData and Reasons attributes.

Stat Server also considers the type of action in its analysis of a filter condition:

- For durable actions and statuses, Stat Server uses the number of times that a filter condition was true on an action (or status) and the duration of time for which the filter was true.
- For retrospective (instantaneous) actions, Stat Server evaluates a filter at the moment of action completion. If the filter condition is true, the statistic uses the entire duration of the action (and the number is 1).

This implementation does not change how Stat Server calculates Current statistics, but it does alter the calculation of historical statistics. Now, for example, instead of Stat Server returning the entire duration when an agent is NotReady with a particular reason only at the end of the NotReady state, Stat Server more accurately returns only that duration of time within the NotReady state for which the filter condition was true. The following example and Figure 6 both illustrate this point.

Example Assume that an agent has placed himself in the NotReady state for 50 minutes. During that state, he selected four reason codes for the following durations, respectively, on the phone set:

- ReasonCode 1—5 minutes
- ReasonCode 2—15 minutes
- ReasonCode 3—5 minutes
- ReasonCode 1—25 minutes

Using filter=Reason Code 1, the current Stat Server implementation returns 2 as the number of times that filter=ReasonCode 1 or 30 minutes as the duration for which the filter condition was true during the NotReady state.

Previous implementations returned 1 as the number of times that the filter condition was true—only if filter = ReasonCode 1 was true at the moment that the agent left the NotReady state. Stat Server also returned 50 minutes, in this example, as the duration of time for which filter=ReasonCode 1.

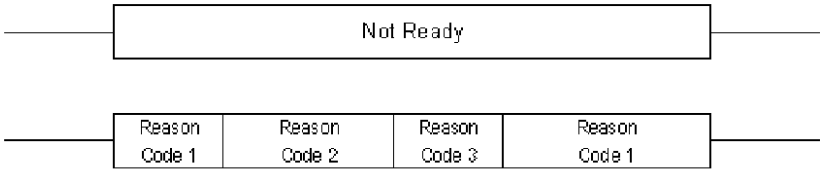


Figure 6: Filter Example

The filters that you configure in Stat Server appear under the Statistical Parameters folder in Data Modeling Assistant (DMA), and among a particular statistic's properties within CCPulse+. (You can use DMA also to configure new filters.) If a Stat Server client requests a particular statistic with a filter, and that filter has been deleted from the configuration environment, Stat Server continues to calculate the statistic and sends the client an unfiltered value. Client applications can submit a statistic request that has no more than one filter applied.

Each opened statistic can have its own specific filter represented as a text string that contains a logical condition. The logical condition has to be proven for each call or device property. The result is either `true,` which includes the considered activity in the calculation, or `false,` which excludes the considered activity from the calculation. The logical condition has references to call or device properties, as well as to numeric and string constants, all of which are combined by operators.

Options in the `Filters` section consist of the following:

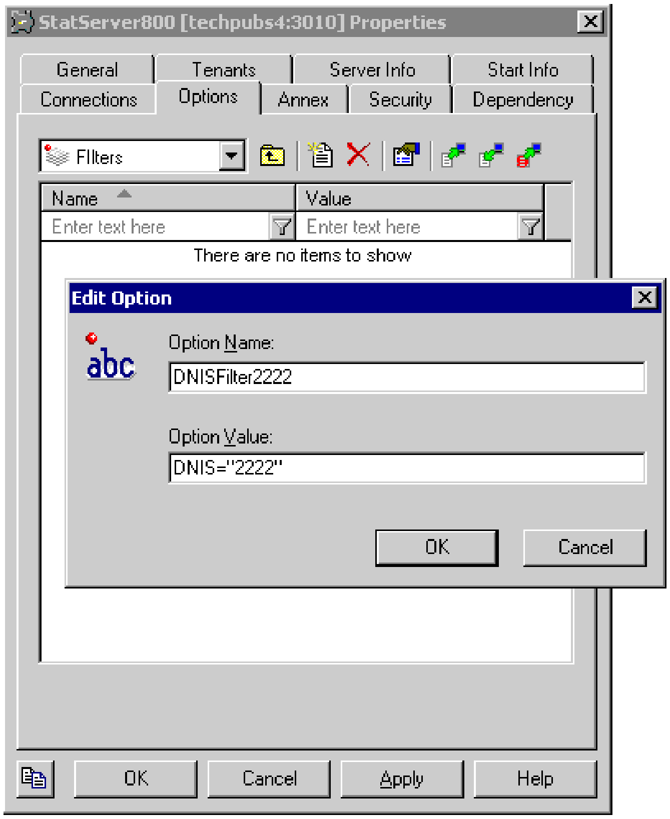
`option name` Any character string that represents the name of the filter.

`option value` A logical condition that contains call or device properties (see Tables 4 and 5) and numeric or string constants that are combined by an operator (see Table 6). You can use the `?` and/or `*` wild-card characters in the designation of the option's value. Stat Server matches `?` in a wild-card string to any single character. Stat Server matches `*` to zero or more characters. The default value uses the `PairExists` function (see Table 3).

Table 3: Configuration Option for Filters Section

Option	Description
<FilterName>	Defines a filter for filtering out call- and non-call-related activity, based on certain criteria that are specified in a logical condition. The logical expression is composed of: • Call or device properties (see Tables 4 and 5) • Operators (see to Table 6). • Values that consist of numerics, character string constants, or empty strings, depending on the call or device property. You must specify a value for this option; otherwise, Stat Server uses its default: Default Value: `PairExists(“filtername”, “*”)` Valid Values: A logical expression (see preceding description). Changes Take Effect: When Stat Server restarts Stat Server recognizes the following functions as aliases for `PairExists:` • `PairExist` • `TKVListPairExist` • `TKVListPairExists`

Example Suppose that you want to set up a filter (`DNISFilter2222`) that considers calls whose Dialed Number Identification Service (DNIS) is `2222`. To do so, enter `DNISFilter2222` in the `Name` field and `DNIS="2222"` in the `Value` field, as shown in Figure 7. In this example, the call property is `DNIS`, the operator is the equal sign (=), and the constant is `2222`.



Option Name: DNISFilter2222

Option Value: DNIS="2222"

Figure 7: Defining a Filter

Call and Device Properties

You can exploit call and device properties by using the keywords provided in Tables 4 and 5. The keyword listings are not comprehensive; T-Events describe other call and device properties. Refer to the *Genesys 7 Models and Events Reference Manual* for additional information.

Table 4: Call Properties

Property Name	Operand Type	Description
DNIS	string	`DNIS` is the Dialed Number Identification Service. The `DNIS` is all or part of the telephone number that was dialed to make a call.
ANI	string	`ANI` is the Automated Number Identification. The `ANI` is all or part of the caller's telephone number.

Table 4: Call Properties (Continued)

Property Name	Operand Type	Description
CustomerID	string	`CustomerID` is the tenant identification number as defined in the Configuration Layer.
MediaType	integer	`MediaType` is the kind of environment through which an interaction (for example, a call) is distributed. The predefined media types are as follows: • `0` (`Voice`) • `1` (`VoIP`) • `2` (`EMail`) • `3` (`VMail`) • `4` (`SMail`) • `5` (`Chat`) • `6` (`Video`) • `9` (`Cobrowsing`) • `8` (`Whiteboard`) • `9` (`AppSharing`) • `10` (`WebForm`) • `11` (`WorkItem`) • `100+` (Custom) An elementary filter condition can contain either an integer value or a string with the predefined media type (for example, `MediaType=5` or `MediaType=Chat`)
ThisQueue	string	`ThisQueue` is the number of the queue.
TreatmentType	string	`Treatment` is the type of the treatment applied to a call, such as `Silence`, `Music`, `Busy`, and so forth.

Table 4: Call Properties (Continued)

Property Name	Operand Type	Description
UserData	string (TKVList)	`UserData` refers to the data that is attached to an interaction. An IVR might attach data to a call, for example, by collecting the numbers that callers press on their telephone keypads in response to a prompt. Or, an agent might attach data to a call from a desktop application. The `TKVList` refers to a set of functions that perform actions on `UserData` properties. *K* stands for *Key* and *V* stands for *Value*. T-Server sends attached data in key-value pairs; that is, one pair element specifies the key that describes the value, and the second element specifies the key's actual value. For example, `AfterCall` could be the name of a key, and the text `Processed the call for 10 minutes` could be the key's value. For memory, performance, and security reasons, Stat Server's processing of `UserData` strips the following types of `UserData` keys, which are not used for internal computations: • Keys included in at least one filter. • Keys coinciding with the names of business attributes. • Keys associated with the `EventUserEvent`, `EventPrivateInfo`, `EventError`, or `EventPartyInfo` TEvents. • `GSW_RECORD_HANDLE`, a predefined key used in the processing of user events for campaign-related statistic computations. **Note:** Stat Server does not strip `UserData` containing the `GSW_CALL_TYPE` key because Stat Server uses this information to generate the `ASM_Engaged` action for agent DNs involved in outbound predictive dialing interactions. `UserData` that Stat Server uses for internal processing is packed into the values of `CurrentState` statistics.

Table 4: Call Properties (Continued)

Property Name	Operand Type	Description
Extensions	string (TKVList)	This property enables Stat Server to filter switch-specific and other features on any specified key-value pair recorded in the `Attribute Extensions` attribute of select TEvents. A filter using this property must be specified in the following format: `PairExists( Extensions,` *<key>*, *<value>* `)` where: `Extensions` is the hard-coded name of the `TKVList` function. *<key>* is a string representing the key of a key-value pair. *<value>* is an integer or string representing the *<key>*'s values. Stat Server applies a filter having this definition to a statistic for the following noncall-related TEvents that Stat Server receives from an agent's DN: • `EventAgentLogin` • `EventAgentLogout` • `EventAgentReady` • `EventAgentNotReady` • `EventDNDOn` • `EventDNDOff` • `EventRegistered` • `EventAddressInfo` Stat Server also applies a filter with this definition to the following call-related TEvents that Stat Server receives from regular DNs: • `EventAbandoned` • `EventAttachedDataChanged` • `EventDialing` • `EventEstablished` • `EventHeld` • `EventNetworkCallStatus` • `EventPartyAdded` • `EventPartyChanged` • `EventPartyDeleted` • `EventPartyInfo` (handled as `EventEstablished`) • `EventQueued` (handled as `EventRinging`) • `EventReleased` • `EventRetrieved` • `EventRinging` For call-related TEvents, filters using this property apply toward any associated actions. For noncall-related TEvents, only the following actions can be impacted by `Extensions` filtering: • `AfterCallWork` • `LoggedIn` • `NotReadyForNextCall` • `WaitForNextCall`

Table 5: Device Properties

Property Name	Operand Type	Description

Reasons	string (TKVList)	Refers to additional data that is included in the TEvent to provide reasons for and results of actions taken by an agent. These reasons can originate from software- or hardware-related reasons—Stat Server does not differentiate between the two. Stat Server uses the value of the `Reasons` attribute in combination with the values of the `UserData` attribute when processing filters:
		`PairExists( UserData, key, value )` and `PairExists( key, value )`
		Stat Server uses the value of the `Reasons` attribute only in filters:
		`PairExists( Reasons, key, value )`
		When specified as such, Stat Server ignores any attached data that has `UserData` defined as the key in order to avoid consuming additional memory for its storage.
		Note: Do not confuse this `Reasons` property with `Reason,` which serves as an alias for the `ExtensionReasonCode` property described in the next row.

Extension ReasonCode		Refers to the reason code that T-Server propagates in its `Attribute Extensions` attribute of a TEvent. T-Server uses this key-value pair to gather switch-specific hardware reason codes that mostly accompany `Ready` and `NotReady` TEvents. Despite the fact that Stat Server does not restrict use of such variables in filters, Genesys recommends that you use filters with this variable only for accessing switch-related reason codes in non-call-related agent or DN states. Values of hardware reasons are switch-specific and must be configured on the customer side. In the event that T-Server propagates no reason code, Stat Server reports the value of this condition as `Unknown` and any filters using this property evaluate as `False.` Stat Server packs `Reason` attached data into the values of `CurrentState` statistics. Stat Server recognizes `Reason` as an alias of `ExtensionReason Code.` This should not be confused with the `Reasons` property described in the row above.

Table 5: Device Properties (Continued)

Property Name	Operand Type	Description
Extension ReasonCode (continued)		For example: • `ExtensionReasonCode = "lunch"` (or `Reason = “lunch”`) returns a `True` value if the value of the key-value pair returned by the `ReasonCode` key is equal to `lunch.` • `ExtensionReasonCode != 12` (or `Reason != 12`) returns `True` if the `Extension` TEvent returns a key-value pair of `ReasonCode` (the key) and its accompanying value which is equal to a value other than `12.` From 8.0 release, Stat Server supports less than or equal, greater than or equal, greater than, less than expressions for numeric operands. For example: • `ExtensionReasonCode >= 12` (or `Reason >= 12`) • `ExtensionReasonCode < 12` (or `Reason < 12`) **Note**: Software reasons (propagated by the `Reasons` attribute) are still provided using the `PairExists` function. Stat Server does not differentiate between hardware- and software-related reasons.

Operators

Table 6 shows the operators that you can use for a logical condition in a filter.

Table 6: Operators in Filters

Operators	Description
=	Equal (for strings or numeric operands)
!=	Not equal (for strings or numeric operands)
>=	Greater than or equal to (for numeric operands only)
<=	Less than or equal to (for numeric operands only)
>	Greater than (for numeric operands only)
<	Less than (for numeric operands only)
&	Logical AND
|	Logical OR
~	Logical NOT
( )	Parentheses (for changing operators’ priorities)

Filter Expression Evaluations

The results of a filter expression can be TRUE, FALSE, or NULL; however, Stat Server returns to its clients either TRUE or FALSE depending on the expression's construction.

Filter sub-expressions, such as GetNumber(), may be evaluated to NULL if, for example, the referenced key in the key-value list does not exist—Stat Server cannot retrieve its value. NULL can also appear as a result of propagation. When evaluating filter expressions, Stat Server propagates NULL according to the following rules:

- Any arithmetical sub-expression having NULL as one of the operands, is evaluated to NULL (for example, NULL+2 yields NULL).
- Any comparison sub-expression having NULL as one of the operands, is evaluated to NULL (NULL=2 yields NULL).

In logical sub-expressions:

- NULL | TRUE yields TRUE.
- NULL & FALSE yields FALSE.

If the whole filter expression is evaluated to NULL, Stat Server returns FALSE as the final result.

UserData

The key-value list UserData cannot be an operand of any operator. Instead, it can be listed as the first parameter of any one of the TKVList family of functions shown in Table 7, or it can be left out. For example, PairExists(UserData,“key”,“value”) and PairExists(“key”,”value”) are equivalent.

These filter function names can be preceded with TKVList, as was the case in previous versions of Stat Server. TKVListPairExists and PairExists are both valid names, for example.

Use the wildcard character * (asterisk) in place of the value in filter functions. PairExists(“Key”,”*”) would return 1 for true if any key-value pair exists where the key equals “Key”, regardless of the value of that pair.

Table 7: UserData Properties

Property	Description
PairExists(“Key”, “Value”)	Performs search for the specified pair. Returns a number: `1` (true) or `0` (false).
GetNumber(“Key”, Index)	Returns the numeric value of the occurrence of the given key as specified by `Index:` • If `Index` is `-1,` the last occurrence is used. • If `Index` is a positive integer *`n,`* the *`nth`* occurrence is used. When `Index` exceeds the total number of occurrences of the given key in the list, or the key does not occur in the list at all, the returned value is `NULL.` `Index` is an optional attribute for this property. If not specified, Stat Server substitutes `-1` for its value; hence, `GetNumber(“Key”)` is equivalent to `GetNumber(“Key”, -1).`
GetString(“Key”, Index)	Returns the string of the value of the given key as specified by `Index:` • If `Index` is `-1,` the string of the last value is used. • If `Index` is a positive integer *`n,`* the string of the *`nth`* value is used. When `Index` exceeds the total number of occurrences of the given key in the list or the key does not occur in the list at all, the returned value is `NULL.` `Index` is an optional attribute for this property. If not specified, Stat Server substitutes `-1` for its value; hence, `GetString(“Key”)` is equivalent to `GetString(“Key”, -1).`
GetMax(“Key”)	Returns the maximum value among all pairs with this `“Key”`. A value of `NULL` means that there are no pairs with the `“Key”`.
GetMin(“Key”)	Returns the minimum value among all pairs with this `“Key”`. A value of `NULL` means that there are no pairs with the `“Key”.`
GetSum(“Key”)	Returns the sum of all values with this `“Key”`. A value of `0` means that there are no pairs with the `“Key”.`
GetAver(“Key”)	Returns the average of all values with this `“Key”`. A value of `0` means that there are no pairs with the `“Key”.`

For all functions dealing with numbers, the value of the key-value pair is evaluated as either an integer or a floating point. If the key type is an integer, the value is evaluated as an integer with no modifications. If the key type is a string, the value is a floating point. Constants for a logical condition can be either strings in double quotation marks `(“English”,“3333”)` or numbers `(100, 3.14).` Numbers (constants and function return values) are floating-point values.

Starting with release 7.0, Stat Server ignores any attached data if no corresponding filter or custom-value formula has been defined within Stat Server that uses the specific key. This is done for performance and security reasons. Stat Server, furthermore, does not output attached data to the Stat Server log under this circumstance.

Example Suppose that you want to filter calls based on language. If the enterprise set up the key "`Language`" to identify language and the value "`Spanish`" for callers who speak Spanish, you could use the `PairExists UserData` function to search for calls with attached data in the key-value pair form of `Language/Spanish.`

On the `Options` tab of the Stat Server `Properties` dialog box, you could add a `SpanishLanguage` option in the `Filters` section and specify filtering for calls with attached data containing the key "`Language`" and the value "`Spanish`". The example would have `SpanishLanguage` in the `Name` field and `PairExists("Language","Spanish")` in the `Value` field.

Now, when an agent attaches the "`Spanish/Language`" key-value pair to calls from a desktop application, the calls are filtered out of statistical calculcations.

TimeRanges Section

The `TimeRanges` section of the Stat Server application defines the time ranges that Stat Server uses for collecting data. If used, this section must be named `TimeRanges.` Time ranges can only be used for the following statistical categories:

- `CurrentNumberInTimeRange`
- `CurrentNumberInTimeRangePercentage`
- `TotalNumberInTimeRange`
- `TotalNumberInTimeRangePercentage`
- `TotalTimeInTimeRange`
- `ServiceFactor1`
- `RelativeNumberPercentage`

For more information about statistical categories, see Chapter 6 on page 137.

The `TimeRanges` section contains one or more `<TimeRangeName>` configuration options. Table 8 describes the one configuration option applicable for this section.

Table 8: Configuration Option for TimeRanges Section

Option	Description
<TimeRangeName>	Defines a time range for collecting data. The time range name is any character string that represents the time range. The time range value is composed of two digits separated by a hyphen (`-`): the starting point and the end of the range in seconds, such as `0-20`. You must specify a value for this option. Note that: • Specifying a time range of `0-20` results in Stat Server collecting data from 0.00 seconds to 19.99999... seconds. • Specifying a time range of `20-50` results in Stat Server collecting data from 20.00 seconds to 49.99999... seconds. Thus, if you configure two time ranges (`0-20` and `20-50`), Stat Server attributes the call that lasts exactly 20 seconds to the second time range only. **Note:** When you have configured no options in the `TimeRanges` section, but a statistic that requires a time range is requested, Stat Server calculates this statistic with the predefined time range of `0-20.` Default Value: No default value Valid Value: Any value specified in the described format above Changes Take Effect: When Stat Server restarts

Example Suppose that you want to calculate the total number of calls answered within 30 seconds based on a specified time range. To do so, enter `Range0-30` in the `Name` field and `0-30` in the `Value` field.

In this example, a statistic that calculates the total number of calls would be based on the time range “`Range0-30`” if configured so in CCPulse+. If one call is answered after being in a queue for 25 seconds, a second call after 40 seconds, and a third call after 10 seconds, Stat Server counts only the first and third calls.

Statistical Type Sections

A *statistical type* (stat type) is a collection of actions, object types, category, and one subject that all help define the structure of a metric. Other factors may contribute to a metric’s definition, such as a time profile, an optional time range, and an optional filter, all described earlier in this chapter.

Each stat type definition consists of:

- A user-defined section name, which represents the name of the stat type.

  Use either the predefined statistical types (see page 189) or campaign statistical types (see, page 177), or create your own custom-value stat types (see page 45).

- Configuration options that apply to that section.

  Most stat type configuration options can be classified as one of the following:

  - Options for core stat types, as described in Table 9
  - Options for Java stat types, as described in Table 10

  A small number of the options serve both categories but have differing permissible values.

  Statistics based on core stat type definitions are calculated directly within Stat Server. Statistical values of Java stat types are provided to Stat Server by another Genesys server, such as Interaction Server or Outbound Contact Server.

Table 9: Configuration Options for Core Stat Types

Option	Description
Objects	Specifies a list of comma-separated Stat Server object types to which statistics apply. The list must consist of objects of the same compatibility group. (This term is defined on page 51). You must include this option in a stat type definition and specify a value. Default Value: No default value Valid Values: Refer to Table 14, “Stat Server Object Types and Descriptions,” on page 49 and “Campaign Objects” on page 175. Changes Take Effect: When Stat Server restarts.

Table 9: Configuration Options for Core Stat Types (Continued)

Option	Description
MainMask	Specifies a list of comma-separated actions (or statuses) that indicate which contact center events that will be measured. This list comprises members from one of five groups: regular DN actions, mediation DN actions, media-channel actions, campaign actions, or statuses. This option is mandatory for core stat types and you must specify one or more values. Use the wild-card (`*`) character to specify all actions; use the logical NOT (~) character to exclude the action it precedes. Use parentheses around each action (or status) that you want Stat Server to exclude from consideration of being filtered. You cannot, however, use parentheses in conjunction with * or ~. For example: `MainMask=CallInbound,(CallOutbound)` If a filter were applied to a statistic having this `MainMask` designation, Stat Server would only apply the filter to `CallInbound` actions. `CallOutbound` actions would continue to contribute to the tally of this statistic unfiltered. It is also possible to use the `*` and ~ characters in selective filtering. Default Value: No default value Valid Values: Refer to “Stat Server Actions” on page 57, “Object Statuses” on page 125, and “Campaign Operational Actions” on page 178 for a listing and description of these actions and statuses. Changes Take Effect: When Stat Server restarts.
RelMask	Specifies a list of comma-separated actions (or statuses) that indicate the superset of contact center events against which the listing of actions (or statuses) provided in the main mask will be measured. This list comprises members from one of five groups: regular DN actions, mediation DN actions, media-channel actions, campaign actions, or statuses. Specifying this option is not mandatory, but if you do use it, you must supply one or more values. Use the wild-card (`*`) character to specify all actions; use the logical NOT (~) character to exclude the action it precedes; and, use parentheses around each mask that you want Stat Server to exclude from consideration of being filtered. You cannot, however, use parentheses in conjunction with * or ~. Default Value: No default value Valid Values: Refer to “Stat Server Actions” on page 57, “Object Statuses” on page 125, and “Campaign Operational Actions” on page 178 for a listing and description of these actions and statuses. Changes Take Effect: When Stat Server restarts.

Table 9: Configuration Options for Core Stat Types (Continued)

Option	Description
Category	Informs Stat Server how to calculate statistics. This section is mandatory for both core and Java stat types; and, you must supply one and only one value. Default Value: No default value Valid Values: Refer to “Statistical Categories” on page 137. Changes Take Effect: When Stat Server restarts.
Subject	Specifies the object type for statistics calculation that, when changed, affect the statistics. This section is mandatory for core stat types and you must supply one and only one value. Default Value: No default value Valid Values: `DNAction`, `DNStatus`, `AgentStatus`, `GroupStatus`, `PlaceStatus`, `CampaignAction`, `Action`. Refer to “Statistical Subjects” on page 165 for a description of these values. Changes Take Effect: When Stat Server restarts. **Note:** The `AgentStatus` and `PlaceStatus` objects were synonymous in releases 5.1, 6.0, and 6.1. However, they are independent in 6.5 and later releases.
<business attribute>	Specifies one, and only one, business attribute that Stat Server applies as a filter during its computation of statistics. Starting with release 7.1, Stat Server supports only the `MediaType` business attribute. Specifying this option is not mandatory. Default Value: No default value Valid Values: Non-empty string Changes Take Effect: When Stat Server restarts. The name of the business attribute must be a valid business attribute that is already defined to a particular tenant before Stat Server starts. This name cannot coincide with the reserved names for other Stat Server configuration options, such as `Subject`, `Category`, and `Filter.` Furthermore, the name must not contain special symbols (such as `|`, `=`, or `;`) or spaces.

Table 9: Configuration Options for Core Stat Types (Continued)

Option	Description

UseSourceTimeStamps	For those metrics that qualify, this option specifies whether Stat Server uses the actual time that events were transmitted to Stat Server (source timestamp) or the time that Stat Server acknowledges receipt of the events (the default behavior) when calculating metric duration. Setting this option to `yes` enables better consistency with the metrics provided by Interaction Concentrator (ICON) and other downstream Genesys applications of ICON.
	Qualifying metrics have both of the following characteristics:
	• `[`*`TimeProfileName`*`]=Selection` or `Growing`
	• `[`*`StatTypeDef`*`]`
	`Subject=DNAction` or `CampaignAction`
	`MainMask=`one or more durable and/or retrospective actions (including instantaneous actions that carry an associated duration, like `AgentLogin`).
	`Category=`one that is historical, cumulative, and measures duration, such as `TotalTime` and `LoadBalance.`
	Stat Server ignores a `yes` value for this option if the metric fails the qualification test.
	Default Value: `no`
	Valid Values: `no, yes`
	Changes Take Effect: When Stat Server restarts.
	Refer to “Stat Server Timestamps” on page 169 for an extended discussion of Stat Server’s use of source timestamps.

Formula	The `DistByConnID` specifier affects Stat Server’s mechanism of aggregating statistics for the call-related actions specified in a statistic’s main mask. `DistByConnID` is applicable only to the limited number of statistical categories: • `TotalNumber` • `TotalAdjustedNumber` • `CurrentNumber` • `TotalTime`

Table 9: Configuration Options for Core Stat Types (Continued)

Option	Description
Formula (continued)	When specified for a statistic, Stat Server groups the statistic's actions by connection ID (`ConnID`). In general, the contribution of a group of actions, differs from that of the sum of contributions of the individual actions in that group—as is the case when `DistByConnID` is not specified for a statistic. Stat Server's procedure of grouping actions by connection ID is applicable to actions for the objects associated with the statistic, which types belong to statistic's `MainMask.` The procedures differ for each statistical category and are described as follows: • For the `TotalNumber/TotalAdjustedNumber` statistical categories, when any of the following conditions are true, Stat Server increments the statistic at the end of an action or the start of status respectively: ◆ An action or status with particular `ConnID` starts. ◆ There are no actions or statuses in progress for the same `ConnID.` ◆ No such actions or statuses were in progress for less than one minute ago (1 minute is hard-coded). If the action or status is unrelated to a call, then aggregation functions in the same manner as when `DistByConnID` is not specified. **Note:** `DCID` is not applicable in `TotalNumber/TotalAdjustedNumber` statistics where filters or time ranges are also used. • For the `CurrentNumber` statistical category, when either of the following conditions is true, Stat Server increments the statistic: ◆ An action or status with a particular `ConnID` starts. ◆ There are no actions or statuses in progress for the same `ConnID.` When the action or status with the particular `ConnID` ends, Stat Server decrements the statistic only if there are no more actions or statuses in progress for that `ConnID.` If the action is either not call-related or not durable, Stat Server ignores this action in statistic calculations. • For the `TotalTime` statistical category, the group of actions or statuses in progress for a particular `ConnID` yields a one second increment to the statistic for each second of the group's existence. Where the statistic's `Subject` is other than `DNAction,` Stat Server immediately reflects this increment in the statistical value. Where `Subject=DNAction,` Stat Server updates the statistic's value in a stepwise fashion, incrementing the statistic when the oldest action belonging to group of actions ends. If an action is either not call-related or not durable, Stat Server ignores this action in statistic calculations.

Table 9: Configuration Options for Core Stat Types (Continued)

Option	Description
Formula (continued)	**Notes:** • If you use the `DistByConnID` qualifier, you must list it first among the `Formula` values in the following format: `Formula=DistByConnID,...` • Stat Server recognizes the following aliases for `DistByConnID`: ◆ `DistinguishByConnID` ◆ `DCID` • Any filtering that may be used in conjunction with a statistic, such as the designation of a `MediaType,` is applied *prior* to Stat Server's processing of `DistByConnID.` Default Value: `no` Valid Values: `no, yes` Changes Take Effect: When Stat Server restarts.
Description	Specifies a description for this stat type. Specifying this option is discretionary; Stat Server ignores any value that you set for this option. Default Value: No default value Valid Values: String of fewer than 256 characters Changes Take Effect: When Stat Server restarts.

Table 10: Configuration Options for Java Stat Types

Option	Description
JavaSubCategory	The name of the Java subclass that implements statistic calculation. Default Value: no default value Valid Values: string specified in the following format: *`jarfile`*:*`subclass`* Changes Take Effect: When Stat Server restarts.
Category	Informs Stat Server how to calculate statistics. Default Value: No default value Valid Value: `JavaCategory` Changes Take Effect: When Stat Server restarts.

Table 10: Configuration Options for Java Stat Types (Continued)

Option	Description
Objects	Specifies a list of comma-separated Stat Server object types to which statistics apply. The list must consist of objects of the same compatibility group. You must include this option in a stat type definition and specify a value. Default Value: No default value Valid Values: `Agent`, `Place`, `GroupAgents`, `GroupPlaces`, `RoutePoint`, `Queue`, `GroupQueues`, `Tenant`, `Staging Area`, `InteractionQueue`. Refer to Table 14, "Stat Server Object Types and Descriptions," on page 49 for more information. Changes Take Effect: When Stat Server restarts.
<business attribute>	Same as the `<business attribute>` configuration option for core stat types.
Description	Same as the `Description` configuration option for core stat types.

Notes: If you want to change the definition of a stat type during runtime, you must first delete the entire stat-type definition and then re-create it with its new definition. Otherwise, if you change the definition during runtime, Stat Server recognizes the change only upon restart.

Stat Server clients may recognize other options for stat types that are not listed in Tables 9 and 10. For instance, Data Sourcer requires that `AggregationType` be specified for statistics derived from a Stat Server Java extension. This information is processed by the client; Stat Server ignores such options.

Classification of Statistical Types

Statistical types can be classified in distinct groups—for example:

- Status-based statistics.
- Interaction-related statistics.

Status-based statistics reflect changes in object statuses and generally contain the word *status* in their names. Interaction-related statistics reflect the telephony or multimedia information applied to specific objects, and characterize the interaction flow passing through the objects. Additional statistics, such as `ExpectedWaitTime` and `LoadBalance` statistics, reflect other characteristics of the contact center that are not related to status changes or telephony object information.

In addition, you can classify statistics based on any part of their stat type definition, such as their type of filter, object, and/or subject, or, on any other criteria that you specify.

Custom-Value Statistical Types

Custom-value stat types improve business data reporting by enabling you to define statistics that use formulas specific to your needs. Using your own formulas, you can create statistics that calculate average sales revenue per call and the total sales revenue for a specific time interval. The custom-value stat types that you define then become available to client applications that request them.

The format of custom-value stat types is similar to the format of Genesys-provided stat types. Custom-value stat types, however, lack the `RelMask` option and always contain the `Formula` option for which you must supply a value. See "Statistical Type Sections" on page 37 for a description of the predefined statistical type format.

Table 11 shows the statistical categories that apply to custom-value statistics.

Table 11: List of Custom-Value Categories

Historical Categories	Current Categories
TotalCustomValue	CurrentCustomValue
AverageCustomValue	CurrentAverageCustomValue
MinCustomValue	CurrentMinCustomValue
MaxCustomValue	CurrentMaxCustomValue

These categories are described on pages 150 and 151.

Example Suppose that you want to define a custom-value stat type that calculates the average sales revenue generated for every inbound call received by an agent. To accomplish this, within Configuration Manager, create and define a new stat type section in the Stat Server `Application` object as follows:

1. Open the `Options` tab of the Stat Server application.
2. Create a new section and name it `AverSalesAmountPerInboundCall`, for example.
3. Within this section, add the `Objects` option and set its value to:

   ```
   Agent, Place, GroupAgents, GroupPlaces
   ```

4. Add the `Category` option to this section and set its value to `AverageCustomValue`.
5. Add the `MainMask` option and set its value to `CallInbound`.
6. Add the `Subject` option and set its value to `DNAction`.
7. Add the `Formula` option and set its value to:

   ```
   GetNumber("Price", 1) * GetSum("Amount")
   ```

   (Refer to "Custom Formulas" below for an explanation of this formula.)
8. Apply the changes.

A configuration-file export of this section, as defined, appears as follows:

```
[AverSalesAmountPerInboundCall]
Objects=Agent, Place, GroupAgents, GroupPlaces
Category=AverageCustomValue
MainMask=CallInbound
Subject=DNAction
Formula=GetNumber("Price", 1) * GetSum("Amount")
```

Custom Formulas

Note: For an evaluation of custom formulas, refer to page 181.

Custom formulas define custom values from an action or a status on the basis of attached data. Attached data can be attached to the call by different T-Server clients. An IVR might attach data to a call, for example, by collecting the numbers that callers press on their telephone keypads in response to a prompt. An agent might also attach data to a call using a desktop application.

The language used in custom formulas is similar to that used in filters. Each formula is an arithmetic expression built from function calls and numeric constants, consisting of:

- Function calls. Custom formulas can use values from the key-value `UserData` lists received with TEvents related to Stat Server actions. Access to these values is provided by the functions listed in Table 13. This table also shows the rules for converting the string values of key-value lists to numbers. Note that the list can include more than one pair with the same key.
- Operators (see Table 12), as well as parentheses (for suppressing standard precedence rules).

Table 12: Operators in Custom Formulas

Operator	Description
+	Addition
-	Subtraction
/	Division
*	Multiplication

- Numeric constants.

Custom formulas always return a value of type `float.` The returned value is used in statistical calculations for each category.

Note: You can apply filters to custom-formula statistics too.

Table 13 lists functions to access key-value UserData lists. Note that a list can include more than one pair with the same key. Local key-value lists function with data attached at the DN where the action occurs. Global key-value lists function with data attached at all participating DNs during the call.

Note: For momentary actions, the GetGlobalMax function returns the same value as the GetMax function.

Table 13: Key-Value List Functions in Custom Formulas

Function	Description
Local Functions (Used for Local Key-Value List Calculations)	
GetNumber(“Key”, Index)	Returns the numeric value of the occurrence of the given key as specified by Index: • If Index is -1, the last occurrence is used. • If Index is a positive integer *n,* the *nth* occurrence is used. When Index exceeds the total number of occurrences of the given key in the list, or the key does not occur in the list at all, the returned value is 0. Index is an optional attribute for this property. If not specified, Stat Server substitutes -1 for its value; hence, GetNumber(“Key”) is equivalent to GetNumber(“Key”, -1).
GetMax(“Key”)	Returns the maximum value among all the values of pairs with the given key. When there are no such pairs, 0 is returned.
GetMin(“Key”)	Returns the minimum value among all the values of pairs with the given key. When there are no such pairs, 0 is returned.
GetSum(“Key”)	Returns the sum of all the values of pairs with the given key. When there are no such pairs, 0 is returned.
GetAver(“Key”)	Returns the average of all the values of pairs with the given key. When there are no such pairs, 0 is returned.
Global Functions (Used for Global Key-Value List Calculations)	
GetGlobalNumber(“Key”, Index)	Returns the numeric value of the occurrence of the given key, attached at any DN, which is a member of the call, as specified by Index: • If Index is -1, the last occurrence is used. • If Index is a positive integer *n,* the *nth* occurrence is used. When Index exceeds the total number of occurrences of the given key in the list, or the key does not occur in the list at all, 0 is the returned value.

Table 13: Key-Value List Functions in Custom Formulas (Continued)

Function	Description
GetGlobalMax("Key")	Returns the maximum value among all the values of pairs, attached at any DN, which is a member of the call, with the given key. When there are no such pairs, `0` is returned.
GetGlobalMin("Key")	Returns the minimum value among all the values of pairs, attached at any DN, which is a member of the call, with the given key. When there are no such pairs, `0` is returned.
GetGlobalSum("Key")	Returns the sum of all the values of pairs, attached at any DN, which is a member of the call, with the given key. When there are no such pairs, `0` is returned.
GetGlobalAver("Key")	Returns the average of all the values of pairs, attached at any DN, which is a member of the call, with the given key. When there are no such pairs, `0` is returned.

Example Suppose you want to multiply 99.99 by the sum of all the values of key-value pairs with key `“Amount”.` To do so, enter the following formula:

```
99.99 * GetSum( "Amount" )
```

Chapter

3 Stat Server Object Types

Stat Server gathers information about contact center objects defined in Configuration Server and supplies statistical data about these objects to Stat Server clients. In the following sections, this chapter describes the object types that Stat Server monitors and how they relate to each other:

Refer to "Campaign Objects" on page 175 for descriptions of the Stat Server objects that are used to monitor agents and campaigns involved with the Outbound Contact Solution.

Object Descriptions

Object types each provide one aspect of a *statistical type* (stat type). Stat types are used to define a statistic. You specify objects within the `Objects` option of stat types (described on page 38). Object-type specification identifies which internal event model Stat Server uses in the acquisition of statistical values. Table 14 describes the types of objects that Stat Server monitors.

Table 14: Stat Server Object Types and Descriptions

Stat Server Object Type	Description
RegDN	Regular DN (directory number) applies to the following DN types: `data`, `music`, `mixed`, `extension`, `ACD position`, `Voice Treatment port`, `voice mail`, `cellular`, `CP` (call-processing equipment), and fax. Except for extensions and Voice Treatment ports, all of these DN types require login events.

Table 14: Stat Server Object Types and Descriptions (Continued)

Stat Server Object Type	Description
Agent	Stat Server tracks agents by a unique `Agent ID`, which is defined as `Employee ID` of `Person` objects in Configuration Manager. Even if the agent changes places, Stat Server can record the agent's activity. At any given time, each agent can be at only one place, and each place can be occupied by only one agent.
Place	Stat Server tracks the activity of a place by using a unique `PlaceID`. Even if various agents move in and out of a place, Stat Server can record the total activity for the place.
Queue	ACD-associated points at which calls wait for agent availability.
RoutePoint	Calls wait at these points for routing. These points have different names on different switching platforms (for example, `CDN`, `VDN`, and so forth).
GroupAgents	Designates a group of agents that is identified by a `GroupID`. An agent can be included in several groups. No matter where agents log in, their activity can be monitored as part of the group.
GroupPlaces	Designates a group of places. Each place that is part of the group has a unique `PlaceID`, which is associated with the `GroupID`.
GroupQueues	Designates a group that includes queues, routing points, virtual queues, and virtual routing points.
Virtual Queue	Designates a special type of DN that is maintained by a CTI installation and whose behavior is identical to that of a Routing Point.
Virtual Routing Point	Designates a special type of DN that is not associated with any particular target and where customer interactions wait while Universal Routing Server (URS) makes routing decisions.
RoutingStrategy	Designates a routing strategy that is deployed by the Interaction Routing Designer Genesys tool and is manifested in Configuration Server as a `Script` object of type, `CFGSimple Routing` or `CFGEnhancedRouting` subtypes.
StagingArea	Analogous to the concept of queues for the Multimedia (formerly known as Multi-Channel Routing, or MCR) solution in which customer interactions may reside while they are processed by Multimedia. This object type corresponds to the `Script` Configuration Server type, `CFGInteraction Queue` or `CFGInteractionWorkBin` subtypes.
Switch	A switch. You can collect only one piece of information for this object type; namely, the total number of hardware errors that occurred at the switch. Refer to Appendix A, "Predefined Statistical Types," on page 189 for an example of how to define this statistic.

Table 14: Stat Server Object Types and Descriptions (Continued)

Stat Server Object Type	Description
Tenant	An object that represents a business entity within the Configuration Server.
Workbin	A queue-like entity for storing multimedia interactions through which agents explicitly pull interactions for further processing. In Configuration Server, workbins are managed as `Script` objects of type `InteractionWorkbin`.

Object Hierarchy

Relationships are defined between various contact center objects within Configuration Server. DNs are defined to a switch. Queues might be assigned to queue groups. Agents might be affiliated with places, and so forth. Relationships can exist only between compatible objects. The listing of objects for which interrelationships could exist form an object's *compatibility group*. Table 15 shows those groups of objects whose members Stat Server considers to be potentially compatible.

Table 15: Stat Server Compatibility Groups

Regular DN Compatibility Group	Mediation DN Compatibility Group	Campaign Compatibility Group
RegDN Agent Place GroupAgents GroupPlaces Tenant	RoutePoint Queue GroupQueues Switch StagingArea	Campaign CallingList CampaignGroup CampaignCallingList

For telephony objects, some of these relationships are illustrated in Figure 8.

Objects' relationships defined within an Outbound campaign are illustrated in Figure 21 on page 176. The Stat Server Java Extensions calculate statistics for multimedia objects, such as `Workbin`, `StagingArea`, `Tenant`, and `Routing Strategy`. These objects are discussed no further in this document.

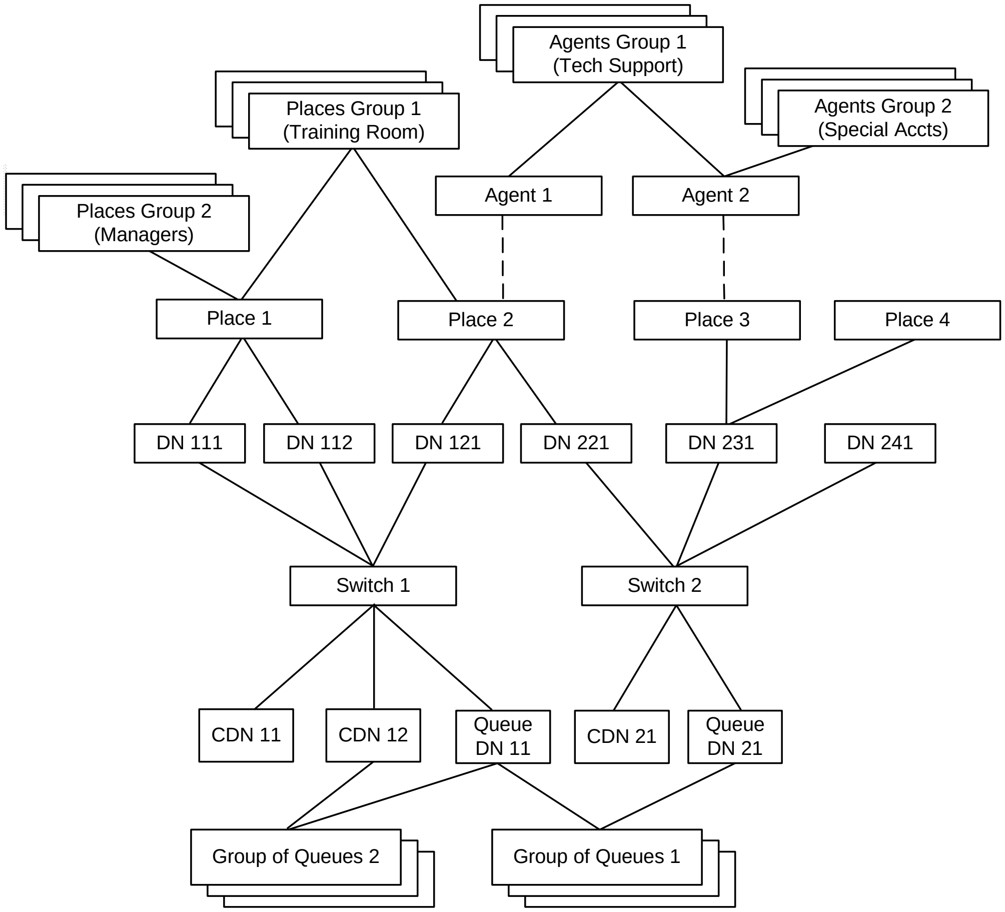


Figure 8: Hierarchy of Stat Server Telephony Objects

Associations Between Agents and Places

Stat Server creates an association between an agent and a place using both the configuration data for the corresponding objects in the Configuration Layer and the real-time events in the contact center. When an agent logs in to at least one DN or media-channel that is associated with a place, the following sequence takes place:

1. Stat Server takes a `LoginID` value from `EventAgentLogin`.
2. Stat Server compares the `LoginID` value against the agent login objects that are configured for the corresponding switch in the Configuration Layer:
   - If a matching `Agent Login` object exists, Stat Server checks whether it is assigned to any `Person` configuration object that has been configured as an agent (that is, with the `Is Agent` check box selected). The agent whose configuration contains the specified `Agent Login` is linked with the place.
   - If no matching `Agent Login` object exists, or if it exists without an association with any `Person` configuration object, Stat Server checks the configuration of all `Person` objects that have the `Is Agent` check

box selected. The agent whose configuration contains an `Employee ID` that matches the `LoginID` value from `EventAgentLogin` is linked with the place.

- If neither `Agent Login` nor `Employee ID` in the configuration matches the `LoginID` value from `EventAgentLogin`, Stat Server does not associate any agent with the place.

DN Association with Queues

For every queue, the login correspondence defines the list of DNs that currently are logged in to the queue. This correspondence also can be considered to define, for every regular DN, the list of queues to which the DN is currently logged in. The login correspondence between queues and regular DNs is updated whenever Stat Server receives `EventAgentLogin` with a nonnull value specified for the `ThisQueue` attribute, `EventQueueLogout`, or `EventAgentLogout` from T-Server.

When Stat Server receives `EventAgentLogin` for a DN, the list of queues becomes the union of the list of queues to which the DN was logged in before the event was received plus the set of queues that include the following:

- Any queue that is received if the `ThisQueue` attribute was received with the event.
- All queues that are listed in the Configuration Database as `OriginationDN` objects for groups of places that contain a place that is linked to the DN that received `EventAgentLogin` TEvent.
- All queues that are listed in the Configuration Database as `OriginationDN` objects for groups of agents that contain an agent who is logged in (after the event is received) at a place that is linked to the DN that received `EventAgentLogin`. (See page 66 for information about how Stat Server determines when an agent is logged in at a place.)

When Stat Server receives `EventQueueLogout`, it:

1. Adds a record to the `LOGIN` table of the Stat Server database.

   Stat Server only logs out the queue, and preserves the DN's association with other queues, if any, as well as its association with a particular agent.

2. Updates the affected, "logged-in" virtual agent groups by removing the agent from such groups.
3. Unlinks the `Queue` object from the `Place` object that is associated with the DN, by updating the `AgentLogin`, `AgentReady`, and `AgentActive` actions for the affected queue.
4. Unlinks the queue object from the DN that received the `EventAgentLogin` TEvent, by updating `DNLogin`, `DNReady`, and `DNActive` actions for the affected queue.

When Stat Server receives `EventAgentLogout` for a DN, the logged-in list of queues becomes empty.

Stat Server's support of the `EventQueueLogout` TEvent was introduced in the 7.0.3 release. The scenario below illustrates what Stat Server records to its database given different releases of T-Server and Stat Server.

Sample Database Entries Given Differing Component Versions

The records Stat Server writes to its `LOGIN` table differ, depending on the versions of T-Server and Stat Server deployed in this environment. Tables 16 through 18 illustrate the differences, given the following scenario:

On the `G3` switch, Agent Ryan has three login IDs which are assigned only to him:

- `2124` for logging into system
- `2126` for logging in to queue `8001`
- `2128` for logging in to queue `8002`

He is usually stationed at place `Sales21,` which has a phone with one DN configured—`601.`

On one particular day, Ryan arrives at work and logs in to DN `601` at 10:00 AM. At 10:01, he logs in to queue `8001` to start receiving the calls from this queue. Four minutes pass before he determines that he can handle calls from an additional queue, so he logs into to queue `8002` at 10:05. At 10:40, however, he concludes that he can no longer handle calls from both queues, so he immediately logs out of `8002.` At 10:50, he breaks for lunch and logs out of the system.

Stat Server writes records 4 and 5 (in Table 16) to the `LOGIN` table, because in T-Server 6.5, there was no notion of logout from just one queue—the `Event QueueLogout` TEvent did not exist. Instead, the model, at that time, called for the logging out of all queues (by sending the `EventQueueLogout` TEvent) and then the logging back in of the remaining queue(s).

Note: Tables 16 through 18 are presented in pseudo-table format. Refer to Refer to "The LOGIN Table" section in the Appendix of the *Framework 8.0 Stat Server Deployment Guide* for the actual format of this table.

Table 16: LOGIN Entries Given T-Server 6.5 and Stat Server 7.0.2 (or prior)

Record#	Switch	DN	Queue	Agent	Place	Status	Time	LoginID
1	G3	601		Ryan	Sales21	LoggedIn	10:00	2124
2	G3	601	8001	Ryan	Sales21	LoggedIn	10:01	2126
3	G3	601	8002	Ryan	Sales21	LoggedIn	10:05	2128
4	G3	601		Ryan	Sales21	LoggedOut	10:40	
5	G3	601	8001	Ryan	Sales21	LoggedIn	10:40	2126
6	G3	601		Ryan	Sales21	LoggedOut	10:50	

The latest version of T-Server 7.0 and forward releases, however, does send the `EventQueueLogout` TEvent—but Stat Server versions prior to 7.0.3 did not recognize it as noted in Table 17.

Table 17: LOGIN Entries Given T-Server 7.0+ and Stat Server 7.0.2 (or prior)

Record#	Switch	DN	Queue	Agent	Place	Status	Time	LoginID
1	G3	601		Ryan	Sales21	LoggedIn	10:00	2124
2	G3	601	8001	Ryan	Sales21	LoggedIn	10:01	2126
3	G3	601	8002	Ryan	Sales21	LoggedIn	10:05	2128
4	G3	601		Ryan	Sales21	LoggedOut	10:50	

In Table 18, notice that Stat Server does add record # 4 upon receipt of the `Event QueueLogout` TEvent. In doing so, Stat Server logs out neither the DN nor the place.

Table 18: LOGIN Entries Given T-Server 7.0+ and Stat Server 7.0.3+

Record#	Switch	DN	Queue	Agent	Place	Status	Time	LoginID
1	G3	601		Ryan	Sales21	LoggedIn	10:00	2124
2	G3	601	8001	Ryan	Sales21	LoggedIn	10:01	2126
3	G3	601	8002	Ryan	Sales21	LoggedIn	10:05	2128
4	G3	601	8002	Ryan	Sales21	LoggedOut	10:40	
5	G3	601	0	Ryan	Sales21	LoggedOut	10:50	

Stat Server's receipt of the `EventAgentLogout` TEvent logs out all queues and DNs to which Ryan was logged in. Stat Server does not write a queue value to the record upon receipt of `EventAgentLogout`.

General Notes About Objects

Queue DNs and Routing Points

Queue DN support is limited for some switches and environments. Please contact Genesys Technical Support for details. Routing points support much of the same statistics as do queues, although Stat Server generates actions for routing points based on a different set of events.

Groups of Queues and Routing Points

Within Configuration Manager, you can combine into groups DNs of the following types: `Routing Point`, `Queue`, `Virtual Routing Point`, `Virtual Queue`, and `Service Number`. You can include each DN in more than one group. A queue group object, `SObjectGroupQueues`, has the same set of statistics as a single queue or Routing Point object.

Chapter

4 Stat Server Actions

Any sequence of events that T-Server reports causes Stat Server to generate an *action*. The same is true for a limited number of events that Interaction Server reports. This chapter explains how Stat Server actions are classified and defined. This information pertains to the values that you might specify in the `MainMask` and/or `RelMask` option of a stat type (described on page 39.)

Information in this chapter is divided among the following topics:

Overview

Actions are the “information atoms” of Stat Server, all statistical values are ultimately based on:

- Data about the occurrence of Stat Server actions.
- Data attached to TEvents starting an action or occurring during an action.
- Where applicable, an action’s duration.

To make sense of any Stat Server statistic, you need to understand which actions are mapped to it and how they exist within a telephony environment.

This chapter classifies the general subdivisions of Stat Server actions and describes individual actions. For information about Stat Server actions related to campaigns, see Chapter 9.

Classifying DN Actions

At the uppermost level, actions can be segmented into one of the following three groups:

- Regular DN actions (generated from TEvents that are spawned from either T-Server or SIP Server)
- Mediation DN actions (generated from TEvents that are spawned from either T-Server or SIP Server)
- Media-channel actions (exclusively generated from events that are spawned from an Interaction Server)

Within each group, actions can be subclassified further as having the following properties:

- *Durable* or *instantaneous*
- Related to an interaction or not related to an interaction

The `CallRinging` action, for example, can be classified as a durable, interaction-related, regular DN action.

Regular DN actions, generated on multimedia DNs, are subdivided further into *media-dependent* and *media-independent* actions. An action is media-dependent if the `MediaType` attribute is applicable to the action and media-independent otherwise. `LoggedIn` is media-independent while all call-related actions, like `CallInbound,` are media-dependent.

Media-dependent actions are either *media-unique* or *media-common.* An action is media-unique if only one action per supported media type can exist on a multimedia device and media-common otherwise. These terms apply only within the context of multimedia DNs and were introduced in Stat Server release 7.6.1 to illustrate the difference between actions generated on regular DNs and actions sharing the same name which Stat Server generates on multimedia DNs.

The following sections describe these classifications.

Uppermost Classification of DN Actions

Stat Server generates actions for the following high-level classifications of DNs.

- *Regular DNs* are DNs such as telephony DNs (`Extension` and `ACD Position`), Internet DNs (`Email`, `VoIP`, `Video`, `Chat`, and `CoBrowse`), and special types of telephony DNs (`EAPort`, `VoiceMail`, `Cellular`, `CP`, `FAX`, `Data`, `Music`, `Mixed`).

- *Multimedia DNs,* controlled by a SIP Server, enable more than one simultaneous interaction, of the same or differing media type, to occur at the same DN. This can be exemplified by an agent handling multiple chat sessions simultaneously. Stat Server recognizes a DN as a Multimedia DN if:
  - The DN's type is `Extension` (`CFGExtension` in Configuration Server).
  - The DN's switch is `SIPSwitch` or `VoIPCMCPSwitch`.
  - The value of the DN's [Tserver]\`multimedia` configuration option value has been set to `yes` (This option is defined under the `Annex` tab in Configuration Manager).

    Note: Changing the value of the `multimedia` option in a DN's properties from `yes` to `no`, and vice versa, leads to a change in the DN's type—from multimedia DN to regular DN, and vice versa. Any such reconfiguration *must* be performed while Stat Server is not running.
- *Mediation DNs* are DNs that regularly distribute interactions, such as ACD queues, routing points, virtual queues, virtual routing points, external routing points, service numbers, and workbins.

Media-channel actions are all sourced from the Genesys Multimedia solution, which follows an entirely different, non-DN-based interaction model.

The special agent group and place group actions, which occur only at the group level, reflect events from origination DNs. They are formally classified with regular DN actions, because all other agent or place group actions are propagated regular-DN actions (see page 66).

Durable Actions Versus Instantaneous Actions

Durable actions occur over time; they have a starting moment and an ending moment.

Status can only be based on durable actions.

Instantaneous actions occur at a single moment and are divided into two groups: retrospective and momentary:

- *Retrospective actions* are generally derived from durable actions and are determined by the termination of the corresponding durable actions. Thus, a durable action's total duration is also a retrospective action's total duration, but a retrospective action's occurrence is unknown until the durable action ends. For instance, the termination of the `CallOnHold` durable action can result in one of three retrospective actions: `CallRetrievedFromHold`, `TransferredFromHold`, or `CallAbandonedFromHold`. However, these three actions can occur only when the interaction is no longer on hold.

  The only retrospective actions that do not derive from durable actions are the following mediation DN actions:
  - `CallTreatmentCompleted` (see page 103)—This action's duration is taken from the parameters of `EventTreatmentCompleted`.

  - All actions reflecting events at mediation DNs that have been distributed from other mediation DNs. Refer to the `CallDistributedToQueue` actions that are described on page 111.
  - All actions reflecting events at regular DNs receiving calls distributed from the mediation DN (see page 104)—these actions take their duration either from the preceding `CallWait` durable action (see page 100) or from a related regular-DN durable action.

> Note: All actions specifically called out as *retrospective* are instantaneous actions.

- *Momentary actions* are generally not derived from durable actions, and their duration is always `0.` An interaction-related durable action generally has a corresponding momentary action that marks the beginning of the durable action. For instance, the momentary `CallHeld` action marks the beginning of the `CallOnHold` durable action.

Interaction-Related Actions Versus Non–Interaction-Related Actions

Actions that reflect events arising from particular interactions (events that carry connection ID information from T-Server or interaction ID information from Interaction Server) are called *interaction-related actions*. Because Stat Server remembers the connection ID (or interaction ID) of the interaction, and because the connection ID (interaction ID) provides a criterion for distinguishing between such actions, more than one interaction-related action of the same kind can occur at the same time on the same DN.

Non–interaction-related actions are caused by events that do not arise from particular interactions. Only one non–interaction-related action can occur at any moment at a DN. Filtered and custom-formula statistics cannot be based on non–interaction-related actions.

Logical Clusters of Interaction-Related Actions

Almost every interaction-related durable action forms the core of a cluster of logically related actions. This cluster comprises the durable action itself, the momentary action that marks the starting point of the durable action, and one or more retrospective actions that carry information about the outcome of the durable action.

The interaction-related durable actions that do *not* form a cluster of logically related actions include:

- The two complementary call-type actions of `CallInternal`: `CallInternalOriginated` and `CallInternalReceived.`

- The two complementary call-type actions of CallConsult: CallConsultOriginated and CallConsultReceived.
- The AfterCallWork action.

Table 19 lists many of the basic actions that make up clusters of logically related actions. Each row in the table comprises all the actions in a single cluster.

Table 19: Logical Clusters of Interaction-Related Actions

Durable Action	Initial Momentary Action	Terminal Retrospective Actions
Regular DN Actions		
CallDialing (page 81)	CallDialingStarted (page 92)	CallDialed (page 88) CallAbandonedFromDialing (page 86) CallDialTransferred (page 88—only possible for consult calls) CallDialConferenced (page 88—only possible for consult calls)
CallRinging (page 84)	CallRingingStarted (page 94)	CallAnswered (page 87) CallAbandonedFromRinging (page 87) CallRingingPartyChanged (page 90—only possible for consult calls) CallForwarded (page 89)
CallOnHold (page 84)	CallHeld (page 93)	CallRetrievedFromHold (page 90) CallAbandonedFromHold (page 86) TransferredFromHold (page 91)
CallConsult (page 80) CallConsultOriginated (page 81) CallConsultReceived (page 81)	CallConsultStarted (page 92)	CallPartyChanged (page 90)
CallInbound (page 82)	CallInboundStarted (page 93)	CallInboundCompleted (page 89)
CallInternal (page 82)	CallInternalStarted (page 93)	CallInternalCompleted (page 89)
CallOutbound (page 84)	CallOutboundStarted (page 93)	CallOutboundCompleted (page 89)
CallUnknown (page 85)	CallUnknownStarted (page 95)	CallUnknownCompleted (page 90)
Group Actions Reflecting Origination DNs		
OrigDNCallWait (page 96)	OrigDNCallEntered (page 97)	OrigDNCallDistributed (page 97) OrigDNCallAbandoned (page 97)

Table 19: Logical Clusters of Interaction-Related Actions (Continued)

Durable Action	Initial Momentary Action	Terminal Retrospective Actions
Mediation DN Actions		
CallWait (page 100)	CallEntered (page 101)	CallDistributed (page 103) CallAbandoned (page 102) CallCleared (page 102)
Media Actions		
DeliveringStarted (page 116)	Delivering (page 113)	Accepted (page 119) Rejected (page 119)
HandlingInboundStarted (page 116)	HandlingInbound (page 114)	Accepted (page 119)
HandlingInternalStarted (page 116)	HandlingInternal (page 114)	Accepted (page 119)
HandlingOutboundStarted (page 116)	HandlingOutbound (page 114)	Accepted (page 119)
HandlingUnknownStarted (page 117)	HandlingUnknown (page 114)	Accepted (page 119)

For every cluster of logically related actions shown in Table 19 (except the Media actions), there are five clusters of interaction-type specific actions whose names are the same as those actions in the basic cluster, but with `Unknown`, `Inbound`, `Consult`, `Internal`, or `Outbound` appended to the end. The `CallDialTransferred` and `CallDial Conferenced` actions are specific to consult calls, so they occur without name modification in the cluster based on `CallDialingConsult`, and have no counterpart in the clusters specific to other call-type actions. The same is true for `CallRingingPartyChanged`. Each of the clusters corresponding to the `CallRingingConsult`, `CallWaitConsult`, and `OrigDNCallWaitConsult` durable actions has an additional terminating retrospective action (`CallRingingParty Changed`, `CallWaitPartyChanged`, and `OrigDNCallWaitPartyChanged`, respectively). These actions account for the possible termination of a consult call during two-step transfers. `CallDial Transferred` can occur only for a consult call.

Normally, at the end of a cluster's durable action, the durable action ends (and thus can be used for historical statistics), and a retrospective action that has the same duration occurs. However, when an interaction-related durable action ends because of a lost connection to T-Server or between T-Server and the switch (in either case, the `NotMonitored` action starts), none of the terminating retrospective actions of the same cluster occurs.

Summary of Stat Server Actions

Table 20 summarizes the actions that Stat Server recognizes and processes, and it provides the high-level classifications for each action. In this table, regular DN, SIP DN, and media-channel objects refer to the following object types:

- `Agent`
- `GroupAgents`
- `Place`
- `GroupPlaces`
- `RegDN`

Mediation DN objects refer to the following object types:

- `Queue`
- `GroupQueues`
- `Routepoint`
- `Workbin`

Campaign-related Stat Server actions are provided on page 177. Names in Table 20 that are followed by `[c-t]` represent a group of actions that include one or more of the following call (or interaction) types: `Consult`, `Inbound`, `Internal`, `Outbound`, and `Unknown`.

Table 20: Stat Server Actions

Action Name	Page(s)	Applies To...			Ixn-Related	Durable	Instantaneous	
		Regular DN/SIP DN Objs	Agt/Place Media Channels	Mediation DN Objects			Moment	Retro
Accepted	119		✓		✓			✓
Active	112		✓			✓		
ACWCompleted[a]	105			✓	✓			✓
ACWMissed*	105			✓	✓			✓
AfterCallWork [c-t]	73	✓				✓		
AgentActive	99			✓		✓		
AgentLogin	99			✓		✓		
AgentLogin	95	✓					✓	
AgentLogout	95	✓						✓
AgentReady	99			✓		✓		
ASM_Engaged	78	✓			✓	✓		
ASM_Outbound	79	✓			✓	✓		
Available	112		✓			✓		
BeingCoached	115	✓			✓		✓	
BeingMonitored	115		✓		✓		✓	
Blocked	113		✓			✓		
CallAbandoned [c-t]	102			✓	✓			✓
CallAbandonedFromDialing [c-t]	86	✓			✓			✓
CallAbandonedFromHold [c-t]	86	✓			✓			✓
CallAbandonedFromRinging [c-t]	87	✓			✓			✓
CallAbandonedFromRinging [c-t]	105, 121			✓	✓			✓
CallAnswered [c-t]	87	✓			✓			✓
CallAnswered [c-t]	105, 121			✓	✓			✓
CallCleared [c-t]	102			✓	✓			✓
CallConferenceJoined	92	✓			✓		✓	
CallConferenceMade	92	✓			✓		✓	
CallConferenceOriginated	80	✓			✓	✓		

Table 20: Stat Server Actions (Continued)

Action Name	Page(s)	Applies To...			Ixn-Related	Durable	Instantaneous	
		Regular DN/SIP DN Objs	Agt/Place Media Channels	Mediation DN Objects			Moment	Retro
CallConferencePartyAdded	92	✓			✓		✓	
CallConferencePartyDeleted	92	✓			✓		✓	
CallConsult	80	✓			✓	✓		
CallConsultCompleted	88	✓			✓			✓
CallConsultOriginated	81	✓			✓	✓		
CallConsultReceived	81	✓			✓	✓		
CallConsultStarted	92	✓			✓		✓	
CallDialConferenced	88	✓			✓			✓
CallDialed [c-t]	88	✓			✓			✓
CallDialing	81	✓			✓	✓		
CallDialingStarted [c-t]	92	✓			✓		✓	
CallDialTransferred	88	✓			✓			✓
CallDistributed [c-t]	103			✓	✓			✓
CallDistributedToQueue [c-t][b]	111			✓	✓			✓
CallEntered [c-t]	101			✓	✓		✓	
CallForwarded [c-t]	89	✓			✓			✓
CallForwarded [c-t]	106			✓	✓			✓
CallHeld	93	✓			✓		✓	
CallInbound	82	✓			✓	✓		
CallInboundCompleted	89	✓			✓			✓
CallInboundStarted	93	✓			✓		✓	
CallInternal	82	✓			✓	✓		
CallInternalCompleted	89	✓			✓			✓
CallInternalOriginated	82	✓			✓	✓		
CallInternalReceived	83	✓			✓	✓		
CallInternalStarted	93	✓			✓		✓	
CallMissed	106			✓	✓			✓
CallObserved...	83	✓			✓	✓		
CallOnHold	84	✓			✓	✓		
CallOutbound	84	✓			✓	✓		
CallOutboundCompleted	89	✓			✓			✓
CallOutboundStarted	93	✓			✓		✓	
CallPartyChanged	90	✓			✓			✓
CallReleased	106, 121			✓	✓			✓
CallRetrievedFromHold	90	✓			✓			✓
CallRinging [c-t]	84	✓			✓	✓		
CallRingingPartyChanged	90	✓			✓			✓
CallRingingStarted [c-t]	94	✓			✓		✓	
CallTransferMade [c-t]	94	✓			✓		✓	
CallTransferPartyChanged	94	✓			✓		✓	
CallTransferTaken	94	✓			✓		✓	
CallTreatmentCompleted	103			✓	✓			✓
CallTreatmentNotStarted	101			✓	✓		✓	
CallTreatmentStarted	101			✓	✓		✓	
CallUnknown	85	✓			✓	✓		
CallUnknownCompleted	90	✓			✓			✓
CallUnknownStarted	95	✓			✓		✓	

Table 20: Stat Server Actions (Continued)

Action Name	Page(s)	Applies To...			Ixn-Related	Durable	Instantaneous	
		Regular DN/SIP DN Objs	Agt/Place Media Channels	Mediation DN Objects			Moment	Retro
CallWait [c-t]	100			✓	✓	✓		
CoachingByIntrusionInitiated	115		✓		✓		✓	
CoachingByRequestInitiated	115		✓		✓		✓	
CoachingRequested	115		✓		✓		✓	
ConferenceJoined	116		✓		✓		✓	
ConferenceJoinedByIntrusion	116		✓		✓		✓	
ConferenceMade	116		✓		✓		✓	
Delivering	113		✓		✓	✓		
DeliveringStarted	116		✓		✓		✓	
DNActive	99			✓		✓		
DNLogin	99			✓		✓		
DNReady	99			✓		✓		
Handling	113		✓		✓	✓		
HandlingInbound	114		✓		✓	✓		
HandlingInboundStarted	116		✓		✓		✓	
HandlingInternal	114		✓		✓	✓		
HandlingInternalStarted	116		✓		✓		✓	
HandlingOutbound	114		✓		✓	✓		
HandlingOutboundStarted	116		✓		✓		✓	
HandlingStarted	117		✓		✓		✓	
LoggedIn	70	✓				✓		
LoggedOut	70	✓				✓		
Monitored	70	✓				✓		
Monitored	100			✓		✓		
MonitoringInitiated	117		✓		✓		✓	
NotAvailable	113		✓			✓		
NotMonitored	70	✓				✓		
NotMonitored	100			✓		✓		
NotReadyForNextCall	72	✓				✓		
OffHook	71	✓				✓		
OnHook	70	✓				✓		
OrigDNCallAbandoned [c-t][c]	97	✓			✓			✓
OrigDNCallDistributed [c-t][‡]	97	✓			✓	✓		✓
OrigDNCallEntered [c-t][‡]	97	✓			✓	✓		✓
OrigDNCallWait [c-t][‡]	96	✓				✓		
Pulled	117		✓		✓		✓	
Rejected	119		✓		✓			✓
Revoked	119		✓		✓			✓
Started [c-t]	117		✓		✓		✓	
Stopped [c-t]	120		✓		✓			✓
StuckCallCleaned	103			✓	✓			✓
StuckCallCleanedWhileRinging	91	✓			✓			✓
StuckCallCleanedWhileRinging	107			✓	✓			✓
TransferMade	118		✓		✓		✓	
TransferredFromHold	91	✓			✓		✓	
TransferTaken	119		✓		✓		✓	
UserEvent	96	✓					✓	

Table 20: Stat Server Actions (Continued)

Action Name	Page(s)	Applies To...			Ixn-Related	Durable	Instantaneous	
		Regular DN/SIP DN Objs	Agt/Place Media Channels	Mediation DN Objects			Moment	Retro
UserEvent	102			✓			✓	
WaitForNextCall	71	✓				✓		

a. This reflects actions occurring at regular DNs that Stat Server generates to mediation DNs.

b. This reflects actions occurring at mediation DNs that Stat Server generates to another mediation DN.

c. This group action reflects actions occurring at origination DNs that Stat Server generates to regular DNs.

Note: Although the names of many actions presume the processing of a voice interaction, these actions might also apply to other types of interactions —for example, chat sessions and e-mail.

Propagation of DN Actions

Every DN action propagates to higher-level objects.

For a regular DN, a DN action propagates to the place to which the DN is linked in the configuration. From the place, the action propagates to:

- The agent logged in at that place if there is such an agent.
- Place and agent groups comprising the place or the agent.

The action is considered to occur at the DN and at all objects above it, as illustrated in Figure 9.

A mediation DN action propagates to all groups of queues comprising the DN where the action occurs.

Connections Between Agents and Places

The propagation scheme shown in Figure 9 includes the dynamic connection between agent and place. The general rule is that when an agent is logged in at a place, the identical actions that occur for the agent also occur for the place. When an agent is not logged in anywhere, no actions are attributed to that agent.

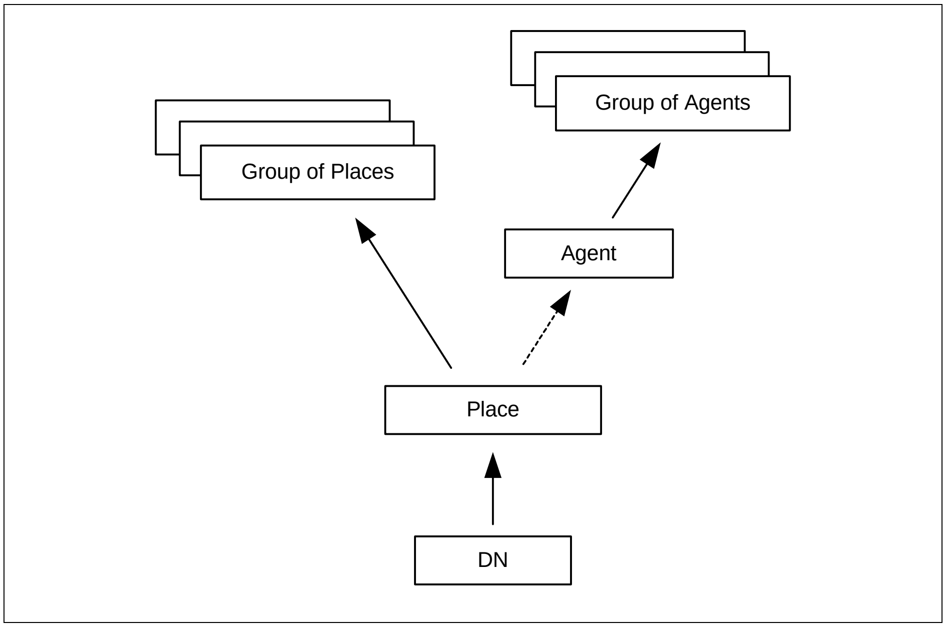


Figure 9: Propagation Hierarchy of Regular DN Action

In the Stat Server model, an agent can only be logged in at one place and only one agent can be logged in at one place at the same time. Stat Server uses the following rules in tracking the agent-place connection:

- When an agent who is not logged in at a place logs in at a place's DN, he or she becomes logged in at that place.
- When an agent logged in at a place logs in at another place, s/he is no longer logged in at the former place.
- When an agent logs in at a place where another agent is already logged in, the latter agent is no longer logged in at the place.
- When an agent is logged in at a place, and s/he logs out from the place's last DN where s/he has been logged in, the agent is no longer logged in anywhere.

`Warning!` Do not configure your system to allow more than one agent to log in at the same place or to allow the same agent to log in at more than one place; otherwise, Stat Server might fail to collect accurate information at the agent level.

Validity of Statistics

Stat Server reports a statistic as *invalid:*

- Whenever a DN propagated to that object changes its status to `NotMonitored` after all DNs propagated to the object had been in `Monitored` status.

- Whenever a statistical request is received for an object for which the last report was a status of invalid.

Stat Server reports a statistic as *valid* when the status of all DNs propagated to the object returns to Monitored.

Action Descriptions

Action descriptions contain information on how T-Server events cause Stat Server to generate actions. The actions, which start after DNs newly register, are determined by the data received with EventRegistered and, possibly, EventAddressInfo. This initialization, described in the next section, applies when Stat Server connects to T-Server for the first time, and when a lost connection is restored between Stat Server and T-Server or between T-Server and its switch.

DN Actions at Newly Registered DNs

When the Monitored action starts at a switch's DNs, Stat Server expects to receive the EventRegistered TEvent for every DN. If Stat Server receives an error instead of EventRegistered for a particular DN, Stat Server waits for any non-error event on behalf of this DN before resuming normal handling of event processing on this DN. Prior to the 7.0.3 release, Stat Server would not monitor such DNs at all.

If Stat Server receives EventRegistered for a DN without the Extensions attribute, Stat Server issues the TQueryAddress T-Library request for that DN and expects EventAddressInfo with info_type equal to AddressInfoDNStatus. The following regular DN actions can be affected by these events:

- LoggedOut—if LoggedOut is going on and EventRegistered or EventAddressInfo reports an AgentID for the DN, LoggedOut ends.
- WaitForNextCall, NotReadyForNextCall, and AfterCallWork:
    - If NotReadyForNextCall or AfterCallWork is going on at a DN for which EventRegistered or EventAddressInfo reports an AgentStatus of 2 (READY), then NotReadyForNextCall or AfterCallWork ends and WaitForNextCall starts.
    - If WaitForNextCall or AfterCallWork is going on at a DN for which EventRegistered or EventAddressInfo reports an AgentStatus of 3 (NOT_READY), then WaitForNextCall or AfterCallWork ends and NotReadyForNextCall starts.
    - If WaitForNextCall or NotReadyForNextCall is going on at a DN for which EventRegistered or EventAddressInfo reports an AgentStatus of 4 (ACW), then WaitForNextCall or NotReadyForNextCall ends and AfterCallWork starts. In this case, AfterCallWork is not interaction-related—that is, it has no attached ConnID and no corresponding call-type action.

  - If WaitForNextCall or AfterCallWork is going on at a DN for which EventRegistered or EventAddressInfo reports an AgentStatus of 5 (Walk_Away), then WaitForNextCall or AfterCallWork ends and NotReadyForNextCall starts.
- CallUnknown, CallInternal, CallInternalOriginated, CallInternalReceived, CallInbound, CallOutbound, CallConsult, CallConsultOriginated, CallConsultReceived, CallUnknownStarted, CallInternalStarted, CallInboundStarted, CallOutboundStarted, and CallConsultStarted—one of the five momentary actions occurs and its corresponding durable action starts as soon as EventAddressInfo with info_type equal to AddressInfoCallsQuery reports the ongoing call type.

Regular DN Actions

Regular DN actions fall into the following categories:

- Durable, non–interaction-related actions (page 69)
- Durable, interaction-related actions (page 78)
- Retrospective, interaction-related actions (page 86)
- Momentary, interaction-related actions (page 91)
- Momentary, non–interaction-related actions (page 95)
- Durable group actions reflecting origination DNs (page 96)
- Retrospective group actions reflecting origination DNs (page 96)
- Momentary group action reflecting origination DNs (page 97)

The subsections below describe the one or more actions comprising each category.

Durable, Non–Interaction-Related Actions

The following are the durable, non–interaction-related actions that Stat Server generates to regular DNs:

- NotMonitored
- Monitored
- LoggedIn
- LoggedOut
- OnHook
- OffHook
- WaitForNextCall
- NotReadyForNextCall
- AfterCallWork

Note: AfterCallWork can be related to an interaction or not. Hence, this action is listed both in this section and in the “Durable, Interaction-Related Actions” section on page 78.

NotMonitored

This durable action begins whenever Stat Server is not connected to the T-Server or SIP Server controlling the switch where the DN is located (Stat Server receives the `EventServerDisconnected` TEvent in this case), or when the link between the T-Server (or SIP Server) and the switch is down (T-Server sends the `EventLinkDisconnected` TEvent). `NotMonitored` ends when both connections are up and running. Its complementary action is `Monitored`—one and only one of these actions can occur for any DN at any given moment. The `NotMonitored` action terminates every other DN action; no other DN action can start while `NotMonitored` is occurring.

Of special note, if Stat Server receives `EventOutOfService` for a particular DN (such as might be the case if the DN's switch is being reconfigured), the `Not Monitored` action occurs, and it persists until Stat Server detects `EventBackIn Service` for that DN. At that point, the `NotMonitored` action ceases.

Monitored

This durable action starts whenever `NotMonitored` terminates—that is, when Stat Server is connected to T-Server or SIP Server and the link between T-Server (or SIP Server) and the switch is up. This action ends when the `NotMonitored` action starts.

LoggedIn

This durable action starts when Stat Server detects agent login on a DN:

- The `EventAgentLogin` TEvent is received on a DN.
- Either the `EventRegistered` or `EventQueryAddress` TEvent is received on a DN for which the `Extensions` attribute contains the pair, `(“AgentStatus”, *value*),` where *`value`* is greater than zero (`0` signifies `LoggedOut`).

This action ends with `EventAgentLogout` or when the `NotMonitored` action starts.

LoggedOut

This durable action starts with `EventAgentLogout` and ends either with `EventAgentLogin` or when the `NotMonitored` action starts. For multimedia DNs, this action is classified as media-independent.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

OnHook

This durable action starts when Stat Server receives `EventOnHook` from T-Server or SIP Server, and ends when Stat Server receives `EventOffHook` or the NotMonitored action starts. This action is specific to a limited number of switches, and only DNs corresponding to physical telephones should be set to generate the corresponding TEvents. For such DNs, OnHook and OffHook are complementary while Monitored occurs. For multimedia DNs, this action is classified as media-independent.

Note: Stat Server ignores EventOnHook TEvent notifications if the ignore-off-hook-on-position Stat Server configuration option is set to true and the DN's type is Position.

OffHook

This durable action starts when Stat Server receives EventOffHook from T-Server or SIP Server, and ends when Stat Server receives EventOnHook or the NotMonitored action starts. For DNs that generate these events, OnHook and OffHook are complementary while Monitored occurs. For multimedia DNs, this action is classified as media-independent.

Note: Stat Server ignores EventOffHook TEvent notifications if the ignore-off-hook-on-position Stat Server configuration option is set to true and the DN's type is Position.

WaitForNextCall

This durable action occurs for a particular DN, regardless of media channel, if all of the following conditions are met:

- Monitored occurs.
- The last TEvent to arrive after any of the following TEvents is EventAgent Ready:
    - EventAgentLogin
    - EventAgentNotReady
    - EventRegistered
    - EventAddressInfo reports agent status 2 (Ready)
- Either EventDNDOn is never received, or the last event from the pair EventDNDOn and EventDNDOff is EventDNDOff.

The only exceptions to this rule are the DNs of type Extension (not multimedia DNs, see definition of multimedia DN on page 58) or Voice Treatment port, for which the WaitForNextCall action starts as soon as the DN is registered.

Note: See also "DN Actions at Newly Registered DNs" on page 68.

WaitForNextCall ends on a DN when any of the following occurs:

- Stat Server receives EventRegistered or EventAddressInfo with reports of agent status equal to any of the following:
  - 0 (LoggedOut)
  - 3 (NOT_READY)
  - 4 (ACW)
  - 5 (Walk_Away)
- Stat Server receives EventDNDon.
- Stat Server receives EventDNOutOfService.
- Stat Server receives EventAgentNotReady with any work mode.
- Stat Server receives EventAgentLogout.
- The NotMonitored action starts.

While Monitored occurs, the actions WaitForNextCall, NotReadyForNextCall, and AfterCallWork are complementary.

For multimedia DNs, this action is classified as media-dependent, media-unique.

Note: Agents cannot selectively make some media channels of a DN ready or not ready. These states apply to all of a DN's media channels.

NotReadyForNextCall

This durable action is complementary to WaitForNextCall while the Monitored action occurs at the DN in question. Thus, NotReadyForNextCall occurs when Monitored occurs and one of the following conditions is met:

- Stat Server receives EventRegistered or EventAddressInfo with reports of agent status equal to either of the following:
  - 1 (LOGGED_IN)
  - 3 (NOT_READY)
- Stat Server receives EventAgentLogin.
- Stat Server receives EventAgentNotReady with Workmode!=ACW while the agent is logged in.
- Stat Server receives EventDNDon.

The NotReadyForNextCall action ends when any of the following occur:

- Stat Server receives EventAgentReady (the WaitForNextCall action begins).
- Stat Server receives EventAgentNotReady with WorkMode=ACW (after-call work begins).
- Stat Server receives EventDNDoff while the agent is logged out, ready, or not ready with Workmode=ACW.
- The NotMonitored action starts.

For multimedia DNs, this action is classified as media-dependent, media unique.

Note: Agents cannot selectively make some media channels of a DN ready or not ready. These states apply to all of a DN's media channels. For multimedia DNs, when conditions are met, Stat Server globally generates or ends the NotReadyForNextCall action for all enabled media channels supported by that DN.

AfterCallWork

This durable action is specific to particular switches and T-Server or SIP Server applications. For multimedia DNs, this action is classified as media-dependent, media unique. While an agent is not involved in calls, this action starts when Stat Server receives EventAgentNotReady with a WorkMode attribute of AfterCallWork on any of the enabled media channels of a DN. Stat Server cancels generation of an AfterCallWork action (where it was previously postponed) if any of the following occur:

- Stat Server receives the EventAgentNotReady TEvent with a work mode other than AfterCallWork.
- Stat Server receives the EventAgentReady or EventDNDOn TEvents.
- Stat Server receives the EventAgentLogout TEvent (the agent logs out).

If a switch permits an agent to enter AfterCallWork mode while still involved in calls, any call ending with this agent will invoke after-call work. Stat Server generates the AfterCallWork action upon completion of the interaction. This behavior occurs even if Stat Server receives EventNotReady TEvent with Workmode=ACW from T-Server. Stat Server postpones the AfterCallWork action upon termination of the interaction.

While AfterCallWork persists on a media channel of a multimedia DN, no routing is possible to that channel. (Stat Server marks the media_state component of the DN's capacity vector NR [NotReady].) Stat Server considers the actions occurring on all media channels when determining the DN's status. A DN's status is the highest ranking action occurring on all enabled media channels according to Stat Server's status priority tables.

If Stat Server receives EventNotReady TEvent with Workmode=ACW while the interaction is active, this action is simultaneous with one of the following call-type actions:

- AfterCallWorkUnknown
- AfterCallWorkInternal
- AfterCallWorkInbound
- AfterCallWorkOutbound
- AfterCallWorkConsult

The interaction type that Stat Server receives from T-Server together with EventReleased, determines which of the preceding five actions occurs simultaneously with `AfterCallWork`. If `AfterCallWork` starts after an interaction is released, none of these call-type actions occurs.

Starting with Release 7.0, Stat Server generates `AfterCallWork` actions only upon completion of the interaction. This behavior allows several statistics to be more independent from a T-Server (and/or a Desktop) implementation; one such statistic is that requested with the `ACWCompleted` action for queues.

Note: These changes do not affect Stat Server's MLink ACW emulation functionality, which is based on an entirely different TEvent set.

The diagram below illustrates the changes in the ACW calculation schema.

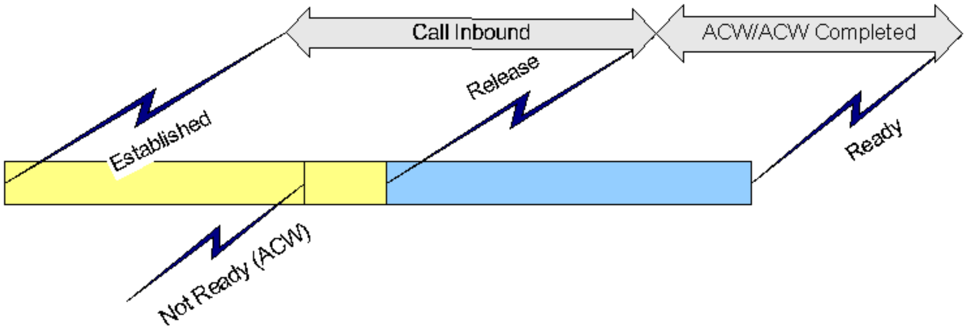


Figure 10: ACW Action Generated After Interaction Completion

Note: See also "DN Actions at Newly Registered DNs" on page 68.

After-Call Work on the Nortel Meridian T-Servers

This section describes how Stat Server processes ACW-related events when operating with the following T-Servers, subsequently referred to only as Meridian or Meridian-like T-Servers:

- Release 7.0$^+$ of T-Server for Nortel Meridian 1
- Release 7.0$^+$ of T-Server for Nortel Symposium Call Center
- Release 7.1$^+$ of T-Server for Nortel Communication Server 1000 with SCCS/MLS
- NEC-2400

Note: The switch type you set in the Configuration Layer when working with these T-Server applications depends on your PBX type and can be `Nortel Meridian 1`, `Nortel Meridian CallCenter/Symposium`, or `Nortel Communication Server 1000 with SCCS/MLS`.

Starting with release 7.0 of Meridian-like T-Servers, their ACW-related events are processed differently than they are with other T-Server types. The reason for the difference in processing is that the Meridian-like DN model is different from other DN models that Genesys supports. Unlike other models, this model consists of a Position and Extension DNs linked together.

- To indicate ACW, Meridian T-Server applications propagate an `EventAgentNotReady` TEvent with workmode=`ACW` the moment an agent requests after-call work functionality (that is, when he or she presses the ACW button). (Other T-Server applications propagate this TEvent upon completion or redirection of the interaction). Meridian T-Servers send this TEvent only for Position DN types—it does not send the event for Extension DNs. If no more than one Position/Extension pair is configured on a place, Stat Server logic links together Position and Extension DNs based on how the corresponding Place object is configured in Configuration Server.
- Meridian T-Servers propagate an additional `EventAgentNotReady` TEvent (workmode=ACW) if the agent changes the reason for being in ACW state.
- After-call work terminates when Stat Server receives from T-Server one of the following TEvents:
  - `EventAgentReady`
  - `EventAgentNotReady` (workmode!= `ACW`)
  - `EventAgentLogout`

Based on the `EventAgentNotReady` TEvent (with workmode=`ACW`), Stat Server generates an `AfterCallWork` action on the Position DN and links the action with the appropriate telephony interaction, if applicable. In addition, if Stat Server recognizes this after-call work as associated with a particular telephony interaction, Stat Server postpones generation of the `AfterCallWork` action until the interaction is released. Furthermore, Stat Server inherits UserData keys and their values from the interaction and allows filtering of `AfterCallWork` action through these keys. If reasons are attached to `EventAgentNotReady` TEvent (workmode= ACW), then Stat Server can use them in filtering. Furthermore, Stat Server reacts when reasons change, such as upon receipt of the subsequent `EventAgentNotReady` ACW TEvent.

Stat Server generates an `ACWCompleted` or `ACWMissed` action on the mediation DN when the interaction is directed to the Position or Extension DN via a queue or routing point.

The following examples illustrate the actions Stat Server generates following receipt of certain TEvents from a Meridian T-Server.

ACW with No Associated Interaction

Figure 11 illustrates a scenario where Stat Server immediately starts an `AfterCallWork` action on the Position DN upon receipt of the `EventAgentNotReady` TEvent (with workmode=`ACW`) from Meridian T-Server, and when there are no telephony interactions on the Position (or Extension) DN.

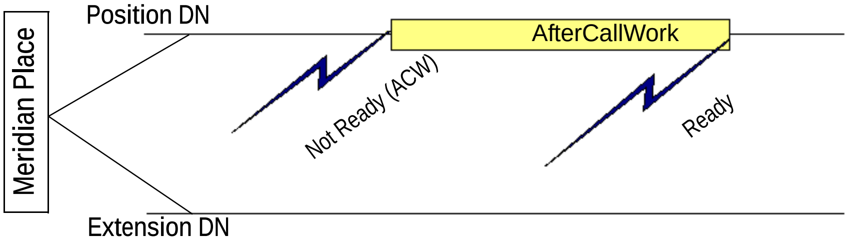


Figure 11: ACW Given No Telephony Interaction

The figure shows the events occurring on the Position DN where Stat Server starts an `AfterCallWork` action. Stat Server terminates it, in this example, upon receipt of an `EventAgentReady` TEvent. (The `EventAgentLogout` or `EventAgent NotReady` TEvents with workmode!= `ACW` would also terminate the action.)

Note: This scenario also applies for other T-Server applications that have only Position or only Extension DNs.

ACW Request During an Interaction

If a telephony interaction is currently in progress on a Position DN, the related Extension DN, or both, when Stat Server receives an ACW-related TEvent on that Position DN, Stat Server generates an `AfterCallWork` action on the Position DN only, and only after all calls complete on the Position and/or Extension DNs. Furthermore, Stat Server associates this action only with the last released interaction. Stat Server does not generate an `AfterCallWork` action on the Extension DN, regardless of where the last interaction took place.

Figure 12 illustrates this scenario when an inbound interaction is underway on the Position DN. An ACW-related event takes place during the interaction. Stat Server starts an `AfterCallWork` action when the interaction is released. Figure 13 illustrates the same scenario, but with the interaction taking place on the Extension DN.

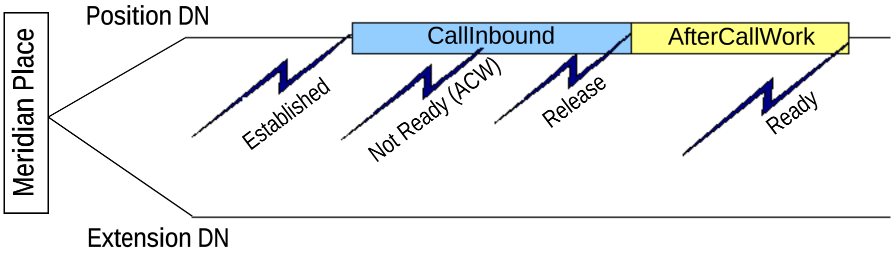


Figure 12: ACW Request on a Position DN During a Telephony Interaction

During the interaction on the Position DN, the agent presses the ACW button (workmode=`ACW`). Upon release of the interaction, Stat Server starts an `AfterCallWork` action on the Position DN. When Stat Server then receives the `EventAgent Ready` TEvent, Stat Server terminates the `AfterCallWork` action. (`EventAgentLogout` and `EventAgentNotReady` with a workmode other than `ACW` would also terminate the action.)

In Figure 13, the agent presses the ACW button (workmode=`ACW`) while conducting an interaction on the Extension DN. Stat Server starts an `AfterCallWork` action *on the Position DN* upon release of the interaction, and terminates it under the same circumstances as those stated above. This termination occurs regardless of the DN from which the `AfterCallWork` action is generated.

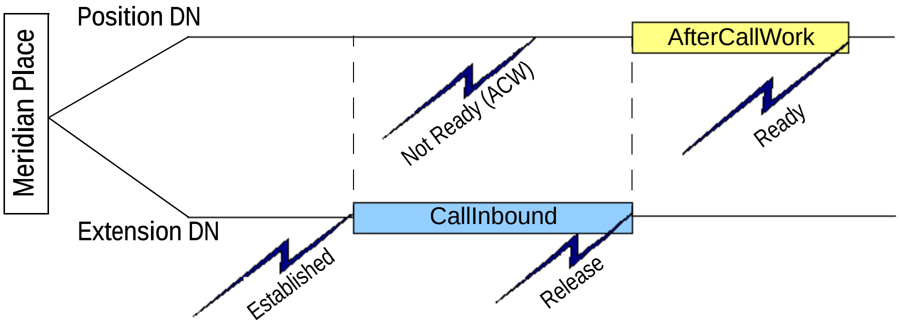


Figure 13: ACW Request on an Extension DN During a Telephony Interaction

Clearing ACW During an Interaction

As a special provision, if, after having previously received an `EventAgentNotReady` TEvent (with workmode=`ACW`) while one or more calls are in progress on either the Position or Extension DN, Stat Server receives an `EventAgentReady` or `EventAgentNotReady` TEvent (with workmode!=`ACW`) while one or more calls are still in progress, Stat Server does not generate an `AfterCallWork` action upon release of the subsequent interaction(s). Figure 14 illustrates this scenario.

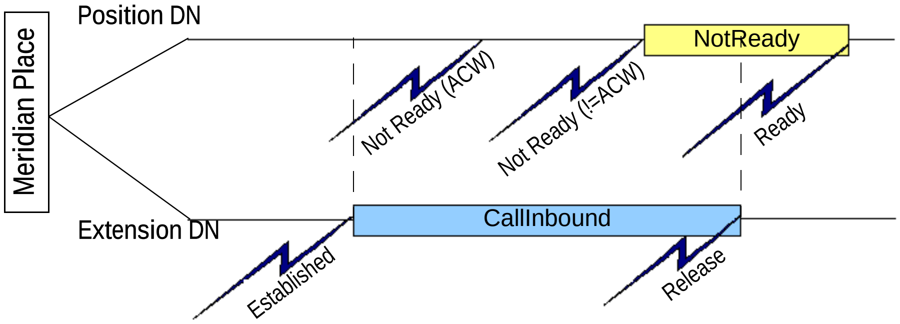


Figure 14: Clearing an ACW Request During an Interaction

The figures shows an interaction occurring on the Extension DN. During the interaction, the agent presses the ACW button. T-Server sends an `EventAgentNotReady` TEvent (workmode=`ACW`), and Stat Server registers it on the Position DN. Later, during the same interaction, the agent presses the `NotReady` button. T-Server sends an `EventAgentNotReady` TEvent (with workmode!=`ACW`), and Stat Server acknowledges it on the Position DN. As this TEvent and workmode combination terminate after-call work, Stat Server does not start an `AfterCallWork` action when the interaction terminates, but rather immediately starts a `NotReady` action on the Position DN when the `NotReady` button is pressed.

Note: This ACW model applies when Stat Server 7.0$^+$ is used in conjunction with Meridian T-Server 7.0. Contact Genesys Technical Support to understand Stat Server 7.0 behavior if the version of your Meridian T-Server is less than 7.0.

Durable, Interaction-Related Actions

The following are the durable, interaction-related actions that Stat Server generates to regular DNs:

- AfterCallWork
- ASM_Engaged
- ASM_Outbound
- CallConferenceOriginated
- CallConsult
- CallConsultOriginated
- CallConsultReceived
- CallDialing
- CallInbound
- CallInternal
- CallInternalOriginated
- CallInternalReceived
- CallObservedConsult
- CallObservedInternal
- CallObservedOutbound
- CallObservedInbound
- CallObservedUnknown
- CallOnHold
- CallOutbound
- CallRinging
- CallUnknown

Note: AfterCallWork can be related to an interaction or not. Hence, this action is listed both in this section and in the “Durable, Non–Interaction-Related Actions” section on page 69.

AfterCallWork

For more information about this durable action, see “AfterCallWork” on page 73.

ASM_Engaged

This durable action is specific to DNs of the Extension or Position type that are involved with the outbound predictive dialing, which runs in Predictive with seizing mode and is based on the Active Switching Matrix (ASM) call model.

This action starts upon Stat Server’s receipt of:

- EventEstablished on the communication port DN (CPDN).
- EventEstablished on the agent DN where its UserData attribute contains the <"GSW_CALL_TYPE","ENGAGING"> key-value pair.

Prior to Stat Server 7.6, this action started upon receipt of `EventRinging.` Now, upon receiving `EventRinging` with `ANI/OtherDN` pointing to the CPDN, Stat Server generates the `CallRinging` action.

N-Dialer makes a predictive dialing call to a customer number and delivers an engaging call (of the `Inbound` or `Internal` type) to an agent via a CPDN. The action indicates that the agent on a particular DN is waiting for the customer to be connected.

This action ends for communication port DNs when any of the following occur:

- The `ASM_Outbound` action starts on the CPDN.
- The customer is connected to the agent.
- Either the predictive dialing or the engaging call is released (through receipt of `EventReleased` or `EventAbandoned`) before the agent and the customer are connected to each other.
- The `NotMonitored` action starts.

This action ends for agent DNs when any of the following occurs:

- The `ASM_Outbound` action starts on the agent DN.
- Either the predictive dialing or the engaging call is released (through receipt of `EventReleased` or `EventAbandoned)` before the agent and the customer are connected to each other.
- The `NotMonitored` action starts.

`Note:` Refer to the *Outbound Contact 8.0 Deployment Guide* for information on the ASM call model.

ASM_Outbound

This durable action is specific to DNs of the `Extension` or `Position` type that are involved with the outbound predictive dialing, which runs in `Predictive with seizing` mode and is based on the Active Switching Matrix (ASM) call model.

This action starts upon Stat Server's receipt of:

- `EventAttachedDataChanged` on the CPDN with `UserData` containing the `(GSW_RECORD_HANDLE,`*`<any value>`*) key-value pair.
- `EventPartyChanged` on the agent DN with `PreviousConnID` pointing to a call that Stat Server recognizes as ASM-engaged and `UserData` containing the `<“GSW_CALL_TYPE,”REGULAR>` key-value pair.

This action ends on the CPDN when either the agent or the customer releases the call or if the `NotMonitored` action starts. On the agent DN, this action ends when the call ends on the agent's DN or when the `NotMonitored` action starts.

Note: Refer to the *Outbound Contact 8.0 Deployment Guide* for information on the ASM call model.

CallConferenceOriginated

This durable action measures the amount of time that an agent spent in a three-party conference. In regular PBX scenarios, this action starts when the originating agent invites another agent to a call (`EventPartyChanged`) and stops when the originating agent leaves the conference (`EventReleased`). The `CallConferenceOriginated` action is not supported in blind conferences when a conference completes while the call is at a routing point or ACD queue.

In network-attended transfer and conference scenarios, this action starts when Stat Server receives `NetworkCallStateConferenced` as the value of the `AttributeNetworkCallState` attribute for the originating agent and stops when this attribute's value becomes `NetworkCallStateReconnected`, `NetworkCallStateDisconnected`, `NetworkCallStateTransferred` or `NetworkCallStateConferenced` for the originating agent or when the `NotMonitored` action starts.

Note: When specified in the `MainMask` of a stat type, Stat Server ignores `DistByConnID` (`DCID`) `Formula` assignments, since, by definition, this action may occur only once for a given connection ID.

Statistics based on this action include the originating agent's continued involvement in conferenced calls, regardless of whether this involvement is active or inactive.

Note: Using this action to measure the originating agent's time in a three-party conference presumes that the originating agent leaves the conference first. If the customer or the conferenced-in agent leaves the conference, Stat Server continues to tally this metric until the originating agent leaves the transaction.

CallConsult

This durable action starts when Stat Server receives `EventEstablished` from a DN with a value of `Consult` for the interaction-type parameter. Call origination, whether from within the contact center or outside, is not indicated. This action's corresponding initial momentary action is `CallConsultStarted` (see page 92).

`CallConsult` ends with `EventReleased` or `EventPartyChanged` for the same call or when the `NotMonitored` action starts. When `CallConsult` ends with `EventReleased`, it causes the `CallConsultCompleted` retrospective action to occur. When `CallConsult` ends with `EventPartyChanged`, it causes the `CallPartyChanged` retrospective action (see page 90) to occur.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallConsultOriginated

This durable action starts when Stat Server receives `EventEstablished` from a DN with a value of `Consult` for the interaction-type parameter. This action is similar to a `CallConsult` action providing additional information about call origination—namely, from an agent’s DN. Its corresponding initial momentary action is `CallConsultStarted` (see page 92).

`CallConsultOriginated` ends with `EventReleased` or `EventPartyChanged` for the same call or when the `NotMonitored` action starts. When `CallConsultOriginated` ends with `EventPartyChanged`, this action causes Stat Server to generate the `CallPartyChanged` retrospective action (see page 90).

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallConsultReceived

This durable action starts when Stat Server receives `EventEstablished` from a DN with a value of `Consult` for the interaction-type parameter. This action is similar to a `CallConsult` action providing additional information about call origination—namely, from a DN outside the contact center. Its corresponding initial momentary action is `CallConsultStarted` (see page 92).

`CallConsultReceived` ends with `EventReleased` or `EventPartyChanged` for the same call or when the `NotMonitored` action starts. When it ends with `EventPartyChanged`, it causes the retrospective action `CallPartyChanged` (see page 90).

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallDialing

This durable action starts when Stat Server receives `EventDialing` from T-Server for a DN. Its corresponding initial momentary action is `CallDialingStarted` (see page 92).

This action lasts until Stat Server receives either `EventEstablished` or `EventReleased` for the same call, or until the `NotMonitored` action starts. If `EventEstablished` or `EventReleased` is received, and, in the latter case, if the interaction is a consult call with a call state of `Transferred` or `Conferenced`, the termination of `CallDialing` produces the retrospective action `CallDialed` (page 88), `CallAbandonedFromDialing` (page 86), `CallDialTransferred` (page 88), or `CallDialConferenced` (page 88).

CallDialing is always simultaneous with one of the following call-type actions:

- CallDialingUnknown
- CallDialingInternal
- CallDialingInbound
- CallDialingOutbound
- CallDialingConsult

The interaction type that Stat Server receives from T-Server with EventDialing determines which of the preceding five actions occurs simultaneously with CallDialing.

CallInbound

This durable action starts when Stat Server receives:

- EventEstablished.
- EventPartyChanged from a DN with a value of Inbound for the interaction-type parameter.

Its corresponding initial momentary action, upon receipt of EventEstablished, is CallInboundStarted (which is described on page 93). Stat Server generates this action upon receipt of EventPartyChanged when T-Server configuration causes T-Server to transmit an Inbound interaction type with the TEvent rather than Consult. This can happen, for example, when the use-data-from T-Server configuration option is set to consult-user-data.

CallInbound ends with EventReleased for the same interaction, causing the CallInboundCompleted retrospective action to occur, when EventPartyChanged is received for a different party, or when the NotMonitored action starts.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallInternal

This durable action starts when Stat Server receives EventEstablished from a DN with a value of Internal for the interaction-type parameter. Its corresponding initial momentary action is CallInternalStarted (see page 93).

CallInternal ends with EventReleased for the same interaction, causing the CallInternalCompleted retrospective action to occur, or when the NotMonitored action starts.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallInternalOriginated

This durable action starts when Stat Server receives EventEstablished from a DN with a value of Internal for the interaction-type parameter. This action is similar to a CallInternal action, providing additional information about interaction origination—namely, from an agent's DN. Its corresponding initial momentary action is CallInternalStarted (see page 93).

CallInternalOriginated ends with EventReleased for the same interaction or when the NotMonitored action starts.

Note: See also "DN Actions at Newly Registered DNs" on page 68.

CallInternalReceived

This durable action starts when Stat Server receives EventEstablished from a DN with a value of Internal for the interaction-type parameter. This action is similar to a CallInternal action, providing additional information about origination of the interaction—namely, from a DN not belonging to the agent. Its corresponding initial momentary action is CallInternalStarted (see page 93).

CallInternalReceived ends with EventReleased for the same interaction or when the NotMonitored action starts.

Note: See also "DN Actions at Newly Registered DNs" on page 68.

CallObserved...

The CallObserved... actions include the following:

- CallObservedUnknown
- CallObservedInternal
- CallObservedInbound
- CallObservedOutbound
- CallObservedConsult

One of these durable actions starts when Stat Server receives EventPartyAdded with ThisDNRole equal to Destination and OtherDNRole equal to Observer. The action terminates when T-Server reports EventPartyDeleted for the agent's DN with OtherDNRole equal to Observer, when it reports EventReleased for the interaction, or when the NotMonitored action starts.

Supervisor participation in an interaction does not affect the Service Observed statistics.

Note: For information on the T-Server call model, refer to the "Service Observing an Agent" section in the *T-Library SDK C Developer's Guide.* See also "DN Actions at Newly Registered DNs" on page 68.

CallOnHold

This durable action starts when Stat Server receives EventHeld from T-Server for a DN. Its initial momentary action is CallHeld (see page 93).

This action lasts until Stat Server receives either EventRetrieved or Event Released for the same interaction, or until the NotMonitored action starts. If Stat Server receives EventRetrieved or EventReleased and, in the latter case, if the interaction state is Transferred, termination of CallOnHold produces one of the following retrospective actions:

- CallRetrievedFromHold (page 90)
- TransferredFromHold (page 91)
- CallAbandonedFromHold (page 86).

CallOnHold is always simultaneous with one of the following call-type actions:

- CallOnHoldUnknown
- CallOnHoldInternal
- CallOnHoldInbound_
- CallOnHoldOutbound
- CallOnHoldConsult

The interaction type that Stat Server receives from T-Server with EventHeld determines which of the above five actions occurs simultaneously with CallOnHold.

When determining status, Stat Server temporarily hides from consideration the corresponding DN action (CallInternal, CallInbound, CallOutbound, or CallUnknown) of an established telephony interaction on the same DN for the duration that the interaction is on hold.

CallOutbound

This durable action starts when Stat Server receives EventEstablished from a DN with a value of Outbound for the interaction-type parameter. Its corresponding initial momentary action is CallOutboundStarted (see page 93).

CallOutbound ends with EventReleased for the same interaction, causing the CallOutboundCompleted retrospective action to occur, or when the NotMonitored action starts.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallRinging

This durable action starts when Stat Server receives either:

- EventRinging from T-Server for a DN or, for an interaction derived from a consult call, when Stat Server receives EventPartyChanged.

- EventPartyChanged in circumstances where T-Server configuration causes T-Server to transmit an Inbound interaction type with the TEvent rather than Consult, such as may be the case when the use-data-from T-Server configuration option is set to "consult-user-data".

Its initial momentary action is CallRingingStarted (see page 94).

CallRinging lasts until Stat Server receives:

- EventEstablished
- EventReleased
- EventPartyChanged for a consult call and for the same interaction

Or, until the NotMonitored action starts.

If EventEstablished, EventReleased, or, for a consult call, EventPartyChanged is received, the termination of CallRinging produces the retrospective action CallAnswered (page 87), CallAbandonedFromRinging (page 87), or CallRingingPartyChanged (page 90).

CallRinging is always simultaneous with one of the following call-type actions:

- CallRingingUnknown
- CallRingingInternal
- CallRingingInbound
- CallRingingOutbound
- CallRingingConsult

The interaction type that Stat Server receives from T-Server with EventRinging determines which of the above five actions occurs simultaneously with CallRinging.

CallUnknown

This durable action starts when Stat Server receives EventEstablished from a DN with a value of Unknown for the interaction-type parameter. Its corresponding initial momentary action is CallUnknownStarted (see page 95).

CallUnknown ends with EventReleased for the same interaction, causing the CallUnknownCompleted retrospective action to occur, or when the NotMonitored action starts.

Note: See also "DN Actions at Newly Registered DNs" on page 68.

Retrospective, Interaction-Related Actions

The following are the retrospective, interaction-related actions that Stat Server generates to regular DNs:

- `CallAbandonedFromDialing`
- `CallAbandonedFromHold`
- `CallAbandonedFromRinging`
- `CallAnswered`
- `CallConsultCompleted`
- `CallDialConferenced`
- `CallDialed`
- `CallDialTransferred`
- `CallForwarded`
- `CallInboundCompleted`
- `CallInternalCompleted`
- `CallOutboundCompleted`
- `CallPartyChanged`
- `CallRetrievedFromHold`
- `CallRingingPartyChanged`
- `CallUnknownCompleted`
- `StuckCallCleanedWhileRinging`
- `TransferredFromHold`

Note that all actions specifically called out as *retrospective* are instantaneous actions.

CallAbandonedFromDialing

This retrospective action derives from the `CallDialing` durable action (see page 81) if `CallDialing` terminates because of `EventReleased` and if the interaction is not a consult call with an interaction state of `Transferred` or `Conferenced`.

`CallAbandonedFromDialing` is always simultaneous with one of the following call-type actions:

- `CallAbandonedFromDialingUnknown`
- `CallAbandonedFromDialingInternal`
- `CallAbandonedFromDialingInbound`
- `CallAbandonedFromDialingOutbound`
- `CallAbandonedFromDialingConsult`

The interaction type that Stat Server receives from T-Server with `EventReleased` determines which of the above five actions occurs simultaneously with `CallAbandonedFromDialing`.

CallAbandonedFromHold

This retrospective action derives from the `CallOnHold` durable action (see page 84) if `CallOnHold` terminates because of `EventReleased` with an interaction state other than `Transferred`.

`CallAbandonedFromHold` is always simultaneous with one of the following call-type actions:

- `CallAbandonedFromHoldUnknown`
- `CallAbandonedFromHoldInternal`
- `CallAbandonedFromHoldInbound`
- `CallAbandonedFromHoldOutbound`
- `CallAbandonedFromHoldConsult`

The interaction type that Stat Server receives from T-Server with `EventReleased` determines which of the above five actions occurs simultaneously with `Call AbandonedFromHold`.

CallAbandonedFromRinging

This retrospective action derives from the `CallRinging` durable action (see page 84) if `CallRinging` terminates because of `EventReleased` or `Event Abandoned` (specifically, without interaction state `22-Redirected` or `23-Forwarded`). The `AttributeReliability` attribute, a new attribute provided with 7.1 T-Servers, must accompany `EventAbandoned` and this attribute's value must equal `TReliabilityOk`.

`CallAbandonedFromRinging` is always simultaneous with one of the following call-type actions:

- `CallAbandonedFromRingingUnknown`
- `CallAbandonedFromRingingInternal`
- `CallAbandonedFromRingingInbound`
- `CallAbandonedFromRingingOutbound`
- `CallAbandonedFromRingingConsult`

The interaction type that Stat Server receives from T-Server with `EventReleased` or `EventAbandoned` determines which of the above five actions occurs simultaneously with `CallAbandonedFromRinging`.

This action may occur simultaneously with the `CallAbandonedFromRinging` retrospective mediation DN action, which is described on page 105.

CallAnswered

This retrospective action derives from the `CallRinging` durable action (see page 84) if `CallRinging` terminates because of `EventEstablished`.

`CallAnswered` is always simultaneous with one of the following call-type actions:

- `CallAnsweredUnknown`
- `CallAnsweredInternal`
- `CallAnsweredInbound`
- `CallAnsweredOutbound`
- `CallAnsweredConsult`

This action may occur simultaneously with the CallAnswered retrospective, mediation DN action, which is described on page 105.

CallConsultCompleted

This retrospective action derives from the CallConsult durable action (see page 80). CallConsultCompleted is generated when a consultation call completes.

Use CallConsultCompleted instead of CallConsult for filtering attached data at the end of actions.

CallDialConferenced

This retrospective action derives from the CallDialing durable action (see page 81) if CallDialing terminates because of EventReleased for a consult call with an interaction state of Conferenced. CallDialConferenced is interaction-type specific, so it can also be considered to derive from CallDialingConsult.

Note: Previously, in multiple-site scenarios in which a given queue, routing point, or virtual queue was on one site and the agent's DN was on another, you had to set the use_orig_connID configuration option. In the release 7.0 and forward, this is no longer necessary.

CallDialed

This retrospective action derives from the CallDialing durable action (see page 81) if CallDialing terminates because of EventEstablished.

CallDialed is always simultaneous with one of the following call-type actions:

- CallDialedUnknown
- CallDialedInternal
- CallDialedInbound
- CallDialedOutbound
- CallDialedConsult

The interaction type that Stat Server receives from T-Server with EventDialing determines which of the above five actions occurs simultaneously with CallDialed.

CallDialTransferred

This retrospective action derives from the CallDialing durable action (see page 81) if CallDialing terminates because of EventReleased for a consult call with an interaction state of Transferred. CallDialTransferred is interaction-type specific, so it can also be considered to derive from CallDialingConsult.

CallForwarded

This retrospective action derives from the CallRinging durable action (see page 84) if CallRinging terminates because of EventReleased with an interaction state of Forwarded or Redirected (when the forwarding functionality is enabled on a DN).

CallForwarded is always simultaneous with one of the following call-type actions:

- CallForwardedUnknown
- CallForwardedInternal
- CallForwardedInbound
- CallForwardedOutbound
- CallForwardedConsult

This action may occur simultaneously with the CallForwarded retrospective, mediation DN action, which is described on page 106.

CallInboundCompleted

This retrospective action derives from the CallInbound durable action (see page 82). CallInboundCompleted is generated when an inbound interaction completes.

Use CallInboundCompleted instead of CallInbound for filtering attached data at the end of actions.

CallInternalCompleted

This retrospective action derives from the CallInternal durable action (see page 82). CallInternalCompleted is generated when an internal interaction completes.

Use CallInternalCompleted instead of CallInternal for filtering attached data at the end of actions.

CallOutboundCompleted

This retrospective action derives from the CallOutbound durable action (see page 84). CallOutboundCompleted is generated when an outbound interaction completes.

Use CallOutboundCompleted instead of CallOutbound for filtering attached data at the end of actions.

CallPartyChanged

This retrospective action derives from the following:

- The CallConsult, the CallConsult Originated, or the CallConsultReceived durable actions (see page 80) if any of these actions terminate because of EventPartyChanged.
- The CallInbound action (see page 82), in circumstances where T-Server configuration causes T-Server to transmit an Inbound interaction type with the TEvent rather than Consult, such as may be the case when the use-data-from T-Server configuration option is set to "consult-user-data"

CallRetrievedFromHold

This retrospective action derives from the CallOnHold durable action (see page 84) if CallOnHold terminates because of EventRetrieved.

CallRetrievedFromHold is always simultaneous with one of the following call-type actions:

- RetrievedFromHoldUnknown
- RetrievedFromHoldInternal
- RetrievedFromHoldInbound
- RetrievedFromHoldOutbound
- RetrievedFromHoldConsult

The interaction type that Stat Server receives from T-Server with EventEstablished determines which of the above five actions occurs simultaneously with CallRetrievedFromHold.

CallRingingPartyChanged

This retrospective action derives from the following:

- The CallRinging durable action (see page 84), if CallRinging terminates because of EventPartyChanged for a consult call.
- The CallRingingConsult action, as CallRingingPartyChanged is interaction-type-specific.
- The CallInbound action (see page 82), in circumstances where T-Server configuration causes T-Server to transmit an Inbound interaction type with the TEvent instead of Consult, such as may be the case when the use-data-from T-Server configuration option is set to "consult-user-data".

CallUnknownCompleted

This retrospective action derives from the CallUnknown durable action (see page 85). CallUnknownCompleted is generated when an unknown interaction completes.

Use CallOutboundCompleted instead of CallUnknown for filtering attached data at the end of actions.

StuckCallCleanedWhileRinging

This retrospective action derives from the CallRinging durable action (see page 84) if Stat Server receives EventAbandoned with an AttributeReliability attribute not equal to TReliabilityOk for the DN. This action's corresponding initial momentary action is CallRingingStarted (see page 94).

StuckCallCleanedWhileRinging is always simultaneous with one of the following call-type actions:

- StuckCallCleanedWhileRingingUnknown
- StuckCallCleanedWhileRingingInternal
- StuckCallCleanedWhileRingingInbound
- StuckCallCleanedWhileRingingOutbound
- StuckCallCleanedWhileRingingConsult

The interaction type that Stat Server receives from T-Server with Event Abandoned (with AttributeReliability!=TReliabilityOk) determines which of the above five actions occurs simultaneously with StuckCallCleanedWhile Ringing.

TransferredFromHold

This retrospective action derives from the CallOnHold durable action (see page 84) if CallOnHold terminates because of EventReleased with an interaction state of Transferred.

TransferredFromHold is always simultaneous with one of the following call-type actions:

- TransferredFromHoldUnknown
- TransferredFromHoldInternal
- TransferredFromHoldInbound
- TransferredFromHoldOutbound
- TransferredFromHoldConsult

The interaction type that Stat Server receives from T-Server with EventReleased determines which of the above five actions occurs simultaneously with TransferredFromHold.

Momentary, Interaction-Related Actions

The following are the momentary, interaction-related actions that Stat Server generates to regular DNs:

- CallConferenceJoined
- CallConferenceMade
- CallConferencePartyAdded
- CallConferencePartyDeleted
- CallConsultStarted
- CallDialingStarted
- CallHeld
- CallInboundStarted
- CallInternalStarted
- CallOutboundStarted

- `CallRingingStarted`
- `CallTransferMade`
- `CallTransferPartyChanged`
- `CallTransferTaken`
- `CallUnknownStarted`

CallConferenceJoined

This momentary action occurs in a conference call at a DN that was added to the conference. `CallConferenceJoined` derives from `EventPartyChanged` with an interaction state of `Conferenced`.

CallConferenceMade

Once the transfer completes, this momentary action occurs at the DN that initiated the conference. `CallConferenceMade` derives from `EventPartyAdded` with an interaction state of `Conferenced`, a `ThirdPartyDNRole` of `AddedBy`, and a `ThirdPartyDN` equal to `ThisDN`.

CallConferencePartyAdded

This momentary action occurs at all DNs participating in a conference call when a new DN joins the conference. `CallConference PartyAdded` derives from `EventPartyAdded` with a `ThirdPartyDNRole` of `AddedBy` and a `ThirdPartyDN` different from `ThisDN`.

CallConferencePartyDeleted

This momentary action occurs in a conference call at all DNs left in the conference when a DN ends its participation in the conference. It derives from `EventPartyDeleted`.

CallConsultStarted

This momentary action occurs whenever the `CallConsult`, `CallConsult Originated`, or `CallConsultReceived` durable action (see page 80) starts.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallDialingStarted

This momentary action occurs whenever the `CallDialing` durable action (see page 81) starts.

CallDialingStarted is always simultaneous with one of the following call-type actions:

- CallDialingStartedUnknown
- CallDialingStartedInternal
- CallDialingStartedInbound
- CallDialingStartedOutbound
- CallDialingStartedConsult

The interaction type that Stat Server receives from T-Server with EventDialing determines which of the above five actions occurs simultaneously with CallDialingStarted.

CallHeld

This momentary action occurs whenever the CallOnHold durable action (see page 84) starts.

CallHeld is always simultaneous with one of the following call-type actions:

- CallHeldUnknown
- CallHeldInternal
- CallHeldInbound
- CallHeldOutbound
- CallHeldConsult

The interaction type that Stat Server receives from T-Server with EventHeld determines which of the above five actions occurs simultaneously with CallHeld.

CallInboundStarted

This momentary action occurs whenever the CallInbound durable action (see page 82) starts.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallInternalStarted

This momentary action occurs whenever the CallInternal durable action (see page 82) starts.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallOutboundStarted

This momentary action occurs whenever the CallOutbound durable action (see page 84) starts.

Note: See also “DN Actions at Newly Registered DNs” on page 68.

CallRingingStarted

This momentary action occurs whenever the CallRinging durable action (see page 84) starts.

CallRingingStarted is always simultaneous with one of the following call-type actions:

- CallRingingStartedUnknown
- CallRingingStartedInternal
- CallRingingStartedInbound
- CallRingingStartedOutbound
- CallRingingStartedConsult

The interaction type that Stat Server receives from T-Server with EventRinging determines which of the above five actions occurs simultaneously with CallRingingStarted.

CallTransferMade

This momentary action occurs at the DN from which a transfer was initiated (by TInitiateTransfer, TSingleStepTransfer, or TMuteTransfer, or by TMergeCalls) once the transfer is completed (EventReleased is received with an interaction state of Transferred).

CallTransferMade is always simultaneous with one of the following call-type actions:

- CallTransferMadeUnknown
- CallTransferMadeInternal
- CallTransferMadeInbound
- CallTransferMadeOutbound
- CallTransferMadeConsult

CallTransferPartyChanged

Once the transfer completes, this momentary action occurs at the DN of the first party for a call transferred from a second party to a third. CallTransfer PartyChanged derives from EventPartyChanged with an interaction state of Transferred and a ConnID equal to PreviousConnID.

CallTransferTaken

This momentary action occurs:

- At the DN to which a transfer was made, once the transfer completes. CallTransferTaken derives from one of the following:
  - EventPartyChanged with an interaction state of Transferred and a ConnID different from PreviousConnID
  - EventRinging with an interaction state of Transferred.
- At the RegDN, if PartyChanged has been received for this interaction on some mediation DN prior to distribution to a regular DN.

Note: Transfers performed by an IVR (DN type VTO) are no longer counted as TransferTaken. Stat Server now counts transfers that are initiated from an agent's DN and completed on a queue or routepoint as Transfer Taken for the agent receiving this call. Previously, transfers initiated by an IVR were also counted as TransferTaken.

CallUnknownStarted

This momentary action occurs whenever the CallUnknown durable action (see page 85) starts.

Note: See also "DN Actions at Newly Registered DNs" on page 68.

Instantaneous, Non–Interaction-Related Actions

Stat Server generates the following the instantaneous, non–interaction-related actions to regular DN objects:

- AgentLogin
- AgentLogout
- UserEvent

The TEvents that trigger these actions carry attached data that you can use and reference when you define filtered and custom-formula statistics based on these actions. Stat Server inherits UserData and Reasons values from the triggering TEvent.

AgentLogin

This momentary action occurs when Stat Server detects agent login to a DN through either of the following:

- Stat Server receives EventAgentLogin on the DN.
- Stat Server receives EventRegistered or EventAddressInfo for the DN indicating agent login.

Stat Server generates this media-independent action when Stat Server detects login to the device—not to a particular media channel on the device.

AgentLogout

EventAgentLogout triggers this retrospective, instantaneous action. Furthermore, this action inherits its attributes (such as Reasons) from this TEvent, which can be useful, for example, for tallying the number of agent logout actions that occurred during a particular time frame because of a particular Reason (using Reason-based filtering [page 32] introduced in the 7.6 release).

The duration of this action coincides with the duration of the agent's login on the DN. Stat Server generates this media-independent action when Stat Server detects:

- EventAgentLogout on a device—not when the agent logs off of a particular media channel.
- EventLinkDisconnected on a regular logged-in DN.

UserEvent

The EventUserEvent TEvent triggers this momentary, instantaneous action.

Durable Group Actions Reflecting Origination DNs

You can list origination DNs on the Advanced tab of the Properties dialog box of an Agent Group or Place Group object in Configuration Manager. If you list queues and routing points from which calls are delivered to a given Group object as origination DNs for that group, you can use events occurring at such DNs in agent group and place group statistics. For this purpose, Stat Server reflects some mediation DN actions as a special set of agent and place group actions.

OrigDNCallWait is a durable group action reflecting origination DNs that Stat Server generates to agent and place groups.

OrigDNCallWait

This agent group and place group action starts and ends at the same time as a CallWait action (see page 100), which starts and ends at a mediation DN. configured as an origination DN for the group. OrigDNCallWait relates to the same interaction as the corresponding CallWait action.

OrigDNCallWait is always simultaneous with one of the following call-type actions:

- OrigDNCallWaitUnknown
- OrigDNCallWaitInternal
- OrigDNCallWaitInbound
- OrigDNCallWaitOutbound
- OrigDNCallWaitConsult

The interaction type that Stat Server receives from T-Server with EventQueued or EventRouteRequest determines which of the above five actions occurs simultaneously with OrigDNCallWait.

Retrospective Group Actions Reflecting Origination DNs

The following are retrospective group actions reflecting origination DNs that Stat Server generates to agent and place groups:

- OrigDNCallAbandoned
- OrigDNCallDistributed

OrigDNCallAbandoned

This agent group and place group action occurs at the same time as a Call Abandoned action (see page 102), which occurs at a mediation DN configured as an origination DN for the group. OrigDNCallAbandoned relates to the same interaction as the corresponding CallAbandoned action.

OrigDNCallAbandoned is always simultaneous with one of the following call-type actions:

- OrigDNCallAbandonedUnknown
- OrigDNCallAbandonedInternal
- OrigDNCallAbandonedInbound
- OrigDNCallAbandonedOutbound
- OrigDNCallAbandonedConsult

The interaction type that Stat Server receives from T-Server with EventAbandoned determines which of the above five actions occurs simultaneously with OrigDN CallAbandoned.

OrigDNCallDistributed

This agent group and place group action occurs at the same time as a CallDistributed action (see page 103), which occurs at a mediation DN configured as an origination DN for the group. OrigDNCallDistributed relates to the same interaction as the corresponding CallDistributed action.

OrigDNCallDistributed is always simultaneous with one of the following call-type actions:

- OrigDNCallDistributedUnknown
- OrigDNCallDistributedInternal
- OrigDNCallDistributedInbound
- OrigDNCallDistributedOutbound
- OrigDNCallDistributedConsult

The interaction type that Stat Server receives from T-Server with Event Diverted or EventRouteUsed determines which of the above five actions occurs simultaneously with OrigDNCallDistributed.

Momentary Group Actions Reflecting Origination DNs

OrigDNCallEntered is a momentary group action reflecting origination DNs that Stat Server generates to agent and place groups.

OrigDNCallEntered

This agent group and place group action occurs at the same time as a CallEntered action (see page 101), which occurs at a mediation DN configured as an origination DN for the group. OrigDNCallEntered relates to the same interaction as the corresponding CallEntered action.

OrigDNCallEntered is always simultaneous with one of the following call-type actions:

- OrigDNCallEnteredUnknown
- OrigDNCallEnteredInternal
- OrigDNCallEnteredInbound
- OrigDNCallEnteredOutbound
- OrigDNCallEnteredConsult

The interaction type that Stat Server receives from T-Server with EventQueued or EventRoute Request determines which of the above five actions occurs simultaneously with OrigDNCallEntered.

Mediation DN Actions

Mediation DN actions fall into the following categories:

- Durable, non–interaction-related actions (page 98)
- Durable, interaction-related actions (page 100)
- Momentary, interaction-related actions (page 101)
- Momentary, non–interaction-related actions (page 101)
- Retrospective, interaction-related actions (page 102)
- Retrospective, interaction-related actions reflecting regular DNs (page 104)
- Retrospective, non–interaction-related actions reflecting regular DNs (page 104)
- Retrospective, interaction-related action distributed from another mediation DNs (page 111)

The subsections below describe the one or more actions that comprise each category. Note that all actions specifically called out as *retrospective* are instantaneous actions.

Durable, Non–Interaction-Related Actions

The following are the durable, non–interaction-related actions that Stat Server generates to mediation DNs:

- AgentActive
- AgentLogin
- AgentReady
- DNActive
- DNLogin
- DNReady
- NotMonitored
- Monitored

AgentLogin

This durable action starts when agent logs on to a mediation DN through a regular DN that belongs to a place. This action ends when the agent logs out from the mediation DN or when the NotMonitored action starts.

AgentActive

This durable action starts on a mediation DN when the status of an agent, who is already logged into that mediation DN through a regular DN belonging to a place, changes from NotReadyForNextCall. This action ends when agent status changes to NotReadyForNextCall on that mediation DN, when that agent logs out from the mediation DN, or when the NotMonitored action starts.

AgentReady

This durable action starts on a mediation DN when the status of agent, who is already logged into that mediation DN through a regular DN belonging to a place, changes to WaitForNextCall. This action ends when agent status changes from WaitForNextCall on that mediation DN, when that agent logs out from the mediation DN, or when the NotMonitored action starts. (See page 130 for a definition of agent status).

DNLogin

This durable action starts on a mediation DN when a regular DN logs into the mediation DN. This action ends when that regular DN logs out from mediation DN or when the NotMonitored action starts.

DNActive

This durable action starts on a mediation DN when the status of regular DN, that is already logged in to that mediation DN changes from NotReadyForNextCall. This action ends when the regular DN's status changes to NotReadyForNextCall, when that regular DN logs out from the mediation DN, or when the NotMonitored action starts.

DNReady

This durable action starts on a mediation DN when the status of regular DN, already logged into that mediation DN, becomes WaitForNextCall.

This action ends on mediation DN when the status of regular DN stops being WaitForNextCall, when that regular DN logs out from the corresponding mediation DN, or when the NotMonitored action starts. (See page 126 for a definition of DN status).

NotMonitored

This durable action begins whenever Stat Server is not connected to the T-Server controlling the switch where the DN is located (Stat Server receives the EventServerDisconnected TEvent in this case) or whenever the link between the T-Server and the switch is down (EventLinkDisconnected is received from T-Server). NotMonitored ends when both connections are up. Its complementary action is Monitored. One and only one of these actions occurs for any DN at any moment. NotMonitored terminates every other DN action; no other action can start while NotMonitored is occurring.

Monitored

This durable action starts whenever NotMonitored terminates—that is, when Stat Server is connected to T-Server and the link between the T-Server and the switch is up. This action ends when the NotMonitored action starts.

Durable, Interaction-Related Actions

CallWait is a durable, interaction-related action that Stat Server generates to mediation DNs.

CallWait

Stat Server generates this durable action depending on the object's type:

- Upon receipt of EventQueued (for ACD and virtual queue objects).
- Upon receipt of EventRouteRequest (for routing points objects).
- Upon receipt of _we_event_placed_in_workbin (for workbin objects).

Its corresponding initial momentary action is CallEntered (see page 101).

This action ends:

- Upon receipt of the following TEvents (for routing points and ACD and virtual queue objects)
  - EventRouteUsed
  - EventDiverted
  - EventAbandoned
  - EventPartyChanged
  - EventReleased
- Upon receipt of EventAddressInfo (for queue and routing point objects)
- When the NotMonitored action starts (such as when T-Server disconnects).

For T-Server-originating events, CallWait is always simultaneous with one of the following call-type actions:

- CallWaitUnknown
- CallWaitInternal
- CallWaitInbound
- CallWaitOutbound
- CallWaitConsult

The interaction type that Stat Server receives from T-Server with EventQueued or EventRouteRequest determines which of the above five actions occurs simultaneously with CallWait.

Momentary, Interaction-Related Actions

The following are the momentary, interaction-related actions that Stat Server generates to mediation DNs:

- CallEntered
- CallTreatmentStarted
- CallTreatmentNotStarted

CallEntered

This momentary action occurs, depending on the type of the DN, when Stat Server receives:

- EventQueued or EventRouteRequest from T-Server.

For T-Server-originating events, CallEntered is always simultaneous with one of the following call-type actions:

- CallEnteredUnknown
- CallEnteredInternal
- CallEnteredInbound
- CallEnteredOutbound
- CallEnteredConsult

The interaction type that Stat Server receives from T-Server with EventQueued or EventRouteRequest determines which of the above five actions occurs simultaneously with CallEntered.

CallTreatmentStarted

This momentary action occurs when Stat Server receives EventTreatment Applied from T-Server.

CallTreatmentNotStarted

This momentary action occurs when Stat Server receives EventTreatment NotApplied from T-Server.

Momentary, Non-Interaction-Related Action

UserEvent is a momentary, non–interaction-related action that Stat Server generates to mediation DNs.

UserEvent

The EventUserEvent TEvent triggers the UserEvent action, which is not related to an interaction, but which, like interaction-related actions, carries data that accompanies the TEvent. This means you can use this action in defining filtered statistics and custom-formula statistics.

Retrospective, Interaction-Related Actions

The following are the retrospective, interaction-related actions that Stat Server generates to mediation DNs:

- CallAbandoned
- CallCleared
- CallDistributed
- CallTreatmentCompleted
- StuckCallCleaned

CallAbandoned

This retrospective action derives from the CallWait durable action (see page 100) if CallWait terminates because of EventAbandoned with an AttributeReliability attribute equal to TReliabilityOk.

CallAbandoned is always simultaneous with one of the following call-type actions:

- CallAbandonedUnknown
- CallAbandonedInternal
- CallAbandonedInbound
- CallAbandonedOutbound
- CallAbandonedConsult

The interaction type that Stat Server receives from T-Server with EventQueued or EventRouteRequest determines which of the above five actions occurs simultaneously with CallAbandoned.

CallCleared

Stat Server generates this retrospective action only for a virtual queue. The action derives from the CallWait durable action (see page 100) if CallWait terminates because of EventDiverted with an interaction state of Redirected. With this event, the Universal Routing Server, by means of T-Server, indicates that an interaction has left this queue and is being delivered to an agent from another virtual queue.

`CallCleared` is always simultaneous with one of the following call-type actions:

- `CallClearedUnknown`
- `CallClearedInternal`
- `CallClearedInbound`
- `CallClearedOutbound`
- `CallClearedConsult`

CallDistributed

This retrospective action derives from the `CallWait` durable action (see page 100) if `CallWait` terminates because:

- Stat Server receives `EventRouteUsed` or `EventDiverted` from T-Server.

In addition, for virtual queue objects, `EventDiverted` must contain an interaction state other than `Redirected`.

For T-Server-originating events, `CallDistributed` is always simultaneous with one of the following call-type actions:

- `CallDistributedUnknown`
- `CallDistributedInternal`
- `CallDistributedInbound`
- `CallDistributedOutbound`
- `CallDistributedConsult`

The interaction type that Stat Server receives from T-Server with `EventQueued` or `EventRoute Request` determines which of the above five actions occurs simultaneously with `CallDistributed`.

CallTreatmentCompleted

This retrospective action is not derived from a durable action. `CallTreatmentCompleted` occurs when Stat Server receives `EventTreatmentCompleted` from T-Server, and the duration of this action is the total duration of the treatment.

StuckCallCleaned

This retrospective action occurs at a mediation DN and derives from the `CallWait` durable action (see page 100) if Stat Server terminates the `CallWait` action because Stat Server receives the `EventAbandoned` TEvent from T-Server with an `AttributeReliability` attribute not equal to `TReliabilityOk`.

`StuckCallCleaned` is always simultaneous with one of the following call-type actions:

- `StuckCallCleanedUnknown`
- `StuckCallCleanedInternal`
- `StuckCallCleanedInbound`
- `StuckCallCleanedOutbound`
- `StuckCallCleanedConsult`

The interaction type that Stat Server receives from T-Server with EventQueued or EventRouteRequest determines which of the above five actions occurs simultaneously with StuckCallCleaned.

Retrospective, Interaction-Related Actions Reflecting Regular DNs

The following are the retrospective, interaction-related actions reflecting regular DNs that Stat Server generates to mediation DNs:

- ACWCompleted
- ACWMissed
- CallAbandonedFromRinging
- CallAnswered
- CallForwarded
- CallMissed
- CallReleased
- StuckCallCleanedWhileRinging

With the exception of the CallAnswered, CallAbandonedFromRinging, CallForwarded, StuckCallCleanedWhileRinging actions, these retrospective, interaction-related actions reflecting regular DNs work, without additional confirmation, only for ACD queues and T-Server applications that propagate the queue parameter in login messages. For T-Server applications that do not do this, you must explicitly configure the association between Agent objects and a Queue object in Configuration Manager, as follows:

1. Select an agent (or place) group and open its Properties dialog box.
2. On the Advanced tab, click Add to add an Origination DN object.
3. In the Browse dialog box, double-click the switch to which the queue belongs.
4. Double-click the DN object that belongs to the switch, and then select the queue that you want to associate with the agent (or place) group.
5. Click OK.
6. In the AgentGroup (or PlaceGroup) Properties dialog box, click OK to save the configured association.

With the exception of the ACWMissed and CallMissed actions in this category, Stat Server propagates retrospective, interaction-related actions that reflect regular DNs, such as CallAnswered and CallAbandonedFromRinging, from an agent’s DN to the last physical mediation DN through which a call passes before being answered or abandoned while ringing in a single-site contact center. Stat Server propagates ACWMissed and CallMissed actions to all mediation DNs involved in the processing of the interaction except the last mediation DN from which the interaction was answered by a handling resource.

Note: If a call is routed to an ACD queue DN from a routing point, Stat Server no longer generates interaction-related actions reflecting regular DNs, such as CallAnswered, on this routing point as Stat Server did in release 7.2 and prior releases. Instead, such actions are generated on the ACD queue. Starting with release 7.5, Stat Server propagates interaction-related actions reflecting regular DNs on mediation DNs only to the last real and virtual mediation DN objects that the interaction passed through.

If, however, the call is queued in parallel to both an ACD queue DN and a routing point, and the ThirdPartyDN attribute of EventRouteUsed shows that the call was answered on some regular DN, then Stat Server will propagate this action to both.

ACWCompleted

This retrospective action occurs at a mediation DN when the regular DN action AfterCallWork is over. Action duration is the same duration as the corresponding AfterCallWork action. If a switch permits agents to enter AfterCallWork mode while they are still involved in calls, Stat Server generates the ACWCompleted action upon completion of the interaction. This behavior was introduced in the 7.0 release. Stat Server postpones the ACWCompleted action until after termination of the interaction.

ACWMissed

This retrospective action occurs at a mediation DN when the regular DN action AfterCallWork (see page 73) is over. ACWMissed applies to calls that have been distributed from a source other than the queue, when an agent logs in and enters the queue. Action duration is the same duration as the corresponding action AfterCallWork.

CallAbandonedFromRinging

This retrospective action occurs at a mediation DN when EventReleased (with an interaction state other than CallForwarded or CallRedirected) is received after EventRinging from a DN to which an interaction was going to be distributed from the mediation DN. It receives as its duration the interval from the moment when the interaction entered the mediation DN (EventQueued or EventRouteRequest) to the moment when the interaction was abandoned (EventReleased).

CallAnswered

This retrospective action occurs at a mediation DN when EventEstablished is received after EventRinging from a DN to which an interaction was distributed from the mediation DN. CallAnswered receives as its duration the interval from the moment when the interaction enters the mediation DN (the latest of the EventQueued, EventRouteRequest or EventPartyChanged TEvents if it occurs while the call is waiting in queue or at the routing point) to the moment when the agent takes the interaction (EventEstablished or EventDiverted, whichever is latest).

Note: If an interaction was accepted at an agent DN at moment T1 and the interaction is subsequently requeued to a mediation DN (at moment T2), Stat Server will not generate the CallAnswered action on all mediation DNs for which the EventDiverted or EventRouteUsed TEvents were delayed (that is, when these events follow T2).

CallAnswered is always simultaneous with one of the following call-type actions

- CallAnsweredUnknown
- CallAnsweredInternal
- CallAnsweredInbound
- CallAnsweredOutbound
- CallAnsweredConsult

This action may occur simultaneously with the CallAnswered retrospective action, which is described on page 87.

CallForwarded

This retrospective action occurs at a mediation DN when Stat Server receives EventReleased (with an interaction state of CallForwarded or CallRedirected) following EventRinging from a DN to which an interaction was going to be distributed from the mediation DN. Action duration is the interval from the moment when the interaction enters the mediation DN (EventQueued or EventRouteRequest) to the moment when the interaction is abandoned (EventReleased).

CallMissed

This retrospective action occurs at a mediation DN when EventReleased comes after EventEstablished. It applies to calls that have been distributed from a source other than the queue, when an agent enters the queue using his or her login. Action duration is the interval beginning with EventEstabished and ending with EventReleased.

CallReleased

This retrospective action occurs at a mediation DN when EventReleased comes after EventEstablished from a regular DN, for an interaction distributed from the mediation DN. Action duration is the interval from EventEstabished to EventReleased.

StuckCallCleanedWhileRinging

This retrospective action derives from the CallRinging durable action (see page 84) and occurs at a mediation DN when Stat Server receives EventAbandoned with an AttributeReliability attribute other than TReliabilityOk from a DN to which an interaction was distributed from the mediation DN. StuckCallCleanedWhileRinging receives as its duration the interval from the moment when the interaction enters the mediation DN (EventQueued or Event RouteRequest) to the moment when Stat Server receives the EventAbandoned TEvent (with AttributeReliability!=TReliabilityOk). This action's corresponding initial momentary action is CallRingingStarted (see page 94).

StuckCallCleanedWhileRinging is always simultaneous with one of the following call-type actions:

- StuckCallCleanedWhileRingingUnknown
- StuckCallCleanedWhileRingingInternal
- StuckCallCleanedWhileRingingInbound
- StuckCallCleanedWhileRingingOutbound
- StuckCallCleanedWhileRingingConsult

The interaction type that Stat Server receives from T-Server with Event Released (with AttributeReliability!=TReliabilityOk) determines which of the above five actions occurs simultaneously with CallRetrievedFromHold.

Generation of Retrospective, Interaction-Related Actions Reflecting Regular DNs for Virtual Queue Mediation DN objects

For virtual queue mediation DN objects, Stat Server generates retrospective, interaction-related actions reflecting regular DNs depending on the combination of settings of the following Stat Server configuration options, which are defined in the *Framework Stat Sever Deployment Guide*:

- vq-ignore-third-party-dn
- vq-treat-unknown-third-party-dn-as-agent-dn

Using these options, you can change the algorithm for Stat Server's generation of CallAnswered actions on virtual queue objects to meet your requirements. If you set vq-ignore-third-party-dn to true (the default value), Stat Server generates CallAnswered action for all virtual queue objects through which a call passes before it is answered. If you set the option vq-ignore-third-party-dn to false, Stat Server references the ThirdPartyDN attribute in EventDiverted TEvents that Stat Server receives from Universal Routing Server for CallAnswered action generation. At this case, the rules of CallAnswered action generation depends on settings of the vq-treat-unknown-third-party-dn-as-agent-dn option. Introduction of this option allows Stat Server to generate this action only on the last virtual queue object through which a call passes before being answered in single-site call monitoring scenarios (inbound call enters monitoring site, Inbound call is queued on routing point associated with Virtual Queue on the same site and is routed to the target on the same site). Multi-site Call monitoring scenarios have some limitations in this rule because there are cases where `ThirdPartyDN` does not contain reliable information about DN to which the call was diverted.

Table 21 describes how Stat Server behaves given the setting of the `vq-treat-unknown-third-party-dn-as-agent-dn` option and the following scenarios:

Scenario A: The value of the `ThirdPartyDN` attribute contains the ID of a DN belonging to the same switch as the virtual queue.

1. `ThirdPartyDN` points to a mediation DN that is not an ACD queue.
2. `ThirdPartyDN` points to an ACD queue.
3. `ThirdPartyDN` points to an agent's DN.

Scenario B: The value of the `ThirdPartyDN` attribute is not empty, but contains the ID of a DN that is not monitored by the same switch to which the virtual queue belongs. The `vq-treat-unknown-third-party-dn-as-agent-dn` configuration option is set to:

1. `True` (the default value).
2. `False`, and the ID of DN answering the call coincides with the value of the `ThirdPartyDN` attribute.
3. `False`, and the ID of DN answering the call differs from the value of the `ThirdPartyDN` attribute.

Scenario C: The value of the `ThirdPartyDN` attribute is null.

It is assumed that after having been diverted from the virtual queue, the call was finally answered by an agent; and that, in multi-site scenarios, Stat Server may receive events out of chronological order, such that a call may first be seen as being answered before Stat Server sees that it was diverted from a virtual queue. Table 21 shows whether Stat Server will generate a `CallAnswered` action given the above three scenarios:

Table 21:

Scenario	Does Stat Server Generate CallAnswered?
A1	No
A2	Yes
A3	Yes
B1	Yes
B2	Yes
B3	No
C	Yes

Configuring the routing strategies and associated virtual queue objects to control and monitor call for multi-site routing you have to take into account the specifics in `CallAnswered` generation in case you are using settings `vq-treat-unknown-third-party-dn-as-agent-dn` in following scenarios:

1. `vq-treat-unknown-third-party-dn-as-agent-dn=no`
   - Call queued on Routing Point 1 Site 1 and Virtual Queue 1 Site 1.
   - Call diverted from Routing Point 1 Site 1 and Virtual Queue 1 Site 1 to Routing Point 2 Site 1 and Virtual Queue 2 Site 1.
   - Call diverted from Routing Point 2 Site 1 and Virtual Queue 2 Site 1 to Routing Point 1 Site 2 and Virtual Queue 1 Site 2.
   - Call diverted from Routing Point 1 Site 2 and Virtual Queue 1 Site 2 to Routing Point 2 Site 2 and Virtual Queue 2 Site 2.
   - Call diverted from Routing Point 2 Site 2 and Virtual Queue 2 Site 2 to Agent 1 Site 2.

   `CallAnswered` and related actions will be generated for Virtual Queue 2 Site 2 only.

   Note: This scenario corresponds to Scenario B2.

2. `vq-treat-unknown-third-party-dn-as-agent-dn=no`
   - Call queued on Routing Point 1 Site 1 and Virtual Queue 1 Site 1.
   - Call diverted from Routing Point 1 Site 1 and Virtual Queue 1 Site 1 to Routing Point 2 Site 1 and Virtual Queue 2 Site 1.
   - Call diverted from Routing Point 2 Site 1 and Virtual Queue 2 Site 1 to Agent 1 Site 2.

   `CallAnswered` and related actions will not be generated for any virtual queue.

   Note: This scenario corresponds to Scenario B3.

3. `vq-treat-unknown-third-party-dn-as-agent-dn=no`
   - Call queued on Routing Point 1 Site 1 and Virtual Queue 1 Site 2
   - Call diverted from Routing Point 1 Site 1 and Virtual Queue 1 Site 2 to Agent 1 Site 2.

   `CallAnswered` and related actions will not be generated for any virtual queue.

   Note: This scenario corresponds to Scenario A1.

4. `vq-treat-unknown-third-party-dn-as-agent-dn=no`
   - Call queued on Routing Point 1 Site 1 and Virtual Queue 1 Site 2.
   - Call diverted from Routing Point 1 Site 1 and Virtual Queue 1 Site 2 to Routing Point 2 Site 1 and Virtual Queue 2 Site 2.

- Call diverted from Routing Point 2 Site 1 and Virtual Queue 2 Site 2 to Routing Point 1 Site 2 and Virtual Queue 1 Site 1.
- Call diverted from Routing Point 1 Site 2 and Virtual Queue 1 Site 1 to Routing Point 2 Site 2 and Virtual Queue 2 Site 1.
- Call diverted from Routing Point 2 Site 2 and Virtual Queue 2 Site 1 to Agent 1 Site 2.

`CallAnswered` and related actions will be generated for Virtual Queue 2 Site 1 only.

Note: This scenario corresponds to Scenario B2.

5. `vq-treat-unknown-third-party-dn-as-agent-dn=yes`
   - Call queued on Routing Point 1 Site 1 and Virtual Queue 1 Site 1.
   - Call diverted from Routing Point 1 Site 1 and Virtual Queue 1 Site 1 to Routing Point 2 Site 1 and Virtual Queue 2 Site 1.
   - Call diverted from Routing Point 2 Site 1 and Virtual Queue 2 Site 1 to Routing Point 1 Site 2 and Virtual Queue 1 Site 2.
   - Call diverted from Routing Point 1 Site 2 and Virtual Queue 1 Site 2 to Routing Point 2 Site 2 and Virtual Queue 2 Site 2.
   - Call diverted from Routing Point 2 Site 2 and Virtual Queue 2 Site 2 to Agent 1 Site 2.

   `CallAnswered` and related actions will be generated for Virtual Queue 2 Site 1 and Virtual Queue 2 Site 2.

   Note: This scenario corresponds to Scenario A3 for Virtual Queue 2 Site 2 and to Scenario B1 for Virtual Queue 2 Site 1.

6. `vq-treat-unknown-third-party-dn-as-agent-dn=yes`
   - Call queued on Routing Point 1 Site 1 and Virtual Queue 1 Site 1.
   - Call diverted from Routing Point 1 Site 1 and Virtual Queue 1 Site 1 to Routing Point 2 Site 1 and Virtual Queue 2 Site 1.
   - Call diverted from Routing Point 2 Site 1 and Virtual Queue 2 Site 1 to Agent 1 Site 2.

   `CallAnswered` and related actions will not be generated for Virtual Queue 2 Site 1.

   Note: This scenario corresponds to Scenario B1.

7. `vq-treat-unknown-third-party-dn-as-agent-dn=yes`
   - Call queued on Routing Point 1 Site 1 and Virtual Queue 1 Site 2.
   - Call diverted from Routing Point 1 Site 1 and Virtual Queue 1 Site 2 to Agent 1 Site 2.

`CallAnswered` and related actions will be generated for Virtual Queue 1 Site 2.

Note: In this scenario, `ThirdPartyDN` points to External Routing Point Site 2 which contains switch access codes is not recognizable for Stat Server.

8. `vq-treat-unknown-third-party-dn-as-agent-dn=yes`

- Call queued on Routing Point 1 Site 1 and Virtual Queue 1 Site 2.
- Call diverted from Routing Point 1 Site 1 and Virtual Queue 1 Site 2 to Routing Point 2 Site 1 and Virtual Queue 2 Site 2.
- Call diverted from Routing Point 2 Site 1 and Virtual Queue 2 Site 2 to Routing Point 1 Site 2 and Virtual Queue 1 Site 1.
- Call diverted from Routing Point 1 Site 2 and Virtual Queue 1 Site 1 to Routing Point 2 Site 2 and Virtual Queue 2 Site 1.
- Call diverted from Routing Point 2 Site 2 and Virtual Queue 2 Site 1 to Agent 1 Site 2.

`CallAnswered` and related actions will be generated for all four virtual queues participated in scenario.

Note: This scenario corresponds to Scenario B1.

Retrospective, Interaction-Related Action Distributed from Another Mediation DN

`CallDistributedToQueue` is a retrospective, interaction-related action distributed from another mediation DN that Stat Server generates to mediation DNs.

CallDistributedToQueue

Stat Server generates this retrospective action on a mediation DN (`DN1`) if an interaction is distributed from this DN after entering a second DN (`DN2`). The duration of this action is equal to the time from receipt of an `EventQueued` or `Event RouteRequest` TEvent on `DN1` until the receipt of an `EventQueued` or `EventRouteRequest` on `DN2`. Stat Server does not generate this action if an interaction enters `DN2` but has not yet been distributed from `DN1`. Stat Server also does not generate this action if an interaction is distributed from `DN1` to a nonmediation DN, such as to an agent's DN. After Stat Server generates `CallDistributedTo Queue` for `DN1`, `DN1` is cleared from the list of DNs from which the interaction can be distributed.

`CallDistributedToQueue` is always simultaneous with one of the following call-type actions:

- `CallDistributedToQueueInternal`

- `CallDistributedToQueueInbound`
- `CallDistributedToQueueOutbound`
- `CallDistributedToQueueConsult`
- `CallDistributedToQueueUnknown`

Media-Channel Actions

Media-channel actions originate from an Interaction Server that is configured in Genesys eServices (previously called Multimedia). Media-channel actions are separated into the following two groups:

- Interaction-related actions, which reflect events arising from particular stages of interaction processing (identified by the `InteractionID`).
- Non–interaction-related actions, which are caused by events not stemming from any particular interaction.

Media-channel actions also can be categorized as durable or instantaneous.

Note that Stat Server retains the interaction ID of an interaction in memory, because this ID provides the criterion for distinguishing between actions.

Refer to the *Open Media Interaction Model Reference Guide* for information about Reporting protocol events.

Durable, Non-Interaction-Related Actions

The following are the durable, non–interaction-related actions that Stat Server generates:

- `Active`
- `Available`
- `Blocked`
- `NotAvailable`

Active

This durable action tracks how long a media channel has been active for a particular agent (or place).

Available

This durable action indicates that an agent (or place) is ready to receive interactions on a particular media channel. This action is similar to `WaitForNextCall` in the telephony model.

Blocked

This durable action indicates that an agent (or place) has put himself or herself into the `NotReady` state for a particular media, and/or that he or she has selected `DoNotDisturb`. This action is similar to the `NotReadyForNextCall` action.

NotAvailable

This durable action indicates that a particular media channel is not ready to receive interactions from a router, solely because of capacity constraints. The media channel is otherwise ready to receive interactions—that is, the agent is in the `Ready` state and has not selected `DoNotDisturb` (`DNDoff`) for this media channel.

Durable, Interaction-Related Actions

The following are the durable, interaction-related actions that Stat Server generates for agent and place media channels:

- `Delivering`
- `Handling`
- `HandlingInbound`
- `HandlingInternal`
- `HandlingOutbound`
- `HandlingUnknown`

Delivering

Stat Server generates this durable action, also called `InteractionDelivering`, for all interactions in the `Delivering` phase for a particular media on agent and/or place objects. `Delivering` follows `EventInvite`, and precedes receipt of `EventPartyChanged`, `EventRevoked`, and `EventRejected` for a particular interaction, agent, and media. This action is similar to `CallRinging` in the telephony model.

Handling

Stat Server generates this durable action, also called `InteractionHandling`, when an agent (or place) accepts an inbound, outbound, or internal interaction on a particular media. This action follows `EventAccepted` and has no equivalent in the telephony model. This action terminates when the agent leaves the interaction or when the `NotMonitored` action starts. This action can be considered a combination of all five interaction-type actions.

Handling is always simultaneous with one of the following interaction-type actions:

- HandlingInbound
- HandlingInternal
- HandlingOutbound
- HandlingUnknown

The interaction type that Stat Server receives from Interaction Server with EventAccepted determines which of the above four actions occurs simultaneously with Handling.

HandlingInbound

Stat Server generates this durable action, also called InteractionHandling Inbound, when an agent (or place) accepts an inbound interaction on a particular media. This action terminates when the agent leaves the interaction or when the NotMonitored action starts. HandlingInbound is similar to CallInbound in the telephony model.

HandlingInternal

Stat Server generates this durable action, also called InteractionHandling Internal, when an agent (or place) accepts an internal interaction on a particular media. This action terminates when the agent leaves the interaction or when the NotMonitored action starts. HandlingInternal and is similar to CallInternal in the telephony model.

HandlingOutbound

Stat Server generates this durable action, also called InteractionHandling Outbound, when an agent (or place) accepts an outbound interaction on a particular media. This action terminates when the agent leaves the interaction or when the NotMonitored action starts. HandlingOutbound and is similar to CallOutbound in the telephony model.

HandlingUnknown

Stat Server generates this durable action, also called InteractionHandling Unknown, when an agent (or place) accepts an interaction of unknown type on a particular media. This action terminates when the agent leaves the interaction or when the NotMonitored action starts. HandlingUnknown and is similar to CallUnknown in the telephony model.

Momentary, Interaction-Related Actions

The following are the momentary, interaction-related actions that Stat Server generates:

- `BeingCoached`
- `BeingMonitored`
- `CoachingByIntrusionInitiated`
- `CoachingByRequestInitiated`
- `CoachingRequested`
- `ConferenceJoined`
- `ConferenceJoinedByIntrusion`
- `ConferenceMade`
- `DeliveringStarted`
- `HandlingInboundStarted`
- `HandlingInternalStarted`
- `HandlingOutboundStarted`
- `HandlingUnknownStarted`
- `HandlingStarted`
- `MonitoringInitiated`
- `Pulled`
- `Started`
- `StartedInbound`
- `StartedInternal`
- `StartedOutbound`
- `StartedUnknown`
- `TransferMade`
- `TransferTaken`

BeingCoached

Stat Server generates this momentary action when coaching begins on a chat interaction, whether by invitation or not.

BeingMonitored

Stat Server generates this momentary action when monitoring begins on a chat interaction.

CoachingByIntrusionInitiated

This momentary action indicates that a resource has begun coaching a chat interaction without the invitation of the agent who is conducting the chat session.

CoachingByRequestInitiated

This momentary action indicates that a resource has begun coaching a chat interaction at the request of the agent who is conducting the chat session.

CoachingRequested

This momentary action indicates that an agent requested coaching regardless of whether a coaching session was actually granted.

ConferenceJoined

This momentary action, also called `InteractionConferenceJoined`, indicates that an agent has accepted and joined a conference. This action is similar to `CallConferenceJoined` in the telephony model.

ConferenceJoinedByIntrusion

Stat Server generates this momentary action when a resource joins a conference without the invitation from the agent who is conducting the conference.

ConferenceMade

This momentary action, also called `InteractionConferenceMade`, indicates that an agent has initiated a conference. This action is similar to `CallConference Made` in the telephony model.

DeliveringStarted

This momentary action, also called `InteractionDeliveringStarted`, marks the onset of interaction delivery (`Delivering`) for any interaction type, and it occurs when an agent is invited to an interaction. This action is similar to `RingingStarted` in the telephony model.

HandlingInboundStarted

Stat Server generates this momentary action, also called `InteractionHandling InboundStarted`, when an agent accepts an inbound interaction. `Handling InboundStarted` is similar to `CallInboundStarted` in the telephony model.

HandlingInternalStarted

Stat Server generates this momentary action, also called `InteractionHandling InternalStarted`, when an agent accepts an internal interaction. `Handling InternalStarted` is similar to `CallInternalStarted` in the telephony model.

HandlingOutboundStarted

Also called `InteractionHandlingOutboundStarted`, Stat Server generates this momentary action when an agent accepts an outbound interaction. `Handling OutboundStarted` is similar to `CallOutbound Started` in the telephony model.

HandlingUnknownStarted

Also called `InteractionHandlingUnknownStarted`, Stat Server generates this momentary action when an agent accepts an interaction of unknown type. `HandlingUnknownStarted` is similar to `CallUnknownStarted` in the telephony model.

HandlingStarted

This momentary action, also called `InteractionHandlingStarted`, marks the onset of interaction handling (`Handling`) for any interaction type, and it occurs when an agent accepts an inbound, outbound, or internal interaction. This action has no equivalent in the telephony model.

`HandlingStarted` is always simultaneous with one of the following interaction-type actions:

- `HandlingInboundStarted`
- `HandlingInternalStarted`
- `HandlingOutboundStarted`
- `HandlingUnknownStarted`

The interaction type that Stat Server receives from Interaction Server with `EventEstablished` determines which of the above four actions occurs simultaneously with `HandlingStarted`.

MonitoringInitiated

Stat Server generates this momentary action when an agent monitors an interaction.

Pulled

Stat Server generates this momentary action, also called `InteractionPulled`, every time it detects that an interaction has been pulled from the interaction queue and directed to be delivered to a resource.

Started

This momentary action, also called `InteractionStarted`, indicates that an agent has initiated an interaction (of any type). This action has no equivalent in the telephony model.

`Started` is always simultaneous with one of the following interaction-type actions:

- `StartedInbound`
- `StartedInternal`
- `StartedOutbound`
- `StartedUnknown`

The interaction type that Stat Server receives from Interaction Server with `EventEstablished` determines which of the above four actions occurs simultaneously with `Started`.

StartedInbound

This momentary action, also called `InteractionStartedInbound`, indicates that an agent has accepted an inbound interaction. This action has no equivalent in the telephony model.

StartedInternal

This momentary action, also called `InteractionStartedInternal`, indicates that an agent has initiated an internal interaction. This action has no equivalent in the telephony model.

StartedOutbound

This momentary action, also called `InteractionStartedOutbound`, indicates that an agent has initiated an outbound interaction. This action has no equivalent in the telephony model.

StartedUnknown

This momentary action, also called `InteractionStartedUnknown`, indicates that an agent has initiated an interaction of unknown type. This action has no equivalent in the telephony model.

TransferMade

This momentary action, also called `InteractionTransferMade`, indicates that an agent has transferred the interaction to another agent directly; that is, the transfer does not occur through a mediation DN. This action is similar to `Call TransferMade` in the telephony model.

`TransferMade` is always simultaneous with one of the following interaction-type actions:

- `TransferMadeInbound`
- `TransferMadeInternal`
- `TransferMadeOutbound`

The interaction type that Stat Server receives from Interaction Server with `EventEstablished` determines which of the above three actions occurs simultaneously with `TransferMade`.

TransferTaken

This momentary action, also called `InteractionTransferTaken`, indicates that an agent has received the transferred interaction. This action is similar to `CallTransferTaken` in the telephony model.

Retrospective, Interaction-Related Actions

The following are the retrospective, interaction-related actions that Stat Server generates for agent and place media channels originating from Interaction Server:

- `Accepted`
- `Rejected`
- `Revoked`
- `Stopped`
- `StoppedInbound`
- `StoppedInternal`
- `StoppedOutbound`
- `StoppedUnknown`

Accepted

This retrospective action, also called `InteractionAccepted`, indicates that an agent (or place) has accepted a delivered interaction. This action terminates the `Delivering` action, and it is similar to `CallAnswered` in the telephony model.

Rejected

This retrospective action, also called `InteractionRejected`, indicates that an agent has rejected the delivered interaction. This action terminates `Delivering` actions.

Revoked

This retrospective action, also called `InteractionRevoked`, indicates that the system has revoked the interaction at the agent’s desktop. This action has no equivalent in the telephony model.

Stopped

This retrospective action, also called `InteractionStopped`, indicates that an agent has terminated an interaction (of any type). This action has no equivalent in the telephony model.

`Stopped` is always simultaneous with one of the following interaction-type actions:

- `StoppedInbound`
- `StoppedInternal`
- `StoppedOutbound`
- `StoppedUnknown`

The interaction type that Stat Server receives from Interaction Server with `EventEstablished` determines which of the above four actions occurs simultaneously with `Stopped`.

StoppedInbound

This retrospective action, also called `InteractionStoppedInbound`, indicates that an agent has terminated an inbound interaction. This action has no equivalent in the telephony model.

StoppedInternal

This retrospective action, also called `InteractionStoppedInternal`, indicates that an agent has terminated an internal interaction. This action has no equivalent in the telephony model.

StoppedOutbound

This retrospective action, also called `InteractionStoppedOutbound`, indicates that an agent has terminated an outbound interaction. This action has no equivalent in the telephony model.

StoppedUnknown

This retrospective action, also called `InteractionStoppedUnknown`, indicates that an agent has terminated an interaction of unknown type. This action has no equivalent in the telephony model.

Retrospective, Interaction-Related Actions Generated on Virtual Queues

The following are the retrospective, interaction-related actions that Stat Server generates for virtual queue objects that are controlled by a Multimedia-monitored switch:

- CallAbandonedFromRinging
- CallAnswered
- CallReleased

Note: In order to properly count any media-related interactions passing through a virtual queue, the virtual queue must be configured on multi-media switch.

CallAbandonedFromRinging

This retrospective action occurs when Stat Server receives from Interaction Server either of the following events along with the Abandoned reason:

- event_rejected, as a result of an agent rejecting the invitation to participate in interaction processing.
- event_revoked, as the result of Interaction Server revoking the interaction from the agent.

The duration that Stat Server prescribes to this action is the interval from EventQueued to event_rejected or from EventQueued to event_revoked.

This action is similar to CallAbandonedFromRinging (page 105) in the telephony model.

CallAnswered

This retrospective action occurs when Stat Server receives event_party_added as a result of an agent accepting the interaction. The duration that Stat Server prescribes to this action is the interval from EventQueued to event_party_added.

This action is similar to CallAnswered (page 105) in the telephony model.

CallReleased

This retrospective action occurs when Stat Server receives event_party_removed as a result of an agent finishing an interaction. Stat Server calculates the duration from the moment of acceptance of an interaction (event_party_added) until the moment that the last involved party of the interaction leaves it (event_party_removed).

Stat Server does not generate this action if an interaction is offered to a contact-center handling resource but the resource does not explicitly accept or answer it. Such may be the case where the configured time interval for acceptance times out and Stat Server receives the `event_revoked` event from Interaction Server.

This action is similar to `CallReleased` (page 106) in the telephony model.

Media-Channel Action Attributes

Table 22 lists all the possible action attributes that can be included with media-channel actions. These attributes deliver specific information that enables Stat Server to identify the objects that are related to each action, as well as additional information such as `Reason` codes.

Table 22: Media-Channel Action Attributes

Parameter Name	Description
InteractionID	The unique identifier assigned to the interaction by the Universal Contact Server (UCS) database or by another application that created the interaction.
MediaTypeID	The type of media used in the interaction.
EventTime	The time at which the event occurred, expressed as a UTC (Universal Time Coordinated) value.
PlaceID	The unique identifier of the place with which the agent who issued the request that resulted in this event is associated. This parameter is mandatory if the change of condition reported by this event was caused by a request from the agent.
TenantID	The unique identifier of the tenant associated with this event.
AgentID	The unique identifier of the agent who issued the request that resulted in this event. This parameter is mandatory if the change of condition reported by this event was caused by a request from the agent.
RouterID	The unique identifier of the router that issued the request that resulted in this event; or the unique identifier of the router to which this interaction is submitted (in the case of an `EventRouting` event). This parameter is mandatory if the change of condition reported by this event was caused by a request from the router.

Table 22: Media-Channel Action Attributes (Continued)

Parameter Name	Description
StrategyID	The unique identifier of a strategy, the execution of which caused the router to issue the request that resulted in this event; or the unique identifier of a strategy to which this interaction is submitted (in the case of an `EventRouting` event). This attribute is mandatory if the change of condition reported by this event was caused by a request from the router.
MediaServerID	The Media Server that issued the request that resulted in this event.
Queue	The queue in which the interaction should be placed.
ParentInteractionID	The identifier stored in the UCS database for the parent interaction of the current interaction. This attribute is mandatory if the interaction is a child interaction.
Reason	The reason for the condition reported by this event.
UserData	The user-entered data attached to the interaction.
AddedProperties	The list of added properties.
ChangedProperties	The list of changed properties.
DeletedProperties	The list of deleted properties.
WorkbinTypeID	The type of workbin in which the interaction should be placed.
WorkbinAgentID	The Agent ID of the workbin in which the interaction should be placed. This attribute is mandatory if a workbin is defined for an agent.
WorkbinGroupID	The Agent Group ID of the workbin in which the interaction should be placed. This attribute is mandatory if a workbin is defined for a group of agents.
ViewID	The view that the agent used to pull the interaction.
TargetAgentID	The agent who pulled this interaction.
TargetPlaceID	The Place to which this interaction was pulled.

Chapter

5 Object Statuses

The state of an object can be described within the Genesys Statistics Model by a set of nonoverlapping statuses. A *status* is the highest-priority action out of all ongoing durable actions occurring at an object, according to Status Priority tables or other rules. Stat Server ascribes only one status to an object at any particular time.

This chapter describes object statuses with respect to Stat Server and how they are classified, defined, and determined.

Notes:

- Only qualified Genesys personnel should change the options that define Status Priority tables (`DefaultDNSPT`, `DefaultAgentSPT`, and `DefaultRPSPT`).
- Values specified for the `DefaultGroupSPT` configuration option no longer impact Stat Server operation.
- The following DN statuses can co-exist with one of their respective interaction-type statuses: `CallDialing`, `CallRinging`, `AfterCallWork`, and `CallOnHold`.

Regular DN Status

The following subsections describe the statuses that are applicable to directory numbers:

- `NotMonitored`
- `Monitored`
- `LoggedIn`
- `OnHook`
- `AfterCallWork`
- `CallConsult`
- `CallDialing`
- `CallInbound`
- `CallInternal`
- `CallOutbound`
- `CallOnHold`
- `CallRinging`
- `CallUnknown`
- `NotReadyForNextCall`
- `OffHook`
- `WaitForNextCall`
- `LoggedOut`

NotMonitored

This status coincides with the `NotMonitored` action, as well as when Stat Server cannot not receive data from one or more T-Servers for a particular DN. This status also appears if you disable a particular DN within Configuration Manager.

Monitored

This status coincides with the `Monitored` action and appears only after initial connection to T-Server. This action disappears when Stat Server receives the `EventRegistered` TEvent from T-Server.

LoggedIn

This special status appears when Stat Server detects synchronization problems between T-Server and the PBX. This status's appearance indicates that T-Server was able to reconstruct agent login on a particular DN, but that T-Server was unable to obtain DN status from the PBX. Stat Server does not derive this status from actions, although the `LoggedIn` action does coincide with `LoggedIn` status. Only a few T-Server types generate this status.

To resolve these synchronization problems, you must manually clear this status by logging out of the DN for which this status appears, and then logging back in. Failure to do so causes Stat Server to calculate unreliable statistics. This status usually appears immediately following link disconnection of T-Server from the PBX.

When working with T-Server or SIP Server for some types of switches, Stat Server reports the `LoggedIn` status for an agent, a DN, or a place given the following conditions:

1. T-Server or SIP Server starts or restores its connection with the switch while an agent handles an interaction.
2. In response to a T-Server (or SIP Server) status query, the switch returns the `Ready` status for the agent and the `NOT_IDLE` status for the agent's DN; however, the switch does not provide any interaction identifiers.
3. When Stat Server registers for this DN, Stat Server receives `Event Registered` with the agent status `Ready` and with the DN status `NOT_IDLE`, but without interaction information.

Given these sequence of events, Stat Server then starts the `LoggedIn` status for this agent, DN, and/or place, which lasts until T-Server or SIP Server reports one of the following events for this DN:

- `EventAgentNotReady`
- `EventAgentReady`
- `EventDialing`
- `EventDNDOn`
- `EventDNDOff`
- `EventOnHook`
- `EventOffHook`
- `EventAgentLogin`
- `EventAgentLogout`
- `EventLinkConnected`
- `EventLinkDisconnected`
- `EventRinging`

`Note:` See also "DN Actions at Newly Registered DNs" on page 68.

New to Release 7.5 of Stat Server is its added ability to detect the login status of switches named in virtual agent group (VAG) scripts. Previously, with regard to VAG scripts, Stat Server detected the login status only of queues.

OnHook

This status appears on a DN under the following circumstances:

- The receiver is put back on the hook after having been previously off the hook.
- There is no activity on the DN.

This status explicitly precedes `WaitForNextCall` status in the `DefaultDNSPT` configuration option.

AfterCallWork

This status appears when the agent sets a particular DN to a special post-interaction-processing mode, and no already-established telephony interactions are currently occurring on the DN.

CallConsult

This status appears when at least one telephony interaction of `consult` interaction type is currently established on a particular DN, and no other already-established telephony interactions of `Internal`, `Outbound`, or `Inbound` interaction type are currently occurring on the DN.

CallDialing

This status appears when a particular DN (phone receiver) is off-hook, the DN is in `Ready` state, dialing is in progress, and no other telephony activity is taking place on the DN.

CallInbound

This status appears when at least one telephony interaction of `Inbound` interaction type is currently occurring on the DN.

CallInternal

This status appears when at least one telephony interaction of `Internal` interaction type and no other already-established telephony interactions of `Outbound` or `Inbound` type are currently occurring on the DN.

CallOutbound

This status appears when at least one telephony interaction of `Outbound` interaction type and no other already-established telephony interactions of `Inbound` interaction type are currently occurring on the DN.

CallOnHold

This status appears when a telephony interaction—of any origin—is on hold at a particular DN, and no other already established telephony interactions, which are not on hold, are currently occurring on the DN.

Stat Server removes from consideration the underlying DN action of an established telephony interaction while the interaction is on hold, thereby allowing:

- The `CallOnHold` status to prevail against other occurring DN actions (except `CallConsult`) when Stat Server determines DN status *on the same DN.*

- CallInbound, CallOutbound, CallInternal, or CallUnknown statuses to prevail when Stat Server determines overall agent or place status, which includes the status consideration of other DNs associated with the agent or place.

CallRinging

This status appears when the PBX alerts a particular DN of an incoming interaction, the DN is in Ready state, and no other already-established telephony interactions are currently in progress on the DN.

CallUnknown

This status appears when at least one telephony interaction of unknown origin is established, and no other already established telephony interactions of known origin are currently occurring on the DN.

NotReadyForNextCall

This status appears when the agent sets a particular DN to a NotReady state (for example, the agent presses the Not Ready button), no other already established telephony interactions are currently in progress for the DN, and the agent has not placed the DN in AfterCallWork mode.

OffHook

This status appears when the agent sets a particular DN to Ready state, and the only activity on the DN is that the phone receiver is off the hook.

WaitForNextCall

This status appears when no activity is currently in progress on a particular DN, and the agent has placed the DN in Ready state—for example, the agent presses the Ready button.

LoggedOut

Though LoggedOut is a valid value in the DefaultDNSPT option, this status never appears on a DN.

Place and Agent Status

`Place status` is the status of a DN linked to the place with the highest priority according to the Agent Status Priority Table, specified as the `DefaultAgentSPT` configuration option found in the `statserver` section. Place status is computed from the actions occurring on all DNs and/or media-channels belonging to that place using the following algorithm:

1. A place that has no devices or media channels, has `NotMonitored` status.
2. The voice-only status of place is computed from the statuses of voice devices (for example, voice DNs or voice-enabled multimedia DNs) belonging to that place, according to the algorithm described in the section below.
   - The status of a voice DN is computed according to the DN status priority table (see page 133).
   - The status of voice-enabled multimedia DN is computed based on media-independent actions and voice actions only, according to the DN status priority table (see page 133).
3. The other-than-voice status, or nonvoice status, of a place is computed from the statuses occurring on media channels and logged-in media-enabled multimedia DNs:
   - The status of a media-channel is computed according to the status priority table for multimedia (see page 135).
   - The status of media-enabled multimedia DN is computed based on media-independent actions and non-voice actions only, according to the DN status priority table (see page 133).
4. For a place that has neither media channels nor media-enabled multimedia DNs but does have voice devices, place status is equivalent to voice-only status.
5. For a place that has no voice devices but does have media channels and/or nonvoice-enabled multimedia DNs, place status is equivalent to nonvoice status.
6. For a place that has both voice devices and media channels and/or non-voice-enabled multimedia DNs, place status is computed as follows. The first satisfied condition defines place status:
   a. If voice status is higher than `NotReadyForNextCall`, then place status is equivalent to voice status.
   b. If nonvoice status is higher than voice status, then place status is equivalent to nonvoice status.
   c. If the place has media channels, voice devices, and no login to a voice device occurs at the place, then place status is equivalent to nonvoice status.
   d. Place status is equivalent to voice status.

If agent is logged into a place, agent status inherits the place's status; otherwise, agent status is `LoggedOut`.

Agent Status Priority Table

The standard Agent Status Priority Table is the same as the standard Regular DN Status Priority Table on page 133, with one more status, `LoggedOut`, that is supported for agents. The `LoggedOut` status has the highest priority out of all agent statuses.

The required value format for the `DefaultAgentSPT` option is the same as that for `DefaultDNSPT`.

For an agent who is logged in at a place (see page 66), the agent status coincides with the place status of the place where the agent is logged in.

Note: Reconfigure a `Place` object only when no agent is logged in to the corresponding place. Dynamic reconfiguration of a `Place` object with a logged-in agent might affect Stat Server reports on the place status.

When several DNs of any DN type are associated with the same `Place` object, Stat Server uses the following algorithm to determine the voice-only place status:

1. If an agent is currently logged in at the extension or position, and if the status of the `Extension` or `Position` has a higher priority than `NotReadyForNextCall`, Stat Server uses only statuses of DNs of the `Position` or `Extension` type in calculating place status.

Note: For the status of an `Extension` DN to affect the status of a place, an agent must be logged in at a position if there is a position DN that belongs to the same switch.

Stat Server treats a position DN accompanied with one or more extensions that belong to the same switch as a single multi-line phone. In other words, Stat Server models a place with a single position and one or more extensions as a multi-line phone.

To prevent Stat Server, in the calculation of the place status, from using the status of an extension that does not have an agent currently logged in, set the `position-extension-linked` configuration option to `no`.

2. If an agent is currently logged in at the extension or position, and if the status of the `Extension` or `Position` has a lower or the same priority as `NotReadyForNextCall`, Stat Server uses statuses of type `Extension` and `Position`, and the statuses of all other types of DNs at which agents are currently logged in, in calculating place status.
3. If an agent is currently logged in at a DN of a type other than `Extension` or `Position`, Stat Server uses only statuses of DNs at which agents are currently logged in, in calculating place status.
4. If no agents are currently logged in at the DNs associated with a `Place` object, Stat Server uses statuses of all DNs in calculating place status.

When the resulting status is WaitForNextCall, and if the place does not contain DNs of the Voice Treatment Port type, Stat Server substitutes the place status to NotReadyForNextCall.

5. If the agent, place, or DN is disabled, Stat Server sets the status of the disabled object to Monitored, regardless of the value of the ignore-disabled-objects-in-group-statistics configuration option.

Note: On the Meridian 1 switch, Stat Server might incorrectly report the status of a Place object when that place contains two physical phones and an agent is assigned two login IDs. In this case, when the agent logs in to one of the two phones, the agent status might be reported as NotReady. The status will be incorrect until the agent logs in to the DNs of the Position type on both phones and the WaitForNextCall action starts for both DNs.

For nonvoice-only interactions occurring at a place, Stat Server assigns the highest priority status among all media channels that are registered at the place.

Group Status

Place Group and Agent Group objects can hold one of these statuses:

- Monitored
- NotMonitored
- WaitForNextCall
- NotReadyForNextCall

In addition, Agent Group objects can also hold a LoggedOut status.

Stat Server determines the status of place groups according to these rules:

1. If every place in the group has NotMonitored status, the group has NotMonitored status.
2. If at least one place in the group has Monitored status, and every place in the group has either Monitored or NotMonitored status, the group has Monitored status.
3. If at least one place in the group has WaitForNextCall status, the group has WaitForNextCall status.
4. In all other cases, the group has NotReadyForNextCall status.
5. An empty place or agent group has Monitored status.

Stat Server determines the status of agent groups according to the preceding rules 1–4 applied to all the places where an agent is logged in (see page 52).

You cannot affect place group status or agent group status by modifying the status priority tables, which are described in the following section.

Status Priority Tables

Regular DN Status Priority Table

The standard Regular DN Status Priority Table, specified by the `DefaultDNSPT` configuration option, defines the priority level and lists actions (separated by commas) in order of increasing priority, as follows:

```
NotMonitored,
Monitored,
LoggedIn,
OnHook
WaitForNextCall,
OffHook,
CallDialing,
CallRinging,
NotReadyForNextCall,
AfterCallWork,
CallOnHold,
CallUnknown,
CallConsult,
CallInternal,
CallOutbound,
CallInbound,
LoggedOut
```

Two additional statuses, `ASM_Engaged` and `ASM_Outbound,` may appear if you have activated the active switching matrix (ASM). `ASM_Engaged` appears if an agent is waiting for a customer. `ASM_Outbound` call is similar to `CallOutbound` with the call being initiated within the ASM.

It is possible to change the appearance of statuses by rearranging their order in the status priority tables, but this action changes the definition of many of the statuses, and Genesys does not recommend this action. Contact Genesys Technical Support for further information.

Stat Server uses the Regular DN Status Priority Table if the `DefaultDNSPT` option is not specified or if the option's value consists of an ellipsis (three consecutive dots).

The `DefaultDNSPT` option must consist of a string consisting of these actions (in any order, separated by commas) or a subset of these actions (with a single occurrence of an ellipsis in the comma-separated list). In the latter case, all missing actions in the list have greater priority than actions preceding the ellipsis, and lesser priority than actions following the ellipsis. The missing actions are prioritized as specified in the standard Regular DN Status Priority Table.

Mediation DN Status Priority Table

The standard Mediation DN Status Priority Table defines the priority level and lists actions (separated by commas) in order of increasing priority, as follows:

```
NotMonitored,
Monitored
```

Stat Server uses this table for mediation DNs if the `DefaultRPSPT` option is not specified or if its value consists of an ellipsis.

The `DefaultRPSPT` option must be a string consisting of actions (in any order, separated by commas) or of a subset of actions (with a single occurrence of an ellipsis in the comma-separated list). In the latter case, all missing actions have greater priority than actions preceding the ellipsis in the list, and lesser priority than actions following the ellipsis. The missing actions prioritize as specified in the standard Mediation DN Status Priority Table.

Note: Call-type actions that are not listed in the Regular DN Status Priority Table or Mediation DN Status Priority Table are not used to determine status. The regular DN actions `LoggedIn` and `LoggedOut` do not affect DN status either.

DN status inherits the attached data from the highest-priority action. You can use filters on the attached data, but you cannot apply custom formulas to it.

Keep in mind that:

- Because more than one action of the same kind can occur on a DN at one time, when such an action determines status, the attached data of the status cannot be predicted. Therefore, use filters cautiously with attached data for statuses.
- The duration of a status, in general, differs from the duration of underlying actions. A status begins when an action becomes the highest-priority current action. A status ends when another action becomes the new highest-priority current action. Therefore, for the duration of the same status, several similar actions may have succeeded one another.

Media-Channel Status Priorities

For DN types that enable the handling of media-channel interactions from Interaction Server (from the Genesys Multimedia Solution), Stat Server observes the following ranking, from lowest to highest:

```
Active
Available
NotAvailable
Blocked
InteractionDelivering
InteractionHandlingUnknown
InteractionHandlingInternal
InteractionHandlingOutbound
InteractionHandlingInbound
```

This ranking is inherent to Stat Server and cannot be reconfigured otherwise.

Multimedia DN Status Priorities

For multimedia DNs, such as those that are controlled by a SIP Server, Stat Server observes the following algorithm for determining status:

- *Voice-only status* is computed on the basis of media-independent actions and voice-related actions.
- *Nonvoice status* is computed on the basis of media-independent actions and nonvoice-related actions.
- The final status is computed as a composition of voice-only status and nonvoice status. If the two are equal, Stat Server assigns the corresponding voice action as the DN’s status action.

This algorithm is inherent to Stat Server and cannot be changed or reconfigured otherwise.

Chapter

6 Statistical Categories

This chapter introduces Stat Server statistical categories and explains how Stat Server calculates statistics that are defined using these categories. This information pertains to the values that you might specify in the `Category` option of a stat type (described on page 40.)

Information in this chapter is divided among the following topics:

Categories and Masks

A *statistical category* is a general definition that determines how to calculate a statistic on the basis of one or two lists of actions (masks) supplied as separate elements of a statistical type.

Subject of Calculation

The aggregated values discussed here are calculated on the basis of a subject specified in the definition of a statistical type, which can be either a DN action or the status of an object. Because statuses are merely highest-priority actions, the computations are the same for any subject, except for the `TotalNumber`, `TotalTime`, `MaxTime`, `MinTime`, and `TotalAdjustedTime` aggregated values (see the next section). *Aggregated custom values* cannot be computed on the basis of status.

Aggregated Values

The actions listed in a mask are used to maintain *aggregated values*. Every kind of aggregated value is available as a category. Other categories calculate statistics by using an additional computation that is based on aggregated values.

Historical aggregated values are based on statuses and actions during a specified interval (configured as a time profile). *Current aggregated values* are based only on statuses and durable actions that occur at the current moment; instantaneous actions that are listed in the mask are ignored. These values do not depend on computation intervals.

Table 23: Historical and Current Aggregated Values

Historical	Current
TotalNumber (page 145) TotalAdjustedNumber (page 145)	CurrentNumber (page 149)
TotalTime (page 146) TotalAdjustedTime (page 145)	CurrentTime (page 150) CurrentContinuousTime (page 148)
MaxTime (page 143)	CurrentMaxTime (page 148)
MaxNumber (page 143)	
MinTime (page 144)	CurrentMinTime (page 148)
MinNumber (page 143)	
	CurrentAverageTime (page 147)

Aggregated Values using TimeRanges

For historical and current statistical categories that use time ranges (namely, `TotalNumberIn TimeRange`, `TotalNumberInTimeRangePercentage`, `Service Factor1`, `CurrentNumberIn TimeRange`, and `CurrentNumberInTimeRange Percentage`), Stat Server maintains the restricted aggregated values listed in Table 24:

Table 24: Aggregated Values using TimeRanges

Historical	Current
TotalNumberInTimeRange (page 146)	CurrentNumberInTimeRange (page 149)
TotalTimeInTimeRange (page 147)	
ServiceFactor1 (page 152)	

Averages of Current Values

Averages of current values are based on an average number or duration of durable actions going on at the current moment; instantaneous actions listed in the mask are ignored.

The two kinds of averages of current values are:

- `AverageOfCurrentNumber` (see page 141)
- `AverageOfCurrentTime` (see page 142)

Aggregated Custom Values

Aggregated custom values are used for computing custom-value statistical categories. They parallel aggregated time values; however, they do not aggregate duration values. Values are obtained from evaluating custom formulas on the `UserData` structure of the interaction to which an action is related, or on the data attached to the `EventUserEvent` TEvent for the `UserEvent` action. The syntax of custom formulas is described in “Custom-Value Statistical Types” on page 45.

The evaluation of custom formulas, which is conducted according to different rules for different classes of actions, is described in detail in Chapter 10. Aggregated custom values depend not only on the mask and, for historical values, the interval from which the statistic is calculated, but also on the specified custom formula.

Accordingly, there are three kinds of historical aggregated custom values and three kinds of current aggregated custom values, which are provided in Table 25:

Table 25: Aggregated Custom Values

Historical	Current
TotalCustomValue (page 151)	CurrentCustomValue (page 151)

Table 25: Aggregated Custom Values (Continued)

Historical	Current
MaxCustomValue (page 150)	CurrentMaxCustomValue (page 152)
MinCustomValue (page 151)	Current MinCustomValue (page 152)

Filtered, Aggregated Values

When a filter is set for a statistical type, only those actions for which the evaluated filter expression is true are considered when calculating the aggregated values.

The filter expression is evaluated over the `UserData` structure belonging to the action or status. For instantaneous actions or durable actions that have ended, the `UserData` structure is the same as the last `UserData` structure received from T-Server with one of the following T-Library events:

- The event that caused the action's occurrence or start
- The event that caused the action to end
- Event `EventAttachedDataChanged` received for the DN while the action was in progress (for durable actions only)

A DN status (see page 126) inherits the `UserData` structure of the action that causes the status. Because more than one action of the same kind can occur simultaneously for the same DN, the definition of the `UserData` belonging to a status cannot be predetermined. Caution is advised when filtered, aggregated values are computed with a subject of DN status.

Similarly, an agent or place status (see page 130) inherits the `UserData` structure of the DN status that causes it. The `LoggedOut` agent status has no attached `UserData`. This definition is even less predetermined than the previous one; computing filtered, aggregated values with a subject of agent or place status is strongly discouraged.

Group status (see page 132) carries no `UserData` structure; if a filtered, aggregated value is requested at this subject level, the filter is ignored.

The `CurrentState` category does not take filters into account. Although you can use the `EstimWaitTime` statistical category with filters, its values lose any intuitive meaning.

Historical Categories

This section describes how Stat Server calculates statistics for the following historical statistical categories:

- `AverageNumberPerRelativeHour`
- `AverageOfCurrentNumber`
- `AverageOfCurrentTime`
- `AverageTime`
- `ElapsedTimePercentage`
- `MaxNumber`
- `MaxTime`
- `MinNumber`
- `MinTime`
- `RelativeNumberPercentage`
- `RelativeTimePercentage`
- `TotalAdjustedNumber`
- `TotalAdjustedTime`
- `TotalNumber`
- `TotalNumberInTimeRange`
- `TotalNumberInTimeRangePercentage`
- `TotalNumberPerSecond`
- `TotalTime`
- `TotalTimeInTimeRange`

AverageNumberPerRelativeHour

This statistical category returns the number of events of a particular type that occurred during an average hour. Here is the formula:

$$\text{Value} = \frac{\sum \text{TotalNumber(MainMask)}}{\sum \text{TotalTime(RelMask)}} \text{X} 3600$$

A relative mask specification is mandatory for this category. The subject applies to both `MainMask` and `RelMask`. Filters, however, can only be applied to the `MainMask`.

Here is an example of a stat-type definition using the `AverageNumberPerRelativeHour` statistical category:

```
[Average_Calls_Per_Hour]
MainMask =
CallInternal,CallInbound,CallOutbound,CallConsult,CallUnknown
RelMask = *,~LoggedOut,~NotMonitored
Subject = AgentStatus
Category = AverageNumberPerRelativeHour
Objects = GroupAgents, GroupPlaces, Agent, Place B
```

A statistic based on this stat-type definition collectively averages all types of calls that agents receive over the time they are both monitored (`~NotMonitored`) and not logged out (`~LoggedOut`)—or, in other words, logged in.

AverageOfCurrentNumber

The value represents the average of current-number measurements over a specified time interval. Unlike the behavior in releases prior to 7.0, Stat Server take current-number measurements every 2 seconds; for example, Stat Server now only notes the time whenever the current number changes—for example, when one inbound interaction enters or exits the contact center.

Also, different from prior releases is Stat Server's use of the first-observed current number in its average calculation. Prior to release 7.0, the first value was not used in the formula. The new formula:

$$\frac{\sum(N_i \times (t_i - t_{i-1}))}{t - t_0}$$

considers this value. When the denominator is `0,` Stat Server returns `0.`

Figure 15 illustrates how an `AverageOfCurrentNumber` statistic might be depicted in a graph. In this figure, *N* denotes the number of interactions currently underway. When a new current interaction, *N*+1, enters the contact center, Stat Server timestamps it, represented by the corresponding time value along the x-axis. The variable *i* serves as a change index in the number of interactions entering or leaving the contact center. Point *(N_i,t_i)*, then, denotes the number of interactions currently underway at a specific time during the *ith* change in the number of current observations.

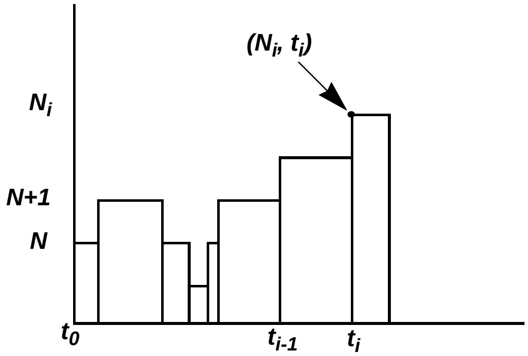


Figure 15: AverageOfCurrentNumber Statistic in Graphic Form

AverageOfCurrentTime

The value is equal to the average of current-time measurements and is calculated similarly to how `AverageOfCurrentNumber` (see previous subsection) is measured. Unlike the behavior in releases prior to 7.0, Stat Server now uses the first-observed current time in its average calculation.

AverageTime

$$\text{Value} = \frac{\text{TotalTime(ActiveMask,Interval)}}{\text{TotalNumber(RelativeMask,Interval)}}$$

When the denominator is `0,` the returned value is `0.`

This category results when the values for the main and relative masks are the same. A relative mask is required for this category. The value is the quotient of the `TotalTime` aggregated value (see page 146) for the main mask and the `TotalNumber` aggregated value (see page 145) for the relative mask.

ElapsedTimePercentage

$$\text{Value} = 100 \times \frac{\text{TotalTime(MainMask,Interval)}}{\text{IntervalDuration}}$$

This category returns, as a percentage, the quotient of the `TotalTime` aggregated value (see page 146) and the entire duration of the interval from which the statistic is calculated. Note that this percentage can exceed 100 if some of the actions in the main mask occur simultaneously on the object for which a statistic for this category is calculated. It can also exceed 100 when actions that start before the beginning of the interval from which the statistic is calculated end during that interval.

MaxNumber

The `MaxNumber` statistical category returns an aggregated value that represents the maximum number of durable actions or statuses that occur simultaneously during an interval.

```
Value = MaxNumber(MainMask,Interval)
```

MaxTime

The `MaxTime` statistical category returns an aggregated value that represents the maximum duration among all durations of durable and retrospective actions or of statuses listed in the mask that, during the interval from which the statistic is calculated:

- Ended (for durable actions).
- Occurred (for retrospective actions).
- Either started or are in progress (for statuses).

Momentary actions listed in the mask are ignored, because they do not have a duration. If a statistic is requested for statuses, Stat Server uses the status duration within the statistical interval for calculation; otherwise, Stat Server uses the entire action duration.

```
Value = MaxTime(MainMask,Interval)
```

MinNumber

The `MinNumber` statistical category returns an aggregated value that represents the minimum number of durable actions or statuses that occur simultaneously during an interval.

```
Value = MinNumber(MainMask,Interval)
```

MinTime

The `MinTime` statistical category returns an aggregated value that represents the minimum duration among all durations of durable and retrospective actions or of statuses listed in the mask that, during the interval from which the statistic is calculated:

- Ended (for durable actions).
- Occurred (for retrospective actions).
- Either started or are in progress (for statuses).

Momentary actions listed in the mask are ignored, because they do not have a duration. If a statistic is requested for statuses, Stat Server uses the status duration within the statistical interval for calculation; otherwise, Stat Server uses the entire action duration.

```
Value = MinTime(MainMask,Interval)
```

RelativeNumberPercentage

$$\text{Value} = 100\text{x}\frac{\text{TotalNumber(MainMask,Interval)}}{\text{TotalNumber(RelativeMask)}}$$

When the denominator is `0`, the returned value is `0`.

A relative mask is required for this category. It returns, as a percentage, the quotient of the `TotalNumber` aggregated value (see page 145) for the main mask and the quotient of the `TotalNumber` aggregated value for the relative mask. Note that if the main mask contains actions absent from the relative mask, the percentage can exceed 100.

In addition, you can specify a time range for a statistic of this category. When specified, Stat Server applies the time range to the actions/statuses that define the main mask for this statistic. So, in this case, the numerator for this category is calculated as:

```
TotalNumberInTimeRange(MainMask,Interval)
```

Refer to page 138 for additional information about historical statistical categories using time ranges.

RelativeTimePercentage

A relative mask is required for this category. It returns, as a percentage, the quotient of the `TotalTime` aggregated value (see page 146) for the main mask and the quotient of the `TotalTime` aggregated value for the relative mask. Note that if the main mask contains actions absent from the relative mask, the percentage can exceed 100.

$$\text{Value} = 100\text{x}\frac{\text{TotalTime(MainMask, Interval)}}{\text{TotalTime(RelativeMask, Interval)}}$$

When the denominator is `0`, the returned value is `0`.

TotalNumber

For statistics based on stat type definitions where `Subject=DNAction`, the `TotalNumber` statistical category returns an aggregated value that represents the total number of actions listed in the mask that ended (for durable actions) or occurred (for instantaneous actions) during the interval from which the statistic is calculated. For statistics based on stat type definitions where `Subject=DNStatus` (and, respectively, `AgentStatus` and `PlaceStatus`), this is the total number of statuses listed in the mask that either started or are in progress during the interval from which the statistic is calculated.

```
Value = TotalNumber(MainMask,Interval)
```

TotalAdjustedNumber

The `TotalAdjustedNumber` statistical category sums the total number of occurrences of actions or statuses listed in the main mask that ended during the interval from which the statistic is calculated.

Note: The `TotalAdjustedNumber` category differs from `TotalNumber` only if reset-based notification is used for the statistic. For all other notification modes, `TotalAdjustedNumber` values are identical to `TotalNumber` values.

TotalAdjustedTime

If a statistic is requested with the DN action specified, `TotalAdjustedTime` represents the sum of all durations of durable and retrospective actions listed in the mask that, during the interval from which the statistic is calculated:

- Either ended or are in progress (for durable actions)
- Occurred (for retrospective actions)

Momentary actions listed in the mask are ignored, because they do not have a duration. Only the duration time that is within the interval is used in this calculation.

For status-based statistics, `TotalAdjustedTime` is the sum of all durations of durable and retrospective actions listed in the mask that ended or occurred (for retrospective actions) during the interval from which the statistic is calculated.

Stat Server uses the overall status duration in this calculation. A statistic of this category must be requested with the reset-based notification; that is, a statistic is reset to `0` when a new interval starts.

Note: The TotalAdjustedTime category differs from TotalTime only if reset-based notification is used for the statistic. For all other notification modes, TotalAdjustedTime values are identical to TotalTime values.

TotalNumberInTimeRange

TotalNumberInTimeRange returns a restricted aggregated value that represents the total number of all durable and retrospective actions, or of statuses listed in the mask that ended (for durable actions or for statuses) or occurred (for retrospective actions) during the interval from which the statistic is calculated, and whose duration is within the specified time range.

`Value = TotalNumberInTimeRange(MainMask,Interval,TimeRange)`

TotalNumberInTimeRangePercentage

$$\text{Value} = 100\text{x}\frac{\text{TotalNumberInTimeRange(MainMask,Interval, TimeRange)}}{\text{TotalNumber(MainMask,Interval)}}$$

When the denominator is 0, the returned value is 0.

A time range is required for this category. It returns, as a percentage, the ratio of the restricted TotalNumberInTimeRange aggregated value (see page 138) and the TotalNumber aggregated value (see page 145)—that is, the percentage of times the actions in the main mask had a duration within the specified time range over the total number of times actions from the main mask occurred or ended during the specified interval.

TotalNumberPerSecond

$$\text{Value} = \frac{\text{TotalNumber(MainMask,Interval)}}{\text{IntervalDuration}}$$

This category returns the ratio of the TotalNumber aggregated value (see page 145) to the entire duration of the interval divided by the statistic.

TotalTime

The TotalTime statistical category returns an aggregated value that represents the sum of all durations of durable and retrospective actions or of statuses listed in the mask that, during the interval from which the statistic is calculated:

- Ended (for durable actions).
- Occurred (for retrospective actions).
- Either started or are in progress (for statuses).

Momentary actions listed in the mask are ignored, because they do not have a duration. If a statistic is requested for statuses, Stat Server uses the status duration within the statistical interval for calculation; otherwise, Stat Server uses the entire action duration.

`Value = TotalTime(MainMask,Interval)`

This category returns the `TotalTime` aggregated value (see page 146).

TotalTimeInTimeRange

The `TotalTimeInTimeRange` statistical category returns an aggregated value that represents the total duration of all durable and retrospective actions, or of statuses listed in the mask that ended (for durable actions or for statuses) or occurred (for retrospective actions) during the interval from which the statistic is calculated, and whose duration is within the specified time range. Unlike other historical aggregated values, these values depend not only on the mask and the interval from which the statistic is computed, but also on the time range.

`Value = TotalTimeInTimeRange(MainMask,Interval,TimeRange)`

A time range is required for this category. It returns the restricted `TotalTimeInTimeRange` aggregated value (see page 138).

Current Categories

This section describes how Stat Server calculates statistics of the following current categories:

- `CurrentAverageTime`
- `CurrentContinuousTime`
- `CurrentMaxTime`
- `CurrentMinTime`
- `CurrentNumber`
- `CurrentNumberInTimeRange`
- `CurrentNumberInTimeRangePercentage`
- `CurrentRelativeNumberPercentage`
- `CurrentRelativeTimePercentage`
- `CurrentTime`

CurrentAverageTime

The `CurrentAverageTime` statistical category provides the average of all durations of durable actions or of statuses that are listed in the mask that are occurring at that time. The durations are interpreted as the time from the beginning of a durable action or a status until the present moment.

$$\text{Value} = \frac{\text{CurrentTime(MainMask)}}{\text{CurrentNumber(RelativeMask)}}$$

When the denominator is `0`, the returned value is `0`.

This category results when the main mask and the relative mask coincide. A relative mask is required for this category. It returns, as a percentage, the quotient of the `CurrentTime` aggregated value for the main mask and the `CurrentNumber` aggregated value for the relative mask. Note that if the main mask contains actions absent from the relative mask, this percentage can exceed 100.

CurrentContinuousTime

`CurrentContinuousTime` returns an aggregated value that provides the duration of time, in seconds, during which an object status belonged to the `MainMask`, or zero, if the current object status does not belong to the `MainMask`. Stat Server increments `CurrentContinuousTime` as soon as the object status is listed in the `MainMask`. Stat Server continues to increment `CurrentContinuousTime` if the object status changes but it still belongs to the `MainMask`. As soon as the object status changes and it is no longer part of the `MainMask`, `CurrentContinuousTime` statistics reset to zero.

```
Value = CurrentContinuousTime(MainMask)
```

Similar to `CurrentTime`, this statistical category is classified as current within the Genesys call model, even though it has an accumulation component.

CurrentMaxTime

The `CurrentMaxTime` statistical category returns an aggregated value that provides the maximum duration among all durations of durable actions or of statuses that are listed in the mask that are occurring currently. The durations are interpreted as the time from the beginning of a durable action or a status until the present moment.

```
Value = CurrentMaxTime(MainMask)
```

CurrentMinTime

The `CurrentMinTime` statistical category returns an aggregated value that provides the minimum duration among all durations of durable actions or of statuses that are listed in the mask that are occurring currently. The durations are interpreted as the time from the beginning of a durable action or a status until the present moment.

```
Value = CurrentMinTime(MainMask)
```

CurrentNumber

The `CurrentNumber` statistical category returns an aggregated value that represents the total number of durable actions or of statuses that are listed in the mask that are occurring currently.

`Value = CurrentNumber(MainMask)`

CurrentNumberInTimeRange

The `CurrentNumberInTimeRange` statistical category returns an aggregated value that represents the total number of all durable actions or statuses listed in the mask that are occurring currently and whose duration is within the specified time range. Unlike other current aggregated values, this value depends not only on the mask, but also on the time range (see page 36).

`Value = CurrentNumberInTimeRange(MainMask,TimeRange)`

CurrentNumberInTimeRangePercentage

A time range is required for this category. It returns, as a percentage, the ratio of the restricted `CurrentNumberInTimeRange` aggregated value and a percentage of the `CurrentNumber` aggregated value—that is, the percentage of times the actions in the main mask had a duration within the specified time range, divided by the total number of times actions from the main mask occurred or ended during the specified interval.

$$\mathtt{Value} = \frac{\mathtt{CurrentNumberInTimeRange(MainMask, TimeRange)}}{\mathtt{CurrentNumber(MainMask)}}$$

When the denominator is `0`, the returned value is `0`.

CurrentRelativeNumberPercentage

A relative mask is required for this category. It returns, as a percentage, the quotient of the `CurrentNumber` aggregated value for the main mask and the `CurrentNumber` aggregated value for the relative mask. Note that if the main mask includes actions that do not also appear in the relative mask, this percentage can exceed 100.

$$\text{Value} = 100 \times \frac{\mathtt{CurrentNumber(MainMask)}}{\mathtt{CurrentNumber(RelativeMask)}}$$

CurrentRelativeTimePercentage

$$\text{Value} = 100 \times \frac{\mathtt{CurrentTime(MainMask)}}{\mathtt{CurrentTime(RelativeMask)}}$$

When the denominator is `0`, the returned value is `0`.

A relative mask is required for this category. It returns, as a percentage, the quotient of the `CurrentTime` aggregated value for the main mask and the `CurrentTime` aggregated value for the relative mask. Note that if the main mask contains actions absent from the relative mask, this percentage can exceed 100.

CurrentTime

The `CurrentTime` statistical category returns an aggregated value that represents the sum of all durations, in seconds, of durable actions or of statuses listed in the mask that are occurring currently. The durations are interpreted as the time from the beginning of a durable action or a status until the present moment. Stat Server resets `CurrentTime` when different actions or statuses begin, even if they are part of the mask.

```
Value = CurrentTime(MainMask)
```

Historical CustomValue Categories

This section describes how Stat Server calculates statistics of the following historical custom-value categories:

- `AverageCustomValue`
- `MaxCustomValue`
- `MinCustomValue`
- `TotalCustomValue`

AverageCustomValue

$$\text{Value} = \frac{\text{TotalCustomValue(MainMask,Interval, CustomFormula)}}{\text{TotalNumber(MainMask,Interval)}}$$

When the denominator is `0`, the returned value is `0` (the numerator is always `0` in this case).

This category returns the average value of the custom formula evaluated over all actions listed in the main mask and occurring or ending during the specified interval.

MaxCustomValue

The `MaxCustomValue` statistical category returns an aggregated value that represents the greatest of the custom-formula values evaluated over each interaction-related or `UserEvent` action listed in the mask that ended (for durable actions) or occurred (for instantaneous actions) during the interval from which the statistic is calculated.

```
Value = MaxCustomValue(MainMask,Interval,CustomFormula)
```

MinCustomValue

The `MinCustomValue` statistical category returns an aggregated value that represents the smallest of the custom-formula values evaluated over each interaction-related or `UserEvent` action listed in the mask that ended (for durable actions) or occurred (for instantaneous actions) during the interval from which the statistic is calculated.

`Value = MinCustomValue(MainMask,Interval,CustomFormula)`

This category returns the `MinCustomValue` aggregated value (see page 151).

TotalCustomValue

The `TotalCustomValue` statistical category returns an aggregated value that represents the sum of the custom formula values evaluated over each interaction-related or `UserEvent` action listed in the mask that ended (for durable actions) or occurred (for instantaneous actions) during the interval from which the statistic is calculated.

`Value = TotalCustomValue(MainMask,Interval,CustomFormula)`

Current CustomValue Categories

This section describes how Stat Server calculates statistics of the following current custom-value statistical categories:

- `CurrentAverageCustomValue`
- `CurrentCustomValue`
- `CurrentMaxCustomValue`
- `CurrentMinCustomValue`

CurrentAverageCustomValue

$$\text{Value} = \frac{\text{CurrentCustomValue(MainMask,CustomFormula)}}{\text{CurrentNumber(MainMask)}}$$

When the denominator is `0`, the returned value is `0` (the numerator is always `0` in this case).

This category returns the average value of the custom formula of all ongoing actions listed in the main mask.

CurrentCustomValue

The `CurrentCustomValue` statistical category returns an aggregated value that represents the sum of the custom formula values evaluated over each interaction-related, durable action listed in the mask that is occurring currently.

```
Value = CurrentCustomValue(MainMask,CustomFormula)
```

CurrentMaxCustomValue

The `CurrentMaxCustomValue` statistical category returns an aggregated value that represents the greatest of the custom formula values evaluated over each interaction-related, durable action listed in the mask that is occurring currently.

```
Value = CurrentMaxCustomValue(MainMask,CustomFormula)
```

CurrentMinCustomValue

The `CurrentMinCustomValue` statistical category returns an aggregated value that represents the smallest of the custom formula values evaluated over each interaction-related, durable action listed in the mask that is occurring currently.

```
Value = CurrentMinCustomValue(MainMask,CustomFormula)
```

Compound Categories

The following are compound statistical categories:

- `EstimWaitTime`
- `LoadBalance`
- `ServiceFactor1`

All compound statistical categories are historical and, thus, calculated over specified time intervals. Configured stat types for compound statistical categories must contain `Subject=DNAction` and a nonempty main mask. For example:

`EstimWaitTime` or **`LoadBalance`**

```
Subject=DNAction
Main Mask=CallWait
```

`ServiceFactor1`

```
Subject=DNAction
Main Mask=CallAnswered
```

Compound statistical categories do not account for masks but, instead, are based on fixed sets of actions that are hard-coded and that are mentioned here for each category.

Note: The `EstimWaitTime` and `LoadBalance` statistical categories are based on formulas that are valid only for single-media mediation DNs—that is, mediation DNs that satisfy the following conditions:

- All interactions that are queued to such mediation DNs are homogenous, having the same *M* media-type.

- All interactions they are distributed from such mediation DNs are diverted only to agents who handle only *M* media-type interactions.

If they are requested for other than single-media mediation DNs, statistics that are based on the these statistical categories may generate erroneous results.

EstimWaitTime

This category, which is also called `ExpectedWaitTime`, provides an estimate for the waiting time of the last call that has entered a queue. This estimate takes into account the possibility of distributing calls from different queues to the same agents. Genesys recommends that you use the `Sliding` time profile (see page 20) when you request statistics of this category.

Note: Universal Routing Server uses the name `StatExpectedWaitTime` to refer to this statistical category. Also, when you request this statistic from Universal Routing Designer or CCPulse+, use the name `ExpectedWaitTime`.

The value of a statistic that belong to this category is calculated as follows:

$$\text{Value} = \text{AHT}\frac{\text{CIQU}}{\text{AA} \times \text{EP}}$$

where:

a. `AHT` stands for *average handling time*—that is, the time that is spent, on average, in processing a call that comes from the queue and after-call work that follow such a call:

$$\text{AHT} = \frac{\text{TotalTime(Mask1, Interval)}}{\text{TotalNumber(Mask2, Interval)}}$$

where

- `Mask1` is given by the `CallReleased`, `ACWCompleted`, `ACWMissed`, and `CallMissed` actions.
- `Mask2` is given by the `CallReleased` and `CallMissed` actions.
- `Interval` is given by a supplied time profile.

If no calls from the queue have been processed yet, `AHT` is considered to be 90 seconds.

Note: This value is not configurable.

b. `CIQU` stands for *calls in queue unassigned*—that is, the number of calls that currently are waiting in the queue that cannot be distributed to agents immediately. This value is calculated, based both on the number of calls in queue:

`CIQ = CurrentNumber (CallWait)`

and on the number of agents ready (AR)—that is, the number of agents that currently are logged in and have the WaitForNextCall status.

The calculations are based on the following algorithm:

- CIQU equals zero (0) if the number of agents ready is greater than or equal to the number of calls in queue—that is, if all calls from this queue can be distributed to agents immediately.
- CIQU equals the number of calls in queue (CIQ) if no agents are currently ready (AR = 0).
- CIQU equals the difference between the number of calls in queue and the number of agents ready (CIQ–AR) if some agents are currently ready.

c. AA stands for *agents active:*

$$AA = \text{CurrentNumber}(\text{AgentActive})$$

Being active means that an agent is being logged in and is not in NotReadyForNextCall status.

If AA=0, it is replaced by 0.0001.

d. EP stands for *effective portion*—that is, the total time spent, by all agents who process calls from the queue, on calls from the queue and after-call work following such calls divided by the total time spent by these agents on calls from all originations and after-call work following these calls:

$$EP = \frac{\text{TotalTime}(\text{Mask1, Interval})}{\text{TotalTime}(\text{Mask2, Interval})}$$

where:

Mask1 is given by the CallReleased and ACWCompleted actions.

Mask2 is given by the CallReleased, CallMissed, ACWCompleted, and ACWMissed actions.

Interval is given by a supplied time profile.

If no calls coming from the queue have been processed yet, EP is considered to be 1.

The reported value is rounded to the nearest integer and should be interpreted as a number of seconds.

Note: Statistics belonging to the EstimWaitTime category always return a value of 10,000 seconds (that is 2 hours, 46 minutes, and 40 seconds) for queues where no agent is currently logged in.

This statistic works only for ACD queues and only for T-Servers that propagate the queue parameter in login messages. For T-Servers that do not do this, you must configure an association between agents and a queue in Configuration Manager as follows:

1. Select an AgentGroup (or a PlaceGroup) and open its Properties dialog box.
2. Click the Advanced tab, and then click Add to add an Origination DN object.
3. In the Browse window, double-click the switch to which the queue belongs.
4. Double-click the DN object that belongs to the switch, and then select the queue that you want to associate with this AgentGroup.
5. Click OK.
6. In the AgentGroup Properties dialog box, click OK to save the configured association.

LoadBalance

This statistical category is intended to assist clients in balancing the call loads between ACD queues and routing points. Based on the load-balancing values of different queues and routing points (among other factors), Universal Routing Server, for instance, can determine where to route calls.

Stat Server's procedure for computing load-balancing statistics uses aggregated values based on queue actions reflecting regular DNs (see page 104).

Stat Server calculates the value of a statistic belonging to this category as follows:

- 10,000,000,000, if ALI = 0

  Note: This value is not configurable.
- (CIQ – AR) / ALI, if AR > CIQ
- AHT * [(CIQ – AR + 1) / ALI], if AR <= CIQ

where:

a. ALI stands for *agents logged in:*

   ALI = CurrentNumber(AgentLogin)

b. CIQ stands for *calls in queue:*

   CIQ = CurrentNumber(CallWait)

c. AR stands for agents ready—that is, the number of agents that are currently logged in and have the WaitForNextCall status:

   AR = CurrentNumber(AgentReady)

d. AHT stands for *average handling time:*

$$\mathrm{AHT} = \frac{\mathrm{TotalTime(Mask1,Interval)}}{\mathrm{TotalNumber(Mask2,Interval)}}$$

   where:

   - Mask1 is given by the CallReleased, ACWCompleted, ACWMissed, and CallMissed actions.
   - Mask2 is given by the CallReleased and CallMissed actions.

- `Interval` is given by a supplied time profile.

This value can be negative. Its implementation does not require the explicit specification of an agent group. If no calls ever entered this queue or other queues related to this queue by agent-login and/or origination-DN association, Stat Server uses the value of the `load-balance-aht` configuration option (described in the *Stat Server Deployment Guide*) for the average handling time. After the first call has been processed by the associated agent, the new calculated value of average handling time will be applied in load-balancing calculations for all related queues and routing points.

ServiceFactor1

This statistical category is the only one that requires two time ranges. Their names in a stat-type definition must be the same as Stat Server option names for these time ranges.

For example, configure two options, `TimeRange` and `TimeRange2`, in the `TimeRanges` section of the Stat Server configuration before you request statistics in the `ServiceFactor1` category. Then, request this statistic in CCPulse+ and specify `TimeRange` and `TimeRange2` as the time ranges. If you select `Default` or `Not Applied` as a value for either time range in CCPulse+, Stat Server uses the time range of `0-20` seconds.

$$\text{Value} = 100\text{X}\frac{\text{nAnsw(TimeRange)}}{\text{nAnsw} + \text{nAband} - \text{nAband(TimeRange2)}}$$

where:

- `nAnsw(TimeRange)` is the restricted `TotalNumberInTimeRange` aggregated value (see page 138) for the `CallAnswered` mediation DN action (see page 87).
- `nAnsw+nAband` is the `TotalNumber` aggregated value (see page 145) for the list of mediation DN actions `CallAnswered`, `CallAbandoned`, and `Call AbandonedFromRinging`.
- `nAband(TimeRange2)` is the restricted `TotalNumberInTimeRange` aggregated value (see page 138) for the mediation DN actions `CallAbandoned` and `CallAbandonedFromRinging`.

If `TimeRange2` is from `0` to t_1 and `TimeRange` is from `0` to t, where t_1 is small enough, so that calls abandoned within t_1 seconds may be considered “stray” calls, and t is an upper limit, in seconds, for the interval within which calls are considered as answered without excessive delay, then, `Service Factor1` gives the percentage ratio of the calls answered without excessive delay over all calls that have been delivered or abandoned from the queue, less the number of “stray” calls.

CurrentState Categories

Current state statistical categories do not return numeric values, but rather return a structure containing current action and status information for agents, places, and groups against all Genesys-defined media types. There are three current state statistical categories:

- `CurrentState`
- `CurrentStateReasons`
- `CurrentTargetState`

Note: The structure of current-state categories might be of more interest to developers than to other types of end users. For this reason, partial structure definitions are provided to help illustrate functionality.

CurrentState

The format of the returned structure for the `CurrentState` statistical category depends on the object and subject of the statistic and can be represented as a tree. The case shown in Figure 16 applies to an agent group as object and DN action as subject (the most complete case).

The root of the tree always corresponds to the stated object of the statistic, all nodes correspond to underlying objects in the DN Action Propagation Hierarchy (see page 66), and the terminal nodes are always at the level of the stated subject of the statistic.

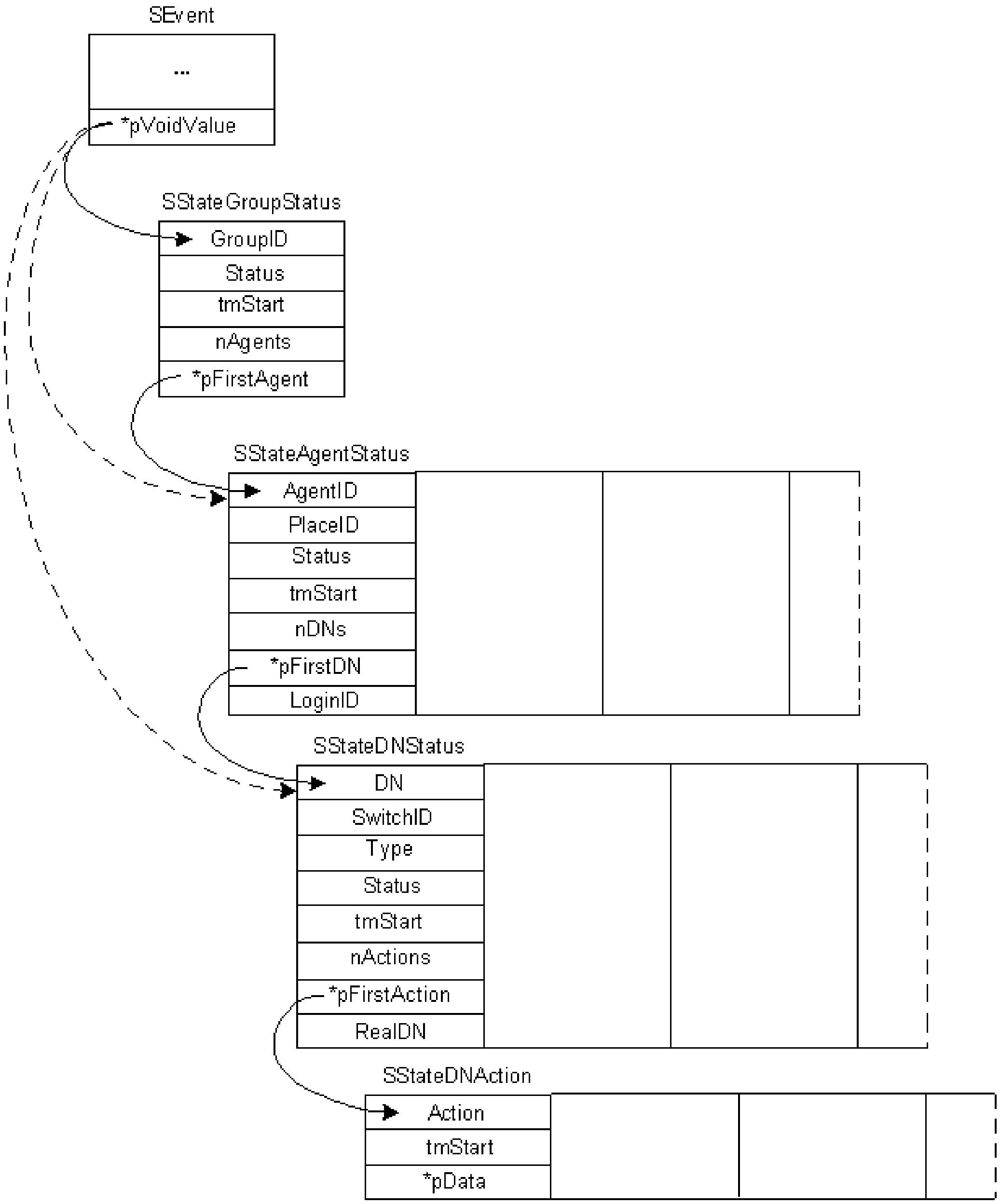


Figure 16: CurrentState Structure for an Agent Group

The `pVoidValue` parameter of the `SEvent` structure contains a pointer to either an `SStateGroupStatus` structure (for agent or place groups as objects) or `SStateAgentStatus` structure (for agents or places as objects). The subject request parameter determines the depth of this tree. In the case of a subject DN action, the tree is expanded up to DN actions (`SStateDNAction`); in the case of a subject DN status, the tree is expanded to DN statuses (`SStateDNStatus`); and so on. `SStateAgentStatus`, `SStateDNStatus`, and `SStateDNAction` structure parameters are similarly organized. The `Status` parameter contains the current object’s status; `tmStart` contains the time of switching to this status; `nAgents` (`nDNs` or `nActions`) contains the number of underlying array elements; and pFirstAgent (pFirstDN, pFirstAction) contains either a pointer to the first element of the array, or NULL in case the array is empty or the tree was cut down to this point.

The pData parameter of the SStateDNAction structure can contain either NULL or a valid pointer to an action-specific data structure. For interaction-related actions, pData points to the SDataCall structure, which contains the ConnID and, possibly, the DNIS, ANI, and UserData.

Stat Server retains the last PlaceID in current state when the association between agent and place is broken, such as when an agent logs out.

CurrentStateReasons

Starting with release 6.5, Stat Server provides the CurrentStateReasons category to support the reasons that agents place themselves in certain agent states. This statistical category applies only to stat types that have Agent and/or GroupAgents objects.

In addition to providing current-state information for agents and places, this statistical category also can store reasons for non–interaction-related statuses in TKVList format, if the underlying T-Server supports reasons. For some agent statuses (Ready, NotReady, AfterCallWork) in which DNs have the same such status, Stat Server collects reasons from the Reason field of the corresponding TEvent and/or the Extension field of the TEvent’s ReasonCode key.

Figure 17 illustrates the structures that support this statistical category.

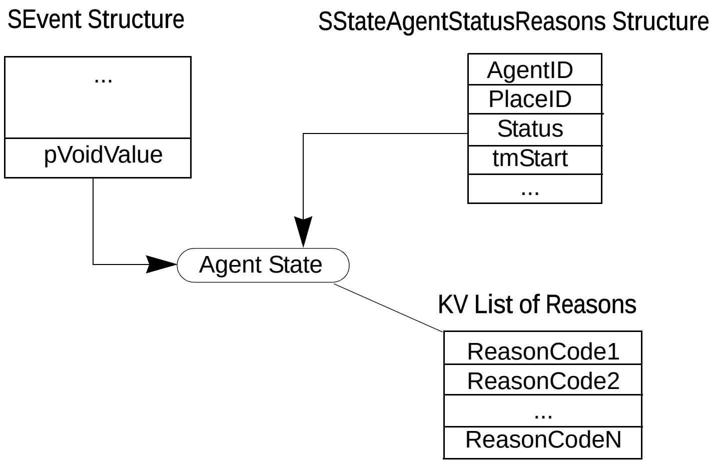


Figure 17: The SStateAgentStatusReasons Structure

Note: Not all T-Servers support reasons. Please refer to the appropriate T-Server manual for more details.

CurrentTargetState

The `CurrentTargetState` statistical category returns one of two structures that include multimedia-capacity information about agent, place, agent group, and place group states:

- `SCurrentTargetStateDelta`
- `SCurrentTargetStateSnapshot`

Stat Server returns the `SCurrentTargetStateSnapshot` structure as its initial response when a client requests a statistic using the `CurrentTargetState` category. Its two fields: `pTargetState` and `nTargets` hold an array of structures each defining the current state of a single target and the number of structures in the array. The term *target* refers to a type of routing object and can be any of the following:

- `ACDQueue`
- `Agent`
- `Agent Group`
- `Destination Label`
- `Place`
- `Place Group`
- `Queue Group`
- `Routing Point`
- `Skill`
- `Variable`

Stat Server sends subsequent notifications using the `SCurrentTargetStateDelta` structure, which contains three fields: `pReqs`, `nReqs`, `pTargetState.` `pReqs` is an array of affected statistical requests that are all based on the `SRequestInfo` structure. Three TEvents determine when the values in these arrays change:

- `SEventCurrentTargetState_TargetAdded`
- `SEventCurrentTargetState_TargetRemoved`
- `SEventCurrentTargetState_TargetUpdated`

The first two apply to agent and place group objects, when members are added to or removed from their groups. When Stat Server receives the `SEventCurrentTargetState_TargetUpdated` TEvent, Stat Server updates the array to note that the target’s state has changed.

Figure 18 illustrates `SCurrentTargetStateSnapshot` and its supporting structures.

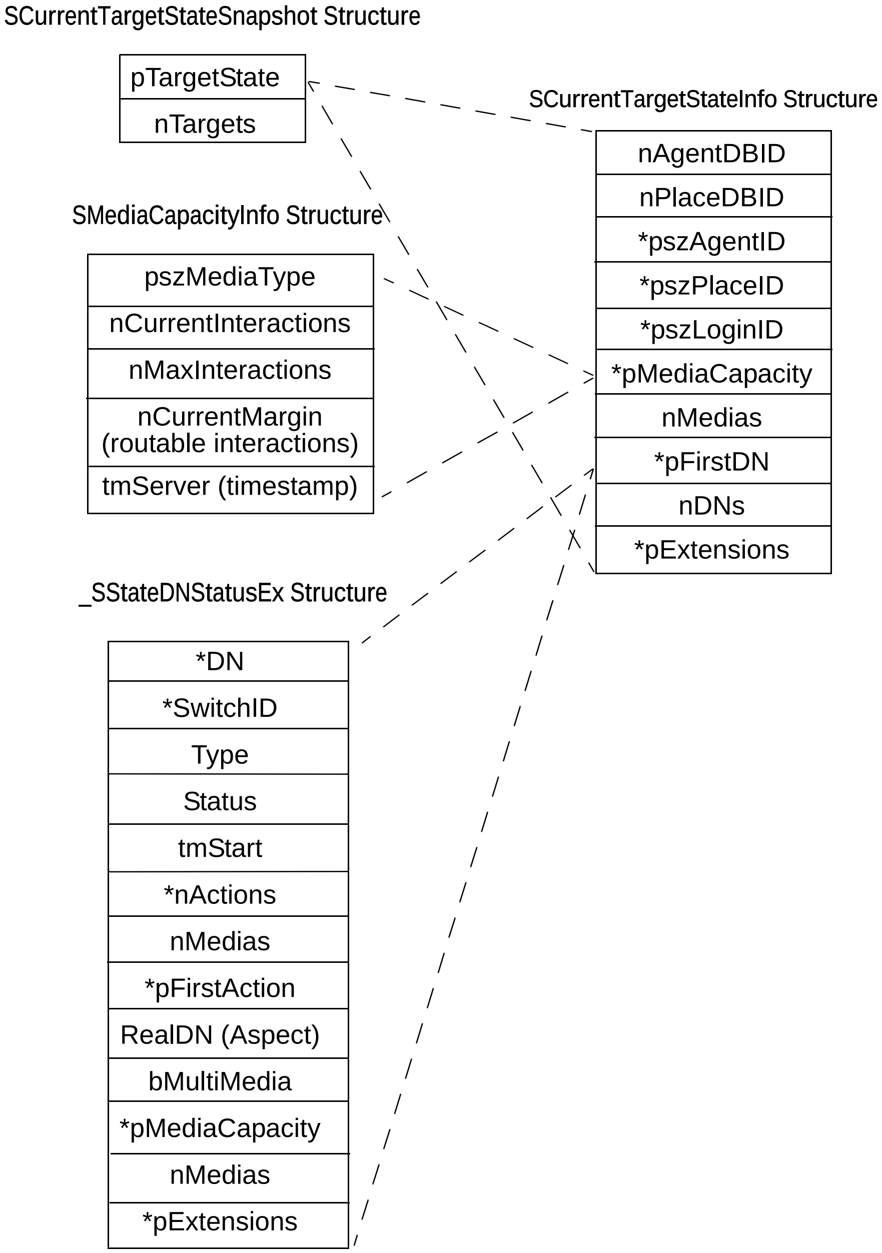


Figure 18: The SCurrentTargetStateSnapshot and Supporting Structures

The method of propagation of agent/place/group state information using this statistical category is somewhat different from that used by the `CurrentState` category. Instead of sending notifications on a statistic-by-statistic basis, Stat Server first determines all of the statistics affected by the change in agent/place/group state and then sends one notification for all of them. In this manner, client decisions—routing strategies, for example—can equally distribute interactions among available resources

Figure 19 illustrates `SCurrentTargetStateDelta` and its supporting structures.

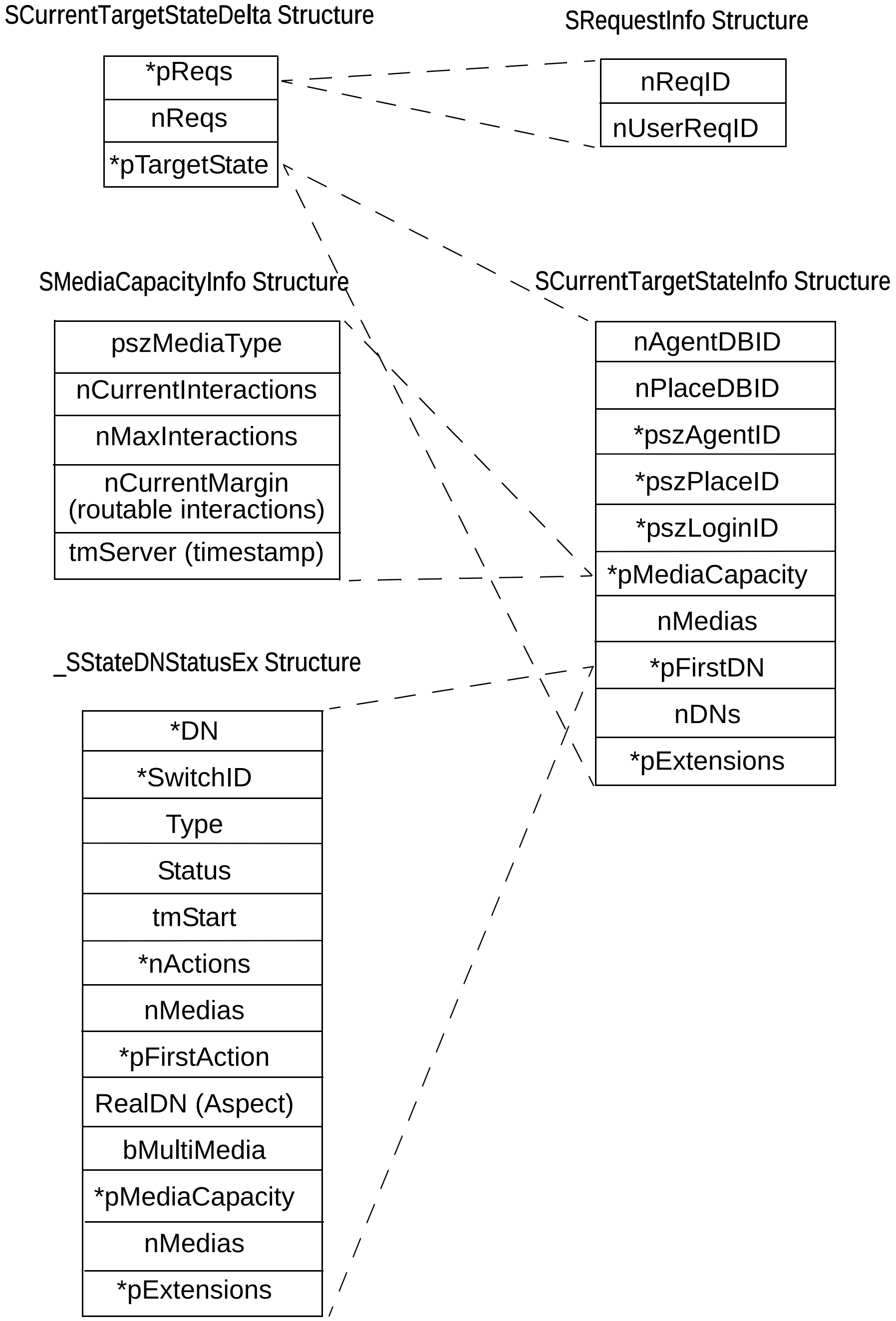


Figure 19: The SCurrentTargetStateDelta and Supporting Structures

Java Category

The `JavaCategory` statistical category must be specified in a stat type's definition to use statistical definitions residing in a Stat Server Java Extensions (SSJE). When loaded, each SSJE passes its own statistical definitions to Stat Server availing them to Stat Server clients. These stat types can be real-time or historical and, unlike regular stat types, are dynamic in nature. This means that they are enabled only if the corresponding SSJE is loaded.

Chapter 6 of the *Reporting Technical Reference Guide* describes stat types provided in the 7.*x* releases of the OCC, Multimedia, and VCB SSJEs.

Chapter

7 Statistical Subjects

The activities that are associated with one contact center interaction can be viewed from many perspectives. For example, when Agent A transfers one inbound call from his extension DN to Agent B, belonging to the same agent group, Stat Server generates:

- Several actions for each agent's DNs including:
  - CallInbound, CallOnHold, CallConsult, CallDialing, and Monitored, for Agent A
  - CallRinging and Monitored, for Agent B
- Several statuses for the Place A, that is associated with Agent A, including:
  - CallInbound for the Extension DN
  - WaitForNextCall on the Position DN
- Several statuses for the agents' group, TierII, including:
  - NotReadyForNextCall, for Agent A
  - WaitForNextCall, for Agent B

To define the perspective from which you want Stat Server to capture data for a statistic, you specify one *statistical subject* in the statistics's underlying stat type definition, by assigning a value to the Subject option. (This option is described on page 40). This chapter introduces the subjects that Stat Server recognizes and explains how they factor into the definition of a statistic.

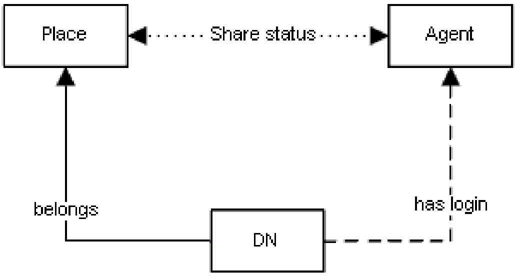
This chapter contains the following sections:

A Recounting of Action Propagation

Stat Server uses an object-centric model to provide statistics for contact center objects. You define and configure the objects that Stat Server monitors within Configuration Server; Stat Server generates actions that report contact center events occurring at these objects. Parent-child relationships can exist between many of these objects where the objects belong to the same compatibility group. Some of these relationships are static—that is, the relationships are defined in configuration. Other parent-child relationships are dynamic, such as the relationship that results when a contact center agent (represented by a `Person` object in Configuration Server) logs in to a DN (represented by a `RegDN` object). The chain of parent-child associations forms a hierarchy, which is illustrated in Figure 8, "Hierarchy of Stat Server Telephony Objects," on page 52.

Stat Server begins by generating all applicable actions for objects that have no children, such as DNs. Stat Server uses these generated actions to compute an object's status. A DN's status, for example, is computed by identifying the highest-priority action occurring at the DN. Stat Server then propagates all of the actions and statuses generated for the child object to its parent objects. Durations are adjusted to reflect the duration of the association between parent and child. To illustrate this adjustment, consider the following:

`Agent` answers a customer call on `DN` and converses with the customer for one hour. Halfway through the call, `Agent` logs in to `Place.` Stat Server generates the `CallInbound` action for `DN` and propagates this action to its parent `Place` and `Person` objects.



However, Stat Server must adjust the duration of the propagated `CallInbound` action for `Agent` to 30 minutes, because `Agent` was associated with neither `Place` nor `DN` during the first half of the call.

For objects that do have children, Stat Server computes status based on the status of the objects' children (adjusted as described above). For additional information on this object-specific algorithm, refer to:

- "Associations Between Agents and Places" on page 52.
- "DN Association with Queues" on page 53.
- "Propagation of DN Actions" on page 66.

The Subject Algorithm

Stat Server can tally statistics with varying levels of granularity for a given object. To narrow the subject of interest, you use only one of the following to define the `Subject` statistic option within a stat type definition:

- `AgentStatus`
- `PlaceStatus`
- `GroupStatus`
- `DNAction`
- `DNStatus`
- `Action`
- `CampaignAction`

The first part of each compound value indicates the data source where Stat Server tracks information: DN, agent, place, group, or campaign. The second part of the compound value indicates whether Stat Server should consider the actions occurring at the data source or the statuses. For example, a `DNAction` subject assignment within a stat type definition informs Stat Server that the actions generated for a regular directory number are the source of statistics for all applicable objects. The `AgentStatus` subject informs Stat Server that the statuses of agents are the source of statistics for all applicable objects. Table 26 maps the objects to which each statistical subject applies. This mapping also indirectly reveals the groups of objects which are compatible.

Table 26: Statistical Subjects and the Objects to Which They Apply

Subject	Applicable to the Following Objects	Description
Action	RegDN Agent Place Tenant GroupAgents GroupPlaces	Analogous to the `DNAction` subject for telephony interactions, specify this value for multimedia statistics.
AgentStatus	Agent Place Tenant GroupAgents GroupPlaces	Indicates that Stat Server should consider only the statuses occurring at agent data sources.
CampaignAction	CampaignGroup CallingList CampaignCallingList	Indicates that Stat Server should consider only the actions occurring at campaign data sources.
DNAction	RegDN Agent Place Queue Tenant RoutePoint GroupAgents GroupPlaces GroupQueues	Indicates that Stat Server should consider only the actions occurring at regular directory number data sources.
DNStatus	RegDN Agent Place Tenant GroupAgents GroupPlaces	Indicates that Stat Server should consider only the statuses occurring at regular directory number data sources.

Table 26: Statistical Subjects and the Objects to Which They Apply (Continued)

Subject	Applicable to the Following Objects	Description
GroupStatus	GroupAgents GroupPlaces Tenant	Indicates that Stat Server should consider only the statuses occurring at place group or agent group data sources.
PlaceStatus	Agent Place Tenant GroupAgents GroupPlaces	Indicates that Stat Server should consider only the statuses occurring at place data sources.

Chapter

8 Stat Server Timestamps

Stat Server internally maintains timestamps for multiple purposes:

- To indicate when Stat Server received event notifications from T-Server, Interaction Server, Outbound Contact Server (OCS)*, or Configuration Server.
- To indicate when T-Server, Interaction Server, OCS, or Configuration Server sent event notifications to Stat Server.
- To track the durations of actions.
- To determine whether and when to clear stuck calls.
- To profile a reporting interval.
- To define time ranges over which actions should be measured.

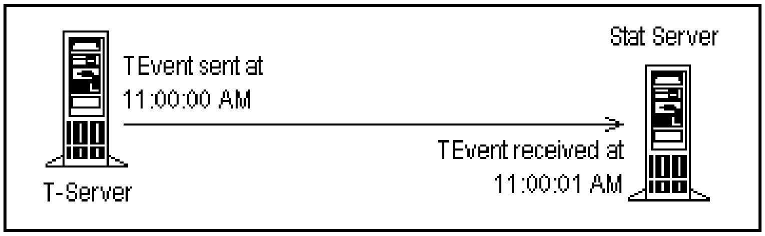
This chapter focuses on the aspects of time that impact the statistical values that Stat Server reports to its clients as they relate to the generation of actions and the measurement of time-sensitive statistics. This discussion can help you decide which value to set for the `UseSourceTimeStamps` option of a stat type's definition, described on page 41.

This chapter contains the following sections:

* There is no direct connection between Stat Server and Outbound Contact Server—T-Server transmits OCS events to Stat Server through communication DNs.

Timestamps, Prior to Release 8.0

Prior to Stat Server release 8.0, Stat Server relied on UTC time measured locally to timestamp events that were received from data sources (such as T-Server) and to update time-sensitive statistics. In practice, this local time could deviate significantly from the data source's UTC time which reflected the moment of event transmission.



Network latencies and load-related delays explain the gap between these two timestamps and largely account for the discrepancies in statistical values reported by other Genesys Reporting applications for comparable statistics.

The setting of the `UseSourceTimeStamps` stat type configuration option (described on page 41) enables you to control which timestamp Stat Server uses; however, it should be noted that this aspect of time alone does not cause all of the differences in reported statistical values. Stat Server uses a mix of local and source timestamps to compute the duration of a terminating action for any statistic.

For example, Stat Server adjusts all actions occurring for a statistic to the local time in which the statistic was opened. But when the actions terminate, their timestamps and, therefore their durations, are measured in:

- Local time, when `UseSourceTimeStamps` has been set to `No`.
- Both local and source time, when `UseSourceTimeStamps` has been set to `Yes`. Stat Server adjusts action duration in this scenario as follows:

  ```
  action_end—max(action_start,statistic_creation_local_timestamp)
  ```

  where `action_end` is measured in source time for the given statistic.

The Pool of Comparable Statistics

Stat Server statistics that are comparable to Interaction Concentrator (ICON) statistics have the following qualities:

- They are purely historical.
- They are time-sensitive. (Counter-based statistics typically do not depend on which timestamp Stat Server uses.)
- Their time profiles are based on the `Selection` or `Growing` interval types (see page 20).

- Their stat type profiles define the following options:
  - `Subject=DNAction`, `CampaignAction`, or `Action`
  - `Category`—equal to one of the following:
    - `AverageNumberPerRelativeHour`
    - `AverageTime`
    - `ElapsedTimePercentage`
    - `EstimWaitingTime`
    - `LoadBalance`
    - `MaxTime`
    - `MinTime`
    - `RelativeTimePercentage`
    - `ServiceFactor1`
    - `TotalTime` (with or without `DCID`)
    - `TotalTimeInTimeRange`
  - `MainMask`—containing only durable and retrospective actions, including instant actions, and carrying associated durations. (`RelMask`, in case of ratios and `Selection`-based statistics, may contain any actions.)

In addition—and this applies to all Genesys products—all of the servers in your environment must be synchronized to hold an accurate GMT setting. This is especially critical to the production of consistent results when Stat Server must monitor several DNs in order to determine when to generate multi-server actions, such as `CallAnswered`.

Source Timestamp Algorithm

In release 8.0, Stat Server maintains the following two new attributes internally for all actions and object relationships:

- Source start—derived from the `Time` attribute of TEvents, the `event_time` attribute of Interaction Server events, and the `cfgTimestamp` attribute of Configuration Server events.
- Source end—derived in the same manner as `Source start`.

Stat Server attributes these to each action regardless of the setting of the `UseSourceTimeStamps` option. When `UseSourceTimeStamps` is set to `true`, Stat Server references the values of these attributes to calculate statistic duration.

Note: The values that Stat Server writes to the Stat Server database do not reflect source timestamps.

Figure 20 illustrates how Stat Server computes the duration of a `CallInbound` action on an agent group when `UseSourceTimeStamps=true`.

- Action source start = maximum value among:
  - T-Server timestamp of agent’s login
  - Configuration Server timestamp when the agent was added to the group.
  - T-Server timestamp associated with `EventEstablished` for the call.

- Action source end = minimum value among:
  - T-Server timestamp of agent logout
  - Configuration Server timestamp when the agent was removed from the group.
  - T-Server timestamp associated with `EventReleased` for the call.

Note that Stat Server references the timestamps from at least two sources to determine duration.

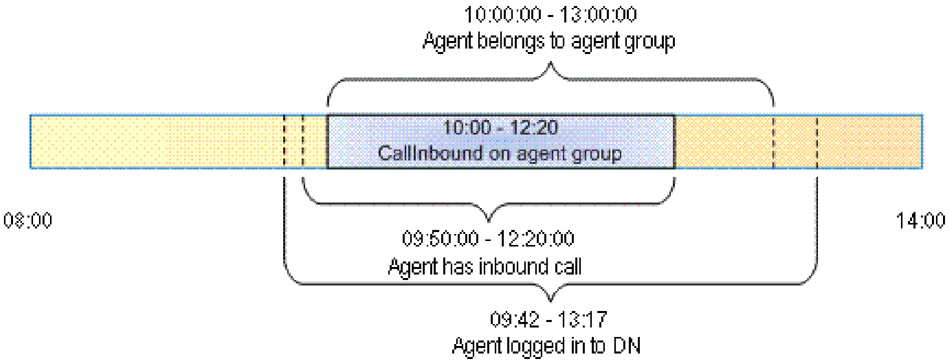


Figure 20: How Stat Server Computes CallInbound Example

The following special circumstances require Stat Server to reference both the source and local timestamps for action generation when `UseSourceTimeStamps` is set to `true`:

- When Stat Server generates actions while reading initial configuration. Stat Server reads configuration data in synchronous mode, so Stat Server sets `Source start` to the value of the local timestamp.
- When actions terminate while the connection to the event-supplying server is lost. Corresponding events do not contain the source timestamp, so Stat Server references its local timestamp to compute action duration.

  For example, T-Library detects that the connection with T-Server is lost and generates `EventServerDisconnected`.
- When statistics use internal timestamps. These internal timestamps might be set to local timestamps as in the case of initial value calculations and resets. For an example of how Stat Server references both local and source time, see page 170.

UseSourceTimeStamps and System Clock Changes

Local system clock changes do not affect the time reported by event-supplying servers, except when these servers are located on the same computer as Stat Server. (Such a deployment, incidentally, is not recommended.) However, if the system time changes on the data source computer, for Daylight Saving

Time, for example, this server may start sending events with timestamps that are earlier than the timestamps of previously transmitted events. Stat Server manages this scenario by holding in memory the timestamp of the last received event, for each data source. If the timestamp of a newly received event is less than the timestamp of the last received timestamp, Stat Server uses the timestamp of the last received event to ensure monotonicity of source time for each data source. If the system time on the data source computer is adjusted forward to a later time, Stat Server applies no adjustments to the new time.

Chapter

9 Campaign Statistics

This chapter introduces statistics that can be calculated for an outbound campaign. Information in this chapter is divided among the following topics:

Overview

Stat Server calculates campaign-related statistics based on the events received from Outbound Contact Server (OCS). Unlike T-Server, Stat Server does not connect directly to OCS to receive these statistics. Instead, OCS sends them in a specific event format to a mediator called a Communication DN, where Stat Server then reads them. During configuration of Outbound Contact Solution, the Communication DN is used exclusively for this purpose. Stat Server needs no additional configuration information to receive campaign-related statistics.

Campaign Objects

Campaign statistics are calculated exclusively for the Outbound Contact Solution to reflect campaign performance. Consult the Outbound Contact Solution 8.0 documentation for information about campaigns. Stat Server provides statistics on groups of agents or groups of places participating in one or more campaigns concurrently, and on one or more calling lists used to run a campaign.

Stat Server bases statistics for a campaign on the following campaign objects:

- `Campaign`

- CampaignGroup
- CallingList
- CampaignCallingList

Stat Server's Campaign and CallingList objects correspond to Configuration Server's Campaign and CallingList objects. Accordingly, these objects must have the same names as Configuration Layer's campaign objects. Campaign Group and CampaignCallingList objects are configured within and are meaningful only to Stat Server. A CampaignGroup object is based on a GroupAgents object that has been assigned to a specific campaign. A Campaign CallingList object is based on a CallingList object that has been assigned to a specific campaign.

CampaignGroup and CampaignCallingList objects must be named *campaign@groupname* and *campaign@callinglist*, respectively, where *groupname* (*callinglist*) is the name of a specific group of agents (CallingList) that has been assigned to the campaign. These Stat Server objects are visible within CCPulse+.

Figure 21 shows the hierarchy of the Stat Server campaign objects (see the Stat Server Telephony Objects schema in Figure 8 on page 52, for lower levels of the hierarchy).

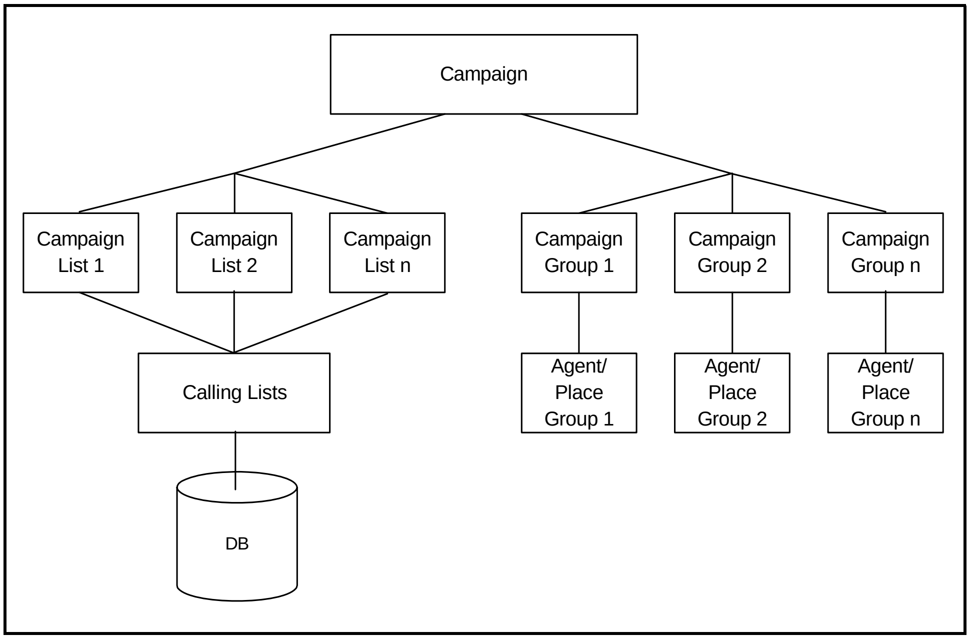


Figure 21: Hierarchy of Stat Server Campaign Objects

Campaign objects and campaign actions are further described in this chapter.

Campaign Statistical Types

Stat Server provides two types of campaign-related statistics: telephony and campaign. The Subject, as defined in a stat type, differentiates the two types.

- *Telephony statistics* are calculated for `Campaign` and `CampaignGroup` objects. They are based on events that are received from T-Server and that concern telephony objects in a contact center. The subjects of these statistics, therefore, are objects other than `Campaign` objects.
- *Campaign statistics* are based on events received from Campaign Server, and their subjects are `Campaign` objects. These types of statistics can be divided into two groups:
    - *General conditions statistics* reflect conditions of the whole campaign and are calculated for `Campaign` and `CampaignGroup` objects.
    - *Operational actions statistics* reflect details of the campaign's progress for all campaign-related objects.

Note: Stat Server ignores filters that are applied to statistics having a `Campaign` subject.

Campaign Actions and Statuses

Campaign Actions

`Campaign` actions are characterized by these durable actions:

- `StatusActivated`
- `StatusRunning`
- `StatusDeactivated`

`StatusActivated` occurs when at least one `CampaignGroup` has `Status Activated`, but none has `StatusRunning`. `StatusRunning` occurs when at least one `CampaignGroup` has `StatusRunning`. `StatusDeactivated` occurs when all campaign groups have `StatusDeactivated`.

CampaignGroup Statuses and Actions

In a specific campaign, a `CampaignGroup` has three statuses: `StatusActivated`, `StatusRunning`, and `StatusDeactivated`.

- `StatusActivated` starts when either a campaign is being loaded on a group or the dialing process stops. `StatusActivated` ends when either the dialing process starts or a campaign is being unloaded.
- `StatusRunning` starts when the dialing process starts, and ends when either the dialing process stops or a campaign is being unloaded.

- `StatusDeactivated` starts when a campaign is being unloaded, and ends when a campaign is being loaded on a group.

Changing these statuses from one to another causes a durable action (`StatusActivated`, `StatusRunning`, or `StatusDeactivated`) to occur.

The `StatusRunning` durable action can be accompanied by the `StatusWaitingRecords`, `StatusWaitingPorts`, `StatusWaitingAgents`, and `StatusSystemError` durable actions.

In parallel with the `StatusRunning` action, one of these dial modes can occur:

- `NoDial`
- `Predict`
- `PredictAndSeize`
- `PredictGVP`
- `Preview`
- `Power`
- `PowerAndSeize`
- `PowerGVP`
- `Progress`
- `ProgressAndSeize`
- `ProgressGVP`
- `PushPreview`

Campaign Operational Actions

Campaign operational actions are calculated for all campaign objects:

- `LeadProcessed` starts when a number of records from calling lists (counting records from the same chain as one) are processed to the point where no further actions are to be taken.
- `CallbackScheduled`.
- `CallbackCompleted`.
- `CallbackMissed`.
- `PersonalCallbackScheduled`.
- `PersonalCallbackCompleted`.
- `PersonalCallbackMissed`.
- `AgentError`.
- `DialAnswer` starts when dialing has been answered.
- `DialMade` starts when dialing is completed—whether successful (`DialAnswer`) or not. When dialing is unsuccessful for any reason, Stat Server starts one of the following actions:
  - `DialAbandoned`
  - `DialAgentCallBackError`
  - `DialAllTrunksBusy`
  - `DialAnswMachine`
  - `DialBusy`
  - `DialCallDropError`
  - `DialCancel`
  - `DialDoNotCall`
  - `DialDropped`
  - `DialDroppedNoAnswer`
  - `DialError`
  - `DialFaxDetected`
  - `DialGeneralError`
  - `DialGroupCallBackError`
  - `DialNoAnswer`
  - `DialNoDialTone`
  - `DialNoEstablished`
  - `DialNoFreePortError`
  - `DialNoPortAvailable`
  - `DialNoProgress`
  - `DialNoRingBack`
  - `DialNUTone`
  - `DialPagerDetected`
  - `DialSilence`
  - `DialSITDetected`
  - `DialSITInvalidNum`
  - `DialSITNoCircuit`
  - `DialSITOperIntercept`
  - `DialSITReorder`
  - `DialSITUnknown`
  - `DialSITVacant`
  - `DialStale`

  - `DialSwitchError`
  - `DialSystemError`
  - `DialTransferError`
  - `DialUnknown`
  - `DialWrongNumber`
  - `DialWrongParty`
- `RecordsNotProcessed`
  Stat Server generates this action for campaign or calling list objects when campaign (or calling list) processing is completed.

> **Note:** Neither the `CampaignCallingList` nor the `CampaignGroup` object type applies to the `RecordsNotProcessed` action.

- `RecordsScheduled`—Stat Server generates this action for campaign objects when Stat Server receives notification that campaign records have been scheduled for processing. Stat Server generates this action only for `CurrentNumber` statistics.
- A *lead* is a set of records from the calling lists related to a specific customer in the Configuration Manager. Stat Server starts a lead action when a number of records from calling lists are processed to the point where no further actions will be taken for the particular lead. Lead actions are calculated as the number of leads processed for every call result. Below is a listing of some lead actions:
  - `LeadAbandoned`
  - `LeadAgentCallBackError`
  - `LeadAllTrunksBusy`
  - `LeadAnswer`
  - `LeadAnswMachine`
  - `LeadBusy`
  - `LeadCallDropError`
  - `LeadCancel`
  - `LeadDoNotCall`
  - `LeadDropped`
  - `LeadDroppedNoAnswer`
  - `LeadError`
  - `LeadFaxDetected`
  - `LeadGeneralError`
  - `LeadGroupCallBackError`
  - `LeadNoAnswer`
  - `LeadNoDialTone`
  - `LeadNoEstablished`
  - `LeadNoFreePortError`
  - `LeadNoProgress`
  - `LeadNoRingBack`
  - `LeadNuTone`
  - `LeadOk`
  - `LeadPagerDetected`
  - `LeadSilence`
  - `LeadSITDetected`
  - `LeadSITInvalidNum`
  - `LeadSITNoCircuit`
  - `LeadSITOperIntercept`
  - `LeadSITReorder`
  - `LeadSITUnknown`
  - `LeadSITVacant`
  - `LeadStale`
  - `LeadSwitchError`
  - `LeadSystemError`
  - `LeadTransferError`
  - `LeadUnknown`
  - `LeadWrongNumber`
  - `LeadWrongParty`

Campaign-Related Statistical Category

In addition to the statistical categories described in Chapter 6, Stat Server supports a statistical category calculated exclusively for the Outbound Contact Solution to reflect an estimated finish time for a particular campaign.

The `EstimTimeToComplete` statistical category (and the statistic with the same name) is based on campaign data propagated from Campaign Manager and is calculated as follows:

1. Stat Server calculates the speed of changes in ready records (that is, records not processed by Campaign Manager). Stat Server measures the difference between two campaign events, which contain a different number of ready records for the same campaign and time between them. From this data, Stat Server calculates the number of records per second (actually, the processing speed).
2. Stat Server divides the number of ready records by the processing speed to yield the number of seconds until this number will become zero (0) (which means that the campaign ends or the calling list has been processed).

`EstimTimeToComplete` is applicable only for `Campaign` and `CallingList` object types.

Chapter

10 Custom Formulas

This chapter defines custom formulas and explains how custom-value statistics are calculated. Information in this chapter is divided between the following topics:

Purpose

You use *custom formulas* to compute user-specific quantities (usually business-related) based on attached data communicated by TEvents.

In a *custom-value statistic*, the values obtained by evaluating custom formulas on individual actions are aggregated much as action durations in time-related values are aggregated.

Before using custom-value statistics, Genesys strongly recommends that you read the following description of the evaluation procedure.

Evaluation

The basic custom-value functions are evaluated on the *relevant key-value list* of an action. For different types of actions, the relevant key-value list is computed differently. Therefore, make sure you thoroughly understand the computation procedure before creating custom-value statistical types.

String values in the relevant key-value list are converted to numbers in the ordinary way if they are in this format:

- Integer: a sequence of digits possibly preceded by the symbol + or –
- Fixed-point decimal: an integer followed by a dot (.), possibly followed by a sequence of digits
- Floating-point decimal: an integer or fixed-point decimal followed by the letter e or E, possibly followed by a sign, followed by one or two digits

A string that is not in any of these formats is converted to `0`.

Evaluation on Momentary Actions

For a momentary action, the relevant key-value list equals the `UserData` list, which is received with the TEvent that caused the action. The same rule holds for the mediation DN action `CallTreatmentCompleted`, which is not derived from a durable action, but is formally classified as a retrospective action (see page 103).

Evaluation on Durable Actions

For a durable action, Stat Server can keep two key-value lists: one relevant to data that is attached at a specific DN during a given interaction (called a *local key-value list*) and the other one relevant to data that is attached at all DNs during the interaction (called a *global key-value list*).

The global key-value list equals the `UserData` list that Stat Server received with the T-Event that triggered the action.

The computation procedure for a local key-value list for a durable action:

1. Separates the specific attached data belonging to the DN in which the action occurs from all the data that is attached to the interaction.

2. Attaches the data while the action occurs.

Calculating a Local Key-Value List

Stat Server calculates a local key-value list from the global key-value list. The local key-value list is recalculated whenever:

- The action starts.
- The `EventAttachedDataChanged` TEvent is received while the action goes on (might be repeated any number of times).
- The action ends.

This section describes how the relevant key-value list is updated during an inductive procedure.

These terms are used:

- A *key-value list* is a finite sequence of ordered pairs of character strings—the first element of a pair is called a *key*, and the second element a *value*.
- A *marked key-value list* is a finite sequence of ordered triples, whose first two elements are strings (*key* and *value*), and whose third element is a *flag* with either a *native* or a *foreign* value.

This notation is used:

- Steps are indexed from `1` to `N`.

- List_k is the `UserData` key-value list received at Step k.
- Δ_k is the `k`-th step value of a marked key-value list defined inductively (referred to as the marked prototype of the relevant key-value list)

These operations are used:

- **List Subtraction.** Let `ListA` and `ListB` be key-value lists. Then `ListA\List B` is defined as the key-value list obtained from `ListA` by removing from it:
  - The first `k` occurrences of any key-value pair that occurs `k` times in `ListB` and more than `k` times in `ListA`.
  - All occurrences of any key-value pair that occurs in `ListA` fewer times than in `ListB`.
- **Marking a List.** Let `ListA` be a key-value list. Then, `Native(ListA)` is the marked key-value list obtained from `ListA` by appending `native` to every pair in the list. `Foreign(ListA)` is the marked key-value list obtained from `ListA` by appending `foreign` to every pair in the list.
- **Marked List Union.** Let `ListA` and `ListB` be marked key-value lists. Then `ListA/ListB` is the marked key-value list obtained by concatenating `ListA` and the list obtained from `ListB` by removing from it:
  - The first `k` occurrences of elements with the same key-value pair occurring `k` times in `ListA` and more than `k` times in `ListB`, regardless of the flags.
  - All occurrences of elements with the same key-value pair occurring in `ListB` fewer times than in `ListA`.

Here is the inductive definition of the marked prototype Δ_k :

1. $\Delta_1 = \varnothing$; that is, the marked prototype contains no elements at Step 1.
2. If Step `k` is caused by `EventAttachedDataChanged` with `ThisDN` equal to `ThirdPartyDN` or at the final step of the action,
   $\Delta_k = \Delta_{k-1} / \text{Native}(\text{List}_k \backslash \text{List}_{k-1})$
3. If Step `k` is caused by `EventAttachedDataChanged` with `ThisDN` different from `ThirdPartyDN`,
   $\Delta_k = \Delta_{k-1} / \text{Foreign}(\text{List}_k \backslash \text{List}_{k-1})$

When a custom formula is evaluated on a durable action for use in a current aggregated value, the relevant key-value list is obtained from Δ_k for the last completed step by removing all pairs flagged by `foreign`, and removing the flag from the remaining pairs.

When a custom formula is evaluated on a durable action for use in a historical aggregated value, the relevant key-value list is obtained from Δ_k for the final step of the action by removing all pairs flagged by `foreign`, and removing the flag from the remaining pairs.

`Note:` This mechanism is best suited for processing attached data if, once a key-value pair is attached, it never gets removed.

Special Note

For group actions reflecting an origination DN, custom-formula evaluation is identical to the evaluation of the custom formula for the corresponding mediation DN action.

Evaluation on Retrospective Actions

As a rule, the value of a custom formula for a retrospective action is the same as the final value of the custom formula for the durable action from which the retrospective action is derived.

Note these exceptions:

- Custom formulas are evaluated for the mediation DN action `CallTreatmentCompleted` in the same way as for instantaneous actions, because this retrospective action is not derived from a durable action.
- The retrospective mediation DN actions `CallAnswered`, `CallAbandonedFromRinging`, `CallReleased`, `CallMissed`, `ACWCompleted`, and `ACWMissed` receive the same value as when evaluated on the corresponding regular DN actions.

Chapter

11 Virtual Agent Groups

This chapter introduces the concepts of *virtual agent groups* and explains how to configure them:

Supported Virtual Agent Group Definitions

Stat Server can provide statistics for a virtual group of agents. A group of agents is considered to be *virtual* if agents do not permanently belong to the group. Instead, Stat Server assigns an agent to the group when an agent meets the criteria specified by the virtual group's definition. Stat Server adds agents to, or removes them from, the group if agent parameters that affect eligibility change or if the specified criteria are modified.

Because the agents are in the group "virtually," they do not appear in the `Group Properties` dialog box in the Configuration Manager. However, you can view the members of the virtual group using CCPulse+, and Stat Server provides the same statistics for this virtual agent group as for a regular agent group.

Use logical expressions to define criteria for a virtual agent group. You can use a parameter defined for an agent in a function in the virtual group definition. As a function with a specific return value, a parameter can be compared with an integer constant or another function. The result of an elementary comparison can be used in a complex logical expression (&, |, ~).

Stat Server currently supports virtual group functionality with three types of agent parameters:

- A skill configured for an agent
- An ACD queue to which an agent is logged in
- A switch into which an agent is logged in

You can simultaneously specify these types of parameters in an expression for a single virtual group.

Note: If you remove the virtual agent group expression from the group's Properties dialog box, the group immediately becomes a regular agent group. Stat Server starts treating the group as a regular agent group and takes into account all Person configuration objects associated with this group in Configuration Manager.

Agent Skill Functions

Configure a Skill object for an agent on the Agent Info tab of the Person Properties dialog box in Configuration Manager. You can use the Skill level as a value of the *Skill* function in the Virtual Group definition. For example, Skill can be "Spanish" with Level 8; this returns an integer value of 8 for a Spanish skill function.

Note: When you fail to define a skill level for an agent, the Skill expression returns the Unknown value.

When Stat Server reads configuration data from Configuration Server, it identifies the agents with the skills and levels of skills that satisfy the expression specified in the Virtual Agent Group Properties dialog box. Stat Server treats these agents, if they belong to the same Tenant object, as belonging to the virtual group. Stat Server updates the Group object, whenever you modify the agent skill or the logical expression.

ACD Queue Functions

Stat Server receives a notification from T-Server that an agent has logged in and identifies to which ACD Queue the agent logged in to. The ACD Queue number could be used as a value of the LoggedIn function in the Virtual Group definition. For example, an agent can log into an ACD Queue whose number is 5253; this returns a true value for this agent if ACD Queue number 5253 is defined in the LoggedIn function for a Virtual Queue.

Keep in mind that:

- Because DN numbers are not unique in a configuration with multiple switches, the ACD queue number must be accompanied by the switch name to make an expression unique.
- When an ACD queue number is unknown for an agent, the LoggedIn expression returns a false value.

When Stat Server receives an agent login notification, it determines whether the agent satisfies the LoggedIn expression specified in the Virtual Agent Group Properties dialog box. Stat Server treats the agents that logged in to the specified queue at the specified switch as belonging to the virtual group. Stat Server updates the Group object as soon as the agent logs out or the logical expression is modified.

Switch Functions

If an agent belonging to a virtual agent group has logged in to a particular switch, Stat Server returns a `true` value to clients that request the agent's `LoggedIn` status on that switch. Agent login to a particular queue on that switch is unnecessary.

Configuring Virtual Agent Groups

From Configuration Manager:

1. Under `Resources` in the main window, select `Agent Groups`.
2. Select `File` > `New Agent Group`.
3. On the `General` tab of the `Agent Group Properties` dialog box, specify the name of the group.
4. On the `Annex` tab, create a section named `virtual`.
5. Within the section, create a new option named `script`.

   An option value must contain the logical expression that defines one or more of the following:

   - Skills and skill levels valid for this group, in the format:

     `Skill("SkillName")=SkillLevel`

     where `SkillName` is the actual name for a `Skill` configuration object; `SkillLevel` is an integer; and one of these operators defines the relationship between `SkillName` and `SkillLevel`: =, !=, >=, <=, >, <.
   - Skills valid for this group, in the format:

     `SkillExists("SkillName")`

     where `SkillName` is the actual name for a `Skill` configuration object.
   - ACD queue numbers and switches valid for this group, in the format:

     `LoggedIn("QueueNumber@SwitchName")`

     where `QueueNumber` is the directory number of an ACD queue and `SwitchName` is the name of the `Switch` configuration object to which this ACD queue belongs. No operators are required within this expression.
   - Switch names, in the format:

     `LoggedIn("SwitchName")`

     where `SwitchName` is the name of a `Switch` configuration object.

   Syntax elements, such as quotation marks and parentheses, are vital for criteria validity.

   Stat Server first tries to validate the `LoggedIn` parameter against the name of switch objects in Configuration Server. If the switch name is in the *`queue@switch`* format (for example, *A@B*), Stat Server will not be able to report logged in status for queue A on switch B under the following conditions:

- Switch object B exists in the configuration.
- Switch object A@B exists in the configuration.
- Queue object A exists in the configuration, and it is defined on switch B.

To avoid this scenario, Genesys recommends that you not use the “@” symbol in the name of your switches.

Note: You can define any number of logical expressions of either type as a value for the same option, as long as these expressions are correctly joined by logical operators & (logical AND), | (logical OR), ~ (logical NOT), and (…) parentheses for changing logical operators’ priorities.

Warning! Do not manually add agents to a virtual agent group.

Examples

If the virtual agent group is meant for agents whose Spanish skill is higher than 5 and whose French skill is higher than 8, the value of the Skill option is:

```
Skill(“Spanish”)>5 & Skill(“French”)>8
```

If the group is meant for agents logged in ACD queue 5253 at the switch named DEFINITY, the option value is:

```
LoggedIn(“5253@DEFINITY”)
```

If the group is meant for agents logged in at the switch named DEFINITY, the option value is:

```
LoggedIn(“DEFINITY”)
```

If the group is meant for agents whose Spanish skill is higher than 5 and who are logged in ACD queue 5253 at the switch named DEFINITY, the option value looks like this:

```
Skill(“Spanish”)> 5 & LoggedIn(“5253@DEFINITY”)
```

Backward Compatibility

The Stat Server 7.*x* and 8.*x* releases are backward compatible with releases 6.1 and 6.5; in particular, you can achieve the same results with regard to Virtual Agent Group functionality.

In release 5.1, however, a virtual group defined by the expression Skill(“SkillName”) < 5 includes both agents with the level of the skill “SkillName” less than 5 and agents without this skill. In releases 6.*x* and 7.*x*, the same virtual group includes only agents with the level of the skill "SkillName" less than 5. To configure a virtual agent group equivalent to that in release 5.1, define the following expression in release 7.*x:*

```
~SkillExists(“SkillName”) | Skill(“SkillName”)<5
```

Appendix

Predefined Statistical Types

Statistical type definitions that are used by the various Genesys applications are either created during the deployment of those applications or are internal to the applications' functionality. This chapter references the predefined stat types that are deployed with Solution Reporting and with the creation of a new Stat Server `Application` object from the Stat Server 8.0 template. It contains the following sections:

Solution Reporting Stat Types

For a current listing, definition, and description of the stat types that are used by Solution Reporting (CCPulse+ and CC Analyzer), refer to the *Reporting Technical Reference Guide.*

Stat Type Definitions in the Stat Server Application Template

The following stat type definitions are preconfigured in the Stat Server 8.0 application template. To read their configurations, open the sections in Configuration Manager. (Or, run a configuration-file export to see all definitions in an ASCII file.) All of these are core stat types—they do not derive their values from a Java extension.

```
[AbandCallsPercentage]
[AverAbandCallTime]
[AverConsultDNStatusTime]
[AverConsultPlaceStatusTime]
[AverConsultStatusTime]
[AverDistribCallTime]
[AverHandleDNStatusTime]
[AverHandlePlaceStatusTime]
[AverHandleStatusTime]
[AverInboundDNStatusTime]
[AverInboundPlaceStatusTime]
[AverInboundStatusTime]
[AverOutboundDNStatusTime]
[AverOutboundPlaceStatusTime]
[AverOutboundStatusTime]
[CurrentAgentState]
[CurrentDNState]
[CurrentGroupState]
[CurrentPlaceState]
[CurrMaxCallWaitingTime]
[CurrNumberACWStatuses]
[CurrNumberConsultStatuses]
[CurrNumberDialingStatuses]
[CurrNumberHoldStatuses]
[CurrNumberInboundStatuses]
[CurrNumberInternalStatuses]
[CurrNumberNotReadyStatuses]
[CurrNumberOutboundStatuses]
[CurrNumberRingingStatuses]
[CurrNumberWaitingCalls]
[CurrNumberWaitStatuses]
[DistribCallsPercentage]
[ServiceFactor]
[TotalAfterCallWorkDNStatusTime]
[TotalAfterCallWorkPlaceStatusTime]
[TotalAfterCallWorkStatusTime]
[TotalLoginTime]
[TotalNotReadyDNStatusTime]
[TotalNotReadyPlaceStatusTime]
[TotalNotReadyStatusTime]
[TotalNumberCallsAband]
[TotalNumberCallsDistrib]
[TotalNumberConsultCalls]
[TotalNumberInboundCalls]
[TotalNumberInternalCalls]
[TotalNumberOutboundCalls]
[TotalTalkDNStatusTime]
[TotalTalkPlaceStatusTime]
[TotalTalkStatusTime]
```

Creating Stat Type Definitions

You define statistical types in Configuration Manager as sections on the `Options` tab within the Stat Server `Application` object. The name of the stat type is the name you assign to the section. Configure core stat types using the following format:

```
[NameOfCoreStatType]
Objects =  One or more objects separated by commas
Category =One and only one statistical category
Subject =  One and only one subject
MainMask =* and/or one or more actions separated by commas
          and optionally preceded by ~ (for NOT)
          [not applicable if Category=JavaCategory]
RelMask =  [optional, applicable if a MainMask is specified]
          * and/or one or more actions separated by commas
          and optionally preceded by ~ (for NOT)
          [not applicable if Category=JavaCategory]
MediaType = media type
Description = [optional] free-form text
Formula =  DCID, <expression>, mandatory for
               CustomValue family statistical categories.
<business attribute name> = <business attribute value>
```

```
UseSourceTimeStamps = yes or no
```

And Java stat types follow this format:

```
[NameOfJavaStatType]
Objects =  One or more objects separated by commas
Category = One and only one statistical category
Subject =One and only one subject
JavaSubCategory = relative path (with respect to the value of
   java-extensions-dir Stat Server configuration option) to the
   .jar file of the loaded Stat Server Java Extension (SSJE) and
   name of the statistical type within that Extension in the
   format <relative path>:<statistical type name>.
AggregationType = [optional]
   One and only one aggregation type, applicable only if a SSJE
   is loaded. Currently used only by Data Sourcer clients.
MediaType = media type
Description = [optional] free-form text
Formula = <expression>, mandatory for CustomValue family statistical
   categories.
<business attribute name> = <business attribute value>
```

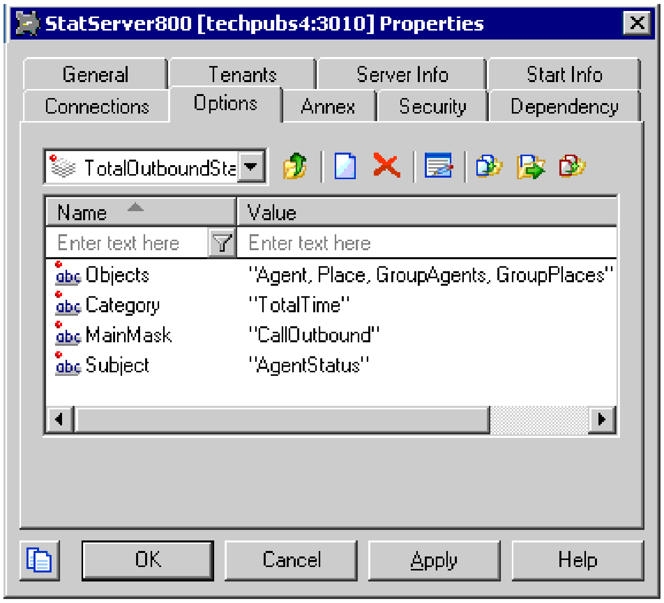
For more detailed descriptions of these configuration options, see the “Statistical Type Sections” on page 37.

Examples

The following examples illustrate the configuration of three sample stat type definitions as they appear in Configuration Manager. TotalOutboundStatusTime, shown in Figure 22, measures the total duration that agents, places, group of agents, or groups of places are in a CallOutbound state.

Full Definition

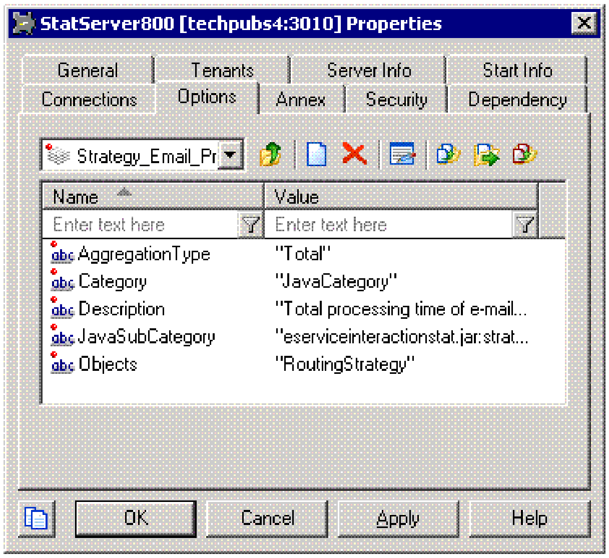
```
[TotalOutboundStatusTime]
   Objects = Agent,Place,GroupAgents,
      GroupPlaces
   Category = TotalTime
   MainMask = CallOutbound
   Subject = AgentStatus
```



Name	Value
Enter text here	Enter text here
Objects	"Agent, Place, GroupAgents, GroupPlaces"
Category	"TotalTime"
MainMask	"CallOutbound"
Subject	"AgentStatus"

Figure 22: Sample Stat Type Definition for Total Duration of Status for CallOutbound Actions

`Strategy_Email_ProcessingTime`, shown in Figure 23, measures the total processing time of e-mail interactions in a simple routing strategy. (Stat Server uses the `RoutingStrategy` object type to monitor `Script` objects in Configuration Server having `Simple Routing` type.)



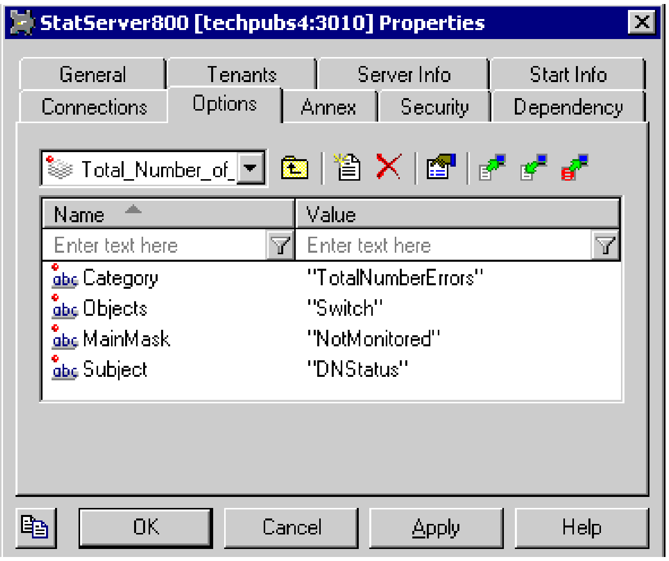
Name	Value
AggregationType	"Total"
Category	"JavaCategory"
Description	"Total processing time of e-mail...
JavaSubCategory	"eserviceinteractionstat.jar:strat...
Objects	"RoutingStrategy"

Full Definition

```
[Strategy_Email_ProcessingTime]
   AggregationType=Total
   Category=JavaCategory
   JavaSubCategory=
      eserviceinteractionstat.jar:
      strategy-total processing time
   MediaType=email
   Objects=RoutingStrategy
```

Figure 23: Sample Stat Type Definition for the Processing Time of Interactions

`Total_Number_of_Errors`, shown in Figure 24, measures the total number of hardware errors that occurred at a switch. This statistic is only meaningful for Network T-Server applications.



Name	Value
Category	"TotalNumberErrors"
Objects	"Switch"
MainMask	"NotMonitored"
Subject	"DNStatus"

Figure 24: Sample Stat Type Definition for a Switch Object

Related Documentation Resources

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

Management Framework

- The *Framework 8.0 Deployment Guide*, which will help you configure and install other Framework components.
- The *Framework 8.0 Stat Server Deployment Guide*, which describes how to configure, install, start, stop, and uninstall Stat Server.

Genesys

- The *Genesys 8.0 Resource Capacity Planning Guide*, which explains how the Genesys model has been expanded to serve agents conducting contact center interactions across several media types.
- The *Genesys 8.0 Instant Messaging Solution Guide*, which describes how to configure media channels to receive instant messages from a Session Initiation Protocol (SIP) Server.
- *Genesys Administrator Help*, for information about configuring Genesys applications using Genesys Administrator.
- Documentation specific to the Genesys solution with which you are using Stat Server.
- The *Reporting Technical Reference Guide for the Genesys X Release*, which describes the stat type definitions provided by Genesys solutions.
- *Genesys Technical Publications Glossary*, which ships on the Genesys Documentation Library DVD and which 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, and which provides documented migration strategies for Genesys product releases. Contact Genesys Technical Support for more information.
- Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at `http://genesyslab.com/support`.

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

- *Genesys Supported Operating Environment Reference Manual*
- *Genesys Supported Media Interfaces Reference Manual*

Consult these additional resources as necessary:

- *Genesys 7 Interoperability Guide,* which provides information on the compatibility of Genesys products with various Configuration Layer Environments; Interoperability of Reporting Templates and Solutions; and Gplus Adapters Interoperability.
- *Genesys Licensing Guide*, which introduces you to the concepts, terminology, and procedures relevant to the Genesys licensing system.

For additional system-wide planning tools and information, see the release-specific listings of System Level Documents on the Genesys Technical Support website, accessible from the `system level documents by release` tab in the Knowledge Base `Browse Documents` Section.

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`.

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:

80fr_us_statserver_11-2009_v8.0.001.00

You will need this number when you are talking with Genesys Technical Support 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 27 describes and illustrates the type conventions that are used in this document.

Table 27: 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 below).	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 . . .
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