Stat Server User's Guide

Stat Server 8.5.1

3/20/2023

Table of Contents

Stat Server 8.5.1 User's Guide

What Is Stat Server?

Stat Server is a Genesys application that 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 eServices, beginning with the 7.0 release, Stat Server architecture was improved to include two new statistical object types: `StagingArea` and `Strategy`. 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 Genesys Administrator Extension using the **Resource Capacity Editor** to create capacity rules. The *Resource Capacity Planning Guide* describes this model and how to use it.

> **Important**
>
> SIP Instant Messaging interactions and e-Services chat interactions use the same media type `chat`, they cannot co-exist for the same Place/Agent. Such deployment is not supported by Genesys Capacity Model.

[+] Stat Server Features

Dynamic Agent Tracking. Stat Server dynamically tracks customer service representatives as they login into DNs and media channels 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 was 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) and to have Stat Server supply them to Genesys applications. The *Framework 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 `TReliability0k`, to its clients upon detecting a stuck call. When determining object actions and statuses, Stat Server considers such events (`EventAbandoned`/`EventReleased` with `AttributeReliability != TReliability0k`) 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 Reference Manual* and the **check-vqstuck-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 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.

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.

Overview

Stat Server Architecture

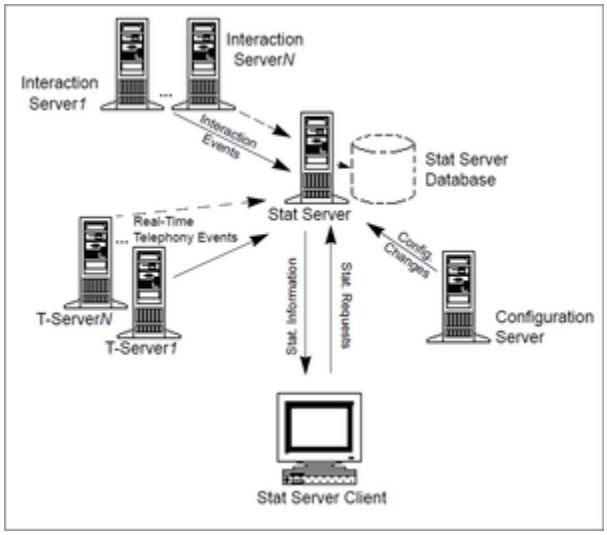
Stat Server supplies statistical information to client applications such as Universal Routing Server (URS), Data Sourcer, and CCPulse+ (formerly known as Call Center Pulse or CC Pulse), upon request. URS, for example, can request information through user-designed strategies.

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.

Starting with release 8.5, Stat Server supports connection to multiple Interaction Servers for the same tenant. The resultant agent/place state is composed of partial states on Stat Server side.

The Figure below shows how Stat Server performs in an actual environment (the dashed entities are optional):



Stat Server Architecture

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 and other parameters.

Regarding Stat Server 7.0.2 and forward releases, when an Internet interaction enters an email, 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 email, chat, or Web callback server, MS T-Server emulates telephony events, so that Stat Server behavior can be described in PBX terms.

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 DN/media that s/he logs in to.

In this example, Stat Server simultaneously monitors each DN/media 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.

> **Important**
>
> 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 disconnects. 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. The backup files, generated by one instance of Stat Server, can be used by any other instance of Stat Server (possibly, on a different platform).

New in This Release

This page highlights new or changed functionality that was introduced in Stat Server release 8.5.1.

New in Release 8.5.112.10

Release 8.5.112.10 introduces the following new features and functionality:

- Stat Server supports the following new Agent Workbin actions:
  - InteractionAbandonedDuringOffering
  - InteractionAccepted
  - InteractionAnswered
  - InteractionCleared
  - InteractionDistributed
  - InteractionEntered
  - InteractionReleased
  - InteractionWait

New in Release 8.5.112.07

Release 8.5.112.07 introduces the following new features and functionality:

- Stat Server supports protected mode in addition to normal mode during configuration reloads.
- The following new configuration option is added in the **[statserver]** section:
  - config-reload-delay-if-primary
- Stat Server supports Kernel Virtual Machine (KVM) on Windows and Linux.
- Stat Server supports Microsoft SQL Server 2017.

New in Release 8.5.110.20

Release 8.5.110.20 introduces the following new features and functionality:

- Stat Server supports changes of the interaction cost for budget-based routing.

New in Release 8.5.110.18

Release 8.5.110.18 introduces the following new features and functionality:

- Stat Server supports the SinceLogin time profile for blended agents.
- Stat Server supports the durable InteractionAgentPartyInProgress action for Tenants.
- Stat Server supports the new DoNotDisturb action on a media channel.
- The following new configuration options are added in the **[statserver]** section:
  - interaction-agent-party-in-progress-on-tenant-max-number
  - interaction-agent-party-in-progress-on-tenant-media-list
  - old-stats-remove

New in Release 8.5.110.05

Release 8.5.110.05 introduces the following new features and functionality:

- Enhanced tracking of a multimedia interaction associated with the virtual queue (VQ).
  Stat Server continues to collect VQ tracking details using the `EventCustom` event, but now does so from all connected Interaction Servers that may participate in processing of the same interaction. This enhancement addresses cases where URS dispatches routing events through one Interaction Server and then offers the interaction with follow-up handling using another Interaction Server.

New in Release 8.5.110.03

Release 8.5.110.03 introduces the following new features and functionality:

- Stat Server supports budget-based routing.
- The following new configuration options, configured in the new **[budget]** section, have been added to support budget-based routing:
  - **`Agent`** or **`Place`** object:
    <media type name>
  - **`MediaType`** object:
    default-agent-budget

    default-cost

    default-cost-consult

    default-cost-inbound

    default-cost-internal

    default-cost-outbound

default-cost-unknown

- **Tenant** object:
  default-agent-budget

  default-cost

  default-cost-consult

  default-cost-inbound

  default-cost-internal

  default-cost-outbound

  default-cost-unknown

  enabled

  interaction-cost-key

New in Release 8.5.109

Release 8.5.109 introduces the following new features and functionality:

- In addition to Warm Standby, Stat Server supports the Hot Standby redundancy mode for Stat Server high availability (HA) pairs on Windows and Linux platforms. See Hot Standby for more information.
- The following new configuration options, configured in the new **[ha]** section, have been added to support Hot Standby redundancy:
  - addp-remote-timeout
  - addp-timeout
  - addp-trace
  - chunk-size
  - chunk-timeout
  - connect-timeout
  - session-expiration-period
  - session-expiration-timeout
- Stat Server maps the content of the `attribute_reason_desc` from Interaction Server events to the value of the `ReasonValue` key in a `UserData` key-value list on Action for the `CurrentState` or in a `Reasons` key-value list for the `CurrentStateReasons`. See CurrentState Categories for more information.

New in Release 8.5.108

Release 8.5.108 introduces the following new features and functionality:

- Stat Server supports the overload protection method of reducing CPU consumption.
- The following new configuration options are added in the **[overload]** section:
  - allow-new-connections-during-overload
  - allow-new-requests-during-overload
  - cpu-cooldown-cycles
  - cpu-poll-timeout
  - cpu-threshold-high
  - cpu-threshold-low
  - cut-debug-log
  - protection
  - qos-default-overload-policy
  - qos-recovery-enable-lms-messages
- Stat Server supports the new `DynamicOverloadPolicy` option in the **[<stat type>]** section.
- Stat Server supports the retrospective InteractionAbandoned action for Tenants.
- Stat Server supports Windows Server 2016 and Windows Server 2016 Hyper-V.

New in Release 8.5.107

Release 8.5.107 introduces the following new features and functionality:

- Stat Server supports Sliding, Selection, and SinceLogin aggregation intervals in the the statistical category Formula.
- Stat Server logs operational statistics and current execution context into a special StatFile log file to help with diagnostics and troubleshooting.
- Stat Server supports the durable InteractionWait action for Tenants. The action is intended to track an interaction until it is actually handled by an agent.
- Stat Server supports the durable DND action on `RegDN`.
- Stat Server supports Red Hat Enterprise Linux 7.
- The following new configuration options are added in the **[statserver]** section:
  - interaction-wait-on-tenant-max-number
  - interaction-wait-on-tenant-media-list
  - stat-file-show-clients-list
  - stat-file-show-options
  - stat-file-timeout

New in Release 8.5.106

Release 8.5.106 introduces the following new features and functionality:

- Stat Server supports the statistical category Formula for the `Growing` time profile.
- Stat Server supports the Ever Growing time profile for a limited set of the float-valued categories.
- Stat Server supports the new DialRemoteRelease action on `Campaign/CallingList` statistical objects.
- Stat Server supports enable-thread, throttle-period, and throttle-threshold options.

New in Release 8.5.105

Release 8.5.105 introduces the following new features and functionality:

- Stat Server can be configured to undertake multiple attempts to register a DN.
- Stat Server supports multiple business attribute values in stat types.
- Stat Server supports new `InteractionResponded` action on a regular DN and on an agent or place.
- The following new configuration options are added in the **[statserver]** section:
  - reg-error-delay
  - reg-error-max-count

New in Release 8.5.104

Release 8.5.104 introduces the following new features and functionality:

- Stat Server supports new Routable and NotRoutable actions on an agent or place.
- New DistinquishBy feature is introduced.
- Stat Server supports new ApplyFilterAtActionEndOnly stat type option.
- Stat Server supports new ExternalServiceRequested and ExternalServiceResponded actions for Tenants.
- Stat Server supports the `ActorType` and `RequestEnvelope` System attributes on selected actions.
- The following new configuration options are added in the **[statserver]** section:
  - consult-acw-mode
  - suppress-user-data

New in Release 8.5.103

Release 8.5.103 introduces the following new features and functionality:

- New GroupBy feature is introduced.
- Support of the time zone specification in a growing `TimeProfile` definition.
- Stat Server supports the following new actions for Tenants:
  - InteractionAccepted
  - InteractionAnswered
  - InteractionReleased
- Stat Server supports new InteractionWait action on StagingArea.
- Stat Server supports new `UserEventReceived` action for a regular DN and mediation DN.
- Stat Server supports the `AgentID` System attribute on selected actions.
- Stat Server supports the `GlobalUserData` key-value list in filters and formulas.
- The following new configuration options are added in the `statserver` section:
  - interaction-wait-on-sa-max-number
  - interaction-wait-on-sa-media-list

New in Release 8.5.102

Release 8.5.102 introduces the following new features and functionality:

- Stat Server supports the following new actions for Tenants:
  - InteractionCreated
  - InteractionAbandonedDuringOffering
  - InteractionAccepted1
  - InteractionDeleted
  - InteractionPastAcceptServiceLevel
  - InteractionPastCompletionServiceLevel
- Stat Server calculates regular statistics on StagingArea for the following new actions:
  - InteractionPastAcceptServiceLevel
  - InteractionPastCompletionServiceLevel
- The `ExpectedWaitTime2` statistical category is now applicable to voice media.
- The new `backup-file-aggregates-store` configuration option is added in the `statserver` section.
- Stat Server supports System key-value list in filters.
- Stat Server supports `SuspendAll`/`ResumeAll`/`PeekAll` batch requests from a client.

New in Release 8.5.101

Release 8.5.101 introduces the following new features and functionality:

- Stat Server supports the following new actions for a regular DN:
  - CallObserving...
  - CallOutboundOriginated
  - CallOutboundReceived
- Stat Server calculates regular statistics on StagingArea for the following new actions:
  - InteractionAbandonedDuringOffering
  - InteractionAccepted
  - InteractionAnswered
  - InteractionReleased
- Absolute paths are now mandatory for the `[java-config]/jvm-path` option. The JVM will not be initialized if the option is set to a relative path.
- Stat Server supports JDK 1.8.
- Stat Server includes functionality for the prevention of database update losses due to commit failures or loss of database server connection.
  Two new database-related options are added:
  - db-timeout
  - db-txn-max-retries

New in Release 8.5.100

Release 8.5.100 introduces the following new features and functionality:

- Stat Server loads configuration data asynchronously. Therefore, during the initialization Stat Server is able to:
  - Immediately process changes of its configuration options.
  - Dynamically recognize configuration changes such as creating, deleting or editing of configuration objects.
  - Dynamically change the list of its connections.
- After a long disconnect from Configuration Server, when the history log expires, Stat Server is able to re-read its configuration. Therefore, Stat Server processes all configuration changes that were done during its disconnect to Configuration Server.
  This new functionality is essential for large environments, where Stat Server initialization takes a long time.
- The `reconnect-timeout` option in the `statserver` section now is also applicable to the Stat Server

connection to Configuration Server.

- Stat Server calculates regular statistics on StagingArea for the following new actions:
  - InteractionCreated
  - InteractionEntered
  - InteractionDistributed
  - InteractionCleared
  - InteractionDistributedToQueue
  - InteractionDeleted

  Stat Server Java Extensions do not need to be loaded for calculation of these statistics.

Statistic Configuration Options

In the Genesys Statistical Model, you configure the metrics that Stat Server should collect for its clients within the Stat Server application itself on the **Options** tab. This chapter describes those options. To learn about the options that you can use to configure other aspects of the Stat Server application apart from metric definitions, refer to the RTME Options Reference.

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

- TimeProfiles Section
- Filters Section
- TimeRanges Section
- Statistical Type Sections

The Genesys Statistical Model is described in the **Overview** book of the *Reporting Technical Reference* series on the Reporting Landing Page.

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. Stat Server begins the calculation of statistics from the moment when they are requested; the reset of statistics to zero (0) depends on the specified `TimeProfile`.

The following table lists the one configuration option that is applicable to the **[TimeProfiles]** section.

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, composed of letters, underscore, numbers, 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. Stat Server uses a special time profile, called `Default`, if a client does not specify a time profile when requesting statistics. The `Default` time profile 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. Default Value: No default value. Valid Values: Dependent on interval type. (See the following subsections.) Changes Take Effect: Immediately.

Stat Server projects actions and statuses onto time intervals (except for the `TotalAdjustedTime` and `TotalAdjustedNumber` statistical categories) 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

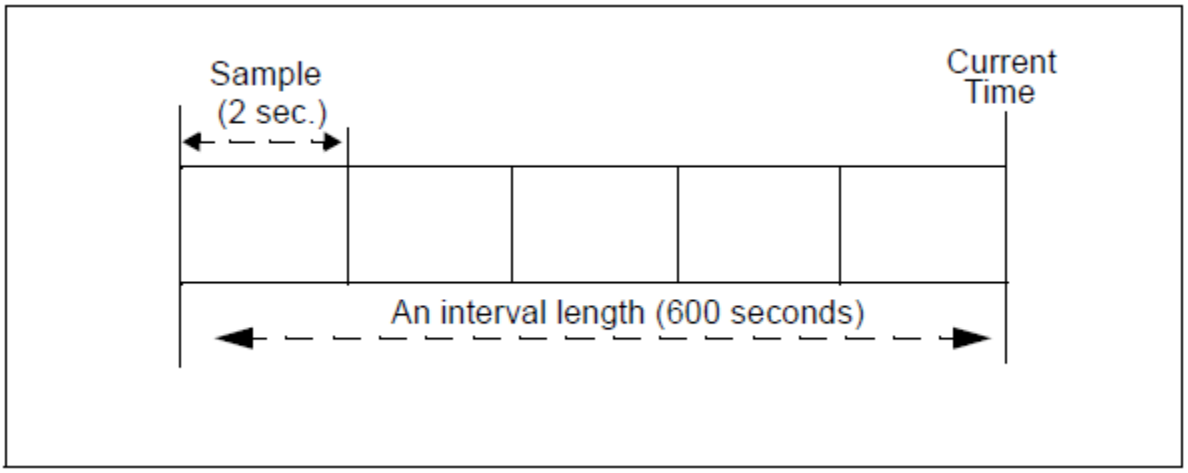
Values for the `Sliding` interval type use the following format:

```
interval:sampling
```

where

`interval` specifies the duration, in seconds, of the reporting interval.
`sampling` (optional) specifies the non-zero duration, in seconds of the sampling. If the 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.



Example of Sliding Interval Type

To create this time profile, 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.

> ### Warning
> Stat Server may actually aggregate on an interval, exceeding the predefined sliding window length by up to the length of a slide.

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. Allows to shorten the **Value** field when there is a pattern between resets.
  The optional increment is also in the 24-hour clock format and is relative to the time to reset statistics to zero.

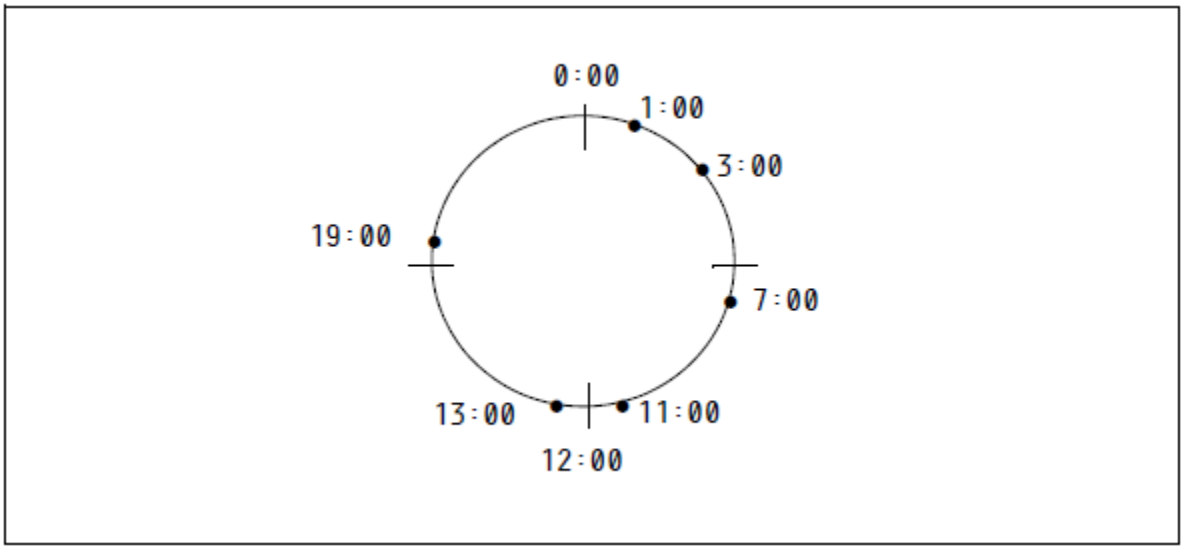
If no time profile is specified for a statistic requested by any client, Stat Server calculates statistics using the `Growing` interval type, which re-sets statistics to zero at `00:00` (midnight) unless a time profile named `Default` in the **[TimeProfiles]** section specifies a different initialization 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.

> **Important**
>
> To specify more than one set of values, separate the sets with commas.

Example 1. 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).

The Figure below illustrates this example.



Example of Growing Interval Type

Example 2. Suppose that you want to set up a time profile (named `Shifts`) that resets statistics to zero when shifts change at 2:00 AM, 5:00 AM, 8:00 AM, 11:00 AM, 1:30 PM, and 8:30 PM. To do so, enter `Shifts,Growing` in the **Name** field and `2:00 +3:00, 13:30 +7:00` in the **Value** field.

In this example, `2:00 +3:00` is translated as reset to zero at 2:00 AM, reset to zero at 2:00 AM plus 3 hours (5:00 AM), reset to zero at 5:00 AM plus 3 hours (8:00 AM), and then reset to zero again at 8:00 AM plus 3 hours (11:00 AM). The setting `13:30 +7:00` is translated as reset to zero at 1:30 PM (or 13:30 on the 24-hour clock), and then reset to zero at 1:30 PM plus 7 hours (8:30 PM, or 20:30 on the 24-hour clock).

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.

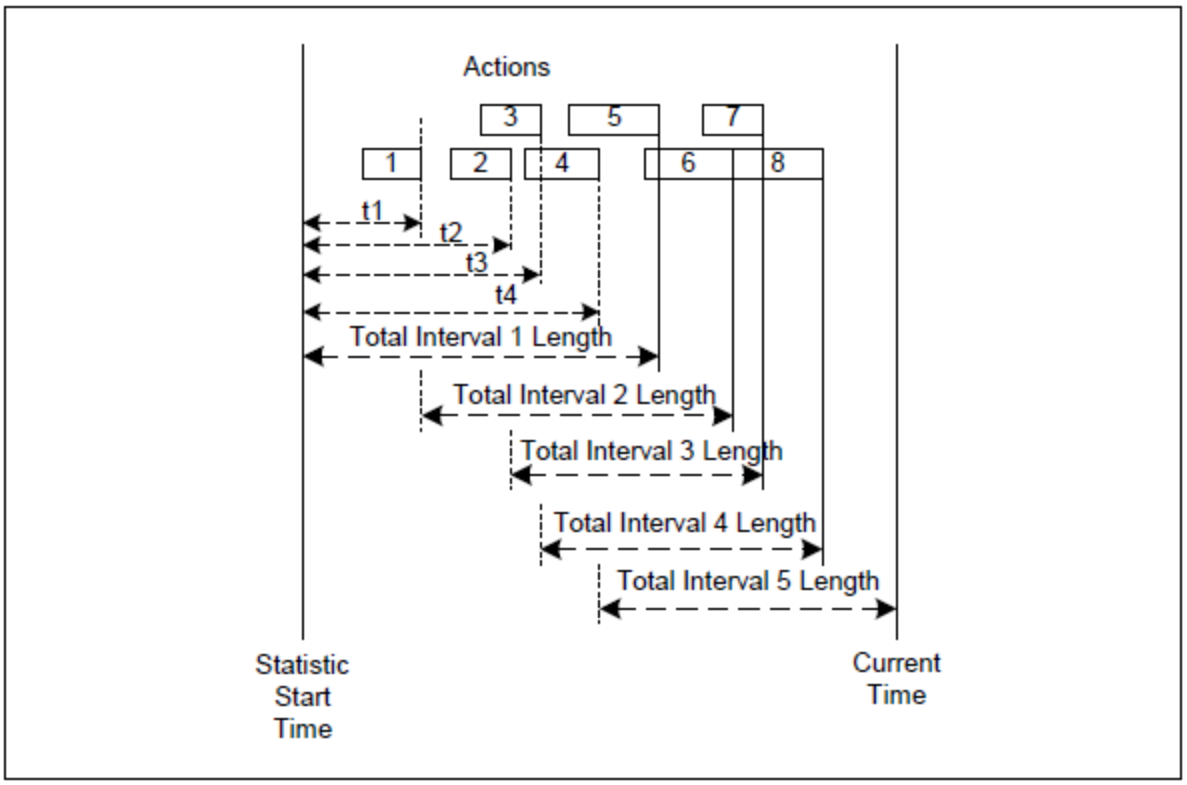
The actions or statuses taken into account are those listed either in the relative mask of the statistical type on which a statistic is based, or in the main mask if no relative mask is specified for the statistical type (see also Statistical Type Sections). The time interval varies depending on the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

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

> ### Important
> 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.

The Figure below 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 `CurrentTime`, the interval only includes durations of four actions (5 through 8).



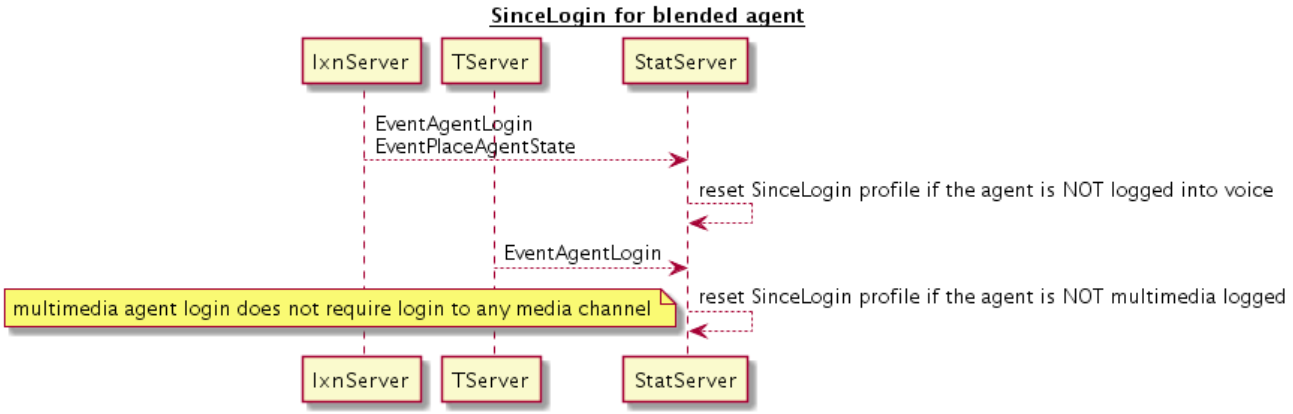
Example of Selection Interval Type

Values for SinceLogin Interval 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. Statistics continue to accumulate as long as the agent is logged into (any) DN. 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.

Starting with release 8.5.100.18, Stat Server supports the `SinceLogin` time profile for blended agents. This accounts requested statistics from the moment of the first-to-come concurrent login to a voice or/and multimedia environment to the moment of the next reset.



Time Zone Specification in a Growing TimeProfile

Starting with release 8.5.103, Stat Server recognizes time zone objects, specified in the `Environment` tenant.

You can specify a time zone in the growing `TimeProfile` definition (for both static and dynamic time profiles) as following:
`[<name of associated time zone object> @ ] <time profile definition, e.g. 0:00>`
where
`[]` is an optional part of the time profile definition. If that part is not provided, the local time zone of the machine where Stat Server runs is used.

Example: `[TimeProfiles]`
`EasternEuropeMidnight,Growing=EET@0:00`

If Stat Server does not recognize the name of the time zone, the time profile is deactivated (not used in statistics). When a time zone is added later, or a time profile is modified, so that the time zone name is recognized, then the time profile is activated (used in statistics).

Important

- Time zones are allowed with `Growing` time profiles only.
- Single Stat Server can handle time profiles for different time zones.
- Stat Server ignores time zone objects in `Tenants`, other than `Environment`.
- There is no need to assign the `Environment` tenant to Stat Server to make it read time zone definitions from it.
- Changes in time zone/time profile definitions take effect immediately.

- Time zone can be used in default time profile definition.
- The @ symbol is allowed in time zone names (Stat Server parses time profile definition backwards, looking for the very last @).
- When a time profile is deactivated, all associated statistics are closed.
- When a client requests the statistic with deactivated time profile, the default time profile is used.
- When the time zone object is modified, the time profile is not deactivated and the next reset time is recalculated for each time profile associated with this time zone.
- When the time profile value is modified, the associated statistics are not closed, if this time profile remained active after the modification.
- Stat Server ignores `State Enabled` flag of the time zone configuration object.

Ever Growing TimeProfile

Starting with release 8.5.106, Stat Server supports `Growing` interval that is never reset. It is defined as a normal `Growing` time profile with an asterisk (*) as a value:

```
[TimeProfiles]
<name>,Growing=*
```

Tip

- Statistical category must be historical, otherwise `EventError` will be sent.
- Ever `Growing` time profile MUST not be used with the reset-based notification, otherwise `EventError` will be sent for such request.
- Statistical category must be float-valued. Below is the list of supported float-valued categories:
  - AverageCustomValue
  - AverageNumberPerRelativeHour
  - AverageOfCurrentNumber
  - AverageOfCurrentTime
  - AverageTime
  - ElapsedTimePercentage
  - Formula (historical only)

- JavaCategory (historical only)
- MaxCustomValue
- MinCustomValue
- RelativeNumber
- RelativeNumberPercentage
- RelativeTimePercentage
- ServiceFactor1
- TotalCustomValue
- TotalNumberInTimeRangePercentage
- TotalNumberPerSecond

Notification Modes

When requesting statistics, clients also specify how often they expect updates on the statistical values. Stat Server sends updates for Historical and Current statistics 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. Starting with Stat Server release 8.5.102, for `Sliding` time profile the notification frequency is replaced with the length of the slide (but not of the window); updates are sent upon slide shift, and also upon change (exceeding the insensitivity).
- `TimeBased` – Stat Server reports the current value at the specified notification frequency (for example, every two seconds). Starting with Stat Server release 8.5.102, for `Sliding` time profile the notification frequency is replaced with the length of the slide (but not of the window); updates are sent only upon slide shift (taking insensitivity into account); the first timer will fire at some point before the time of the specified notification frequency. Starting with release 8.5.108.17, Stat Server uses the notification frequency or sliding window length (not the size of the slide), whichever is smaller, to provide current values of requested statistics to clients.
- `ResetBased` – For `Historical` statistics, Stat Server reports the value right before setting the statistical value to zero (0). For `Current` statistics, the `ResetBased` notification mode is supported with the `Growing` time profile only. At the point of reset, `Current` statistics report current values without setting statistical values to zero (0). `CurrentState` statistics cannot be requested with the `ResetBased` notification mode.
- `NoNotification` – Stat Server does not report updates or updates are turned off.

Tip

It is recommended that for `Sliding` time profile the slide length is a divisor of 3600, e.g. 2, 3, 5, 6, 10, 12, 30, 60, 120 seconds etc.

Important

The minimum frequency of Stat Server updates is 2 seconds.

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).

Tip

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, GlobalUserData, Extensions, 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.

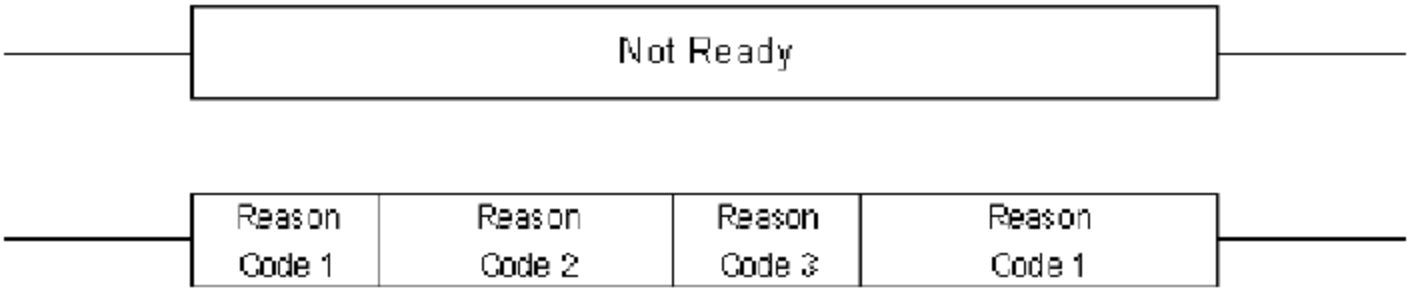
[+] Example

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=ReasonCode 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.



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 Pulse. (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 (no restrictions) that represents the name of the filter.
- *option value* – A logical condition that contains call or device properties and numeric or string constants that are combined by an operator. You can use the ? and/or * wildcard characters in the designation of the option's value. Stat Server matches ? in a wildcard string to any single character. Stat Server matches * to zero or more characters. The default value uses the `PairExists` function.

> **Important**
>
> - Wildcard characters in the designation of the option's value for the `ExtensionReasonCode` are supported only with the 3-argument filters. **Example:** `ExtRC1_a=PairExist(Extensions,"ReasonCode 1","*")`.
> - For media channels, if there are multiple actions with the same priority, use filters with statistics where the Subject is `DNAction` or `Action`.

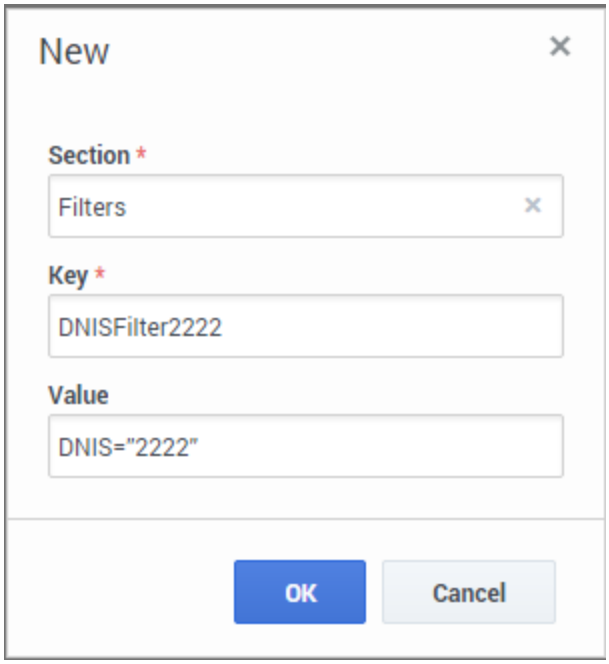
Configuration Option for Filters Section

Option	Description
<FilterName>	Defines a filter for filtering out call- and non-call-

Option	Description
	related activity, based on certain criteria that are specified in a logical condition. The logical expression is composed of: • Call or device properties • Operators • Values that consist of numerics, character string constants, or empty strings, depending on the call or device property. **Important** • The names of filters are unique only to the Stat Server application in which they are defined; they do not inherently reveal the tenant who created them. In multi-tenant environments that share the same Stat Server application, consider implementing a naming convention, such as `TenantName-FilterName`, to help users readily identify those filters that are pertinent to their branch of the business. • In addition, you must specify a value for this option; otherwise, Stat Server uses its default. Default Value: `PairExists("`*filtername*`", "*")` Valid Values: A logical expression Changes Take Effect: Immediately 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, in your Stat Server `Application Options` under the `[Filters]` section enter `DNISFilter2222` in the Key field and `DNIS="2222"` in the `Value` field:



Defining a Filter

In this example, the call property is `DNIS`, the operator is the equal sign (=), and the constant is 2222.

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. Starting with release 8.5.102, `DNIS` can be compared to a variable string expression, in addition to a literal string.
ANI	string	`ANI` is the Automated Number Identification. The `ANI` is all or part of the caller's telephone number.
CustomerID	string	`CustomerID` is the tenant identification number as defined in the Configuration Layer.
MediaType	integer	`MediaType` identifies the media of interaction. For example, the media type of a call is `voice`. The predefined, case-sensitive media types are as follows:

Property Name	Operand Type	Description
		• 0 (voice) • 1 (voip) • 2 (email) • 3 (vmail) • 4 (smail) • 5 (chat) • 6 (video) • 7 (cobrowsing) • 8 (whiteboard) • 9 (appsharing) • 10 (webform) • 11 (workitem) • 12 (callback) • 13 (fax) • 14 (imchat) • 15 (busevent) • 16 (alert) • 17 (sms) • 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. Starting with release 8.5.102, `ThisQueue` can be compared to a variable string expression, in addition to a literal string.
TreatmentType	string	The type of the treatment applied to a call, such as `Silence`, `Music`, `Busy`, and so forth. **Important** Use the `Treatment` key for this attribute in Stat Server filters. For example, `Treatment=Busy`.
UserData	string (TKVList)	`UserData` refers to the data that is attached to an interaction. An

Property Name	Operand Type	Description
		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. Important 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.
GlobalUserData	string (TKVList)	`GlobalUserData` contains all user-data, associated with a given action, unlike `UserData`,

Property Name	Operand Type	Description
		which only contains the user-data, attached by the object, on which the action is generated.
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. For example: `PairExists(Extensions,"Sales",10000)` (if the value is numeric) `PairExists(Extensions,"Color","Green")` (if the value is string) 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`

Property Name	Operand Type	Description
		• 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

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 )`

Property Name	Operand Type	Description
		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. **Important** Do not confuse this `Reasons` property with `Reason`, which serves as an alias for the `ExtensionReasonCode` property described in the next row.
ExtensionReasonCode	string	Refers to the reason code that T-Server propagates in its `AttributeExtensions` 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 `ExtensionReasonCode`. This should not be confused with the `Reasons` property described in the row above. 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`.

Property Name	Operand Type	Description
		• `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`) **Important** 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 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.

Starting with release 8.5.102, the following features are available:

- `GetList` function and `IsNull` predicate in filters and formulas.
- Keywords `true` and `false`.
- Inline `if`: `<boolean expression> ? <result1> : <result2>`.
  If boolean expression is true then `result1` is returned, else `result2` is returned.
  Example:
  ~IsNull(Reason) ? (Reason>5) : false
- `DNIS` and `ThisQueue` can be compared to a variable string expression.
  Example:
  ```
  DNIS=GetGlobalNumber( "DNNumber" )
  ThisQueue != 8001
  ```

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 the UserData Properties table, 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.

UserData Properties

Operators	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".
GetList(<List>, "Key", "Value")	Returns a list of data and can be used in other functions that are able to process any list of data. **Example:** MyFilter1=PairExists(GetList(

Operators	Description
	UserData, "key1"), "key2", "*").
IsNull(<Parameter>)	Returns true or false. The parameter can be one of the following: UserData, ExtensionReasonCode (Reason), Reasons, or Extensions.

Note: Starting with release 8.5.102, GetAver, GetSum, GetMin, GetMax, GetNumber, and GetString are applicable not only to UserData but to any key-value list attributes: UserData, Reasons, ExtensionReasonCode (Reason), Extensions, System, or a list returned by the GetList function.

Examples:
```
GetAver( Reasons, "key" )
GetAver( Extensions, "key" )
GetAver( System, "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.

System Key-Value List

Stat Server supports System key-value list in filters.

The supported key-value list contains the following system attributes:

System Attribute Name	Value	Notes
AgentID	string	Supported on the AgentLogin,

System Attribute Name	Value	Notes
		AgentReady, and AgentActive mediation DN actions when the queue-use-pseudo-actions configuration option is set to false.
ActorType	strategy, agent, media_server	Supported on multimedia actions. The value is taken from the attr_actor_type event attribute. There are no benefits of using the ActorType attribute with durable actions.
budget_avail	integer	Available for Agents, Places, AgentGroups, and PlaceGroups on Routable and NotRoutable actions. Introduced in Stat Server release 8.5.110.03 to support budget-based routing.
budget_timestamp	integer	Available for Agents, Places, AgentGroups, and PlaceGroups on Routable and NotRoutable actions. Introduced in Stat Server release 8.5.110.20 to support budget-based routing.
budget_total	integer	Available for Agents, Places, AgentGroups , and PlaceGroups on Routable and NotRoutable actions. Introduced in Stat Server release 8.5.110.03 to support budget-based routing.
budget_used	integer	Available for Agents, Places, AgentGroups, and PlaceGroups on Routable and NotRoutable actions. Introduced in Stat Server release 8.5.110.03 to support budget-based routing.
ConnectionID	string	Applicable only for voice.
current_number	integer	Current number of interactions of given a media type on an agent/ place. Applicable only to the Routable and NotRoutable actions.
InitialOperation	unknown, transfer, conference, intrude, route, pull, create, reject, timeout, leave, stop, place_in_workbin, place_in_queue, party_disconnect	Can be used in both Filters and UserData formulas. Applicable only for multimedia.
InteractionCost	integer	Available for call actions on DN, Media Channel, Queue, and Routing Point. Introduced in Stat Server release 8.5.110.03 to support budget-based routing.

System Attribute Name	Value	Notes
		The value of the `InteractionCost` attribute can change.
OrigInteractionCost	integer	Available for call actions on `DN`, `Media Channel`, `Queue`, and `Routing Point`. Introduced in Stat Server release 8.5.110.20 to support budget-based routing. The `OrigInteractionCost` immutable system attribute coincides with the value of the `InteractionCost` when the corresponding action is started.
InteractionID	string	Applicable only for multimedia.
InteractionSubtype	InboundNew, OutboundNew, etc.	Applicable only for multimedia.
InteractionType	Unknown, Inbound, Outbound, Consult, Internal	For multimedia, `Unknown` value is not applicable.
IsAccepted	yes, no	Applicable only for multimedia. Can have the `yes` value only in case of the first acceptance of an inbound interaction (Interaction Server event attribute `attr_itx_delivered_at` is equal to `NULL`).
IsOnline	yes, no	Applicable only for multimedia. Starting with release 8.5.104, the `IsOnline` system attribute is **no longer supported**.
max_number	integer	Maximum allowed number (according to a capacity rule) of interactions of a given media type on an agent/place. Applicable only to the Routable and NotRoutable actions.
media_state	1, 0	1 if media is ready and 0 otherwise. Applicable only to the Routable and NotRoutable actions.
MediaType	chat, email, voice, etc.	
Operation	unknown, transfer, conference, intrude, route, pull, create, reject, timeout, leave, stop, place_in_workbin, place_in_queue, party_disconnect	Should only be used in `UserData` formulas. Applicable only for multimedia.
RequestEnvelope	string	Associated with the `attr_esp_request_envelope` event attribute. Applicable only to the ExternalServiceRequested and ExternalServiceResponded actions.

System Attribute Name	Value	Notes
routable	integer	Number of interactions of a given media type that can be routed to an agent/place, according to a capacity rule. Applicable only to the Routable and NotRoutable actions. **Warning** Number of routable interactions for a given media is updated only by an interaction of this given media type.
ServiceObjective CompleteServiceObjective	duration in seconds	
VisibilityMode	unknown, conference, monitor, coach	Applicable only for multimedia.
WorkbinID	string	Name of the agent workbin into which the interaction is placed. It is defined by the `attr_itx_workbin_type_id` attribute from the Interaction Server event.
WorkbinOwnerID	string	Employee ID of the agent who owns the agent workbin into which the interaction is placed. It is defined by the `attr_itx_agent_id` attribute from the Interaction Server event.

Important

- All system attributes, with the exception of the `AgentID` attribute, in the Table above are supported starting with release 8.5.102. The `AgentID` attribute is supported starting with release 8.5.103. The `ActorType`, `current_number`, `max_number`, `media_state`, `RequestEnvelope`, and `routable` attributes are supported starting with release 8.5.104.
- Starting with release 8.5.104, the `IsOnline` system attribute is no longer supported.
- System attribute names are case-sensitive.
- `PairExists()` without a key-value list indication does not search through System attributes.

Example 1:
To calculate interactions with `InteractionType = Inbound` the following filter can be used:

```
F1 = PairExists( System, "InteractionType", "Inbound" )
```

Example 2:

To count interactions that have been placed by an agent in an Interaction Queue the following stat type can be used:

```
Category=TotalCustomValue
Objects=Agent
MainMask=CallInbound,CallOutbound,CallInternal
Formula=PairExists( System, "Operation", "place_in_queue" ) ? 1 : 0
Subject=DNAction
```

Example 3:
To count the number of chats, routable to an agent group the following stat type can be used:

```
Category=CurrentCustomValue
Objects=GroupAgents
MainMask=Routable
Subject=DNAction
Formula=GetNumber( System, "routable" )
```

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`

See Statistical Categories for more information.

The **[TimeRanges]** section contains one or more *<TimeRangeName>* configuration options. The Table below describes the one configuration option applicable for this section.

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 numbers separated by a hyphen (-): the starting point and the end of the range in seconds, such as `0-20`. Default Value: 0-20 Valid Value: Any value specified in the described format above Changes Take Effect: When Stat Server restarts **Important** • 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.

Option	Description
	When a statistic is requested with a `Category` that uses the time range and there is no time range specified in the request, Stat Server calculates this statistic with the `<Default> TimeRange` configured in the **[TimeRanges]** section. If no `<Default> TimeRange` is configured in the **[TimeRanges]** section, Stat Server calculates this statistic with the predefined time range of `0-20`. Stat Server truncates milliseconds from timestamps before determining duration. So, according to Stat Server, the duration of a call that is queued with a timestamp of `05:40:56.949`, for example, and answered at `05:41:07.542` is 11 seconds, and not 10.593 seconds. This difference of as much as one second can affect in which time range the duration of an interaction falls.

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 Range`0-30` 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.
- 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
- Options for Java stat types
- Options for Formula stat types

> **Warning**
> The Formula category (as a value of the `Category` option) and the stat type definition option `Formula` (described in the table below) are two distinct objects.

A small number of the options serve both core and Java stat type classifications, but these options have differing permissible values. The table below lists all configuration options that you can use to define stat types. The third column in the table indicates that you can specify this option for stat types that are used in conjunction with a Stat Server Java extension.

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

Stat Type Configuration Options

Option	Description	Java
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: Refer to Stat Server Object Types and Descriptions and Campaign Objects. Changes Take Effect: When Stat Server restarts.	Yes
MainMask	Specifies a list of comma-separated actions (or statuses in case of status-based Subject option in the Stat Type) that indicate which contact center events will be measured. This list comprises members from the following groups: • regular DN actions • mediation DN actions • media-channel actions • campaign actions • statuses This option is mandatory for core stat types and you must specify one or more values. Use the wildcard (*) 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)` Please note that the logical NOT (~) character does not work	

Option	Description	Java
	with aliases for the group of actions, such as `CallWait` on mediation DNs. To exclude the whole group, each member should be excluded explicitly. For example: `*,~CallWaitUnknown,~CallWaitConsult,~CallWaitInbound,~CallWaitOutbound` 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, Object Statuses, and Campaign Operational Actions 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 in case of status-based Subject option in the Stat Type) 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 the following groups: • regular DN actions • mediation DN actions • media-channel actions • campaign actions • statuses Specifying this option is not mandatory, but if you do use it, you must supply one or more values. Use the wildcard (*) character to specify all actions; use the	

Option	Description	Java
	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 ~. Please note that the logical NOT (~) character does not work with aliases for the group of actions, such as `CallWait` on mediation DNs. To exclude the whole group, each member should be excluded explicitly. For example: `*,~CallWaitUnknown,~CallWaitConsult,~CallWaitInbound,~CallWaitOutbound` Default Value: No default value Valid Values: Refer to Stat Server Actions, Object Statuses, and Campaign Operational Actions for a listing and description of these actions and statuses. Changes Take Effect: When Stat Server restarts.	
Category	Informs Stat Server how to calculate statistics. This section is mandatory for both core and Java stat types. You must supply one and only one value. Default Value: No default value Valid Values: • For Stat Server core stat types, refer to Statistical Categories. • For Java stat types, this value must be: `JavaCategory` Changes Take Effect: When Stat Server restarts.	Yes
JavaSubCategory	The name of the Java subclass that implements statistic calculation. Default Value: No default value	Yes

Option	Description	Java
	Valid Values: String specified in the following format: **jarfile:subclass** Changes Take Effect: When Stat Server restarts.	
Subject	Specifies the subject type for statistics calculation that, when changed, affects the statistical value. 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` Refer to Statistical Subjects for a description of these values. Changes Take Effect: When Stat Server restarts. Important 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 business attributes that Stat Server applies as a filter during its computation of statistics. The value string is recognized as a single value only. Starting with release 7.1, Stat Server supports 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.	Yes

Option	Description	Java
	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. Starting with the 8.5.105 release, Stat Server supports multiple business attribute values, specified as a comma-separated list for a given business attribute, in the stat type definition. Comma in the list of business attribute values is working as `or` in the related filter. Business attributes in different options of the Stat Type are working as different independent filters. This is applicable to all statistical categories, except Compound Categories and `JavaCategory`. Important • There are no restrictions on using business attributes with stat types that have only instant and/or retrospective actions in `MainMask` and `RelMask`. • Mutable business attributes (those that can be changed during the life of interaction), such as Customer Segment, should not be used with stat types that have durable actions in `MainMask` and/or `RelMask` due to unpredictable statistical results. • Modification of business attributes (and/or their values) in the configuration, while Stat Server is running, might cause unpredictable statistical results. • Business attributes usage when `Subject` is set to `DNStatus` or	

Option	Description	Java
	> `PlaceStatus` or `AgentStatus` is not recommended due to unpredictable statistical results.	
ReasonStartOverridesStatusStart	Determines how Stat Server computes current-state statistics. If this option is set to `no`, Stat Server uses the timestamp that is affiliated with the agent's current status, as in prior releases, to determine statistical values. If this option is set to `yes`, Stat Server also considers the timestamp that is affiliated with changes in reason code. Default Value: `no` Valid Values: `yes`, `no` Changes Take Effect: When Stat Server restarts. Setting this option to `yes` enables Stat Server to provide more refined results for those circumstances in which agents designate different reasons for being in the same state. This option is applicable only to the `CurrentStateReasons` statistical category. The `Subject` has to be set to `DNAction` for monitoring reasons changes.	
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.	

Option	Description	Java
	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**=historical categories only, with the exceptions of `JavaCategory` and `TotalAdjustedTime`. 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 for an extended discussion of Stat Server's use of source timestamps.	
Formula	Enables Stat Server to compute user-specific quantities that are based on attached data communicated by TEvents. The Custom Formulas chapter is dedicated to an extended discussion of this subject. You can define a custom formula as described in the Custom Formulas section below. A special specifier—`DistByConnID`—affects Stat Server's mechanism of aggregating statistics for the call-related actions that are listed in the main mask. `DistByConnID` is applicable only to the limited number of statistical categories:	

Option	Description	Java

	• TotalNumber
	• TotalAdjustedNumber
	• CurrentNumber
	• TotalTime
	
	When the DistByConnID specifier is used in a stat type's definition, 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. **For example**, three agents are participating in the internal conference call. The ConnID is the same for all three agents. We want to calculate the value of the CurrentNumber statistic for the CallInternal action for the group of these agents. If DistByConnID is applied, the result is 1, otherwise the result is 3.
	
	Stat Server's procedure of grouping actions by connection ID applies to the actions specified in **MainMask** for the objects that are associated with the statistic. The procedure differs for each statistical category and is 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 a 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).	

Option	Description	Java

	If the action or status is unrelated to a call, then aggregation functions in the same manner as when `DistByConnID` is not specified.
	
	Important
	`DistByConnID` is not applicable in `TotalNumber/TotalAdjustedNumber` statistics where filters or time ranges are also defined. Therefore, using `DistByConnID` with the `TotalNumber/TotalAdjustedNumber` statistical categories together with filters or time ranges may produce wrong results.
	
	• 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	

Option	Description	Java

	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 a group of actions ends. If an action is either not call-related or not durable, Stat Server ignores this action in statistic calculations.
	
	Tip
	• If you use the `DistByConnID` specifier, you must list it first among the **Formula** values as such: `Formula=DistByConnID,...`
	• Stat Server recognizes the following aliases for `DistByConnID`:
	  • `DistinguishByConnID`
	  • `DCID`
	• Any filtering that might be used in conjunction with a statistic, such as the designation of a `MediaType`, is applied *prior* to Stat Server's processing of `DistByConnID`.
	• When used with `DistByConnID`, `CurrentNumber` and `TotalTime` categories ignore actions that are not associated with T-Server calls.	

Option	Description	Java
	• The `TotalNumber` category, used with the `DistByConnID` specifier, considers actions that are not associated with T-Server calls as having a unique `ConnectionID`. Therefore, those actions are taken into account. Default Value: No default value Valid Values: `DistByConnID`, `DCID`, `DistinguishByConnID`, *<custom formula>* Changes Take Effect: When Stat Server restarts. Starting with release 8.5.102, the following features are available in formulas: • `GetList` function and `IsNull` predicate. • Keywords `true` and `false`. • Inline `if`: `<boolean expression> ? <result1> : <result2>`. If boolean expression is true then `result1` is returned, else `result2` is returned. • `DNIS` and `ThisQueue` can be compared to a variable string expression.	
ApplyFilterAtActionEndOnly	With this option set to `yes`, Stat Server checks the filter on a durable action at its end and if it is `true`, then the whole action duration (not a sub-period while the filter was true) is used in a statistical algorithm. This option is only taken into account for `Subject=DNAction` or `Subject=CampaignAction` and only for `Growing` and	

Option	Description	Java

	`SlidingWindow` aggregation intervals.
	Default Value: `no`
	Valid Values: `yes, no`
	Changes Take Effect: When Stat Server restarts.
	Important
	The `ApplyFilterAtActionEndOnly` specifier:
	• Can be used with durable, instant, retrospective actions (for instant and retrospective actions, the statistic behavior with or without the specifier is identical).
	• Is not taken into account for current statistical categories.
	• Is not taken into account for the following (historical) statistical categories:
	  • `ServiceFactor1`
	  • `JavaCategory`
	  • `AverageOfCurrentNumber`
	  • `AverageOfCurrentTime`
	  • `EstimWaitingTime`
	  • `ExpectedWaitTime2`
	  • `LoadBalance`
	  • `MinNumber`	

Option	Description	Java
	• MaxNumber • TotalNumberErrors • TotalAdjustedTime (if mode=ResetBasedNotification only) • TotalCustomValue • MinCustomValue • MaxCustomValue • AverageCustomValue	
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.	Yes
DynamicOverloadPolicy	Defines actions that Stat Server may apply to a given statistic to reduce the overload. Default Value: 0 Valid Values: • 0 - sends and updates for requested statistics can be cut;	

Option	Description	Java
	• 1 - only sends of statistics to Stat Server clients can be cut; • 2 - nothing can be cut, Stat Server updates and sends all requested statistics. Changes Take Effect: When Stat Server restarts.	
<any other name>	Defines a custom parameter (specific option) for the stat-type with **Category** set to `JavaCategory`.	

Important

- 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, Stat Server will recognize the change only upon restart.
- Stat Server clients may recognize other options for stat types that are not listed in the Table above. For instance, Data Sourcer requires that the **AggregationType** option 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. At the top of this page see a description of the predefined statistical type format.

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

List of Custom-Value Categories

Historical	Current

• TotalCustomValue • AverageCustomValue • MinCustomValue • MaxCustomValue	• CurrentCustomValue • CurrentAverageCustomValue • CurrentMinCustomValue • CurrentMaxCustomValue

These categories are described on the Historical CustomValue Categories and Current CustomValue Categories pages.

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, 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 Formula 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

> **Important**
> For an evaluation of custom formulas, refer to Custom Formulas.

Note: Custom formulas can be requested with **Subject**=`DNAction` only.

Custom formulas define custom values from an action 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 the Key-Value List Functions in Custom Formulas table below. Note that the list can include more than one pair with the same key.
- Operators, as well as parentheses (for suppressing standard precedence rules).

Operators in Custom Formulas

Operator	Description
+	Addition
-	Subtraction
/	Division
*	Multiplication
?	Then (in an Inline IF)
:	Else (in an Inline IF)

- Numeric constants.

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

> **Important**
>
> You can apply filters to custom-formula statistics too.

The Table below lists functions to access key-value `UserData` lists. 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.

> **Important**
>
> For momentary actions, the `GetGlobalMax` function returns the same value as the `GetMax` function.

Key-Value List Functions in Custom Formulas

Local Functions (Used for Local Key-Value List Calculations)	
Function	**Description**
GetNumber("Key", Index)	Returns the numeric value of the occurrence of the given key as specified by `Index`:

Local Functions (Used for Local Key-Value List Calculations)	
	• 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 (zero). 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.
GetList(<List>, "Key", "Value")	Returns a list of data and can be used in other functions that are able to process any list of data. **Example:** MyFilter1=PairExists(GetList(UserData, "key1"), "key2", "*").
IsNull(<Parameter>)	Returns true or false. The parameter can be one of the following: UserData, ExtensionReasonCode (Reason), Reasons, or Extensions.
	Note: Starting with release 8.5.102, GetAver, GetSum, GetMin, GetMax, and GetNumber are applicable not only to UserData but to any key-value list attributes: UserData, Reasons, ExtensionReasonCode (Reason), Extensions, System, or a list returned by the GetList function.
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. Index is an optional attribute for this property. If not specified,

Local Functions (Used for Local Key-Value List Calculations)	
	Stat Server substitutes `-1` for its value; hence, `GetGlobalNumber("Key")` is equivalent to `GetGlobalNumber("Key", -1)`.
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 1: Suppose that 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" )
```

Example 2:
To count interactions that have been placed by an agent in an Interaction Queue the following stat type can be used:

```
Category=TotalCustomValue
Objects=Agent
MainMask=CallInbound,CallOutbound,CallInternal
Formula=PairExists( System, "Operation", "place_in_queue" ) ? 1 : 0
Subject=DNAction
```

> **Important**
>
> `PairExists()` without a key-value list indication does not search through System attributes.

GroupBy Feature

Starting with release 8.5.103, Stat Server supports the `GroupBy` stat type specifier. `GroupBy` defines

the mapping of an action to a group. Each action belongs to only one group at any moment of time.

GroupBy statement is the comma separated expression list. An expression can be of types:

- bool, for example the PairExists function.
- float and int, for example GetNumber, GetMin, GetMax, and other functions from the Key-Value List Functions in Custom Formulas table.
- string, for example the GetString function.

The result value for each group is calculated individually and is defined by Category. A special token null, which indicates an absence of a key, is a possible value for an expression.

There are three optional specifiers that can be used with GroupBy in stat types:

- OrderBy
  OrderBy is the comma separated list of tokens. Each token is a positive or negative integer, that refers to an index of the element in the GroupBy expression.

> **Tip**
> Indexes start with 1.
> If OrderBy is not provided, the groups come unordered.
> Use the minus (-) sign to sort in descending order.

- RowCount
  RowCount is an integer stat type specifier that allows you to limit the number of returned groups. Its valid values are from 0 (zero) to 2147483647. If RowCount is not specified, the default value of 2147483647 is used.
  Note that, in some scenarios, Stat Server can ignore the RowCount set limit when loading saved aggregate data from backup files, resulting in additional groups being restored.
- GroupByColumns
  GroupByColumns specifies column names, corresponding to GroupBy. Stat Server sends this information to its clients in the GroupByColumns key in the embedded key-value list, associated with GroupBy StatType.

> **Important**
> - Only the DNAction Subject is allowed for GroupBy.
> - GroupBy is not compatible with DistByConnID in formulas.
> - Legacy clients, such as CCPulse+, do not support GroupBy.
> - Filters can be used in combination with GroupBy.
> - GroupBy stat types are serialized into/from backup file, as well as other statistics.
> - GroupBy is supported with old (dynamic) API as well.

The following categories cannot be used with GroupBy:

- CurrentState
- CurrentStateReasons
- CurrentTargetState
- JavaCategory
- ServiceFactor1
- TotalNumberErrors
- EstimWaitingTime
- LoadBalance
- EstimTimeToComplete
- Formula
- ExpectedWaitTime2
- CurrentDistinctNumber
- TotalDistinctTime

GroupBy cannot be used with the following statistics:

Categories	Object	Subject	Action
CurrentNumber	Campaign	CampaignAction	RecordsNotProcessed
CurrentNumber	CallingList	CampaignAction	RecordsNotProcessed
CurrentNumber	Campaign	CampaignAction	RecordsScheduled
CurrentNumber	CallingList	CampaignAction	RecordsScheduled

If the queue-use-pseudo-actions configuration option is set to true, GroupBy cannot be used with the following statistics:

Categories	Object	Subject	Action
CurrentNumber	Queue, RoutePoint, GroupQueues	DNAction	CallWait
CurrentNumber	Queue, RoutePoint, GroupQueues	DNAction	AgentLogin
CurrentNumber	Queue, RoutePoint, GroupQueues	DNAction	AgentActive
CurrentNumber	Queue, RoutePoint, GroupQueues	DNAction	AgentReady
CurrentNumber	Queue, RoutePoint, GroupQueues	DNAction	DNLogin
CurrentNumber	Queue, RoutePoint, GroupQueues	DNAction	DNActive
CurrentNumber	Queue, RoutePoint, GroupQueues	DNAction	DNReady
CurrentRelativeNumberPercentage	Queue, RoutePoint, GroupQueues	DNAction	any of the above actions

Example 1.

If you want to know the total time each agent was active while logged into a queue, you can use the following stat type:

```
Category = TotalTime
Objects = Queue, GroupQueues
Subject = DNAction
MainMask = AgentActive
GroupBy = GetString(System, "AgentID")
```

For the above statistic you will get a result, similar to below:

Total value:	120 seconds
Agent_1	25 seconds
Agnet_2	50 seconds
Agent_3	45 seconds

Example 2.

If you want to know how many calls came from each customer on some Agent Group, assuming that the customer name is attached as User Data in the `customer_name` key, you can use the following stat type:

```
Category = TotalNumber
Objects = GroupAgents
Subject = DNAction
MainMask = CallAnswered
GroupBy = GetString(UserData, "customer_name")
```

For the above statistic you will get a result, similar to below:

Total value:	10
Customer_1	3
Customer_2	5
Customer_3	2

Example 3.

If User Data contains some additional information about customers, like country and city in corresponding keys and you want to get calls distribution for these parameters, you can use the following stat type:

```
Category = TotalNumber
Objects = GroupAgents
Subject = DNAction
MainMask = CallAnswered
GroupBy = GetString(UserData, "customer_name"), GetString(UserData, "country"),
GetNumber(UserData, "city")
```

For the above statistic you will get a result, similar to below:

Total value:	10		
Customer_1	United States	New York	3
Customer_2	Canada	Toronto	5
Customer_3	United States	San Francisco	2

If you need to order resulted groups by country (the second item in a group) in ascending order add `OrderBy=+2` to the stat type definition.

If you need to see only first 2 groups, add `RowCount=2` to the stat type definition.

DistinguishBy

Starting with release 8.5.104, Stat Server supports the `DistinguishBy` stat type specifier to prevent duplicate counts for the `CurrentNumber` statistical category, based on the action's content.

Stat Server applies an expression, defined by `DistinguishBy`, to any action in the `MainMask`. Therefore, Stat Server maps an action to a group.

Important

- Only the `DNAction Subject` is allowed for the `DistinguishBy` specifier.
- The `DistinguishBy` specifier is only supported for the `CurrentNumber` statistical category.
- `Formula=DCID` is ignored, if specified in addition to `DistinguishBy`.
- The `GroupBy` specifier is ignored, if specified in addition to `DistinguishBy`.
- The `DistinguishBy` specifier can also be used with the dynamic (old) API.

Example 1.

The following statistic calculates the number of agents associated with a `GroupQueues` via the login into a mediation DN, which belongs to this `GroupQueues`. The agent, logged into multiple queues, is counted only once. Make sure, that the **[statserver]**/queue-use-pseudo-actions option is set to `false`.

```
Category=CurrentNumber
Objects=GroupQueues
MainMask=AgentLogin
Subject=DNAction
DistinguishBy=GetString( System, "AgentID" )
```

Example 2.

The following statistic calculates the current number of distinct pairs (<interaction type>, <key value>) extracted from Handling actions on an agent group.

```
Category=CurrentNumber
Objects=GroupAgents
MainMask=Handling
Subject=DNAction
DistinguishBy=GetString( System, "InteractionType" ), GetNumber( GlobalUserData,
"key2" )
```

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.

The following topics describe the object types that Stat Server monitors and how they relate to each other:

- Object Descriptions
- Object Hierarchy
- General Notes About Objects

Refer to Campaign Objects 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 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. Object-type specification identifies which internal event model Stat Server uses in the acquisition of statistical values. The Table below describes all of the types of objects Stat Server monitors.

Stat Server Object Types and Descriptions

Object Type	Description	
RegDN	Regular DN (directory number) applies to the following DN type: `data`, `music`, `mixed`, `extension`, `ACD position`, `Voice Treatment port`, `voice mail`, `cellular`, and CP (call-processing equipment). Except for extensions for Meridian-like T-Servers and Voice Treatment ports, all of these DN types require login events.	
Agent	Stat Server tracks agents by a unique identification `Employee ID`.	
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	Stat Server tracks the activity occurring at: • Automatic Call Distribution (ACD)-associated points at which calls wait for agent availability. • Virtual queue DNs, a special type of DN that is maintained by a CTI installation and whose behavior is identical to that of a routing point.	
RoutePoint	Stat Server tracks the activity occurring at: • Regular routing point DNs,	

Object Type	Description	
	where calls wait for routing. These points might have different names on different switching platforms (for example, CDN, VDN, and so forth). • Virtual routing point DNs, which designate 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.	
GroupAgents	This object type designates a collection of agents that is identified by a GroupID. An agent can be a member of more than one agent group. No matter where agents log in, their activity can be monitored as part of the group. Stat Server also attributes the activity of virtual agent groups to this object type. Virtual agent groups are dynamically generated within Configuration Server. Refer to Supported Virtual Agent Group Definitions for more information.	
GroupPlaces	This object type designates a group of places. Each place that is part of the group has a unique PlaceID, which is associated with the GroupID.	
GroupQueues	This object type designates a group that includes the following Stat Server object types: • Queues (ACD and virtual) • Routing points (regular and virtual)	
RoutingStrategy	This object type 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 Simple Routing or Enhanced	

Object Type	Description	
	`Routing`.	
StagingArea	This object type corresponds to the `Script` Configuration Server of type `Interaction Queue`. It is analogous to the concept of queues for the eServices (formerly known as Multimedia) solution in which customer interactions may reside while they are being processed.	
Switch	This object type names a switch. You can collect only one piece of information for objects having this object type; namely, the total number of hardware errors that occurred at the switch. Refer to Creating Stat Type Definitions for an example of how to define this statistic.	
Tenant	An object that represents a business entity within the Configuration Server.	

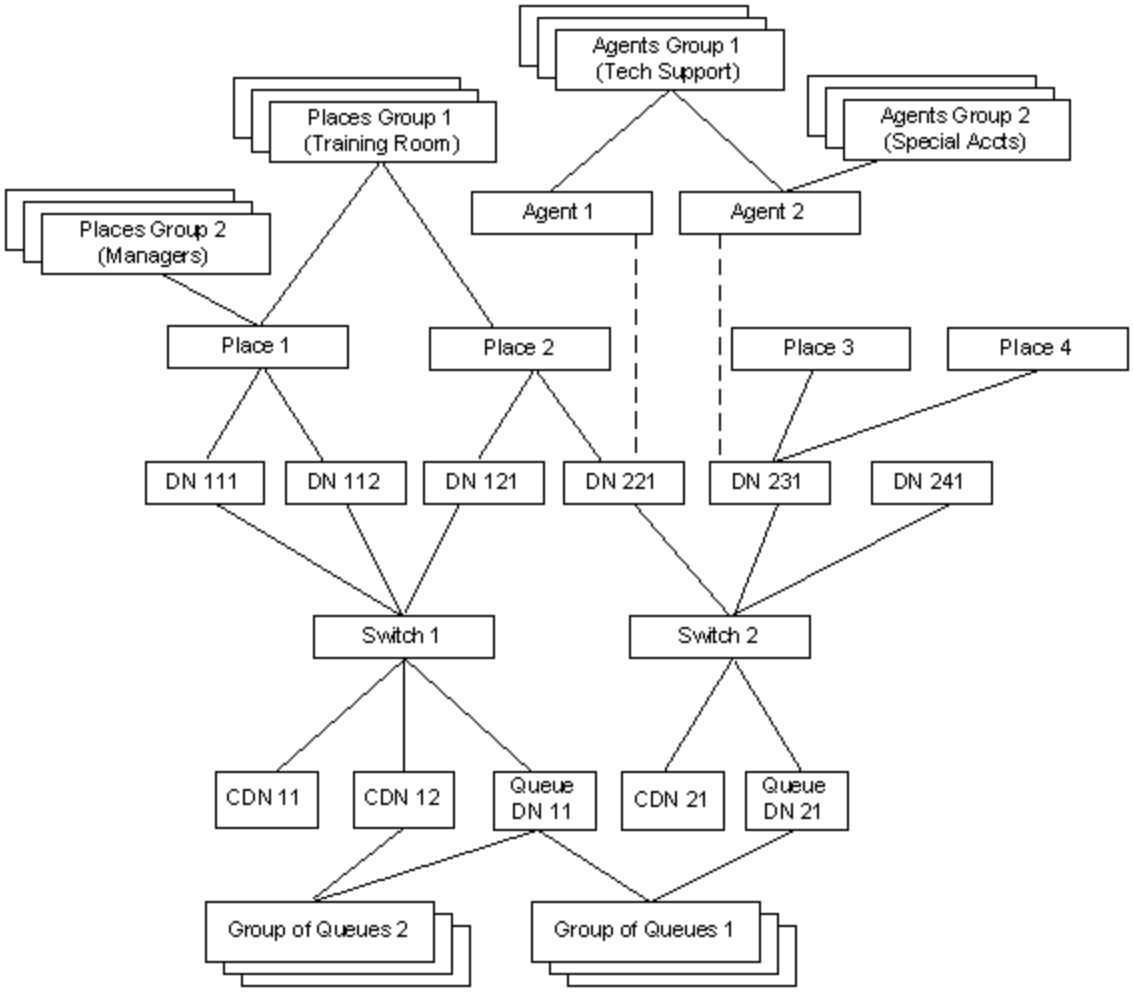
Object Hierarchy

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. The Table below shows those groups of objects whose members Stat Server considers to be potentially compatible.

Stat Server Compatibility Groups				
Regular DN Compatibility Group	**Mediation DN Compatibility Group**	**Campaign Compatibility Group**	**Interaction Queue Compatibility Group**	**Routing Strategy Compatibility Group**
RegDN	RoutePoint	Campaign	StagingArea	RoutingStrategy
Agent	Queue	CallingList		
Place	GroupQueues	CampaignGroup		
GroupAgents	Switch	CampaignCallingList		
GroupPlaces				
Tenant				

For telephony objects, some of these relationships are illustrated below. Objects' relationships defined within an Outbound campaign are illustrated in Hierarchy of Stat Server Campaign Objects. The Stat Server Java Extensions calculate statistics on multimedia objects, such as `StagingArea`, `Tenant`, and `RoutingStrategy`. Starting from release 8.5.1, Stat Server calculates regular statistics on StagingArea object type for InteractionCleared, InteractionCreated, InteractionDeleted, InteractionDistributed, InteractionDistributedToQueue, InteractionEntered, InteractionAbandonedDuringOffering, InteractionAccepted, InteractionAnswered, and InteractionReleased actions. For calculation of these statistics Stat Server Java Extensions do not need to be loaded. Despite similarity between some statistics calculated with Java Extensions on StagingArea and these regular statistics, they are designed differently and serve different purposes.

Hierarchy of Stat Server Telephony Objects



Hierarchy of Stat Server Telephony Objects

Associations Between Agents and DNs/Media Channels

Stat Server creates an association between an agent and a DN/media-channel 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 a DN or media-channel, 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 DN/media-channel.

- 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 DN/media-channel.
- 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 DN/media-channel.

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 the EventAgentLogin TEvent.
- For Stat Server 8.1.0⁻, 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. (Here you can find more information about how Stat Server determines when an agent is logged in at a place for Stat Server 8.1.0⁻.)

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 agent who is logged into 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. The Tables below 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 the 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 in 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 "LOGIN Entries Given T-Server 6.5 and Stat Server 7.0.2 (or prior)") to the LOGIN table, because in T-Server 6.5, there was no notion of logout from just one queue—the EventQueueLogout 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).

> **Tip**
> LOGIN Entries are presented here in pseudo-table format. Refer to “The LOGIN Table” section in the Appendix of the *Framework Stat Server Deployment Guide* for the actual format of this table.

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, do send the EventQueueLogout TEvent—but Stat Server versions prior to 7.0.3 did not recognize it as noted in the table below.

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

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 Customer Care 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

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.

Stat Server Actions

Overview

Any sequence of events that T-Server or SIP Server reports causes Stat Server to generate an *action*. The same is true for a limited number of events that Interaction Server reports. Information on how Stat Server actions are classified and defined pertains to the values that you might specify in the **MainMask** and/or **RelMask** option; see the Stat Type Configuration Options table.

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. Here we classify the general subdivisions of Stat Server actions and describe individual actions.

- Classifying DN Actions
- Summary of Stat Server Actions
- Propagation of DN Actions
- Action Descriptions
- Regular DN Actions
- Mediation DN Actions
- Media-Channel Actions
- StagingArea Actions
- Tenant Actions
- Agent Workbin Actions

For information about Stat Server actions related to campaigns, see Campaign Statistics.

DN Actions at Newly Registered DNs

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.

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 calltype 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.

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 **Options** tab in Genesys Administrator Extension).

> **Tip**
> 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, and service numbers.

Media-channel actions are all sourced from the Genesys eServices 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 Propagation of DN Actions.

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 retrospective actions that do not derive from durable actions are the following actions:

  Mediation DN actions:

  - CallTreatmentCompleted—This action's duration is taken from the parameters of `EventTreatmentCompleted`.
  - All actions reflecting events at mediation DNs receiving calls distributed to other mediation DNs. Refer to the CallDistributedToQueue actions.
  - All actions reflecting events at regular DNs receiving calls distributed from the mediation DN (see Retrospective, Interaction-Related Actions Reflecting Regular DNs). These actions take their duration either from the preceding CallWait durable action or from a related regular DN durable action.

  Media-Channel actions:

  - CoachingByRequestInitiated

  StagingArea actions:

- InteractionCleared—This action's duration is taken from the parameters of `EventTakenFromQueue`.
- InteractionDeleted—This action's duration is taken from the parameters of `EventProcessingStopped`.
- InteractionDistributed—This action's duration is taken from the parameters of `EventTakenFromQueue`.

> **Important**
> 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.

The Table below 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.

Logical Clusters of Interaction-Related Actions

Durable Action	Initial Momentary Action	Terminal Retrospective Actions
Regular DN Actions		
CallDialing	CallDialingStarted	CallDialed
		CallAbandonedFromDialing
		CallDialTransferred (only possible for consult calls)
		CallDialConferenced (only possible for consult calls)
CallRinging	CallRingingStarted	CallAnswered (Regular DNs)
		CallAbandonedFromRinging (Regular DNs)
		CallRingingPartyChanged (Regular DNs) (only possible for consult calls)
		CallForwarded (Regular DNs)
CallOnHold	CallHeld	CallRetrievedFromHold
		CallAbandonedFromHold
		TransferredFromHold
CallConsult	CallConsultStarted	CallPartyChanged
CallConsultOriginated		
CallConsultReceived		
CallInbound	CallInboundStarted	CallInboundCompleted
CallInternal	CallInternalStarted	CallInternalCompleted
CallOutbound	CallOutboundStarted	CallOutboundCompleted
CallUnknown	CallUnknownStarted	CallUnknownCompleted
Group Actions Reflecting Origination DNs		
OrigDNCallWait	OrigDNCallEntered	OrigDNCallDistributed
		OrigDNCallAbandoned
Mediation DN Actions		
CallWait	CallEntered	CallDistributed
		CallAbandoned
		CallCleared
Media Actions		
Delivering	DeliveringStarted	Accepted
		Rejected
HandlingInbound	HandlingInboundStarted	StoppedInbound
HandlingInternal	HandlingInternalStarted	StoppedInternal

HandlingOutbound	HandlingOutboundStarted	StoppedOutbound

For every cluster of logically related actions shown in the table above (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 `CallDialConferenced` 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 (`CallRingingPartyChanged`, `CallWaitPartyChanged`, and `OrigDNCallWaitPartyChanged`, respectively). These actions account for the possible termination of a consult call during two-step transfers. `CallDialTransferred` 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

The table below provides the high-level classifications for each action and summarizes the set of actions applicable for Stat Server. 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

Campaign-related Stat Server actions are provided in Campaign Actions and Statuses.

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
Accepted		✓				✓			✓
Active		✓					✓		
ACWCompleted			✓			✓			✓
ACWMissed			✓			✓			✓
AfterCallWork	✓						✓		
AgentActive			✓				✓		
AgentLogin (Regular DNs)	✓							✓	
AgentLogin (Mediation DNs)			✓				✓		
AgentLogout	✓								✓
AgentReady			✓				✓		
ASM_Engaged	✓					✓	✓		

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
ASM_Outbound	✓					✓	✓		
Available		✓					✓		
BeingCoached		✓				✓		✓	
BeingMonitored		✓				✓		✓	
Blocked		✓					✓		
CallAbandoned			✓			✓			✓
CallAbandonedFromDialing	✓					✓			✓
CallAbandonedFromHold	✓					✓			✓
CallAbandonedFromRinging (Regular DNs)	✓					✓			✓
CallAbandonedFromRinging (Mediation DNs)			✓			✓			✓
CallAbandonedFromRinging (Virtual Queues)			✓			✓			✓

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
CallAnswered (Regular DNs)	✓					✓			✓
CallAnswered (Mediation DNs)			✓			✓			✓
CallAnswered (Virtual Queues)			✓			✓			✓
CallCleared			✓			✓			✓
CallConferenceJoined	✓					✓		✓	
CallConferenceMade	✓					✓		✓	
CallConferenceOriginated	✓					✓	✓		
CallConferencePartyAdded	✓					✓		✓	
CallConferencePartyDeleted	✓					✓		✓	
CallConsult	✓					✓	✓		
CallConsultCompleted	✓					✓			✓

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
CallConsultOriginated	✓					✓	✓		
CallConsultReceived	✓					✓	✓		
CallConsultStarted	✓					✓		✓	
CallDialConferenced	✓					✓			✓
CallDialTransferred	✓					✓			✓
CallDialed	✓					✓			✓
CallDialing	✓					✓	✓		
CallDialingStarted	✓					✓		✓	
CallDistributed			✓			✓			✓
CallDistributedToQueue			✓			✓			✓
CallEntered			✓			✓		✓	
CallForwarded (Regular DNs)	✓					✓			✓

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
CallForwarded (Mediation DNs)			✓			✓			✓
CallHeld	✓					✓		✓	
CallInbound	✓					✓	✓		
CallInboundCompleted	✓					✓			✓
CallInboundStarted	✓					✓		✓	
CallInternal	✓					✓	✓		
CallInternalCompleted	✓					✓			✓
CallInternalOriginated	✓					✓	✓		
CallInternalReceived	✓					✓	✓		
CallInternalStarted	✓					✓		✓	
CallMissed			✓			✓			✓
CallObserved...	✓					✓	✓		

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
CallObserving...	✓					✓	✓		
CallOnHold	✓					✓	✓		
CallOutbound	✓					✓	✓		
CallOutboundCompleted	✓					✓			✓
CallOutboundOriginated	✓						✓		
CallOutboundReceived	✓						✓		
CallOutboundStarted	✓					✓		✓	
CallPartyChanged	✓					✓			✓
CallReleased (Mediation DNs)			✓			✓			✓
CallReleased (Virtual Queues)			✓			✓			✓
CallRetrievedFromHold	✓					✓			✓
CallRinging	✓					✓	✓		

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
CallRingingPartyChanged (Regular DNs)	✓					✓			✓
CallRingingPartyChanged (Mediation DNs)			✓			✓			✓
CallRingingStarted	✓					✓		✓	
CallTransferMade	✓					✓		✓	
CallTransferPartyChanged	✓					✓		✓	
CallTransferTaken	✓					✓		✓	
CallTreatmentCompleted			✓			✓			✓
CallTreatmentNotStarted			✓			✓		✓	
CallTreatmentStarted			✓			✓		✓	
CallUnknown	✓					✓	✓		
CallUnknownCompleted	✓					✓			✓
CallUnknownStarted	✓					✓		✓	

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
CallWait			✓			✓	✓		
CoachingByIntrusionInitiated		✓				✓		✓	
CoachingByRequestInitiated		✓				✓			✓
CoachingRequested		✓				✓		✓	
ConferenceJoined		✓				✓		✓	
ConferenceJoinedByIntrusion		✓				✓		✓	
ConferenceMade		✓				✓		✓	
DND	✓					✓	✓		
Delivering		✓				✓	✓		
DeliveringStarted		✓				✓		✓	
DNActive			✓				✓		
DNLogin			✓				✓		

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
DNReady			✓				✓		
DoNotDisturb		✓				✓	✓		
ExternalServiceRequested (Tenants)					✓	✓		✓	
ExternalServiceResponded (Tenants)					✓	✓		✓	
Handling		✓				✓	✓		
HandlingInbound		✓				✓	✓		
HandlingInboundStarted		✓				✓		✓	
HandlingInternal		✓				✓	✓		
HandlingInternalStarted		✓				✓		✓	
HandlingOutbound		✓				✓	✓		
HandlingOutboundStarted		✓				✓		✓	
HandlingStarted		✓				✓		✓	

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
InteractionAbandoned (Tenants)					✓	✓			✓
InteractionAbandonedDuringOffering (MediaChannels)		✓				✓			✓
InteractionAbandonedDuringOffering (StagingAreas)				✓		✓			✓
InteractionAbandonedDuringOffering (Tenants)					✓	✓			✓
InteractionAbandonedDuringOffering (Virtual Queues)			✓			✓			✓
InteractionAbandonedDuringOffering (Agent Workbin)						✓			✓
InteractionAccepted1					✓	✓			✓
InteractionAccepted (StagingAreas)				✓		✓			✓
InteractionAccepted (Tenants)					✓	✓			✓
InteractionAccepted (Agent Workbin)						✓			✓

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
InteractionAgentPartyInProgress (Tenants)					✓	✓	✓		
InteractionAnswered (StagingAreas)				✓		✓			✓
InteractionAnswered (Tenants)					✓	✓			✓
InteractionAnswered (Agent Workbin)						✓			✓
InteractionCleared				✓		✓			✓
InteractionCleared (Agent Workbin)						✓			✓
InteractionCreated (StagingAreas)				✓		✓		✓	
InteractionCreated (Tenants)					✓	✓		✓	
InteractionDeleted (StagingAreas)				✓		✓			✓
InteractionDeleted (Tenants)					✓	✓			✓

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
InteractionDistributed				✓		✓			✓
InteractionDistributed (Agent Workbin)						✓			✓
InteractionDistributedToQueue				✓		✓			✓
InteractionEntered				✓		✓		✓	
InteractionEntered (Agent Workbin)						✓		✓	
InteractionPastAcceptServiceLevel (StagingAreas)				✓		✓			✓
InteractionPastAcceptServiceLevel (Tenants)					✓	✓			✓
InteractionPastCompletionServiceLevel (StagingAreas)				✓		✓			✓
InteractionPastCompletionServiceLevel (Tenants)					✓	✓			✓
InteractionReleased (StagingAreas)				✓		✓			✓

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
InteractionReleased (Tenants)					✓	✓			✓
InteractionReleased (Agent Workbin)						✓			✓
InteractionResponded (Regular DNs)	✓							✓	
InteractionResponded (MediaChannels)		✓				✓		✓	
InteractionWait (StagingAreas)				✓		✓	✓		
InteractionWait (Tenants)					✓	✓	✓		
InteractionWait (Agent Workbin)						✓	✓		
LoggedIn	✓						✓		
LoggedOut	✓						✓		
Monitored (Regular DNs)	✓						✓		

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
Monitored (Mediation DNs)			✓				✓		
MonitoringInitiated		✓				✓		✓	
NotMonitored (Regular DNs)	✓						✓		
NotMonitored (Mediation DNs)			✓				✓		
NotReadyForNextCall	✓						✓		
NotRoutable		✓				✓	✓		
OffHook	✓						✓		
OnHook	✓						✓		
OrigDNCallAbandoned	✓					✓			✓
OrigDNCallDistributed	✓					✓			✓
OrigDNCallEntered	✓					✓		✓	

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
OrigDNCallWait	✓					✓	✓		
Pulled		✓				✓		✓	
Rejected		✓				✓			✓
Revoked		✓				✓			✓
Routable		✓				✓	✓		
StartedInternal		✓				✓		✓	
StartedOutbound		✓				✓		✓	
StoppedInbound		✓				✓			✓
StoppedInternal		✓				✓			✓
StoppedOutbound		✓				✓			✓
StuckCallCleaned			✓			✓			✓
StuckCallCleanedWhileRinging (Regular DNs)	✓					✓			✓

Action	Regular DN/SIP DN Objects	Agent/Place Media Channels	Mediation DN Objects	Staging Area DN Objects	Tenant Objects	Ixn-Related	Durable	Instantaneous (Momentary)	Instantaneous (Retrospective)
StuckCallCleanedWhileRinging (Mediation DNs)			✓			✓			✓
TransferMade		✓				✓		✓	
TransferTaken		✓				✓		✓	
TransferredFromHold	✓					✓			✓
UserEvent (Regular DNs)	✓							✓	
UserEvent (Mediation DNs)			✓					✓	
UserEventReceived (Regular DNs)	✓						✓		
UserEventReceived (Mediation DNs)			✓				✓		
WaitForNextCall	✓						✓		

Important

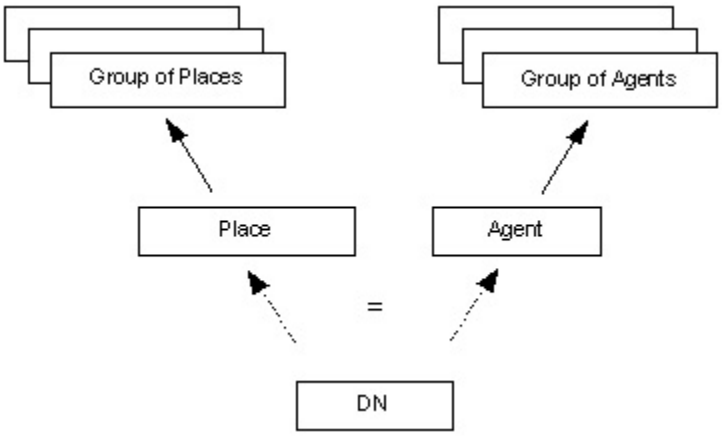
Although the names of many actions presume the processing of a voice interaction, these actions might also apply to other types of interactions—for example, chat sessions and e-mail.

Propagation of DN Actions

Every DN action propagates to higher-level objects. The path propagation takes depends on the Stat Server release.

8.5 and 8.1.2

Stat Server 8.1.2 and higher releases (8.1.2+)



Propagation Hierarchy of Regular DN Action in 8.1.2+

Beginning with Stat Server release 8.1.2, Stat Server propagates a DN action simultaneously to both:

- The place that is associated with the DN and then to the place group comprising the place.
- The agent who is logged in to the DN and then to the agent group comprising the agent.

The Figure illustrates this propagation scheme.

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

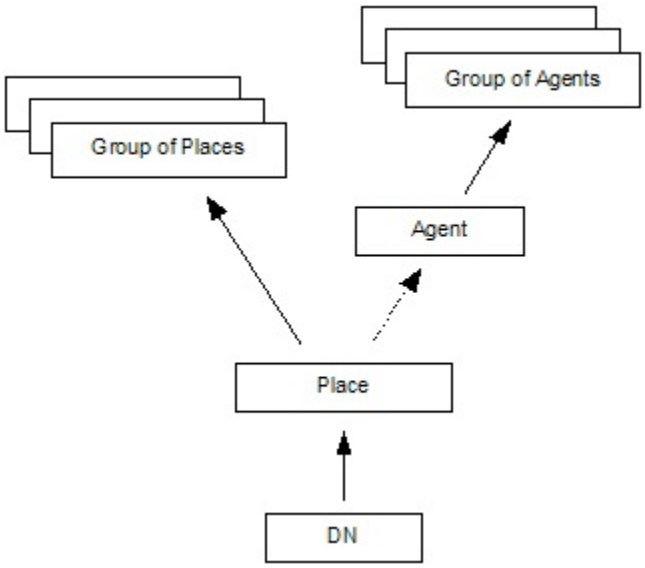
In the 8.1.2+ release, many agents can potentially be logged in to different DNs that are configured on the same place. Genesys, however, will recognize it as misconfiguration.

> **Important**

> In the case of the one-to-one association between the agent and the place ("normal" configuration), the same set of actions is propagated to the agent / agent-groups in 8.1.0^{-} and 8.1.2^{+} releases.

8.1.0

Stat Server 8.1.0 and lower releases (8.1.0^{-})



Propagation Hierarchy of Regular DN Action Prior to 8.1.2

In releases prior to 8.1.2, a DN action propagates to the place to which the DN is linked in the configuration for a regular DN. 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 the Figure.

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

The Figure shows the propagation scheme used by Stat Server 8.1.0 and lower releases (8.1.0^{-})—it illustrates a dynamic connection between agent and place and observes the general rule 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.

In the Stat Server 8.1.0- model, a one-to-one association between the agent and the place is artificially enforced. Stat Server 8.1.0- uses the following rules in tracking the agent-place association:

- When an agent who is not logged in at a place logs in at a place's DN, s/he 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.

> **Important**
>
> 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`.

Validity events are not sent for statistical categories `CurrentState`, `CurrentStateReasons`, `CurrentTargetState`.

Action Descriptions

This page provides a list and descriptions of all the Actions in alphabetical order. If the same action can be generated on different objects, the action name includes the applicable object type in brackets. For lists of the Actions grouped by DN and Action types, see Regular DN Actions, Mediation DN Actions, Media-Channel Actions, StagingArea Actions, and Tenant Actions.

Accepted
Active
ACWCompleted
ACWMissed
AfterCallWork
AgentActive
AgentLogin (Regular DNs)
AgentLogin (Mediation DNs)
AgentLogout
AgentReady
ASM_Engaged
ASM_Outbound
Available
BeingCoached
BeingMonitored
Blocked
CallAbandoned
CallAbandonedFromDialing
CallAbandonedFromHold
CallAbandonedFromRinging (Regular DNs)
CallAbandonedFromRinging (Mediation DNs)
CallAbandonedFromRinging (Virtual Queues)
CallAnswered (Regular DNs)
CallAnswered (Mediation DNs)
CallAnswered (Virtual Queues)
CallCleared
CallConferenceJoined
CallConferenceMade
CallConferenceOriginated
CallConferencePartyAdded
CallConferencePartyDeleted
CallConsult
CallConsultCompleted
CallConsultOriginated
CallConsultReceived
CallConsultStarted
CallDialConferenced
CallDialTransferred
CallDialed
CallDialing
CallDialingStarted
CallDistributed
CallDistributedToQueue
CallEntered
CallForwarded (Regular DNs)
CallForwarded (Mediation DNs)
CallOutbound
CallOutboundCompleted
CallOutboundOriginated
CallOutboundReceived
CallOutboundStarted
CallPartyChanged
CallReleased (Mediation DNs)
CallReleased (Virtual Queues)
CallRetrievedFromHold
CallRinging
CallRingingPartyChanged (Regular DNs)
CallRingingPartyChanged (Mediation DNs)
CallRingingStarted
CallTransferMade
CallTransferPartyChanged
CallTransferTaken
CallTreatmentCompleted
CallTreatmentNotStarted
CallTreatmentStarted
CallUnknown
CallUnknownCompleted
CallUnknownStarted
CallWait
CoachingByIntrusionInitiated
CoachingByRequestInitiated
CoachingRequested
ConferenceJoined
ConferenceJoinedByIntrusion
ConferenceMade
DND
Delivering
DeliveringStarted
DNActive
DNLogin
DNReady
DoNotDisturb
ExternalServiceRequested (Tenants)
ExternalServiceResponded (Tenants)
Handling
HandlingInbound
HandlingInboundStarted
HandlingInternal
HandlingInternalStarted
HandlingOutbound
HandlingOutboundStarted
HandlingStarted
InteractionAnswered (Agent Workbin)
InteractionCleared
InteractionCleared (Agent Workbin)
InteractionCreated (StagingAreas)
InteractionCreated (Tenants)
InteractionDeleted (StagingAreas)
InteractionDeleted (Tenants)
InteractionDistributed
InteractionDistributed (Agent Workbin)
InteractionDistributedToQueue
InteractionEntered
InteractionEntered (Agent Workbin)
InteractionPastAcceptServiceLevel (StagingAreas)
InteractionPastAcceptServiceLevel (Tenants)
InteractionPastCompletionServiceLevel (StagingAreas)
InteractionPastCompletionServiceLevel (Tenants)
InteractionReleased (StagingAreas)
InteractionReleased (Tenants)
InteractionReleased (Agent Workbin)
InteractionResponded (Regular DNs)
InteractionResponded (MediaChannels)
InteractionWait (StagingAreas)
InteractionWait (Tenants)
InteractionWait (Agent Workbin)
LoggedIn
LoggedOut
Monitored (Regular DNs)
Monitored (Mediation DNs)
MonitoringInitiated
NotMonitored (Regular DNs)
NotMonitored (Mediation DNs)
NotReadyForNextCall
NotRoutable
OffHook
OnHook
OrigDNCallAbandoned
OrigDNCallDistributed
OrigDNCallEntered
OrigDNCallWait
Pulled
Rejected
Revoked
Routable

CallHeld CallInbound CallInboundCompleted CallInboundStarted CallInternal CallInternalCompleted CallInternalOriginated CallInternalReceived CallInternalStarted CallMissed CallObserved... CallObserving... CallOnHold	InteractionAbandoned (Tenants) InteractionAbandonedDuringOffering (MediaChannels) InteractionAbandonedDuringOffering (StagingAreas) InteractionAbandonedDuringOffering (Tenants) InteractionAbandonedDuringOffering (Virtual Queues) InteractionAbandonedDuringOffering (Agent Workbin) InteractionAccepted1 InteractionAccepted (StagingAreas) InteractionAccepted (Tenants) InteractionAccepted (Agent Workbin) InteractionAgentPartyInProgress (Tenants) InteractionAnswered (StagingAreas) InteractionAnswered (Tenants)	StartedInternal StartedOutbound StoppedInbound StoppedInternal StoppedOutbound StuckCallCleaned StuckCallCleanedWhileRinging (Regular DNs) StuckCallCleanedWhileRinging (Mediation DNs) TransferMade TransferTaken TransferredFromHold UserEvent (Regular DNs) UserEvent (Mediation DNs) UserEventReceived (Regular DNs) UserEventReceived (Mediation DNs) WaitForNextCall	

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.

Back to top

Active

This durable action tracks how long a media channel has been active for a particular agent (or place). Stat Server generates this action when the `EventMediaAdded` event is received from Interaction Server for the media on a place where an agent is logged in. This action ends with the `EventMediaRemoved` event from Interaction Server.

Back to top

ACWCompleted

This retrospective action occurs on 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 `ACW` on a regular DN upon completion of the interaction. Then, after the `ACW` action is ended, the `ACWCompleted` action is generated on a mediation DN, which distributes the interaction to regular DN. This behavior was introduced in the 7.0 release.

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. This action was introduced in release 7.0.

Back to top

ACWMissed

This retrospective action occurs on a mediation DN when the regular DN action AfterCallWork is over. Action `ACWMissed` is generated on a mediation DN only if an agent enters ACW mode while s/he is on a call that was distributed from a source other than the mediation DN, on which the agent is logged in. Action duration is the same duration as the corresponding action `AfterCallWork`.

Back to top

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.

The UserData, Reasons, and Extensions attributes from the EventDNDOn or EventDNDOff TEvents are not inherited by this action.

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:

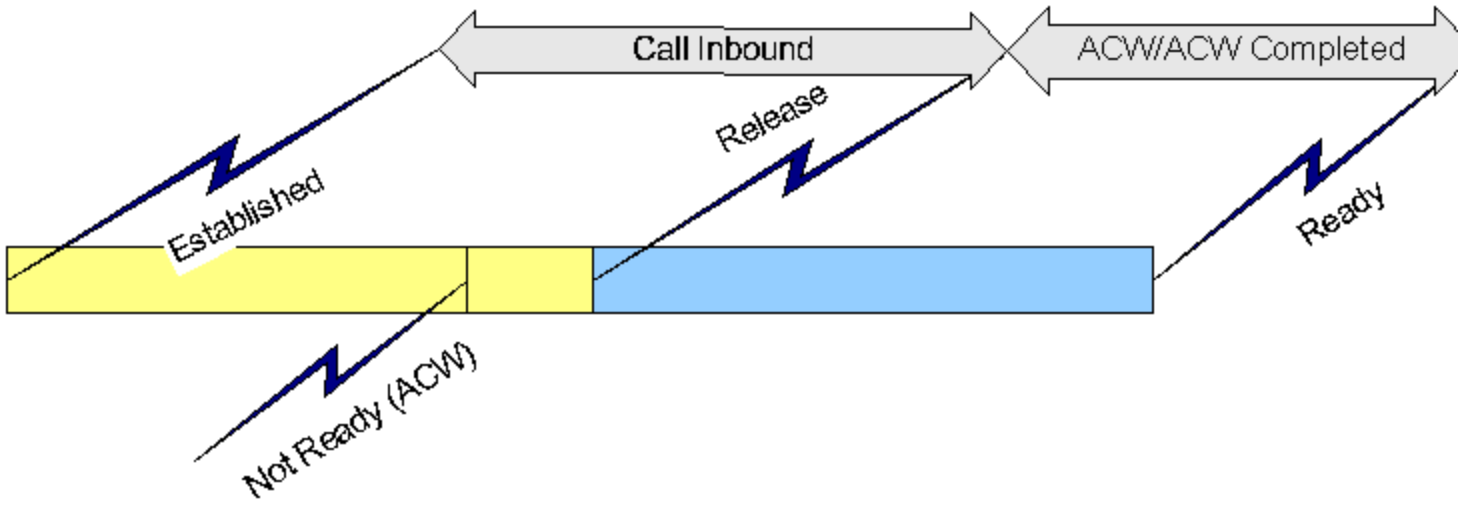
- AfterCallWorkUnknown
- AfterCallWorkInternal
- AfterCallWorkInbound
- AfterCallWorkOutbound
- AfterCallWorkConsult

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.

> **Important**
>
> 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.



ACW Action Generated After Interaction Completion

> **Tip**
> See also DN Actions at Newly Registered DNs.

After-Call Work on the Nortel Meridian T-Servers

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

- NEC-2400
- 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

> **Important**
> 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 Stat Server release 8.5.000.32, the configurable association between position and extension on the switch level is supported for many switches (for more information refer to the position-extension-linked option, which is configurable on the switch Annex). In this case, Stat Server considers related T-Servers as Meridian-like T-Servers.

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-like 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-like 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-like 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`

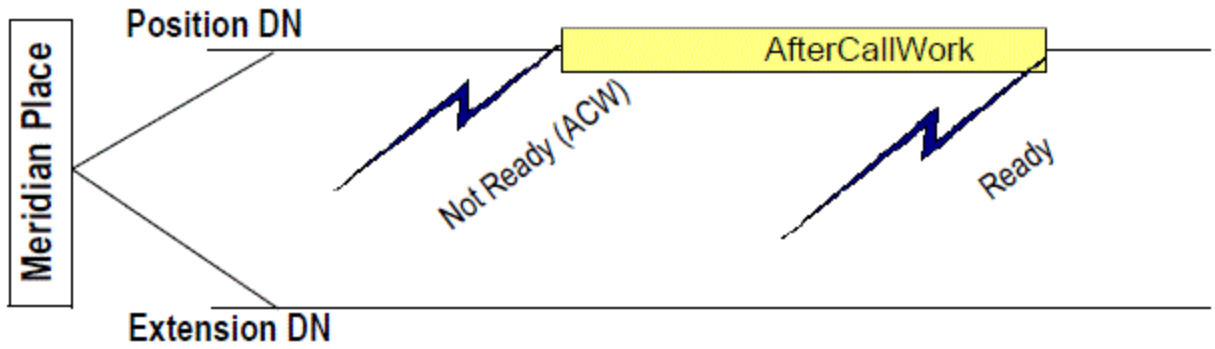
Based on the `EventAgentNotReady` TEvent (with workmode=ACW), Stat Server generates an `AfterCallWork` action on the Position DN and links the action with the appropriate telephony interaction, if applicable. In addition, if Stat Server recognizes this after-call work as associated with a particular 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-like T-Server.

ACW with No Associated Interaction

The diagram below 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-like T-Server, and when there are no telephony interactions on the Position (or Extension) DN.



ACW Given No Telephony Interaction

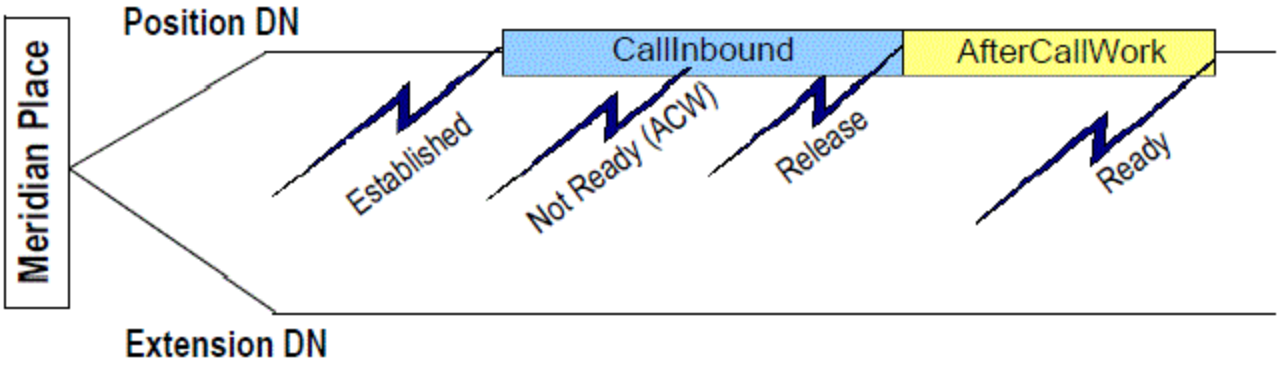
The diagram shows the events occurring on the Position DN where Stat Server starts an `AfterCallWork` action. Stat Server terminates it, in this example, upon receipt of an `EventAgentReady` TEvent. (The `EventAgentLogout` or `EventAgentNotReady` TEvents with workmode!= `ACW` would also terminate the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

> **Important**
>
> 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.

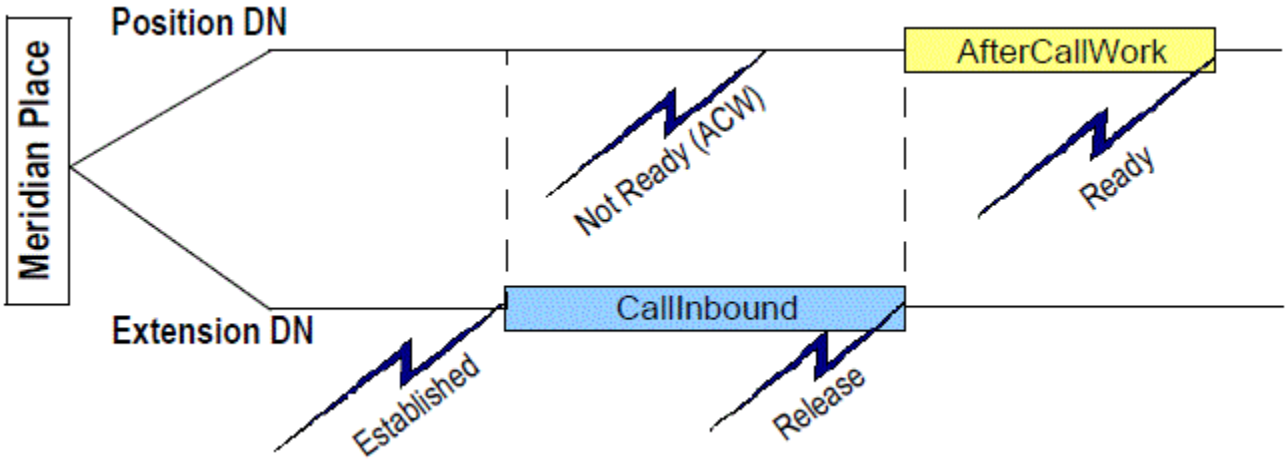
The diagram *ACW Request on a Position DN During a Telephony Interaction* illustrates this scenario when an inbound interaction is underway on the Position DN. An ACW-related event takes place during the interaction. Stat Server starts an `AfterCallWork` action when the interaction is released.



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 `EventAgentReady` TEvent, Stat Server terminates the `AfterCallWork` action. (`EventAgentLogout` and `EventAgentNotReady` with a workmode other than ACW would also terminate the action.)

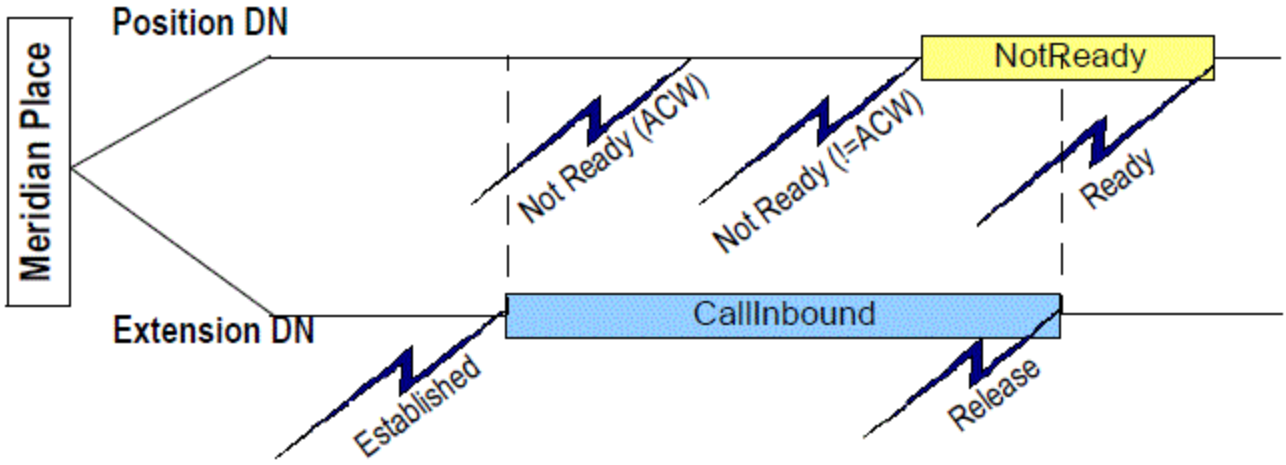
In the diagram *ACW Request on an Extension DN During a Telephony Interaction*, the agent presses the ACW button (workmode=ACW) while conducting an interaction on the Extension DN. Stat Server starts an `AfterCallWork` action *on the Position DN* upon release of the interaction, and terminates it under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



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).



Clearing an ACW Request During an Interaction

The diagram *Clearing an ACW Request During an Interaction* illustrates this scenario. It 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.

Important

This ACW model applies when Stat Server 7.0^{+} is used in conjunction with Meridian T-Server 7.0. Contact Genesys Customer Care to understand Stat Server 7.0 behavior if the version of your Meridian T-Server is less than 7.0.

Back to top

AgentActive

For Stat Server 8.1.0 and lower releases (8.1.0^{-}), 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 that belongs 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. For 8.1.2 and higher releases (8.1.2^{+}), Stat Server generates this action only on DNs on which there is a known agent (the assignment of those DNs to a place is not required).

> **Important**
> In order to use this action in statistics that are accessing action attributes (for example: filters, formulas, `GroupBy` , `DistinguishBy`) set the **[statserver]**/queue-use-pseudo-actions option to `false`.

Back to top

AgentLogin (Regular DNs)

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.

For a description of this action on mediation DNs, see AgentLogin (Mediation DNs).

Back to top

AgentLogin (Mediation DNs)

For Stat Server 8.1.0 and lower (releases $8.1.0^{-}$), this durable action starts when agent logs on to a mediation DN through a regular DN that belongs to a place and for which the agent is known. This action ends when the agent logs out from the mediation DN or when the `NotMonitored` action starts. For 8.1.2 ($8.1.2^{+}$) and higher release, Stat Server generates this action only on DNs for which there is a known agent (the assignment of those DNs to a place is not required).

> **Important**
> In order to use this action in statistics that are accessing action attributes (for example: filters, formulas, `GroupBy` , `DistinguishBy`) set the **[statserver]**/queue-use-pseudo-actions option to `false`.

For a description of this action on regular DNs, see AgentLogin (Regular DNs).

Back to top

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 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.

For 8.1.2 and higher releases (8.1.2^{+}), Stat Server generates this action only on DNs for which there is a known agent.

Back to top

AgentReady

For Stat Server 8.1.0 and lower releases (8.1.0^{-}), 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 Place and Agent Status for a definition of agent status). For 8.1.2 and higher releases (8.1.2^{+}), Stat Server generates this action only on DNs for which there is a known agent (the assignment of those DNs to a place is not required).

> **Important**
>
> In order to use this action in statistics that are accessing action attributes (for example: filters, formulas, GroupBy , DistinguishBy) set the **[statserver]**/queue-use-pseudo-actions option to false.

Back to top

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.

> **Tip**
> Refer to the *Outbound Contact Deployment Guide* for information on the ASM call model.

Back to top

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 agent DN 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.

> **Tip**
> Refer to the *Outbound Contact Deployment Guide* for information on the ASM call model.

Back to top

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.

Back to top

BeingCoached

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

Back to top

BeingMonitored

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

Back to top

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.

Back to top

CallAbandoned

This retrospective action derives from the CallWait durable action 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.

Back to top

CallAbandonedFromDialing

This retrospective action derives from the CallDialing durable action 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.

Back to top

CallAbandonedFromHold

This retrospective action derives from the CallOnHold durable action 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 CallAbandonedFromHold.

Back to top

CallAbandonedFromRinging (Regular DNs)

This retrospective action derives from the CallRinging durable action if CallRinging terminates because of EventReleased or EventAbandoned (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 retrospective CallAbandonedFromRinging (Mediation DNs) action.

Back to top

CallAbandonedFromRinging (Mediation DNs)

For regular interactions, 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).

For hunt-call interactions, this retrospective action occurs at a mediation DN when EventAbandoned is received on that DN, given that EventRinging had been previously received on at least one agent DN, belonging to a hunt group. The resultant action receives as its duration from the moment that the call entered the mediation DN (EventQueued or EventRouteRequest) to the moment when the interaction was abandoned (EventAbandoned).

Back to top

CallAbandonedFromRinging (Virtual Queues)

For virtual queue objects that are controlled by a Multimedia-monitored switch this retrospective action occurs when Stat Server receives from Interaction Server the EventRevoked event with the Abandoned reason.

The duration that Stat Server prescribes to this action is the interval from EventQueued to EventRevoked.

This action is similar to CallAbandonedFromRinging (Mediation DNs) in the telephony model.

Back to top

CallAnswered (Regular DNs)

This retrospective action derives from the CallRinging durable action if CallRinging terminates because of EventEstablished.

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

- CallAnsweredUnknown
- CallAnsweredInternal
- CallAnsweredInbound
- CallAnsweredOutbound
- CallAnsweredConsult

The interaction type that Stat Server receives from T-Server with EventEstablished determines which of the above five actions occurs simultaneously with CallAnswered (Regular DNs).

This action may occur simultaneously with the retrospective mediation DN action CallAnswered (Mediation DNs).

Back to top

CallAnswered (Mediation DNs)

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).

> **Important**
> 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

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

This action may occur simultaneously with the CallAnswered (Regular DNs) retrospective action.

Back to top

CallAnswered (Virtual Queues)

For virtual queue objects that are controlled by a Multimedia-monitored switch this retrospective action occurs when Stat Server receives EventPartyAdded as a result of an agent accepting the interaction. The duration that Stat Server prescribes to this action is the interval from EventQueued to EventPartyAdded.

This action is similar to CallAnswered (Mediation DNs) in the telephony model.

Back to top

CallCleared

Stat Server generates this retrospective action only on a virtual queue. The action derives from the CallWait durable action 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:

- 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 CallCleared.

Back to top

CallConferenceJoined

This momentary action occurs in a conference call at a DN that was added to the conference. CallConferenceJoined derives from:

- EventPartyChanged for a two step conference

- EventEstablished for a single step conference

with an interaction state of Conferenced.

Back to top

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.

Back to top

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.

For CallConferenceOriginated actions that are triggered by the EventPartyChanged TEvent with the CallState attribute set to Conferenced, all attributes (ThisQueue, DNIS, and others) are now taken from this TEvent.

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.

> **Important**
>
> When specified in the MainMask of a stat type, Stat Server ignores DistByConnID 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.

> **Important**
>
> 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.

Back to top

CallConferencePartyAdded

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

Back to top

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`.

Back to top

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`.

`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 to occur.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallConsultCompleted

This retrospective action derives from the `CallConsult` durable action. `CallConsultCompleted` is generated when a consultation call completes.

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

Back to top

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`.

`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.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

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`.

`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.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallConsultStarted

This momentary action occurs whenever the `CallConsult`, `CallConsultOriginated`, or `CallConsultReceived` durable action starts.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallDialConferenced

This retrospective action derives from the CallDialing durable action 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.

Back to top

CallDialTransferred

This retrospective action derives from the CallDialing durable action 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.

Back to top

CallDialed

This retrospective action derives from the CallDialing durable action 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.

Back to top

CallDialing

This durable action starts when Stat Server receives EventDialing from T-Server for a DN. Its corresponding initial momentary action is CallDialingStarted.

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, CallAbandonedFromDialing, CallDialTransferred, or CallDialConferenced.

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.

Back to top

CallDialingStarted

This momentary action occurs whenever the CallDialing durable action 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.

Back to top

CallDistributed

This retrospective action derives from the CallWait durable action 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 EventRouteRequest determines which of the above five actions occurs simultaneously with CallDistributed.

Back to top

CallDistributedToQueue

Stat Server generates this retrospective action on a mediation DN (DN1) if an interaction is distributed from this DN to a second mediation DN (DN2). The duration of this action is equal to the time from receipt of an EventQueued or EventRouteRequest 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 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 CallDistributedToQueue on 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

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

Back to top

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.

Back to top

CallForwarded (Regular DNs)

This retrospective action derives from the CallRinging durable action 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 (Mediation DNs) retrospective action.

Back to top

CallForwarded (Mediation DNs)

For regular interactions, 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).

For hunt-call interactions, Stat Server never generates this action.

Back to top

CallHeld

This momentary action occurs whenever the CallOnHold durable action 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.

Back to top

CallInbound

This durable action starts when Stat Server receives:

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

Its corresponding initial momentary action, upon receipt of EventEstablished, is CallInboundStarted. 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.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallInboundCompleted

This retrospective action derives from the CallInbound durable action. CallInboundCompleted is generated when an inbound interaction completes.

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

Back to top

CallInboundStarted

This momentary action occurs whenever the CallInbound durable action starts.

> **Tip**

> See also DN Actions at Newly Registered DNs.

Back to top

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`.

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

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallInternalCompleted

This retrospective action derives from the `CallInternal` durable action. `CallInternalCompleted` is generated when an internal interaction completes.

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

Back to top

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`.

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

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

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`.

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

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallInternalStarted

This momentary action occurs whenever the `CallInternal` durable action starts.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallMissed

This retrospective action occurs on a mediation DN when `EventReleased` comes after `EventEstablished`. It applies to calls distributed from a source other than the mediation DN, on which the agent is logged in. Action duration is the interval beginning with `EventEstabished` and ending with `EventReleased`.

Action `CallMissed` is not generated on a mediation DN at the time when a call is released on an agent's DN, if at that moment the agent's DN is no longer associated with the mediation DN, either through an agent group or place group.

Back to top

CallObserved...

The `CallObserved...` actions include the following:

- CallObservedUnknown
- CallObservedInternal
- CallObservedInbound
- CallObservedOutbound
- CallObservedConsult

One of these durable actions starts in one the following cases:

- Stat Server receives EventPartyAdded with ThisDNRole equal to Destination and OtherDNRole equal to Observer.
- For ConnID Stat Server receives the EventPartyChanged event with PreviousConnID not equal to ConnID and action CallObserved... existed for the PreviousConnID.

The action terminates in one the following cases:

- Stat Server receives EventPartyDeleted for the agent's DN with OtherDNRole equal to Observer.
- Stat Server receives EventReleased for the interaction.
- For PreviousConnID Stat Server receives the EventPartyChanged event with PreviousConnID not equal to ConnID.
- The NotMonitored action starts.

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

> **Tip**
> 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.

Back to top

CallObserving...

The CallObserving... actions include the following:

- CallObservingUnknown
- CallObservingInternal
- CallObservingInbound
- CallObservingOutbound
- CallObservingConsult

One of these durable actions starts in one the following cases:

- Stat Server receives the EventEstablished event with ThisDNRole equal to RoleObserver and CallState equal to CallStateBridged.
- For ConnID Stat Server receives the EventPartyChanged event with PreviousConnID not equal to ConnID and action CallObserving... existed for the PreviousConnID.

The action terminates in one the following cases:

- Stat Server receives the EventReleased event.
- For PreviousConnID Stat Server receives the EventPartyChanged event with PreviousConnID not equal to ConnID.
- The NotMonitored action starts.

> **Tip**
> Observer actions can be used in statistics with DistByConnID in formulas.

This action was introduced in release 8.5.1.

Back to top

CallOnHold

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

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
- TransferredFromHold
- CallAbandonedFromHold

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.

Back to top

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`.

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

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallOutboundCompleted

This retrospective action derives from the `CallOutbound` durable action. `CallOutboundCompleted` is generated when an outbound interaction completes.

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

Back to top

CallOutboundOriginated

This durable call-related action is started when the `EventEstablished` event is received for a given `ConnID` and there exists the `CallDialing` action for that `ConnID`. This action ends with the `EventReleased` event.

> **Tip**
> The `CallOutboundOriginated` action can be used in statistics with `DistByConnID` in formulas.

This action was introduced in release 8.5.1.

Back to top

CallOutboundReceived

This durable call-related action is started when the EventEstablished event is received for a given ConnID and there exists the CallRinging action for that ConnID. This action ends with the EventReleased event.

> **Tip**
> The CallOutboundReceived action can be used in statistics with DistByConnID in formulas.

This action was introduced in release 8.5.1.

Back to top

CallOutboundStarted

This momentary action occurs whenever the CallOutbound durable action starts.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallPartyChanged

This retrospective action derives from the following:

- The CallConsult, the CallConsultOriginated, or the CallConsultReceived durable actions if any of these actions terminate because of EventPartyChanged.
- The CallInbound action, 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.

Back to top

CallReleased (Mediation DNs)

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.

Back to top

CallReleased (Virtual Queues)

For virtual queue objects that are controlled by a Multimedia-monitored switch this retrospective action occurs when Stat Server receives EventPartyRemoved as a result of an agent finishing an interaction. Stat Server calculates the duration from the moment of acceptance of an interaction (EventPartyAdded) until the moment that the last involved party of the interaction leaves it (EventPartyRemoved).

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 EventRevoked event from Interaction Server.

This action is similar to CallReleased (Mediation DNs) in the telephony model.

Back to top

CallRetrievedFromHold

This retrospective action derives from the CallOnHold durable action 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.

Back to top

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. CallRinging lasts until Stat Server receives:

- EventEstablished

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

Or, until the NotMonitored (Regular DNs) action starts.

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

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.

Back to top

CallRingingPartyChanged (Regular DNs)

This retrospective action derives from the following:

- The CallRinging durable action, if CallRinging terminates because of EventPartyChanged for a consult call.
- The CallRingingConsult action, as CallRingingPartyChanged (Regular DNs) is interaction-type-specific.
- The CallInbound action, 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.

Back to top

CallRingingPartyChanged (Mediation DNs)

CallRingingPartyChanged (Mediation DNs) is a retrospective, interaction-related action reflecting Regular DN actions that Stat Server generates on Mediation DNs. Similar retrospective action is CallRingingPartyChanged (Regular DNs).

Back to top

CallRingingStarted

This momentary action occurs whenever the CallRinging durable action 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.

Back to top

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

Back to top

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. CallTransferPartyChanged derives from EventPartyChanged with an interaction state of Transferred and a ConnID equal to PreviousConnID.

Back to top

CallTransferTaken

This momentary action occurs at the DN when a transfer is made, once the transfer completes (EventEstablished). This action requires one of the following conditions:

- Stat Server receives EventPartyChanged with an interaction state of Transferred and a ConnID different from PreviousConnID.
- Stat Server receives EventPartyChanged for this interaction on some mediation DN prior to distribution

to a regular DN.

- Stat Server receives EventRinging with an interaction state of Transferred. (Refer to the description of the generate-transfer-taken-on-ringing configuration option in the Framework Stat Server Deployment Guide to learn how to control this aspect of CallTransferTaken action generation.)
- Stat Server receives EventRouteRequest with a CallState attribute of OK on a routing point if such event was preceded by EventQueued on the same routing point with a CallState attribute of Transferred.
- Note, that EventQueued will only be handled on a routing point, if the rp-handle-queueing-events configuration option in the [statserver] section has been set to true. (Refer to the option description in the Framework Stat Server Deployment Guide to learn how to control this aspect of CallTransferTaken action generation.)

> **Important**
>
> Stat Server counts transfers that are initiated from an agent's DN and completed on a queue or routepoint as TransferTaken for the agent receiving this call. In 7.x and lower releases, transfers initiated by an IVR were also counted as TransferTaken.

Back to top

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.

> **Important**
>
> Stat Server handles treatment-related events only for Routing Points. In order to generate an appropriate action, a call with ConnID specified in the associated event should currently be waiting on a Routing Point.

Back to top

CallTreatmentNotStarted

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

> **Important**
>
> Stat Server handles treatment-related events only for Routing Points. In order to generate an appropriate action, a call with ConnID specified in the associated event

should currently be waiting on a Routing Point.

Back to top

CallTreatmentStarted

This momentary action occurs when Stat Server receives `EventTreatmentApplied` from T-Server.

> **Important**
> Stat Server handles treatment-related events only for Routing Points. In order to generate an appropriate action, a call with `ConnID` specified in the associated event should currently be waiting on a Routing Point.

Back to top

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.

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

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

CallUnknownCompleted

This retrospective action derives from the CallUnknown durable action. `CallUnknownCompleted` is generated when an unknown interaction completes.

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

Back to top

CallUnknownStarted

This momentary action occurs whenever the CallUnknown durable action starts.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

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).

Its corresponding initial momentary action is CallEntered.

`CallWait` 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.

Back to top

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.

Back to top

CoachingByRequestInitiated

This retrospective action indicates that an agent has begun coaching an interaction and not the initiator of this activity. Stat Server calculates the duration from the moment when coaching is started to the moment when coaching is finished.

This retrospective action is not derived from a durable action.

Back to top

CoachingRequested

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

Back to top

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.

Back to top

ConferenceJoinedByIntrusion

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

Back to top

ConferenceMade

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

Back to top

DND

This durable action starts on RegDN when Stat Server receives:

- The EventRegistered or EventAddressInfo TEvent with the ("dnd",1) pair in the Extensions attribute
- The EventDNDOn TEvent

Previously started DND action ends when Stat Server receives one of the following TEvents:

- EventDNDOff
- EventServerDisconnected
- EventLinkDisconnected
- EventOutOfService
- EventAddressInfo with the ("dnd",0) pair in the Extensions attribute

> **Tip**
>
> - The DND action is ignored in status calculations.
> - The DND action is media-independent (on a multi-media DN there can be only one instance of DND).
> - The DND action is unique on a given DN.

This action was introduced in release 8.5.107.

Back to top

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.

Back to top

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.

Back to top

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.

Back to top

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.

Back to top

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.

The counter that this action designates equals the number of DNs that are currently logged in to the queue when these DNs are in the WaitForNextCall status. That number does not include DN positions for Meridian and Meridian-like switches, for which associated extension DNs are not in WaitForNextCall status. (See Regular DN Status for a definition of DN status.)

Back to top

DoNotDisturb

Durable `DoNotDisturb` action for Agent/Place starts and ends upon relevant events simultaneously for every media-channel of a particular agent and associated by login the corresponding place. At any moment of time only one `DoNotDisturb` action can exist on a specific media-channel for the particular agent or place.

The `DoNotDisturb` action starts when Stat Server receives one of the following Interaction Server events:

- EventDonotDisturbOn
- EventAgentLogin (if the DND flag set to 1)
- EventPlaceAgentState (if the DND flag set to 1)
- EventMediaAdded (if an agent is currently in the DoNotDisturb state)

Previously started `DoNotDisturb` action ends when Stat Server receives one of the following

Interaction Server events:

- EventDonotDisturbOff
- EventAgentLogout

> **Tip**
>
> - An agent can be logged in without logging into any media channel.
> - The `DoNotDisturb` action can be started/ended many times on a particular media-channel during the agent login session.
> - The `DoNotDisturb` action does not participate in the media-channel status calculation. However, if the media-channel status is `Active` or `Available`, Stat Server changes it to `Blocked` upon receiving the `EventDoNotDisturbOn` event from Interaction Server.

This action was introduced in release 8.5.110 .18.

Back to top

ExternalServiceRequested (Tenants)

This momentary action is generated on a Tenant upon receiving the `EventExternalServiceRequested` event.

Mandatory event attribute `attr_actor_tenant_id` specifies the Tenant's dbid.

The `ExternalServiceRequested (Tenants)` action supports the `RequestEnvelope` system attribute, associated with the `attr_esp_request_envelope` event attribute.

> **Important**
>
> The only supported `Subject` for the `ExternalServiceRequested (Tenants)` action is `DNAction`.

This action was introduced in release 8.5.104.

Back to top

ExternalServiceResponded (Tenants)

This momentary action is generated on a Tenant upon receiving the `EventExternalServiceResponded` event.

Mandatory event attribute `attr_actor_tenant_id` specifies the Tenant's dbid.

The ExternalServiceResponded (Tenants) action supports the RequestEnvelope system attribute, associated with the attr_esp_request_envelope event attribute.

> **Important**
> The only supported Subject for the ExternalServiceResponded (Tenants) action is DNAction.

This action was introduced in release 8.5.104.

Back to top

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 EventPartyAdded with attr_party_type = 2 and has no equivalent in the telephony model. This action terminates when the agent leaves the interaction or when the NotMonitored action starts.

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

- HandlingInbound
- HandlingInternal
- HandlingOutbound

The interaction type that Stat Server receives from Interaction Server with EventPartyAdded and attr_party_type = 2 determines which one of the above three actions occurs simultaneously with Handling.

> **Tip**
> Starting with Release 8.5.104, new ApplyFilterAtActionEndOnly stat type option is introduced, which can be used as additional filtering for the Handling action. See example below.

For example.
The stat type:

```
Category=TotalTime
MainMask=InteractionHandling
Objects=Agent, GroupAgents
Subject=DNAction
ApplyFilterAtActionEndOnly=yes
```

applied with filter = PairExists("CustomerSegment", "Gold")

The table below illustrates how the stat type above behaves with and without the ApplyFilterAtActionEndOnly specifier in a specific scenario:

Event	ApplyFilterAtActionEndOnly=no	ApplyFilterAtActionEndOnly=yes
T1: Interaction handling starts, `CustomerSegment=Gold`		
T2: Interaction data changed, `CustomerSegment=Silver`	The value is incremented by (**T2-T1**)	
T3: Interaction data changed, CustomerSegment=Gold		
T4: Interaction handling ends, CustomerSegment=Gold	The value is incremented by (**T4-T3**)	The value is incremented by (**T4-T1**)

Back to top

HandlingInbound

Stat Server generates this durable action, also called `InteractionHandlingInbound`, 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.

Back to top

HandlingInboundStarted

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

Back to top

HandlingInternal

Stat Server generates this durable action, also called `InteractionHandlingInternal`, 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.

Back to top

HandlingInternalStarted

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

Back to top

HandlingOutbound

Stat Server generates this durable action, also called `InteractionHandlingOutbound`, 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.

Back to top

HandlingOutboundStarted

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

Back to top

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`

The interaction type that Stat Server receives from Interaction Server with `EventPartyAdded` where `attr_party_type = 2` determines which one of the above three actions occurs simultaneously with `HandlingStarted`.

Back to top

InteractionAbandoned (Tenants)

This retrospective action is unconditionally generated on a Tenant upon receiving the `EventAbandoned`.

Mandatory event attribute `attr_actor_tenant_id` specifies the Tenant dbid.

Action duration is calculated from the moment when the interaction enters the system (specified by the `attr_itx_received_at` attribute) to the moment specified by the `attr_itx_abandoned_at` attribute. This attribute is initialized when the interaction is abandoned and appears in subsequent events.

As soon as Interaction Server recognizes an interaction as abandoned, it sends `EventAbandoned` to Stat Server, though the abandoned interaction might eventually be handled by an agent.

Interaction Server release 8.5.110.10 (or later) and Chat Media Server release 8.5.201.07 (or later) are required. This action was introduced in release 8.5.108.

Back to top

InteractionAbandonedDuringOffering (MediaChannels)

This retrospective action is unconditionally generated on agent (or place) upon receiving from Interaction Server the `EventRejected` event if the interaction is not accepted by an agent for any reason while offering or the `EventRevoked` event if the interaction is revoked.

Action duration is an interval between the `EventAgentInvited` and `EventRejected`/`EventRevoked` events. This action was introduced in release 8.5.102.

Back to top

InteractionAbandonedDuringOffering (StagingAreas)

This retrospective action is unconditionally generated on a StagingArea (Interaction Queue) upon receiving the `EventRevoked` event if the interaction is revoked or upon receiving the `EventRejected` event if the interaction is not accepted by an agent for any reason.

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue.

Action duration is calculated from the time specified in the `attr_itx_placed_in_queue_at` attribute of the `EventRevoked` or `EventRejected` event to the time when these events occur. This action was introduced in release 8.5.101

> **Important**
>
> Starting with release 8.5.102, introduction of the `InteractionAbandonedDuringOffering` media-action on Agent/Pplace, StagingArea, VirtualQueue and Tenant adds more flexibility on counting all interactions that have been dropped (rejected or revoked) before they were accepted. Filters by reason can be applied to accommodate different conditions.

.

Back to top

InteractionAbandonedDuringOffering (Tenants)

This retrospective action is unconditionally generated on a Tenant upon receiving the `EventRevoked` event if the interaction is revoked or upon receiving the `EventRejected` event if the interaction is not accepted by an agent for any reason.

Mandatory event attribute `attr_itx_tenant_id` specifies the Tenant dbid.

Action duration is calculated from the moment when the interaction enters the system (specified by

the attr_itx_received_at attribute) to the moment when the interaction is revoked or rejected. This action was introduced in release 8.5.102.

Back to top

InteractionAbandonedDuringOffering (Virtual Queues)

For Virtual Queue objects, that are controlled by a Multimedia-monitored switch, the retrospective action InteractionAbandonedDuringOffering is unconditionally generated on an associated virtual queue upon receiving the EventRevoked event if the interaction is revoked or upon receiving the EventRejected event if the interaction is not accepted by an agent for any reason.

A virtual queue, associated with the interaction, is specified by the tracking details initially stored upon receiving the EventCustom event with packed EventQueued TEvent.

Action duration is calculated from the moment when the interaction has entered a virtual queue to the moment when the interaction is revoked or rejected. This action was introduced in release 8.5.102

> **Tip**
> To satisfy the RONA (Revoke On No Answer) requirement a filter by the Expired reason has to be used.

.

Back to top

InteractionAbandonedDuringOffering (Agent Workbin)

This retrospective action is generated on an Agent Workbin upon receiving the EventRevoked or EventRejected event from Interaction Server.

Interaction attributes attr_itx_workbin_type_id and attr_itx_agent_id specify the Agent Workbin object.

Action duration is calculated from the time specified in the attr_itx_placed_in_queue_at attribute of the EventRevoked or EventRejected event to the time when these events occur. This action was introduced in release 8.5.112.10.

Back to top

InteractionAccepted1

This retrospective action is generated on an Agent and a Place independently upon receiving the EventPartyAdded event when the interaction is accepted for the very first time. Therefore, this action is generated only when the mandatory attribute attr_itx_delivered_at is equal to NULL.

Action duration is calculated from the moment when the interaction enters the system (specified by

the attr_itx_received_at attribute) to the moment when the interaction is accepted for handling.

To apply this action to the Tenant, a propagation mechanism is used. A request from the client has to include a name of the Tenant object, but the object type needs to be GroupAgents or GroupPlaces.

> **Tip**
> The InteractionAccepted1 action is different from the Accepted (InteractionAccepted) action, which is also generated on agent/place objects.

This action was introduced in release 8.5.102.

Back to top

InteractionAccepted (StagingAreas)

This retrospective action is generated on a StagingArea (Interaction Queue) upon receiving the EventPartyAdded event when an interaction is accepted by an agent for the first time. It is generated only ones for any interaction.

Mandatory event attribute attr_itx_queue specifies the Interaction Queue.

Interaction related attribute attr_itx_delivered_at provides the time stamp when an interaction is accepted for the first time.

The InteractionAccepted action is generated only if the attr_itx_delivered_at attribute is equal to NULL. **Note:** The attr_itx_delivered_at attribute provides the time stamp when an interaction was accepted for the first time.

Action duration is calculated from the moment when the interaction enters the system (specified by the attr_itx_received_at attribute) to the moment when the interaction is accepted by an agent for handling. This action was introduced in release 8.5.101.

Back to top

InteractionAccepted (Tenants)

This retrospective action is generated on a Tenant upon receiving the EventPartyAdded event when an interaction is accepted by an agent for the first time. It is generated only ones for any interaction.

Mandatory event attribute attr_itx_tenant_id specifies the Tenant's dbid.

Interaction related attribute attr_itx_delivered_at provides the time stamp when an interaction is accepted for the first time.

The InteractionAccepted action is generated only if the attr_itx_delivered_at attribute is equal to NULL. **Note:** The attr_itx_delivered_at attribute provides the time stamp when an interaction was accepted for the first time.

Action duration is calculated from the moment when the interaction enters the system (specified by

the attr_itx_received_at attribute) to the moment when the interaction is accepted by an agent for handling. This action was introduced in release 8.5.103.

Back to top

InteractionAccepted (Agent Workbin)

This retrospective action is generated on an Agent Workbin upon receiving the EventPartyAdded event when an interaction is accepted by an agent for the first time. It is generated only once for any interaction based on the attribute attr_itx_delivered_at equal to NULL.

Interaction attributes attr_itx_workbin_type_id and attr_itx_agent_id specify the Agent Workbin object.

Action duration is calculated from the moment when the interaction enters the system (specified by the attr_itx_received_at attribute) to the moment when the interaction is accepted by an agent for handling. This action was introduced in release 8.5.112.10.

Back to top

InteractionAgentPartyInProgress (Tenants)

This durable action is started on a Tenant when an interaction is initially associated with an agent(s). This action ends when there are no agents associated with the interaction.

The InteractionAgentPartyInProgress action can be started and stopped several times for the same interaction, but at any particular time there can be only one InteractionAgentPartyInProgress action with the specific InteractionID.

The InteractionAgentPartyInProgress action starts when Stat Server receives one of the following Interaction Server events:

- EventAgentInvited
- EventPartyAdded

Previously started InteractionAgentPartyInProgress action ends when Stat Server receives one of the following Interaction Server events:

- EventRejected
- EventRevoked
- EventPartyRemoved
- EventProcessingStopped

The following options are available in Stat Server to control memory usage, associated with the InteractionAgentPartyInProgress action:

- **[statserver]**/interaction-agent-party-in-progress-on-tenant-max-number
- **[statserver]**/interaction-agent-party-in-progress-on-tenant-media-list

This action was introduced in release 8.5.110.18.

Back to top

InteractionAnswered (StagingAreas)

This retrospective action is generated on a StagingArea (Interaction Queue) upon receiving the `EventPartyAdded` event when an interaction is accepted by an agent.

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue.

Action duration is calculated from the moment when the interaction is placed in the Interaction Queue (specified by the `attr_itx_placed_in_queue_at` attribute) to the moment when the interaction is accepted by an agent for handling. This action was introduced in release 8.5.101.

Back to top

InteractionAnswered (Tenants)

This retrospective action is generated on a Tenant upon receiving the `EventPartyAdded` event when an interaction is accepted by an agent.

Mandatory event attribute `attr_itx_tenant_id` specifies the Tenant's dbid.

Action duration is calculated from the moment when the interaction is placed in the Interaction Queue (specified by the `attr_itx_placed_in_queue_at` attribute) to the moment when the interaction is accepted by an agent for handling. This action was introduced in release 8.5.103.

Back to top

InteractionAnswered (Agent Workbin)

This retrospective action is generated on an Agent Workbin upon receiving the `EventPartyAdded` event when an interaction is accepted by an agent.

Interaction attributes `attr_itx_workbin_type_id` and `attr_itx_agent_id` specify the Agent Workbin object.

Action duration is calculated from the moment when the interaction is placed in the Agent Workbin to the moment when the interaction is accepted by an agent for handling. This action was introduced in release 8.5.112.10.

Back to top

InteractionCleared

This retrospective action is generated on a StagingArea (Interaction Queue) upon receiving the `EventTakenFromQueue` event, when the interaction is diverted from the specific Interaction Queue to be processed not by an agent based on Actor information (`attr_party_type != 2`) provided in the event. **Note:** At this point an association between the Interaction Queue and the interaction still exists.

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue.

Action duration is calculated from the moment when the interaction is placed in the Interaction Queue (specified by the `attr_itx_placed_in_queue_at` attribute) to the moment when the interaction leaves the Interaction Queue. This action was introduced in release 8.5.1.

Back to top

InteractionCleared (Agent Workbin)

This retrospective action is generated on an Agent Workbin upon receiving the `EventTakenFromWorkbin` event, when the interaction is diverted not by an agent as specified by event attribute `attr_party_type != 2`.

Interaction attributes `attr_itx_workbin_type_id` and `attr_itx_agent_id` specify the Agent Workbin object.

Action duration is calculated from the moment when the interaction is placed into the Agent Workbin to the moment when the interaction leaves it. This action was introduced in release 8.5.112.10.

Back to top

InteractionCreated (StagingAreas)

This momentary action is generated on a StagingArea (Interaction Queue) upon receiving the `EventInteractionSubmitted` event, when a new interaction is submitted to the Interaction Queue.

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue.

> **Important**
>
> Stat Server option `subscribe-for-all-ixn-server-events` must be set to `true` for this event to be sent by Interaction Server.

This action was introduced in release 8.5.100.

Back to top

InteractionCreated (Tenants)

This momentary action is generated on a Tenant upon receiving the `EventInteractionSubmitted` event, when a new interaction is received by the system.

Mandatory event attribute `attr_itx_tenant_id` specifies the Tenant's dbid.

> **Important**
> Stat Server option `subscribe-for-all-ixn-server-events` must be set to `true` for this event to be sent by Interaction Server.

This action was introduced in release 8.5.102.

Back to top

InteractionDeleted (StagingAreas)

This retrospective action is generated on a StagingArea (Interaction Queue) upon receiving the `EventProcessingStopped` event, when the interaction is finished and no longer exists in the system.

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue.

Action duration is calculated from the moment of accepting the interaction by the system (specified by the `attr_itx_received_at` event attribute) to the moment when the interaction handling is finished by the system. This action was introduced in release 8.5.100.

Back to top

InteractionDeleted (Tenants)

This retrospective action is generated on a Tenant upon receiving the `EventProcessingStopped` event, when the interaction is finished and no longer exists in the system.

Mandatory event attribute `attr_itx_tenant_id` specifies the Tenant's dbid.

Action duration is calculated from the moment when the interaction is accepted by the system (specified by the `attr_itx_received_at` event attribute) to the moment when the handling of this interaction is finished by the system. This action was introduced in release 8.5.102.

Back to top

InteractionDistributed

This retrospective action is generated on a StagingArea (Interaction Queue) upon receiving the `EventTakenFromQueue` event, when the interaction is diverted from the specific Interaction Queue to be processed by an agent based on Actor information (`attr_party_type = 2`) provided in the event.
Note: At this point an association between the Interaction Queue and the interaction still exists.

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue.

Action duration is calculated from the moment when the interaction is placed in the Interaction Queue (specified by the `attr_itx_placed_in_queue_at` event attribute) to the moment when the interaction leaves the Interaction Queue. This action was introduced in release 8.5.1.

Back to top

InteractionDistributed (Agent Workbin)

This retrospective action is generated on an Agent Workbin upon receiving the `EventTakenFromWorkbin` event, when the interaction is diverted by an agent as specified by the event attribute `attr_party_type = 2`.

Interaction attributes `attr_itx_workbin_type_id` and `attr_itx_agent_id` specify the Agent Workbin object.

Action duration is calculated from the moment when the interaction is placed into the Agent Workbin to the moment when the interaction leaves it. This action was introduced in release 8.5.112.10.

Back to top

InteractionDistributedToQueue

This retrospective action is generated on a StagingArea (Interaction Queue1 specified with the `attr_old_queue` attribute) upon receiving the `EventPlacedInQueue` event on an Interaction Queue2 (specified with the `attr_itx_queue` attribute) when the interaction is entered into the Interaction Queue2 from the Interaction Queue1.

The mandatory `attr_old_queue` event attribute specifies the Interaction Queue from where the interaction has been moved and must be different from the mandatory `attr_itx_queue` event attribute.

Action duration is calculated from the moment when the interaction was placed in an Interaction Queue1 (specified by the attribute `attr_itx_placed_in_queue_at` of the `EventPlacedInQueue` event on an Interaction Queue2) to the moment when the interaction was placed in the Interaction Queue2.

> **Important**
>
> Interaction Server v. 8.5.105.00 or later is required for calculation of the `InteractionDistributedToQueue` action duration.

This action was introduced in release 8.5.1.

Back to top

InteractionEntered

This momentary action is generated on a StagingArea (Interaction Queue) upon receiving the `EventPlacedInQueue` event or the `EventSubmitted` event with `attr_itx_state = 0`, when the interaction is entered into the specific Interaction Queue.

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue. This action was introduced in release 8.5.1.

Back to top

InteractionEntered (Agent Workbin)

This momentary action is generated on an Agent Workbin upon receiving the `EventPlacedInWorkbin` event from Interaction Server.

Interaction attributes `attr_itx_workbin_type_id` and `attr_itx_agent_id` specify the Agent Workbin object. This action was introduced in release 8.5.112.10.

Back to top

InteractionPastAcceptServiceLevel (StagingAreas)

This retrospective action is generated on a StagingArea (Interaction Queue) upon receiving the `EventProcessingStopped` event with `UserData` containing the `ServiceObjective` key. A value of the `ServiceObjective` key provides expected duration for an interaction to be accepted for handling.

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue.

The `InteractionPastAcceptServiceLevel` action is generated in the following cases:

- For answered interactions - when the time interval `t1` between the moment of when the interaction is entering the system (specified by the `attr_itx_received_at` attribute) to the moment of the interaction acceptance by an agent (specified by the `attr_itx_delivered_at` attribute) is greater than the value of the `ServiceObjective` key.
- For the interactions without acceptance, such as revoked by customer (the `attr_itx_delivered_at` attribute is not available) - when the time interval `t2` between the moment of when the interaction is entering the system (specified by the `attr_itx_received_at` attribute) to the moment when the interaction is stopped is greater than the value of the `ServiceObjective` key.

For `StagingArea` objects, Stat Server generates this action only for the last StagingArea (Interaction Queue) that is associated with the interaction.

The actual duration of the `InteractionPastAcceptServiceLevel` action is calculated as a difference between `t1` or `t2` and the expected duration, specified by the `ServiceObjective` key. This action was introduced in release 8.5.102.

Back to top

InteractionPastAcceptServiceLevel (Tenants)

This retrospective action is generated on a Tenant upon receiving the `EventProcessingStopped` event with `UserData` containing the `ServiceObjective` key. A value of the `ServiceObjective` key provides expected duration for an interaction to be accepted for handling.

Mandatory event attribute `attr_itx_tenant_id` specifies the Tenant's dbid.

The `InteractionPastAcceptServiceLevel` action is generated in the following cases:

- For answered interactions - when the time interval `t1` between the moment of when the interaction is entering the system (specified by the `attr_itx_received_at` attribute) to the moment of the interaction acceptance by an agent (specified by the `attr_itx_delivered_at` attribute) is greater than the value of the `ServiceObjective` key.

- For the interactions without acceptance, such as revoked by customer (the attr_itx_delivered_at attribute is not available) - when the time interval t2 between the moment of when the interaction is entering the system (specified by the attr_itx_received_at attribute) to the moment when the interaction is stopped is greater than the value of the ServiceObjective key.

The actual duration of the InteractionPastAcceptServiceLevel action is calculated as a difference between t1 or t2 and the expected duration, specified by the ServiceObjective key. This action was introduced in release 8.5.102.

Back to top

InteractionPastCompletionServiceLevel (StagingAreas)

This retrospective action is generated on a StagingArea (Interaction Queue) upon receiving the EventProcessingStopped event with UserData containing the CompleteServiceObjective key. A value of the CompleteServiceObjective key provides expected duration to complete interaction handling.

Mandatory event attribute attr_itx_queue specifies the Interaction Queue.

The t1 time interval is the interval from the moment when the interaction is accepted by the system (specified by the attr_itx_received_at attribute) to the moment when the interaction handling is finished by the system.

The InteractionPastCompletionServiceLevel action is generated only if t1 is greater than the value of the CompleteServiceObjective key.

For StagingArea objects, Stat Server generates this action only for the last StagingArea (Interaction Queue) that is associated with the interaction.

The action duration is calculated as a difference between the t1 time interval and the expected duration, specified by the CompleteServiceObjective key. This action was introduced in release 8.5.102.

Back to top

InteractionPastCompletionServiceLevel (Tenants)

This retrospective action is generated on a Tenant upon receiving the EventProcessingStopped event with UserData containing the CompleteServiceObjective key. A value of the CompleteServiceObjective key provides expected duration to complete interaction handling.

Mandatory event attribute attr_itx_tenant_id specifies the Tenant's dbid.

The t1 time interval is the interval from the moment when the interaction is accepted by the system (specified by the attr_itx_received_at attribute) to the moment when the interaction handling is finished by the system.

The InteractionPastCompletionServiceLevel action is generated only if t1 is greater than the value of the CompleteServiceObjective key.

The action duration is calculated as a difference between the t1 time interval and the expected duration, specified by the CompleteServiceObjective key. This action was introduced in release

8.5.102.

Back to top

InteractionReleased (StagingAreas)

This retrospective action is generated on a StagingArea (Interaction Queue) upon receiving the `EventPartyRemoved` event.

Mandatory event attribute `attr_old_queue` specifies the Interaction Queue.

Action duration is calculated from the moment when the interaction is accepted by an agent (specified by the `attr_itx_assigned_at` attribute) to the moment when this agent no longer handles the interaction (**Note:** The interaction can still be alive).

> ### Important
> Interaction Server v. 8.5.105.00 or later is required for calculation of the `InteractionReleased` action duration.

This action was introduced in release 8.5.101.

Back to top

InteractionReleased (Tenants)

This retrospective action is generated on a Tenant upon receiving the `EventPartyRemoved` event.

Mandatory event attribute `attr_itx_tenant_id` specifies the Tenant's dbid.

Action duration is calculated from the moment when the interaction is accepted by an agent (specified by the `attr_itx_assigned_at` attribute) to the moment when this agent no longer handles the interaction (**Note:** The interaction can still be alive).

> ### Important
> Interaction Server v. 8.5.105.00 or later is required for calculation of the `InteractionReleased` action duration.

This action was introduced in release 8.5.103.

Back to top

InteractionReleased (Agent Workbin)

This retrospective action is generated on an Agent Workbin upon receiving the `EventPartyRemoved` event from Interaction Server.

Interaction attributes `attr_itx_workbin_type_id` and `attr_itx_agent_id` specify the Agent Workbin object.

Action duration is calculated from the moment when the interaction is accepted by an agent (specified by the `attr_itx_assigned_at` attribute) to the moment when this agent no longer handles the interaction (the interaction can still exist). This action was introduced in release 8.5.112.10.

Back to top

InteractionResponded (Regular DNs)

This instant action is generated on a regular DN upon receiving the `EventReleased` TEvent with the following event attributes:

- `CallType != Outbound`
- `ThisDNRole = Destination` or `ThisDNRole = ConferenceMember`
- `CallState = Ok` or `CallState = Transferred`.

This action was introduced in release 8.5.105.

Back to top

InteractionResponded (MediaChannels)

This instant action is generated on an agent (or place) upon receiving the `EventPartyRemoved` event for:

- Any interaction of the `Inbound` or `Internal` type
- Any interaction of the `Outbound` type with the `attr_itx_subtype = OutboundReply`

> **Tip**
> This action is applicable to any agent involved in an interaction of the `Internal` type.

This action was introduced in release 8.5.105.

Back to top

InteractionWait (StagingAreas)

The `InteractionWait (StagingAreas)` durable action starts on a StagingArea when Stat Server receives the following events:

- `EventPlacedInQueue`
- `EventSubmitted` with `attr_itx_state=0`

The `InteractionWait (StagingAreas)` action ends upon the following events:

- `EventTakenFromQueue`
- `EventProcessingStopped`

Mandatory event attribute `attr_itx_queue` specifies the Interaction Queue.

The `interaction-wait-on-sa-max-number` configuration option controls the number of `InteractionWait (StagingAreas)` actions. When the threshold is reached, Stat Server does not create any more instances of the `InteractionWait (StagingAreas)` action, until the number of `InteractionWait (StagingAreas)` actions is below the threshold.

Upon a disconnect from Interaction Server all active `InteractionWait (StagingAreas)` actions across all StagingArea(s) end, which is reflected in the Stat Server log by a single message. This action was introduced in release 8.5.103.

Back to top

InteractionWait (Tenants)

The `InteractionWait (Tenants)` action is tracking an interaction until it is handled by an agent. This action can start and end multiple times for the same interaction.

This durable action starts when Stat Server receives one of the following Interaction Server events:

- EventInteractionSubmitted
- EventPlaceAgentState for a delivering interaction only with the `'Inviting' [int] = 1` attribute
- EventAgentInvited
- EventPartyRemoved with the `attr_operation` not equal to STOP (value 10), `attr_actor_type=AGENT` (value 2), and an agent is the last participant who is leaving the interaction

> **Tip**
> The value of the `attr_operation` attribute, different from STOP, means that an interaction is not finished and intended to be processed further by the system.

Previously started `InteractionWait (Tenants)` action ends when Stat Server receives one of the following Interaction Server events:

- EventProcessingStopped
- EventPartyAdded with the `attr_actor_type=AGENT` (value 2)

Mandatory event attribute `attr_actor_tenant_id` specifies the Tenant.

Upon disconnect from Interaction Server all active `InteractionWait` actions across all monitored Tenants are ended and a single message is printed in the Stat Server log. This action was introduced in release 8.5.107.

Back to top

InteractionWait (Agent Workbin)

The `InteractionWait (Agent Workbin)` durable action starts on an Agent Workbin when Stat Server receives the `EventPlacedInWorkbin` event from Interaction Server.

The `InteractionWait (Agent Workbin)` action ends upon the following events:

- `EventTakenFromWorkbin`
- `EventProcessingStopped`

Interaction attributes `attr_itx_workbin_type_id` and `attr_itx_agent_id` specify the Agent Workbin object.

Upon a disconnect from Interaction Server all active `InteractionWait (Agent Workbin)` actions across all Agent Workbins end, which is reflected in the Stat Server log by a single message. This action was introduced in release 8.5.112.10.

Back to top

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.

Back to top

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.

> **Tip**
> See also DN Actions at Newly Registered DNs.

Back to top

Monitored (Regular DNs)

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.

Back to top

Monitored (Mediation DNs)

`Monitored (Mediation DNs)` is a durable, non–interaction-related actions that Stat Server generates to mediation DNs.

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

For Stat Server 8.5.0 and higher releases (8.5.0^{+}), `Monitored (Mediation DNs)` durable action is generated to a Virtual Queue on the multimedia switch of the tenant if and only if there is at least one connected Interaction Server for that tenant. Otherwise, durable action NotMonitored (Mediation DNs) is generated.

Back to top

MonitoringInitiated

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

Back to top

NotMonitored (Regular DNs)

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 (Regular DNs) —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 `NotMonitored` action occurs, and it persists until

Stat Server detects EventBackInService for that DN. At that point, the NotMonitored action ceases.

Back to top

NotMonitored (Mediation DNs)

For Stat Server 8.1.2 and lower releases (8.1.2^{-}), 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 (Mediation DNs) ends when both connections are up.

Stat Server 8.5.0 and higher releases (8.5.0^{+}) generates this durable action on a Virtual Queue on the multimedia switch of the tenant if no Interaction Servers that are serving that tenant are connected to the Stat Server.

Its complementary action is Monitored (Mediation DNs). One and only one of these actions occurs for any DN at any moment. NotMonitored (Mediation DNs) terminates every other DN action; no other action can start while NotMonitored (Mediation DNs) is occurring.

Back to top

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.

The UserData, Reasons, and Extensions attributes from the EventDNDOn or EventDNDOff TEvents are not inherited by this action.

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

Important

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.

Back to top

NotRoutable

This durable action is generated on an agent or place for a particular `MediaType` if the agent/place has DNs/media channels for this `MediaType` and the agent/place capacity does not allow routing to this agent/place for this `MediaType`. As soon as one of these conditions is not true, the `NotRoutable` action stops.

The `NotRoutable` action is generated on agent/place itself (not inherited from underlying DNs/media-channels) and then is propagated from an agent to the agent group or from a place to the place group.

The `NotRoutable` action is always associated with a `MediaType` and one `NotRoutable` action for a particular `MediaType` may exist on an agent/place. For a particular `MediaType`, the `NotRoutable` action is mutually exclusive with the `Routable` action.

Only `NotRoutable` and `Routable` actions support `current_number`, `max_number`, `media_state` and `routable` system attributes.

Important

- The `NotRoutable` action can only be used with `Subject=DNAction`.
- The `NotRoutable` action does not affect the status of an agent or place.
- To specify a `MediaType` Genesys recommends to use a filter with the corresponding system attribute, for example, `PairExists( System, "MediaType", "chat")`.

This action was introduced in release 8.5.104.

Back to top

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.

> **Important**
> 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.

Back to top

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.

> **Important**
> 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.

Back to top

OrigDNCallAbandoned

This agent group and place group action occurs at the same time as a CallAbandoned action, 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 a retrospective group action reflecting origination DNs that Stat Server generates to agent and place groups.

OrigDNCallAbandoned is always simultaneous with one of the following calltype 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.

Back to top

OrigDNCallDistributed

This agent group and place group action occurs at the same time as a CallDistributed action, 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 a retrospective group action reflecting origination DNs that Stat Server generates to agent and place groups.

`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`.

Back to top

OrigDNCallEntered

This agent group and place group action occurs at the same time as a CallEntered action, 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 a momentary group action reflecting origination DNs that Stat Server generates to agent and place groups.

`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 `EventRouteRequest` determines which of the above five actions occurs simultaneously with `OrigDNCallEntered`.

Back to top

OrigDNCallWait

This agent group and place group action starts and ends at the same time as a CallWait action, 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.

You can list origination DNs on the Advanced tab of the Properties dialog box of an Agent Group or Place Group object. 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 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.

Back to top

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.

Back to top

Rejected

This retrospective action, also called InteractionRejected, is generated on an agent (or place) upon receiving the EventRejected event from Interaction Server and indicates that an agent has rejected the delivered interaction. This action terminates Delivering actions, and it is similar to the CallAbandonedFromRinging action in the telephony model.

Action duration is an interval between EventAgentInvited and EventRejected.

> **Tip**
> The `CallAbandonedFromRinging` action is a legacy alias for the `Rejected` action.

Back to top

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.

Back to top

Routable

This durable action is generated on an agent or place for a particular `MediaType` if the agent/place has DNs/media channels for this `MediaType` and the agent/place capacity allows routing to this agent/place for this `MediaType`. As soon as one of these conditions is not true, the `Routable` action stops.

The `Routable` action is generated on agent/place itself (not inherited from underlying DNs/media-channels) and then is propagated from an agent to the agent group or from a place to the place group.

The `Routable` action is always associated with a `MediaType` and one `Routable` action for a particular `MediaType` may exist on an agent/place. For a particular `MediaType`, the `Routable` action is mutually exclusive with the `NotRoutable` action.

Only `Routable` and `NotRoutable` actions support `current_number`, `max_number`, `media_state` and `routable` system attributes.

> **Important**
> - The `Routable` action can only be used with `Subject=DNAction`.
> - The `Routable` action does not affect the status of an agent or place.
> - To specify a `MediaType` Genesys recommends to use a filter with the corresponding system attribute, for example, `PairExists( System, "MediaType", "chat")`.

This action was introduced in release 8.5.104.

Back to top

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.

Back to top

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.

> **Important**
> There is no such `StartedInbound` action that Stat Server generates.

Back to top

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.

Back to top

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.

Back to top

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.

> **Important**
> Actions `InteractionStoppedInbound`, `InteractionStoppedInternal`, and `InteractionStoppedOutbound` are not generated upon receiving the `EventProcessingStopped` event if before stopping the interaction an agent was logged out of the media (indicated by `EventMediaRemoved` event).

Back to top

StuckCallCleaned

This retrospective action occurs at a mediation DN and derives from the CallWait durable action 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`.

Back to top

StuckCallCleanedWhileRinging (Regular DNs)

This retrospective action derives from the CallRinging durable action 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.

`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 `EventAbandoned` (with `AttributeReliability!=TReliabilityOk`) determines which of the above five actions occurs simultaneously with `StuckCallCleanedWhileRinging`.

Back to top

StuckCallCleanedWhileRinging (Mediation DNs)

This retrospective action derives from the CallRinging durable action 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 EventRouteRequest) to the moment when Stat Server receives the EventAbandoned TEvent (with AttributeReliability!=TReliabilityOk). This action's corresponding initial momentary action is CallRingingStarted.

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 EventReleased (with AttributeReliability!=TReliabilityOk) determines which of the above five actions occurs simultaneously with CallRetrievedFromHold.

Back to top

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 CallTransferMade 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.

Back to top

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.

Back to top

TransferredFromHold

This retrospective action derives from the CallOnHold durable action 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.

Back to top

UserEvent (Regular DNs)

The EventUserEvent TEvent triggers this momentary, instantaneous action.

Starting with Stat Server release 8.5.103, the UserEvent (Regular DNs) action inherits GlobalUserData, Reasons and Extensions key-value lists from the EventUserEvent TEvent.

Back to top

UserEvent (Mediation DNs)

The EventUserEvent TEvent triggers the UserEvent momentary, instantaneous 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.

Starting with Stat Server release 8.5.103, the UserEvent (Mediation DNs) action inherits GlobalUserData, Reasons and Extensions key-value lists from the EventUserEvent TEvent. This action was introduced in release 8.5.0.

Back to top

UserEventReceived (Regular DNs)

The EventUserEvent TEvent triggers the UserEventReceived (Regular DNs) durable action.

Upon receiving the EventUserEvent TEvent on an object, specified by the ThisDN attribute, the following occurs:

- If there is an UserEventReceived (Regular DNs) action in progress, that action is ended.
- New action UserEventReceived (Regular DNs) is started, inheriting GlobalUserData, Reasons and Extensions key-value lists from the event.

Important

- The `EventUserEvent (Regular DNs)` durable action is not used in status calculations on a DN/Queue.
- The only supported `Subject` for the `UserEventReceived (Regular DNs)` action is `DNAction`.
- `UserData` key-value list cannot be used in filters/formulas, that are applied to the `UserEventReceived (Regular DNs)` action.

This action was introduced in release 8.5.103.

Back to top

UserEventReceived (Mediation DNs)

The `EventUserEvent` TEvent triggers the `UserEventReceived (Mediation DNs)` durable action.

Upon receiving the `EventUserEvent` TEvent on an object, specified by the `ThisDN` attribute, the following occurs:

- If there is an `UserEventReceived (Mediation DNs)` action in progress, that action is ended.
- New action `UserEventReceived (Mediation DNs)` is started, inheriting `GlobalUserData`, `Reasons` and `Extensions` key-value lists from the event.

Important

- The `EventUserEvent (Mediation DNs)` durable action is not used in status calculations on a DN/Queue.
- The only supported `Subject` for the `UserEventReceived (Mediation DNs)` action is `DNAction`.
- `UserData` key-value list cannot be used in filters/formulas, that are applied to the `UserEventReceived (Mediation DNs)` action.

This action was introduced in release 8.5.103.

Back to top

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 EventAgentReady:
  - 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 or Voice Treatment Port, for which the WaitForNextCall action starts as soon as the DN is registered.

> **Tip**
> See also DN Actions at Newly Registered DNs.

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.

The UserData, Reasons, and Extensions attributes from the EventDNDOn or EventDNDOff TEvents are not inherited by this action.

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

> **Important**
> 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.

Back to top

Stat Server Actions for Regular DNs

Durable, Non–Interaction-Related Actions

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

- AfterCallWork
- CallOutboundOriginated
- CallOutboundReceived
- LoggedIn
- LoggedOut
- Monitored (Regular DNs)
- NotMonitored (Regular DNs)
- NotReadyForNextCall
- OffHook
- OnHook
- UserEventReceived (Regular DNs)
- WaitForNextCall

> **Tip**
> `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.

Durable, Interaction-Related Actions

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

- AfterCallWork
- ASM_Engaged
- ASM_Outbound
- CallConferenceOriginated
- CallConsult
- CallConsultOriginated
- CallConsultReceived

- CallDialing
- CallInbound
- CallInternal
- CallInternalOriginated
- CallInternalReceived
- CallObserved...
- CallObserving...
- CallOnHold
- CallOutbound
- CallRinging
- CallUnknown
- DND

> **Tip**
> `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.

Retrospective, Interaction-Related Actions

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

- CallAbandonedFromDialing
- CallAbandonedFromHold
- CallAbandonedFromRinging (Regular DNs)
- CallAnswered (Regular DNs)
- CallConsultCompleted
- CallDialConferenced
- CallDialTransferred
- CallDialed
- CallForwarded (Regular DNs)
- CallInboundCompleted
- CallInternalCompleted
- CallOutboundCompleted
- CallPartyChanged
- CallRetrievedFromHold

- CallRingingPartyChanged (Regular DNs)
- CallUnknownCompleted
- StuckCallCleanedWhileRinging (Regular DNs)
- TransferredFromHold

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

Momentary, Interaction-Related Actions

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

- CallConferenceJoined
- CallConferenceMade
- CallConferencePartyAdded
- CallConferencePartyDeleted
- CallConsultStarted
- CallDialingStarted
- CallHeld
- CallInboundStarted
- CallInternalStarted
- CallOutboundStarted
- CallRingingStarted
- CallTransferMade
- CallTransferPartyChanged
- CallTransferTaken
- CallUnknownStarted

Instantaneous, Non–Interaction-Related Actions

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

- AgentLogout
- AgentLogin (Regular DNs)
- InteractionResponded (Regular DNs)
- UserEvent (Regular DNs)

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.

Durable Group Actions Reflecting Origination DNs

You can list origination DNs on the **Origination DNs** tab of an Agent Group or Place Group using Genesys Administrator or GAX. 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 on agent and place groups.

Retrospective Group Actions Reflecting Origination DNs

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

OrigDNCallAbandoned
OrigDNCallDistributed

Momentary Group Action Reflecting Origination DNs

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

Stat Server Actions for Mediation DNs

> **Tip**
> 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 on mediation DNs:

- AgentActive
- AgentLogin (Mediation DNs)
- AgentReady
- DNActive
- DNLogin
- DNReady
- Monitored (Mediation DNs)
- NotMonitored (Mediation DNs)
- UserEventReceived (Mediation DNs)

> **Tip**
> The `AgentID` system attribute is supported on the `AgentLogin`, `AgentReady`, and `AgentActive` mediation DN actions when the queue-use-pseudo-actions configuration option is set to `false`.

Durable, Interaction-Related Actions

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

Momentary, Interaction-Related Actions

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

- CallEntered
- CallTreatmentNotStarted

- CallTreatmentStarted

Momentary, Non-Interaction-Related Action

UserEvent (Mediation DNs) is a momentary, non–interaction-related action that Stat Server generates on mediation DNs.

Retrospective, Interaction-Related Actions

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

- CallAbandoned
- CallCleared
- CallDistributed
- CallTreatmentCompleted
- StuckCallCleaned

Retrospective, Interaction-Related Actions Reflecting Regular DNs

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

- ACWCompleted
- ACWMissed
- CallAbandonedFromRinging (Mediation DNs)
- CallAnswered (Mediation DNs)
- CallForwarded
- CallMissed
- CallReleased (Mediation DNs)
- 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 mediation DN in Genesys Administrator Extension, as follows:

1. Select an agent (or place) group.
2. On the **Origination DNs** tab, click **Add**.
3. In the **Origination DN** dialog box, select the appropriate mediation DN.
4. On the **Origination DNs** tab, click **Save** or **Apply** 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.

Warning

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. The actions that result for other mediation DNs along the path will reflect call diversion.

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.

Retrospective, Interaction-Related Actions Generated on Virtual Queues

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

- CallAbandonedFromRinging (Virtual Queues)
- CallAnswered (Virtual Queues)
- CallReleased (Virtual Queues)
- InteractionAbandonedDuringOffering (Virtual Queues)

Important

- All mediation DN actions can be potentially generated for virtual queues, controlled by the Multimedia-monitored switch except voice actions listed below:
  - CallTreatmentStarted
  - CallTreatmentNotStarted
  - CallTreatmentCompleted
  - ACWCompleted
  - ACWMissed

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

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 the `CallAnswered` action on 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. In this case, the rules of `CallAnswered` action generation depend on settings of the **vq-treat-unknown-third-party-dn-asagent-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 the routing point associated with the 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 the DN to which the call was diverted.

[+] Scenarios

The Table below (Stat Server Generates CallAnswered) describes how Stat Server behaves given the setting of the **vq-treatunknown-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. The Table below shows whether Stat Server will generate a `CallAnswered` action given the above three scenarios:

Stat Server Generates CallAnswered

Scenario	CallAnswered Generated
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 a call for multi-site routing, you have to take into account the specifics in `CallAnswered` generation in case you are using settings **vq-treatunknown-third-party-dn-as-agent-dn** in the 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.

> **Tip**
> 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.

> **Tip**
> 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.

> **Tip**
> 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.

> **Tip**
> 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.

> **Tip**
> 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.

> **Tip**
> 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.

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.

> **Tip**
> This scenario corresponds to Scenario B1.

Retrospective, Interaction-Related Action Generated on Interaction Distribution from One Mediation DN to Another Mediation DN

- CallDistributedToQueue

Stat Server Actions for Media-Channels

Overview

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.

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.

> **Important**
> Some internal aliasing of action names permits voice-related actions to be used for multimedia stat types. For such stat types, however, Genesys recommends that you confine your selection of actions to only those that are listed in this section.

Durable, Non-Interaction-Related Actions

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

- Active
- Available
- Blocked

Durable, Interaction-Related Actions

The following are the durable, interaction-related actions that Stat Server generates on agent and place objects:

- Delivering
- DoNotDisturb

- Handling
- HandlingInbound
- HandlingInternal
- HandlingOutbound
- NotRoutable
- Routable

Momentary, Interaction-Related Actions

The following are the durable, interaction-related actions that Stat Server generates on agent and place objects:

- BeingCoached
- BeingMonitored
- CoachingByIntrusionInitiated
- CoachingRequested
- ConferenceJoined
- ConferenceJoinedByIntrusion
- ConferenceMade
- DeliveringStarted
- HandlingInboundStarted
- HandlingInternalStarted
- HandlingOutboundStarted
- HandlingStarted
- InteractionResponded (MediaChannels)
- MonitoringInitiated
- Pulled
- StartedInternal
- StartedOutbound
- TransferMade
- TransferTaken

Retrospective, Interaction-Related Actions

The following are the retrospective, interaction-related actions, originated from Interaction Server that Stat Server generates on agent and place objects:

- Accepted
- CoachingByRequestInitiated

- InteractionAbandonedDuringOffering (MediaChannels)
- InteractionAccepted1
- Revoked
- StoppedInbound
- StoppedInternal
- StoppedOutbound

Attributes of Media-Channel Actions

The Table below 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.

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.
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.

Media-Channel Action Attributes	
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.
GlobalUserData	The user-data, associated with a given action.
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.

Stat Server Actions for StagingArea

> **Important**
> StagingArea is not compatible with any other statistical objects. Stat type listed other object along with StagingArea will not be accepted by Stat Server.

Durable, Interaction-Related Actions

The following is the durable, interaction-related action that Stat Server generates on StagingArea:

- InteractionWait (StagingAreas)

Momentary, Interaction-Related Actions

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

- InteractionCreated (StagingAreas)
- InteractionEntered

Retrospective, Interaction-Related Actions

The following are the retrospective, interaction-related actions, originated from Interaction Server that Stat Server generates on StagingArea:

- InteractionAbandonedDuringOffering (StagingAreas)
- InteractionAccepted (StagingAreas)
- InteractionAnswered (StagingAreas)
- InteractionCleared
- InteractionDeleted (StagingAreas)
- InteractionDistributed
- InteractionDistributedToQueue
- InteractionPastAcceptServiceLevel (StagingAreas)
- InteractionPastCompletionServiceLevel (StagingAreas)
- InteractionReleased (StagingAreas)

Stat Server Actions for Tenant

Starting with release 8.5.102, Stat Server supports Tenant as a native statistical object.

Durable, Interaction-Related Actions

The following is the durable, interaction-related action that Stat Server generates on Tenant:

- InteractionAgentPartyInProgress (Tenants)
- InteractionWait (Tenants)

Momentary, Interaction-Related Actions

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

- ExternalServiceRequested (Tenants)
- ExternalServiceResponded (Tenants)
- InteractionCreated (Tenants)

Retrospective, Interaction-Related Actions

The following are the retrospective, interaction-related actions that Stat Server generates on Tenant:

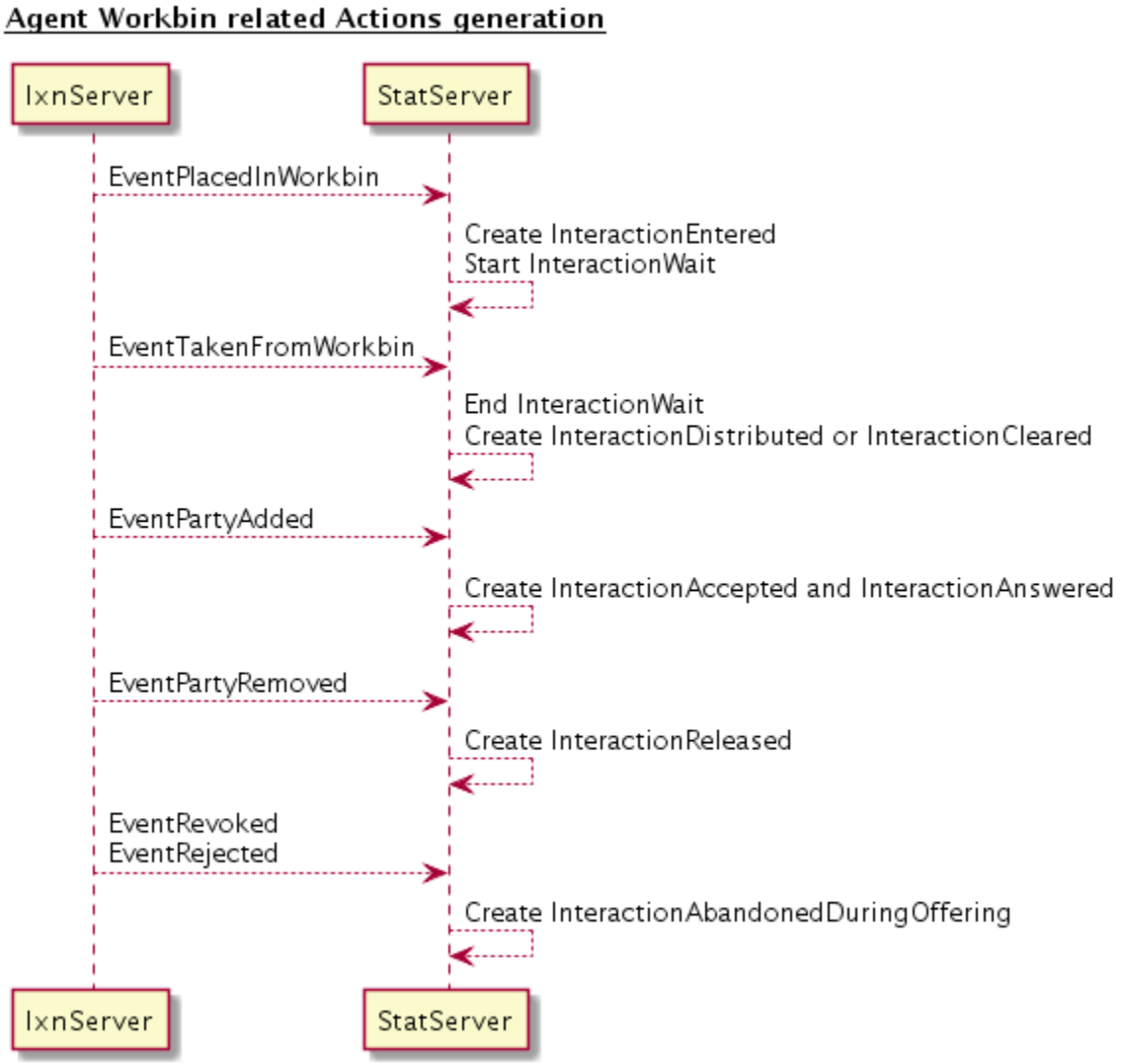
- InteractionAbandoned (Tenants)
- InteractionAbandonedDuringOffering (Tenants)
- InteractionAccepted1
- InteractionAccepted (Tenants)
- InteractionAnswered (Tenants)
- InteractionDeleted (Tenants)
- InteractionPastAcceptServiceLevel (Tenants)
- InteractionPastCompletionServiceLevel (Tenants)
- InteractionReleased (Tenants)

Stat Server Actions for Agent Workbin

Starting with release 8.5.112.10, Stat Server supports Agent Workbin statistics. The agent workbin (<agent_id>@<workbin_type_id>) is defined by the following two attributes from Interaction Server events:

- `attr_itx_workbin_type_id`
- `attr_itx_agent_id`

Both attributes are added to System Key-Value Lists (system attributes) with the `WorkbinID` and `WorkbinOwnerID` key names.



Durable, Interaction-Related Actions

The following is the durable, interaction-related action that Stat Server generates on an Agent Workbin:

- InteractionWait (Agent Workbin)

Momentary, Interaction-Related Actions

The following is the momentary, interaction-related action that Stat Server generates on an Agent Workbin:

- InteractionEntered (Agent Workbin)

Retrospective, Interaction-Related Actions

The following are the retrospective, interaction-related actions, originated from Interaction Server that Stat Server generates on an Agent Workbin:

- InteractionAbandonedDuringOffering (Agent Workbin)
- InteractionAccepted (Agent Workbin)
- InteractionAnswered (Agent Workbin)
- InteractionCleared (Agent Workbin)
- InteractionDistributed (Agent Workbin)
- InteractionReleased (Agent Workbin)

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.

These topics describe object statuses with respect to Stat Server and how they are classified, defined, and determined.

Important

- Options that define Status Priority tables (including the options **DefaultDNSPT**, **DefaultAgentSPT**, and **DefaultRPSPT**) are no longer supported in release 8.5.0 and later. Do not attempt to modify these values.
- The following DN statuses can co-exist with one of their respective interaction-type statuses: `CallDialing`, `CallRinging`, `AfterCallWork`, and `CallOnHold`.

Regular DN Status

NotMonitored

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

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

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

> **Tip**
> See also DN Actions at Newly Registered DNs.

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

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

This status never appears on a DN.

Place and Agent Status

`PlaceStatus` is the status of a DN linked to the place with the highest priority according to the DN Status Priority Table.

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 Tables.
   - The status of a voice-enabled multimedia DN is computed based on media-independent actions and voice actions only, according to the DN Status Priority Tables.
3. The other-than-voice 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 Media-Channel Status Priorities.
   - The status of a media-enabled multimedia DN is computed based on media-independent actions and non-voice actions only, according to the Multimedia DN Status Priorities.
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 nonvoice-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.

> **Important**
>
> For Stat Server 8.1.0⁻, 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 (Stat Server 8.1.0-), 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.

> **Tip**
>
> 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 (Stat Server 8.1.0-) 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 8.1.0-, 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 (Stat Server 8.1.0-), 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 8.1.0- 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 8.1.0-), 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 **ignoredisabled- objects-in-group-statistics** configuration option.

> **Important**
>
> In the 8.1.0- release, 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.

In Stat Server 8.1.0-, if an agent is logged into a place, agent status inherits the status of the place; otherwise, agent status is `Logged0ut`.

In Stat Server 8.1.2+, agent status is `Logged0ut` if the agent is not logged in to any DN/media-channel. Otherwise, the same algorithm as for the place (above) is used to compute agent status based on DNs/media channels (possibly, belonging to different places), where the agent is logged in.

Meridian extensions without agent login are also used for agent status computation, should there be an agent login on the associated position (extension and position must belong to the same 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 Associations Between Agents and DNs/Media Channels).

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 Regular DN Status Priority Table 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`

> **Important**
> 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 the interaction is on hold.

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.

Agent Status Priority Table

The standard Agent Status Priority Table is the same as the standard Regular DN Status Priority Table. The only difference is in the status `Logged0ut` that is listed in the DN Status Priority Table, but never appears on a DN. Instead, the `Logged0ut` status is supported for agents and has the highest priority out of all agent statuses.

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.

> **Important**
> 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 `Logged0ut` 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 eServices Solution), Stat Server observes the ranking shown in the table Media-Channel status priorities in order from lowest to highest. You cannot change the ranking order.

The Media-Channel status priorities table also maps the media-channel statuses and associated regular DN statuses:

Media-Channel status priorities

Media-Channel status (in ascending order)	Associated regular DN status
Active	LoggedIn
Available	WaitForNextCall
Blocked	NotReadyForNextCall
InteractionDelivering	CallRinging
InteractionHandlingInternal	CallInternal
InteractionHandlingOutbound	CallOutbound
InteractionHandlingInbound	CallInbound

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.

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, see the Stat Type Configuration Options table.

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.

Historical and Current Aggregated Values

Historical	Current
TotalNumber TotalAdjustedNumber	CurrentNumber
TotalTime TotalAdjustedTime	CurrentTime CurrentContinuousTime
MaxTime	CurrentMaxTime
MaxNumber	
MinTime	CurrentMinTime
MinNumber	
	CurrentAverageTime

Aggregated Values using TimeRanges

For historical and current statistical categories that use time ranges (namely, `TotalNumberInTimeRange`, `TotalNumberInTimeRangePercentage`, `ServiceFactor1`, `CurrentNumberInTimeRange`, and `CurrentNumberInTimeRangePercentage`), Stat Server maintains the restricted aggregated values listed below.

Aggregated Values using TimeRanges

Historical	Current
TotalNumberInTimeRange	CurrentNumberInTimeRange
TotalTimeInTimeRange	
ServiceFactor1	

Averages of Current Values

Averages of current values are based on an average number or duration of durable actions listed in the mask that are going on at the current moment or ended during the interval from which the statistic is calculated; instantaneous actions listed in the mask are ignored.

The two kinds of averages of current values are:

- AverageOfCurrentNumber
- AverageOfCurrentTime

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.

The evaluation of custom formulas, which is conducted according to different rules for different classes of actions, is described in detail in Custom Formulas. 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 below:

Aggregated Custom Values

Historical	Current
TotalCustomValue	CurrentCustomValue
MaxCustomValue	CurrentMaxCustomValue
MinCustomValue	Current MinCustomValue

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 that belongs 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 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` that belong 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 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 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.

Formula

Starting with the 8.5.106 release, Stat Server supports statistical category `Formula`. This statistical category allows you to create customized versions of the existing hard-coded categories by combining different statistical aggregates. See Formula for more information.

Supported Categories

Stat Server supports the following categories:

Historical	AverageNumberPerRelativeHour
	AverageOfCurrentNumber
	AverageOfCurrentTime
	AverageTime
	ElapsedTimePercentage
	MaxNumber
	MaxTime
	MinNumber
	MinTime
	RelativeNumberPercentage
	RelativeTimePercentage
	TotalAdjustedNumber
	TotalAdjustedTime
	TotalContinuousNumber
	TotalNumber
	TotalNumberInTimeRange
	TotalNumberInTimeRangePercentage
	TotalNumberPerSecond
	TotalTime
	TotalTimeInTimeRange
Current	CurrentAverageTime
	CurrentContinuousTime
	CurrentMaxTime
	CurrentMinTime
	CurrentNumber
	CurrentNumberInTimeRangePercentage
	CurrentRelativeNumberPercentage
	CurrentRelativeTimePercentage
	CurrentTime
	CurrentNumberInTimeRange
Historical CustomValue	AverageCustomValue
	MaxCustomValue
	MinCustomValue

	TotalCustomValue
Current CustomValue	CurrentAverageCustomValue
	CurrentCustomValue
	CurrentMaxCustomValue
	CurrentMinCustomValue
Compound Categories	EstimWaitTime
	ExpectedWaitTime2
	LoadBalance
	ServiceFactor1
Current State	CurrentState
	CurrentStateReasons
	CurrentTargetState
Java	JavaCategory

Historical Categories

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
```

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.

> **Important**
>
> Filter for this category should not be used with `Subject=AgentStatus`, `Subject=DNStatus`, or `Subject=PlaceStatus`. However, you can use a business attribute. For example, the `MediaType=chat` business attribute can be added to a stat type definition to calculate only chat interactions.

AverageOfCurrentNumber

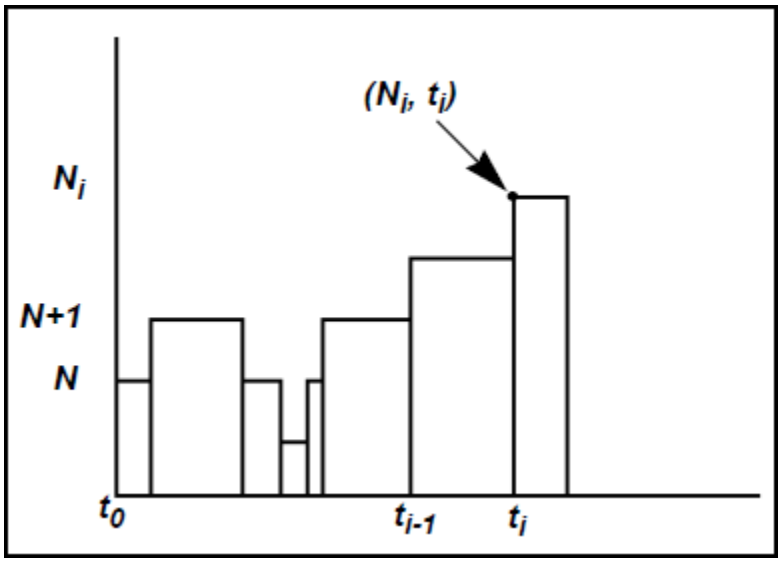
The value represents the average of current-number measurements over a specified time interval. Unlike the behavior in releases prior to 7.0, Stat Server take current-number measurements every 2 seconds; for example, Stat Server now only notes the time whenever the current number changes—for example, 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 below 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.



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

$$\text{Value} = \frac{\text{TotalTime}(\texttt{MainMask}, \texttt{Interval})}{\text{TotalNumber}(\texttt{RelativeMask}, \texttt{Interval})}$$

When the denominator is 0, the returned value is 0. The value is the quotient of the TotalTime aggregated value for the main mask and the TotalNumber aggregated value 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 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

$$Value = 100X\frac{TotalNumber(MainMask,Interval)}{TotalNumber(RelativeMask,Interval)}$$

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 for the main mask and 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 Aggregated Values using TimeRanges 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 for the main mask and 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.

$$\text{Value} = 100 \times \frac{\text{TotalTime(MainMask, Interval)}}{\text{TotalTime(RelativeMask, Interval)}}$$

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.

> **Important**
>
> 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

The `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 for durable actions and whole duration for retrospective

actions are used in this calculation.

For status-based statistics, TotalAdjustedTime is the sum of all durations of statuses listed in the mask that ended 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.

> **Important**
>
> - 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.
> - For a DNAction or Action subject, TotalAdjustedTime reports finished and unfinished actions for the current interval.
> - For the DNStatus and AgentStatus subjects, however, TotalAdjustedTime causes Stat Server to report the entire time a DN or agent status occurs only if it ends within the time interval.
> - TotalAdjustedTime calculation for retrospective actions was corrected in 8.5 Release.

TotalContinuousNumber

At any moment in time, the TotalContinuousNumber is the sum of the current (C) and historical (H) components:

V = C + H,

where

- V is the value of the statistic, sent to the client;
- C coincides with the value of the CurrentNumber statistic with the same MainMask and Subject;
- H is invisible internal counter.

Current component C is equal to 1, if current object status belongs to MainMask, and zero (0), otherwise.

Historical component H increases by 1 every time, when status is changed from value in MainMask to value not in MainMask.

In more details:

- When object status changes from S1 (not in MainMask) to S2 (not in MainMask), it does not affect the C, H;
- When object status changes from S1 (not in MainMask) to S2 (in MainMask), C is 1 and V is V+1;

- When status changes from S1 (in MainMask) to S2 (in MainMask), it does not affect the C, H;
- When object status changes from S1 (in MainMask) to S2 (not in MainMask), C is zero (0), H is H+1, and V is unchanged;
- When interval ends H is zero (0), C is unchanged, and V is equal C.

> **Important**
>
> Filter for this category should not be used with Subject=AgentStatus, Subject=DNStatus, or Subject=PlaceStatus. However, you can use a business attribute. For example, the MediaType=chat business attribute can be added to a stat type definition to calculate only chat interactions.

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

$$\text{Value} = 100 \times \frac{\text{TotalNumberInTimeRange(MainMask,Interval, TimeRange)}}{\text{TotalNumber(MainMask,Interval)}}$$

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 and the TotalNumber aggregated value—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 to the entire duration of the

interval, from which the statistic is calculated.

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.

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.

Current Categories

CurrentAverageTime

The CurrentAverageTime statistical category provides the average of all durations of durable actions or of statuses 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.

```
Value = CurrentTime(MainMask)/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

The 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. This statistical category applies only to stat types that have Agent and/or Place designated as their object—this category is not applicable for GroupAgents or GroupPlaces objects. This category is only applicable to the object status.

CurrentMaxTime

The CurrentMaxTime statistical category returns an aggregated value that provides the maximum duration among all durations 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.

```
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 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 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.

```
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 the `CurrentNumber` aggregated value—that is, the percentage of times the actions or statuses 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.

```
Value = 100 x CurrentNumberInTimeRange(MainMask,TimeRange)/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 or statuses that do not also appear in the relative mask, this percentage can exceed 100.

```
Value = 100 x CurrentNumber(MainMask)/CurrentNumber(RelativeMask)
```

CurrentRelativeTimePercentage

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 or statuses actions absent from the relative mask, this percentage can exceed 100.

```
Value = 100 x CurrentTime(MainMask)/CurrentTime(RelativeMask)
```

When the denominator is `0`, the returned value is `0`.

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

> **Important**
>
> Configured stat types for historical custom-value statistical categories must specify `DNAction` as the `Subject`.

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 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.

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)
```

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)`

> **Important**
>
> To keep the last calculated value and have sustainable results for the `MinCustomValue` and the `AverageCustomValue` categories, the appropriate filter must be applied. The filter should contain `PairExist("key", "*")` validation for all `UserData` for interactions involved in the calculation. The `PairExist("key", "*")` validation prevents formula calculation in cases where there is a missing key in `UserData` (the value for a missing key is always `0`).

Current CustomValue Categories

This section describes how Stat Server calculates statistics of the following current custom-value statistical categories:

- CurrentAverageCustomValue
- CurrentCustomValue
- CurrentMaxCustomValue
- CurrentMinCustomValue

> **Important**
>
> Configured stat types for current custom-value statistical categories must specify `DNAction` as the `Subject`.

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 formulas for Stat Server's compound statistical categories are derived from the formulas of two or more simple statistical categories. Stat Server defines the following compound statistical categories:

- EstimWaitTime
- ExpectedWaitTime2
- LoadBalance
- ServiceFactor1

With the exception of `ServiceFactor1`, all compound 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. The `EstimWaitTime` and `LoadBalance` categories service voice media type (it means that mediation DN must not belong to the multimedia switch); the `ExpectedWaitTime2` category services all media types.
- All interactions that are distributed from such mediation DNs are delivered only to agents who handle *M* media-type interactions only.

Statistics that are based on the these statistical categories might generate results that are difficult to interpret if statistics are requested for other than single-media mediation DNs.

All compound statistical categories are historical and, thus, calculated over specified time intervals. Configured stat types for compound statistical categories must specify `DNAction` as the `Subject` and must specify a nonempty main mask.

For example:

```
[stattype]
Category=EstimWaitTime or ExpectedWaitTime2 or LoadBalance
Subject=DNAction
Main Mask=CallWait

[stattype]
Category=ServiceFactor1
Subject=DNAction
Main Mask=CallAnswered
```

Compound statistical categories are based on fixed sets of actions. Each of the following sections lists the applicable actions for each category.

Genesys recommends to use the Sliding or Selection time profiles when you request statistics that use the `EstimWaitTime`, `ExpectedWaitTime2`, or `LoadBalance` categories.

> **Important**
>
> For switch types such as the Nortel Meridian—in which places are configured with both Position and Extension DNs and agents are required to be logged in to the Position DN—you must set the `position-extension-linked` Stat Server configuration option to yes for Stat Server to properly calculate statistics that are based on these categories.

EstimWaitTime

The `EstimWaitTime` statistical category provides an estimate of the amount of time that the last call that entered the mediation DN must wait before it is distributed from the mediation DN. This estimate takes into account the possibility of distributing calls from different queues to the same agents.

> **Important**
>
> Stat Server recognizes the following aliases for the `EstimWaitTime` statistical category:
>
> - StatExpectedWaitTime—Used by Universal Routing Server.
> - ExpectedWaitTime—Used within the Universal Routing Designer and CCPulse+ user interfaces. Do not confuse this alias with the ExpectedWaitTime2 statistical category.
>
> However, when you are creating stat types, Genesys recommends that you specify the proper category name.

Stat Server calculates the value of a statistic that belongs to this category 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` and `ACWCompleted` actions.
- `Mask2` is given by the `CallReleased` action.
- `Interval` is given by a supplied time profile.

Starting with Star Server release 8.5.103, the ACWMissed and CallMissed actions are not used in AHT calculations.

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 who currently are logged in and have 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 = CurrentNumber(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:

$$EP = \frac{TotalTime(Mask1, Interval)}{TotalTime(Mask2, Interval)}$$

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.

> **Tip**
>
> 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 and virtual 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 Genesys Administrator Extension, as follows:

1. Select an agent (or place) group.

2. On the Origination DNs tab click Add.
3. In the Origination DN dialog box, select the queue that you want to associate with this group.
4. On the Origination DNs tab click Save or Apply to save the configured association.

ExpectedWaitTime2

Similar to the EstimWaitTime statistical category, the ExpectedWaitTime2 category also provides wait-in-queue estimates for the last interaction that entered a virtual queue. This category, however, has been designed for the multimedia model which recognizes that agents can handle more than one simultaneous nonvoice interaction at a time. Starting with the 8.5.102 release of Stat Server, the ExpectedWaitTime2 statistical category is also applicable to voice media.

> **Important**
> The ExpectedWaitTime2 category does not support GroupQueues as an object.

Stat Server's formula for calculating values of statistics that use this category is the same as for the EstimWaitTime statistical category with exceptions along the interpretation of the formula's terms.

$$\text{Value} = \text{AHT}\frac{\text{CIQU}}{\text{AA} \times \text{EP}}$$

First, however, we revisit the definition of the Stat Server capacity vector, [S N_1 N_2 N_3], which Stat Server logs whenever, among other factors, the number of concurrent or assignable interactions for each media type changes at a particular place. Each vector pertains to one particular media type and its definition plays role in understanding why the terms in the preceding formula have different meanings.

- S represents the state of readiness of a particular media at a particular place.
- N_1 represents the current number of interactions that are in progress at a specific target for the particular media.
- N_2 represents the maximum number of interactions of the particular media that can be routed to a specific target according to the current capacity rule given the condition that the number of interactions on each of other medias remains unchanged.
- N_3 represents the number of additional interactions of a particular media that can be routed without violating the capacity rule given the condition that the number of interactions on each of the other medias remains unchanged. N_3 can differ from 0 only when the particular media channel is ready.

Further information about this vector and vector examples are provided in the *Genesys Resource Capacity Planning Guide*.

Below are the reinterpretations of terms in the ExpectedWaitTime2 statistical category:

- CIQU—The AR component of CIQU represents the sum of available capacity, N_3, of all agents who are

logged in to the queue instead of the current number of logged-in agents having `WaitForNextCall` status.

- AA—Represents the sum of maximum capacities (N_2 of the capacity vector) of all active agents for a given media instead of the current number of active agents.
- AHT—*average handling time*—an average time, spent on processing a call that comes from the queue and after-call work that follows it.

$$AHT = \frac{TotalTime(Mask1, Interval)}{TotalNumber(Mask2, Interval)}$$

where:

- `Mask1` is given by the `CallReleased` and `ACWCompleted` actions.
- `Mask2` is given by the `CallReleased` action.
- `Interval` is given by a supplied time profile.

Starting with Star Server release 8.5.102, the `ACWMissed` and `CallMissed` actions are not used in `AHT` calculations.

If no calls from the queue have been processed yet, `AHT` is considered to be 90 seconds.

- EP—*effective portion*—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 specified by the same media type as given queue and after-call work following these calls:

$$EP = \frac{TotalTime(Mask1, Interval)}{TotalTime(Mask2, Interval)}$$

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.

Starting with Star Server release 8.5.102, EP is calculated separately for each media type. EP is calculated only if more than one virtual queue is distributing calls for a specified media type. In other case EP=1.

If no calls from the queue have been processed yet, EP is considered to be 1.

In `Mask2` above only those `CallMissed` and `ACWMissed` actions are accounted for, that have the same media type as the given queue.

> **Tip**
>
> The value of 10000 seconds is assigned for the requested `ExpectedWaitTime2` statistic, regardless of the `CIQU` and `AHT` values, when `AA` is equal to 0 (for example, there are no logged in agents, or all logged in agents are not active or have 0 capacity).

To use this category in the statistics that you define, you must also assign one— and only one—media that describes the type of interactions that the virtual queue will handle. You accomplish this by setting the media-type configuration option (described in the Stat Server Deployment Guide and RTME Options Reference) among the properties of the virtual queue object in the Configuration Layer.

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.

Stat Server calculates the value of a statistic that belongs 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 who currently are logged in and have the `WaitForNextCall` status:
   `AR = CurrentNumber(AgentReady)`

d. `AHT` stands for *average handling time*:

$$AHT = \frac{TotalTime(Mask1, Interval)}{TotalNumber(Mask2, Interval)}$$

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.

$$\text{Value} = 100x\frac{\text{nAnsw(TimeRange)}}{\text{nAnsw} + \text{nAband} - \text{nAband(TimeRange2)}}$$

where:

- nAnsw(TimeRange) is the restricted TotalNumberInTimeRange aggregated value for the CallAnswered (Mediation DNs) action.
- nAnsw+nAband is the TotalNumber aggregated value for the list of mediation DN actions CallAnswered, CallAbandoned, and Call AbandonedFromRinging.
- nAband(TimeRange2) is the restricted TotalNumberInTimeRange aggregated value 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`

> **Tip**
>
> 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

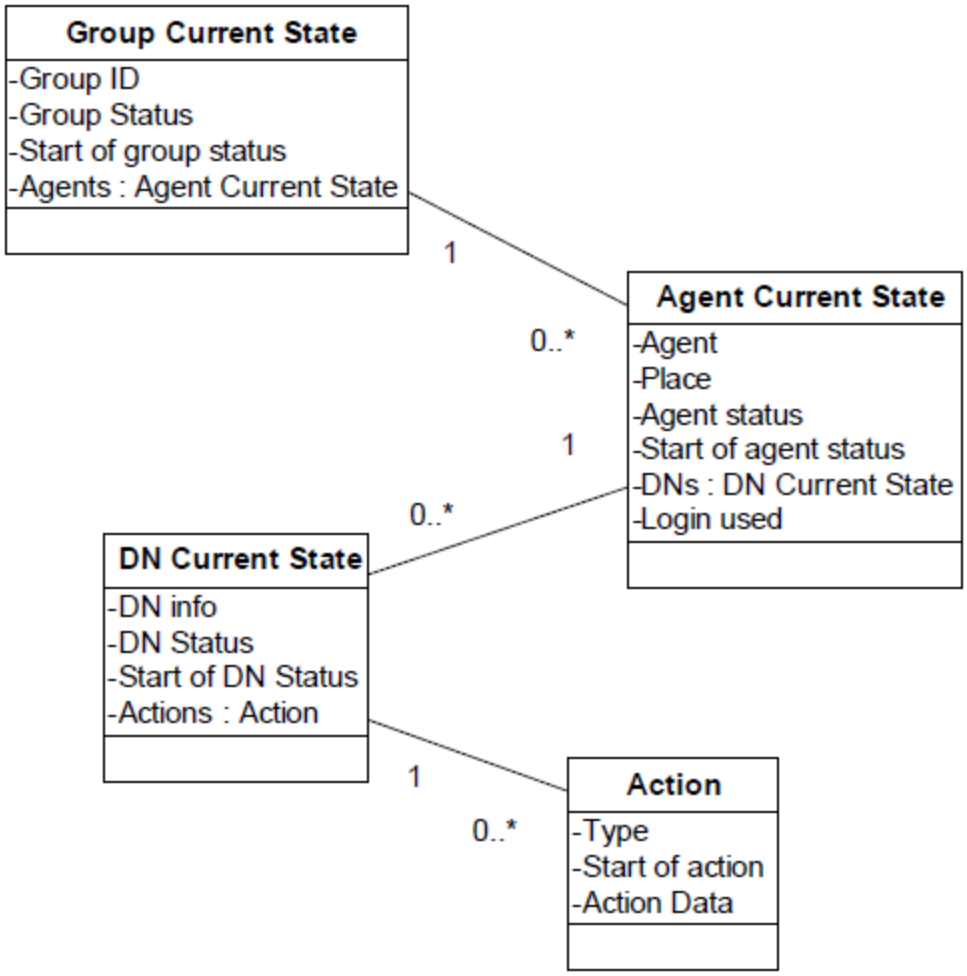
The format of the returned structure for the `CurrentState` statistical category depends on the object and subject of the statistic and can be represented as a tree. See the *CurrentState* figure below.

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, and the terminal nodes are always at the level of the stated subject of the statistic.

The Object statistical parameter determines whether Group `CurrentState` or Agent `CurrentState` is sent. Place-Group `CurrentState` has the same format as Agent-Group `CurrentState`. Place `CurrentState` has the same format as Agent `CurrentState`.

The Subject statistical parameter determines the depth of the tree:

- if `Subject = DNAction`, the tree is expanded down to DN actions;
- if `Subject = DNStatus`, the tree is expanded down to DN statuses; etc.



CurrentState

Starting with release 8.5.109, Stat Server maps the content of the `attribute_reason_desc` from Interaction Server events to the value of the `ReasonValue` key in a `UserData` key-value list with the statistical parameter `Subject` defined as `DNAction` or `Action`.

CurrentStateReasons

Starting with release 6.5, Stat Server provides the `CurrentStateReasons` category to support the reasons that agents place themselves in certain agent statuses (such as `WaitForNextCall`, `NotReadyForNextCall`, and `AfterCallWork`). Reasons can change within the same agent status. If it is likely that the agents within your contact center will change the reason they entered a particular agent status within that same agent status, and if you want Stat Server to measure such changes, consider setting the `ReasonStartOverridesStatusStart` stat type option (described in the Stat Type Configuration Options table) to `yes` to change the timestamp for the `tmStart` field. This statistical category applies only to stat types that have `Agent` and/or `GroupAgents` designated as their objects.

In addition to providing current status information for agents and agent groups, this statistical category also can store reasons for non-interaction-related statuses in key-value list 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 below illustrates the structures that support this statistical category.

The Agent CurrentStateReasons

> **Tip**
>
> - Not all T-Servers support reasons. Please refer to the appropriate T-Server manual for more details.
> - The Subject has to be set to DNAction for monitoring reasons changes.

Starting with release 8.5.109, Stat Server maps the content of the attribute_reason_desc from Interaction Server events to the value of the ReasonValue key in a Reasons key-value list.

CurrentTargetState

The CurrentTargetState statistical category is reported using the following two structures that include multimedia-capacity information about agent, place, agent group, and place group states:

- Snapshot
- Delta

Stat Server returns the Snapshot structure as its initial response when a client requests a statistic using the CurrentTargetState category. In general, it includes the array of Target structures, where each structure corresponds to a single routing Target (Agent or Place) If Object is Agent or Place, the array contains a single Target structure. Stat Server sends subsequent notifications (Target added/changed/removed—the first two are sent only if a statistic is requested for agent group or place group) using the Delta structure, which contains the (changed) Target and the list of associated statistical requests.

> **Important**
>
> The CurrentTargetState statistical category is only available for the OpenStat request with the ChangesBasedNotification mode. The GetStat or the PeekStat request is not applicable to that category.

Consider the following situation:

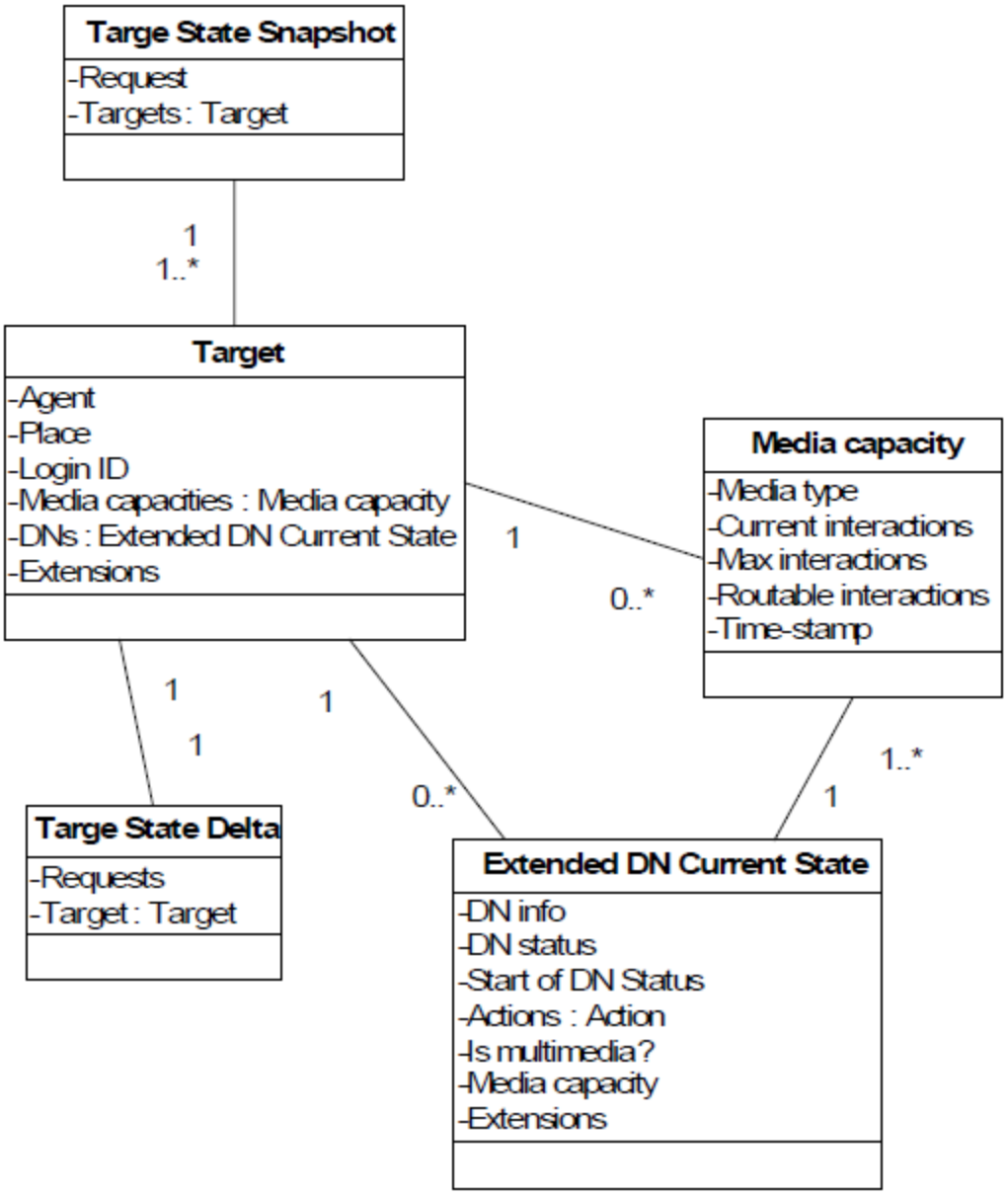
- Agent is associated with place (via login into DN belonging to that place)
- Agent belongs to an agent group(s)
- Place belongs to a place group(s)
- CurrentTargetState statistic is requested for the Agent, Place, agent-group(s), place(groups)

For 8.1.0^{-} release, the single ("atomic") response is sent for all (associated) requests above.

For 8.1.2^{+} release, two responses are sent:

- one for the Place and place groups
- another for the Agent and agent groups

Figure below illustrates CurrentTargetState.



The CurrentTargetState

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. For this reason, the `TimeBased` notification mode does not apply to this category.

Formula

Overview

Starting with the 8.5.106 release, Stat Server supports statistical category `Formula`.

Formulas (defined in SQL-like language) allow you to calculate statistics that could not be calculated using hard-coded categories.
With formulas you can:

- Use/combine statistical Aggregates
- Perform Rank Computations
- Do Sorting and Branching
- Apply Mathematical, String and Key-Value List functions
- Use Action Attributes and System Variables

Historical custom formula aggregates (`COUNT`, `SUM`, `AVG`, `MIN`, `MAX`, and so on) are only updated in the following two situations:

- Upon reset
- Upon action or status end (depending on the `Subject` parameter of the statistic)

Historical aggregates in hard-coded categories are also updated while status is in progress.

Important

- In addition to the `Growing` aggregation interval, supported in the 8.5.106 release, starting with the 8.5.107 release, Stat Server supports `Sliding`, `Selection`, and `SinceLogin` aggregation intervals.
- `Sliding` and `Selection` aggregation intervals are applicable to the `DNAction` subject only.
- Formulas serialization into/from the backup file is supported.
- The logical NOT (~) character does not work with aliases for the group of actions, such as `CallWait` on mediation DNs. To exclude the whole group, each member should be excluded explicitly. For example:
  `*,~CallWaitUnknown,~CallWaitConsult,~CallWaitInbound,~CallWaitOutbound`

Examples

The formula itself has to be specified in the `Expression` option:

```
Category=Formula
Objects=<objects>
Expression=<formula expression>
Subject=<subject>
<business attribute name>=<business attribute value>
```

Stat Type	Description
`Category` = Formula `Subject` = DNAction `Objects` = Queue `Expression` = @COUNT(CallWait; duration > 10)	A statistic with this Stat Type, requested for a Queue, returns the current number of calls on this Queue with duration more than 10 seconds.
`Category` = Formula `Subject` = DNAction `Objects` = Agent `Expression` = AVG(CallInbound; kvnum(udata, "Priority"))	A statistic with this Stat Type, requested for an Agent, returns an average value of the "Priority" User Data key of received by this Agent inbound calls (`CallInbound` actions generated).

Options for Formula Stat Type

Option	Applicability	Comments
Category	Mandatory	Has to be set to `Formula`.
Objects	Mandatory	See the Stat Type Configuration Options table for the description of the `Objects` option.
Subject	Mandatory	See the Stat Type Configuration Options table for the description of this option.
Expression	Mandatory	The formula has to be specified in this option. Stat Server Actions from the `Expression` are used for the `Selection` time profile if `MainMask` and `RelMask` are absent.
MainMask	Special	Applicable only for the `Selection` time profile if the `RelMask` is absent and ignored in all other cases. See the Stat Type Configuration Options table for the description of this option.
RelMask	Special	Applicable only for the `Selection` time profile and ignored in all other cases. See

Option	Applicability	Comments
		the Stat Type Configuration Options table for the description of this option.
<business attribute>	Optional	See the Stat Type Configuration Options table for the description of this option.
UseSourceTimeStamps	Optional	See the Stat Type Configuration Options table for the description of this option.
ApplyFilterAtActionEndOnly	Optional	See the Stat Type Configuration Options table for the description of this option.
Description	Optional	See the Stat Type Configuration Options table for more information.
DynamicOverloadPolicy	Optional	Defines actions that Stat Server may apply to a given statistic to reduce the overload.
JavaSubCategory	N/A	Not applicable.
ReasonStartOverridesStatusStart	N/A	Not applicable.
Formula	N/A	Not applicable.

> **Important**
>
> - `TimeRanges`, if needed, should be included as conditions in the `Expression`.
> - `Filters`, `TimeProfile`, `Object` name are included in a New API Statistic request.

Aggregates

Using formulas, users can create customized versions of the existing hard-coded categories, combine different statistical aggregates, and so on. Aggregates are the basis of any formula and provide the way of calculating values:

Aggregates

Aggregator	Parameters	Description	Example
@COUNT	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. condition** Optional. Boolean expression,	Current aggregator that returns current number of actions or statuses listed in action_types_list that matches the condition if provided.	@COUNT(CallInternal, CallInbound, CallConsult; kvexists(udata, "Language", "English")) – returns current number of actions/

Aggregator	Parameters	Description	Example
	depending on action/ status attributes, interval attributes and system variables		statuses listed, that have "Language", "English" key-value pair in attached UserData.
@SUM	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. expression** Mandatory. Numeric expression, depending on action/status attributes, interval attributes and system variables **3. condition** Optional. Boolean expression, depending on action/ status/st attributes, interval attributes and system variables	Current aggregator that returns current sum of numeric expressions, computed for each action or status, satisfying the condition.	@SUM(CallWait; kvnum(global_udata, "amount", -1)) – returns current sum of the last value in each CallWait action of "amount" key in attached UserData.
@AVG	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. expression** Mandatory. Numeric expression, depending on action/status attributes, interval attributes and system variables **3. condition** Optional. Boolean expression, depending on action/ status attributes, interval attributes and system variables	Current aggregator that returns current average of numeric expressions, computed for each action or status, satisfying the condition.	@AVG(CallOutbound; duration; duration > 0 & duration < 5) – returns current average duration of CallOutbound actions/ statuses with duration less than 5 seconds.
@MIN	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. expression** Mandatory. Numeric expression, depending on action/status attributes, interval attributes and system variables **3. condition** Optional. Boolean expression,	Current aggregator that returns current minimum of numeric expressions, computed for each action or status, satisfying the condition.	@MIN(WaitForNextCall; duration) – returns current minimum duration of WaitForNextCall action/ status

Aggregator	Parameters	Description	Example
	depending on action/ status attributes, interval attributes and system variables		
@MAX	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. expression** Mandatory. Numeric expression, depending on action/status attributes, interval attributes and system variables **3. condition** Optional. Boolean expression, depending on action/ status attributes, interval attributes and system variables	Current aggregator that returns current maximum of numeric expressions, computed for each action or status, satisfying the condition.	@MAX(NotReadyForNextCall; duration; kvexists(reasons, "ReasonCode", "Break")) – returns current maximum duration of NotReadyForNextCall action/status with "ReasonCode" = "Break"
COUNT	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. condition** Optional. Boolean expression, depending on action/ status attributes, interval attributes and system variables	Historical aggregator that returns total number of actions or statuses listed in action_types_list that matches the condition if provided.	COUNT(CallInternal, CallInbound, CallConsult; kvexists(udata, "Language", "English")) – returns total number of actions/statuses listed, that have "Language" = "English" key-value pair in attached UserData.
SUM	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. expression** Mandatory. Numeric expression, depending on action/status attributes, interval attributes and system variables **3. condition** Optional. Boolean expression, depending on action/ status attributes, interval attributes and system variables	Historical aggregator that returns sum of numeric expressions, computed for each action or status, satisfying the condition.	SUM (CallInternal, CallInbound, CallConsult; duration; kvnum(udata, "amount", -1) < 100) – returns sum of actions/ statuses duration if this action/status has last value of "amount" key in attached UserData less than 100.
AVG	**1. action_types_list** Mandatory. Comma separated list of Stat	Historical aggregator that returns average of numeric expressions,	AVG(CallWait; duration; duration >= 10) – returns average

Aggregator	Parameters	Description	Example
	Server actions or statuses **2. expression** Mandatory. Numeric expression, depending on action/status attributes, interval attributes and system variables **3. condition** Optional. Boolean expression, depending on action/ status attributes, interval attributes and system variables	computed for each action or status, satisfying the condition.	duration of CallWait action if action duration was more than or equal to 10 seconds.
MIN	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. expression** Mandatory. Numeric expression, depending on action/status attributes, interval attributes and system variables **3. condition** Optional. Boolean expression, depending on action/ status attributes, interval attributes and system variables	Historical aggregator that returns minimum of numeric expressions, computed for each action or status, satisfying the condition.	MIN(AgentActive; duration) – returns minimum duration of AgentActive action.
MAX	**1. action_types_list** Mandatory. Comma separated list of Stat Server actions or statuses **2. expression** Mandatory. Numeric expression, depending on action/status attributes, interval attributes and system variables **3. condition** Optional. Boolean expression, depending on action/ status attributes, interval attributes and system variables	Historical aggregator that returns maximum of numeric expressions, computed for each action or status, satisfying the condition.	MAX(CallInbound; kvmax(udata, "amount"); kvexists(udata, "Language", "English")) – returns maximum value of "amount" key in UserData of CallInbound actions/ statuses for which key-value pair "Language" = "English" also be presented in UserData.

Action Attributes

The following attributes are used in expression and condition parameters of aggregators:

Attributes

Attribute	Description
duration	A duration of Stat Server action/status.
start	A timestamp of the start time of Stat Server action/status.
end	A timestamp of the end time of Stat Server action/status.
udata	UserData key-value list. List of data that is used in key-value pair functions as the first argument.
global_udata	Global UserData key-value list. List of data that is used in key-value pair functions as the first argument.
reasons	Reasons key-value list. List of data that is used in key-value pair functions as the first argument.
extensions	Extensions key-value list. List of data that is used in key-value pair functions as the first argument.
system	System key-value list. List of data that is used in key-value pair functions as the first argument.

System Variable Attributes

The aggregation interval parameter can be used in current Formulas (those, containing only current aggregates) with the `interval != Growing`. Current Formulas do not aggregate across the `Selection` interval.

The following parameter is available for current statistics:

System Variables

Attribute	Description
$istart	An aggregation interval start. For example, when `interval=Sliding` the `@SUM( CallInternal; end-greatest(start,$istart) )` adjusts the duration of all current actions to the start of the window.

Mathematical Functions

The table below lists mathematical functions that are used in the expression and condition parameters of aggregators as well as in independent clauses:

Mathematical Functions

Function	Description
abs	Absolute value
ceil	Ceiling of the value
floor	Floor of the value
log	Logarithm of the value
exp	Exponent of the value
sqrt	Square root of the value
sin	Sinus of the value
cos	Cosinus of the value
pow	Power of the value
%	Module (for example, 5%2=1, and 64%4=0)

All mathematical functions have one float argument, except for the pow and % functions, which have two float arguments.

String Functions

String functions are applicable to set parameters, for example, `"Language"` and values returned by the `kvstr` function: `kvstr(udata, "Language")`. The table below lists string functions:

String Functions

Function	Description
sub	Substring
upper	Conversion to upper case
lower	Conversion of lower case
concatenation (operator +)	Concatenation of strings
convert	String conversion to a numeric value. For example, *convert("32.58")* returns the 32.58 numeric value.

All string functions have a single argument of the type string, except for the concatenation (+) function, which combines two strings.

Key-Value List Operations

The key-value list functions are similar to those presented in the UserData Properties. The following collections are taken as the first parameter:

- udata
- global_udata
- reasons
- extensions
- system

The functions below can be applied to collections:

Key-Value List Functions

Function	Description	Similar Operator in UserData Properties
`kvstr`	A string extraction from the key-value list.	`GetString`
kvlist	An embedded key-value list extraction from the key-value list.	`GetList`
`kvnum`	A number extraction from the key-value list.	`GetNumber`
`kvsum`	A Sum of numbers in the key-value list.	`GetSum`
`kvmin`	A minimum value for a given key in the key-value list.	`GetMin`
`kvmax`	A maximum value for a given key in the key-value list.	`GetMax`
`kvavg`	An average value for a given key in the key-value list.	`GetAver`
`kvexists`	`True` if a given key or pair `<key,value>` exists in the key-value list. `False` otherwise.	`PairExists`

Rank Computations

Ranking functions (see the table below) are used in conditions of aggregators to rank Stat Server actions by some parameters. There are two arguments for rank functions: a function of actions attributes, and a positive integer (rank).

Ranking Functions

Function	Description
rank_lt	`True`, if the rank of an action, according to a condition, is less than the predefined number.
rank_gt	`True`, if the rank of an action, according to a condition, is greater than the predefined number.

Example

If you want to calculate the (current) average duration of the 5 longest waiting calls in queue use the following expression:

```
@AVG( CallWait; duration; rank_lt( duration, 6 ) )
```

Sorting

Sorting functions:

- greatest - the greatest value in a list.
  `greatest( x1, x2, …, xN )`, where x1, …, xN are numbers.
- least - the least value in a list.
  `least( x1, x2, …, xN )`, where x1, …, xN are numbers.

can take any numeric values (including aggregators) as arguments. Unlike other functions, sorting functions have variable number of comma separated arguments.

Example

Find the largest number of the current Inbound, Outbound or Consult calls:

```
greatest( @COUNT(CallInbound), @COUNT(CallOutbound), @COUNT(CallConsult) )
```

Branching

You can use the branching construction (`<conditions> ? <expression if condition is true> : <expression if condition is false>`) in aggregator, num_expr, and condition parts of a formula. However for aggregators, Genesys recommends that you not use `kvexists()`, ranking computations, or string operations with the branching operator.

Branching `<boolean expression> ? < num_expr1> : <num_expr2>` means that if the `<boolean expression>` is true, then `<num_expr1>` is returned, if the `<boolean expression>` is false `<num_expr2>` is returned.

Operators and their Priorities

Operators and their priority (see the table below) can be used in any part of the formula expression.

Operator	Priority	Associativity (which operator occurrence is applied first when the same priority operators appear in a row)
^	10	Left
- (unary minus)	9	None

Operator	Priority	Associativity (which operator occurrence is applied first when the same priority operators appear in a row)
~	8	Right
*	7	Left
/	7	Left
%	7	Left
+	6	Left
-	6	Left
<	5	Left
>	5	Left
<=	5	Left
>=	5	Left
=	5	Left
!=	5	Left
&	4	Left
|	3	Left
:	2	Right
?	1	Right

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 Extension (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.

The *Solution Reporting Templates* book of the *Reporting Technical Reference* series describes stat types provided in the 7.x and 8.x releases of the Outbound Contact (OCC) and Multimedia (MCR) SSJEs. See Reporting and Reporting Templates.

Starting with release 8.5.1, Stat Server is extended to include support for the Orchestration Server (ORS) Java Extension.

Disclaimer:	SSJE is a closed platform for Genesys use only. No third party Stat Server Java Extensions are supported by Genesys.

Batch Requests

Starting with release 8.5.102, Stat Server supports requests for batch operations on all previously opened statistics (except the `TargetCurrentState` statistic) by the same client, thus providing ways to optimize network traffic between Stat Server and its clients.

Available batch requests are:

- `SuspendAll` - allows a client to suspend all notifications for previously opened statistics until they are resumed by the client one by one or all at once using the `ResumeAll` request.
- `ResumeAll` - resumes notifications for all suspended statistics for the client.
- `PeekAll` - allows to peek all opened, not suspended statistics with just one request.

> **Important**
>
> The `PeekAll` request does not peek the statistics, for which the notification has previously been suspended.
>
> Statistics, which do not support Peek request (such as `CurrentTargetState`), are not be impacted by the `PeekAll` request.
>
> The `CurrentTargetState` statistic is not impacted by `SuspendAll`/`ResumeAll` requests.

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

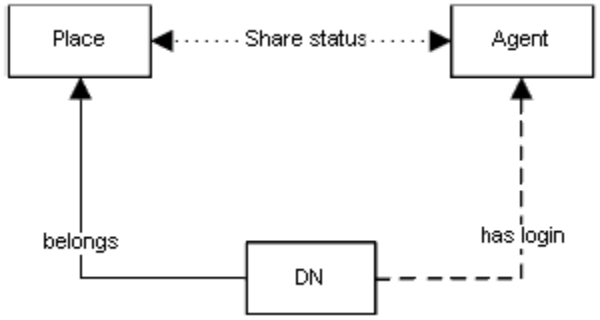
To define the perspective from which you want Stat Server to capture data for a statistic, you specify one *statistical subject* in the statistics's underlying stat type definition, by assigning a value to the `Subject` option. (This option is described in the Stat Type Configuration Options table). 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 Hierarchy of Stat Server Telephony Objects.

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 DNs/Media Channels
- DN Association with Queues
- Propagation of DN Actions

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 status of agents are the source of statistics for all applicable objects.

If you specify an invalid or unsupported subject given, Stat Server will both:

- Refuse to open the affected statistic
- Log an error

> **Tip**
>
> Starting with release 8.1.2, Stat Server automatically changes subject of the `{Place, AgentStatus}` and `{Agent, PlaceStatus}` admissible combinations and validates statistics.

Table below maps the objects to which each statistical subject applies. This mapping also indirectly reveals the groups of objects which are compatible.

Statistical Subjects and the Objects to Which They Apply

Subject	Applicable to the Following Objects	Description
AgentStatus	Agent	Indicates Stat Server should

Subject	Applicable to the Following Objects	Description
	Tenant GroupAgents	consider only the status occurring at agent data sources.
CampaignAction	CampaignGroup CallingList CampaignCallingList	Indicates Stat Server should consider only the actions occurring at campaign data sources.
DNAction (alias Action)	RegDN Agent Place Queue Tenant RoutePoint GroupAgents GroupPlaces GroupQueues	Indicates Stat Server should consider only the actions occurring at regular directory number data sources.
DNStatus	RegDN Agent Place Tenant GroupAgents GroupPlaces	Indicates Stat Server should consider only the status occurring at regular directory number data sources.
GroupStatus	GroupAgents GroupPlaces Tenant	Indicates Stat Server should consider only the status occurring at place group or agent group data sources.
PlaceStatus	Place Tenant GroupPlaces	Indicates Stat Server should consider only the status occurring at place data sources.

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.

> **Tip**
>
> There is no direct connection between Stat Server and Outbound Contact Server—T-Server transmits OCS events to Stat Server through communication DNs.

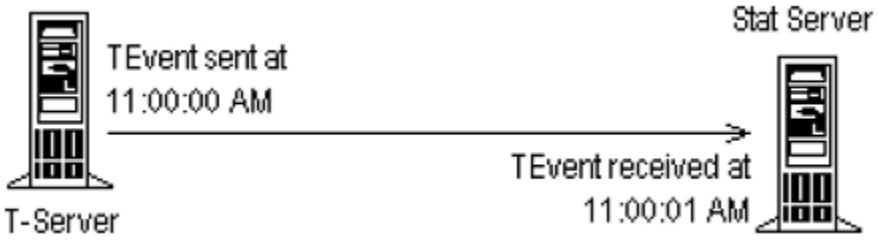
- To indicate when T-Server, Interaction Server, OCS, or Configuration Server sent event notifications to Stat Server.
- To track the durations of actions.
- To determine whether and when to clear stuck calls.
- To profile a reporting interval.
- To define time ranges over which actions should be measured.

This chapter focuses on the aspects of time that impact the statistical values Stat Server reports to clients as the aspects 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 in the Stat Type Configuration Options table).

This chapter contains the following sections:

Timestamps

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.



Timestamp of the TEvent

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 in the Stat Type Configuration Options table) 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.

Source Timestamp Algorithm

Starting with 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.

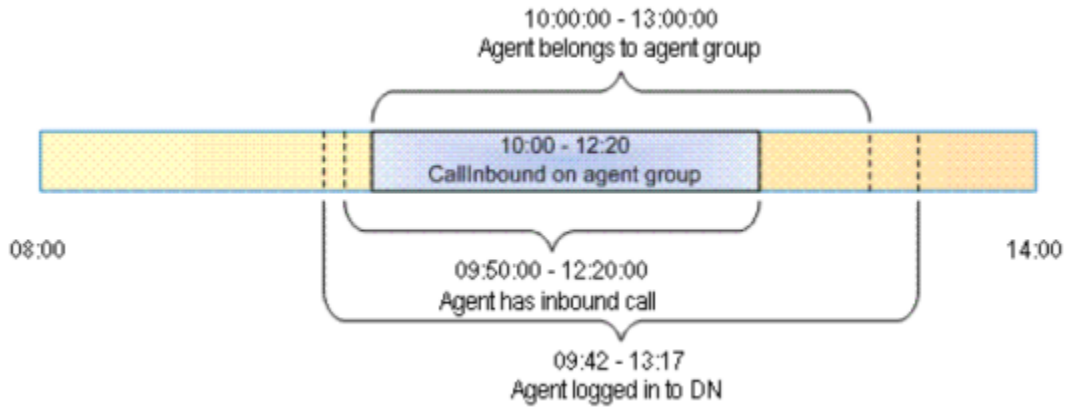
Stat Server attributes these to each action regardless of the setting of the `UseSourceTimeStamps` option. When `UseSourceTimeStamps` is set to `true`, Stat Server references the values of these attributes to calculate statistic duration.

> **Important**
>
> The values that Stat Server writes to the Stat Server database do not reflect source timestamps.

Figure below illustrates how Stat Server computes the duration of a `CallInbound` action on an agent group when `UseSourceTimeStamps=true`. Note that Stat Server uses seconds-level precision, truncating milliseconds before computing duration.

- 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.



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 example at the top of the page.

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.

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 Configuration Option for the TimeProfiles Section).
- 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 `DistByConnID`)
    - `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`.

Campaign Statistics

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.

Further information is divided among the following topics:

Campaign Objects

Campaign statistics are calculated exclusively for the Outbound Contact Solution to reflect campaign performance. Consult the Outbound Contact Solution 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`

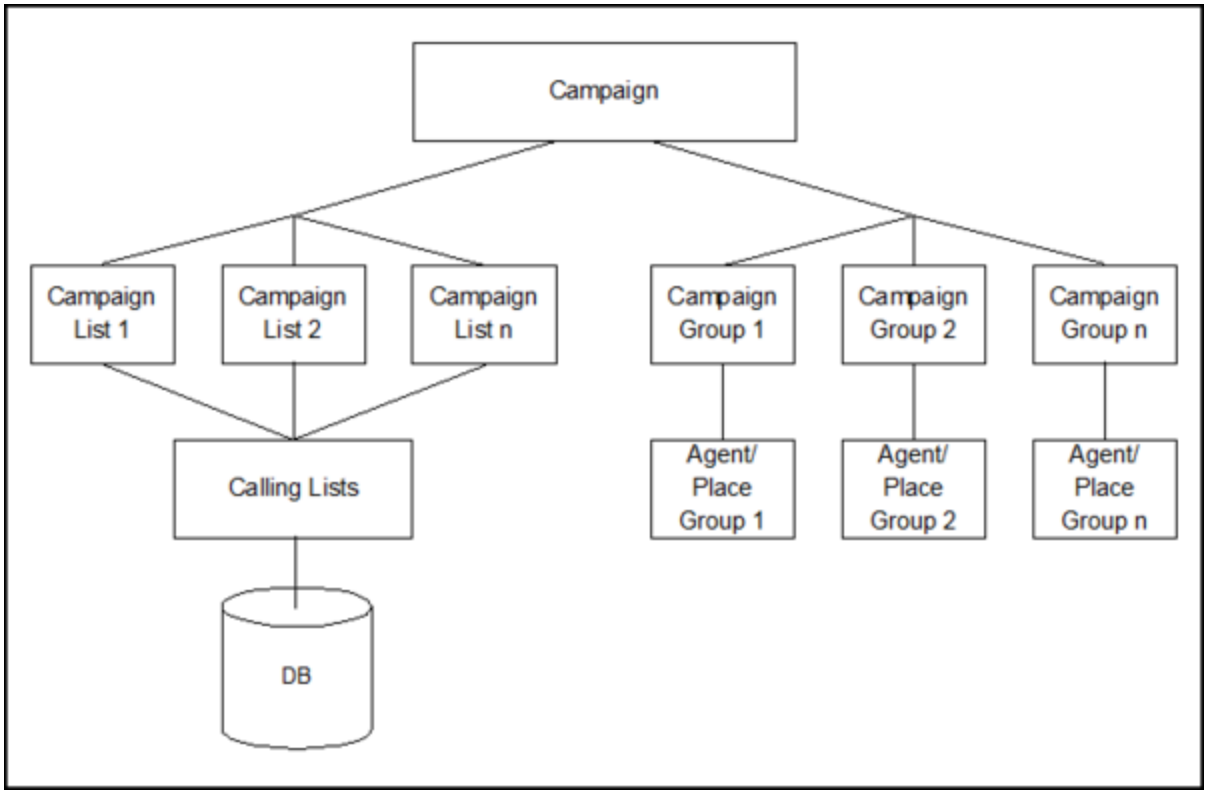
Stat Server's `Campaign` and `CallingList` objects correspond to Configuration Server's `Campaign` and `CallingList` objects. Accordingly, these objects must have the same names as Configuration Layer's campaign objects. `Campaign Group` and `CampaignCallingList` objects are configured within and are 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+.

> ### Warning
>
> Starting with release 7.5, Configuration Server allows an arbitrary name for a `CampaignGroup`, however, Stat Server only recognizes a `CampaignGroup`, named as <campaign name>@<group name>, where <campaign name> is the name of the campaign and <group name> is the name of the group (agent group or place group).

The graphic below shows the hierarchy of the Stat Server campaign objects (see the Stat Server Telephony Objects schema, for lower levels of the hierarchy).

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.

> **Important**
> 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 StatusWaiting Records, StatusWaitingPorts, StatusWaitingAgents, and StatusSystemError durable actions.

In parallel with the StatusRunning action, one of these dial modes can occur:

• NoDial	• Preview	• Progress
• Predict	• Power	• ProgressAndSeize
• PredictAndSeize	• PowerAndSeize	• ProgressGVP
• PredictGVP	• PowerGVP	• 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`
- `DialNoProgress`
- `DialNoRingBack`
- `DialNUTone`
- `DialPagerDetected`
- `DialSilence`
- `DialSITDetected`
- `DialSITInvalidNum`
- `DialSITNoCircuit`
- `DialSITOperIntercept`
- `DialSITReorder`
- `DialSITUnknown`
- `DialSITVacant`
- `DialStale`
- `DialSwitchError`
- `DialSystemError`
- `DialTransferError`
- `DialUnknown`
- `DialWrongNumber`
- `DialWrongParty`

- Starting with release 8.5.106, Stat Server supports new instant action `DialRemoteRelease` on `Campaign/CallingList` statistical objects to count `Dial` attempts finalized with `CALL_RESULT= 5` (Remote Release).
- RecordsNotProcessed—Stat Server generates this action for campaign or calling list objects when campaign (or calling list) processing is completed.

> **Tip**
>
> 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 Layer. 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 StatisticalCategories, 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.

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:

Note: Custom formulas can be requested with `Subject=DNAction` only.

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.

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`.
- $\texttt{List}_k$ 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,
   $\Delta_k = \Delta_{k-1}/\texttt{Native}(\texttt{List}_k\backslash\texttt{List}_{k-1})$
3. If Step k is caused by `EventAttachedDataChanged` with `ThisDN` different from `ThirdPartyDN`,
   $\Delta_k = \Delta_{k-1}/\texttt{Foreign}(\texttt{List}_k\backslash\texttt{List}_{k-1})$

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.

> **Tip**
>
> 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.

Virtual Agent Groups

Concepts of a Virtual Agent Group and explains on how to configure it are provided in the following articles:

- Supported Virtual Agent Group Definitions
- Configuring Virtual Agent Groups

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.

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 to which an agent is logged in.

You can simultaneously specify these types of parameters in an expression for a single virtual group.

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 the Configuration Layer.

Agent Skill Functions

Configure a Skill object for an agent on the **Skills** tab of the **Persons** dialog box in Genesys Administrator Extension. 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.

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. 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 is logged in to a switch, and a client requests the agent's `LoggedIn` status on that switch, Stat Server returns the value `true`. Agent login to a particular queue on that switch is unnecessary.

Configuring Virtual Agent Groups

Procedure

From Genesys Administrator Extension:

1. Under **Configuration**, select **Accounts** module.
2. Open the `Agent Group` configuration object.
3. On the **Options** tab, create a section named `virtual`.
4. Within the section, create a new option named `script`.
5. Enter a valid expression as a value for this option.

   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. Note that `LoggedIn("SwitchName")` is not supported for eServices switches.

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.

> **Important**
>
> - 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.
> - 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 virtual agent group is meant for agents whose Spanish skill is higher than 5 or whose French skill is higher than 8, the value of the Skill option is:

```
Skill("Spanish")>5 | Skill("French")>8
```

If the virtual agent group is meant for agents logged in ACD queue 5253 at the switch named DEFINITY, the option value is:

```
LoggedIn("5253@DEFINITY")
```

If the virtual agent group is meant for agents logged in at the switch named DEFINITY, the option value is:

```
LoggedIn("DEFINITY")
```

If the virtual agent 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")
```

> **Important**
>
> Recommended limit on the number of VAGs to maintain optimal Stat Server performance should be no more than 2000.

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 lists the stat types that are available to you upon creating a new Stat Server Application object using the Stat Server 8.5 template. It contains the following sections:

Stat Type Definitions in the Stat Server Application Template

The following stat type definitions are preconfigured in the Stat Server 8.5.1 application template. 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]
- [IxnQ_Cleared]
- [IxnQ_Cleared_Time]
- [IxnQ_Created]
- [IxnQ_Deleted]
- [IxnQ_Deleted_Time]
- [IxnQ_Distributed]
- [IxnQ_Distributed_Time]
- [IxnQ_Distributed_To_Queue]
- [IxnQ_Entered]
- [ServiceFactor]
- [TotalAfterCallWorkDNStatusTime]
- [TotalAfterCallWorkPlaceStatusTime]
- [TotalAfterCallWorkStatusTime]
- [TotalLoginTime]
- [TotalNotReadyDNStatusTime]
- [TotalNotReadyPlaceStatusTime]
- [TotalNotReadyStatusTime]
- [TotalNumberCallsAband]
- [TotalNumberCallsDistrib]
- [TotalNumberConsultCalls]
- [TotalNumberInboundCalls]
- [TotalNumberInternalCalls]
- [TotalNumberOutboundCalls]
- [TotalTalkDNStatusTime]
- [TotalTalkPlaceStatusTime]
- [TotalTalkStatusTime]

Creating Stat Type Definitions

Stat Type Formats

You define statistical types as sections on the **Options** tab (in Genesys Administrator) or on the **Application Options** (in Genesys Administrator Extension) 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)
        RelMask = [optional, applicable if a MainMask is specified]
                * and/or one or more actions separated by commas
                and optionally preceded by ~ (for NOT)
        MediaType = media type
        UseSourceTimeStamps = yes or no
        ReasonStartOverridesStatusStart = yes or no
        Description = [optional] free-form text
        Formula = DistByConnID, <expression>, mandatory for
                                CustomValue family of statistical categories.
        <business attribute name> = <business attribute value>
```

And Java stat types follow this format:

```
[NameOfJavaStatType]
        Objects = One or more objects separated by commas
        Category = JavaCategory
        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
```

For more detailed descriptions of these configuration options, see the Statistical Type Sections.

Starting with the 8.5.105 release, Stat Server supports multiple business attribute values, specified as a comma-separated list for a given business attribute, in the stat type definition. This is applicable to all statistical categories, except Compound Categories and `JavaCategory`.

Now it is possible to combine multiple business attributes in one stat type.

For example:
`MediaType=chat,email`

```
CustomerSegment=gold,silver
```

Stat Server does not check if values are correct as the same business attribute may have different correct values on different tenants.

> **Important**
> Currently, Stat Server Java Extensions either ignore business attribute specifiers in stat type definitions, or support only a single value in such specifiers.

Examples

The following examples illustrate the configuration of three sample stat type definitions as they appear in the configuration layer.

Example 1: Sample Stat Type Definition for Total Duration of Status for CallOutbound Actions

`TotalOutboundStatusTime` 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
```

Example 2: Sample Stat Type Definition for the Processing Time of Interactions

`Strategy_Email_ProcessingTime` 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.)

```
Full Definition
[Strategy_Email_ProcessingTime]
        AggregationType=Total
        Category=JavaCategory
        JavaSubCategory=eserviceinteractionstat.jar:strategy-total processing time
        MediaType=email
        Objects=RoutingStrategy
```

Example 3: Sample Stat Type Definition for a Switch Object

`Total_Number_of_Errors` measures the total number of hardware errors that occurred at a switch. This statistic is only meaningful for Network T-Server applications.

```
Full Definition
```

```
[Total_Number_of_Errors]
        Category=TotalNumberErrors
        Objects=Switch
        MainMask=NotMonitored
        Subject=DNStatus
```

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 *Solution Reporting Templates* book of the *Reporting Technical Reference* series. See Reporting and Reporting Templates.

Overload Protection

Starting with release 8.5.108, Stat Server supports overload protection.

Introduction

When and why to use overload protection?

The number of opened statistics depends on the client demands. The more statistics are opened or the more incoming events are received, the higher Stat Server CPU consumption. Stat Server application is not scalable and, in certain circumstances, it may start behaving unreliably (disconnect clients, get disconnected from servers, delay computations).

Stat Server load is %CPU, consumed by its main thread. It depends on the rate of incoming events and number (and parameters) of open statistics. `Overload protection` is a method of reducing CPU consumption as a response to Stat Server overload. The load range is defined as `[min,max]`. The `cooldown` is a predefined duration of time, when the load is less then `min`. Stat Server is in `overload`, if the load exceeded `max`, and no `cooldown` happened since then. Stat Server is in `recovery`, if it is in `overload`, and current load is less then `min`.

Overload protection consists of the following load reducing measures:

- Measure 1. Cut debug logging, controlled by the settings of the cut-debug-log option.
- Measure 2. Stat Server cannot skip incoming events and always processes them. However, it can lower the quality of service for some statistics in order to reduce CPU consumption. Also, it can skip some operations in the pipeline above: for some statistics, it may stop recalculating values and sending them.
- Measure 3. For some statistics Stat Server may stop updating aggregate. Please note, that measure 3 includes measure 2.

As soon as Stat Server hits the predefined high CPU threshold, it enters the state of overload. To leave that state, CPU should remain below predefined low threshold for predefined `cooldown` period.

The goal of the overload protection is to skip minimal amount of operations of statistical sends and updates to reduce CPU consumption to the acceptable level.

> **Tip**
>
> The following statistical categories are not affected by overload protection:
>
> - `CurrentTargetState`
> - `CurrentState`
> - `CurrentStateReasons`

Configuration Options

The following new configuration options are added to Stat Server starting with release 8.5.108:

Option	Summary
allow-new-connections-during-overload	Allows new clients to connect during overload.
allow-new-requests-during-overload	Allows opening new statistics during overload.
cpu-cooldown-cycles	The number of cpu-pool-timeout cycles in a cooldown period (Cooldown period / cpu-poll-timeout).
cpu-poll-timeout	Timeout of polling main thread CPU, in seconds.
cpu-threshold-high	The higher boundary of the load range.
cpu-threshold-low	The lower boundary of the CPU range.
cut-debug-log	Controls the debug log in overload.
protection	Enables/disables protection.
qos-default-overload-policy	Default overload policy.
qos-recovery-enable-lms-messages	Enables recovery-related LMS messages.

The above options are configured in the **[overload]** section of the Stat Server application.

The overload policy may vary from statistic to statistic, depending on the end-user preferences. The default overload policy, defined by the qos-default-overload-policy option settings, can be overridden on the stat type level by the `DynamicOverloadPolicy` option in the **[<stat type>]** section:

Option	Values	Description
DynamicOverloadPolicy	• `0` (default) - sends and updates for requested statistics can be cut • 1 - only sends of statistics to Stat Server clients can be cut • 2 - nothing can be cut, Stat Server updates and sends all requested statistics.	Defines actions that Stat Server may apply to a given statistic to reduce the overload

> **Important**
>
> It is recommended to keep default low and high thresholds (60-80) and not raising them as it may make overload protection less effective.

LMS Messages

New LMS messages, associated with overload protection, are listed below:

- 10070|STANDARD|GCTI_SS_OVERLOAD_DETECT|Overload detected on %s (%d current CPU usage)
- 10071|STANDARD|GCTI_SS_OVERLOAD_END|Overload ended on %s (%d current CPU usage)
- 10072|STANDARD|GCTI_SS_OVERLOAD_RECOVERY_STARTED|Overload recovery started on %s (%d current CPU usage)
- 10073|STANDARD|GCTI_SS_OVERLOAD_RECOVERY_FAILED|Overload recovery failed on %s (%d current CPU usage)
- 10074|STANDARD|GCTI_SS_OVERLOAD_PROTECTION_ACTIVATED|Overload protection on %s activated
- 10075|STANDARD|GCTI_SS_OVERLOAD_PROTECTION_DEACTIVATED|Overload protection on %s deactivated

Important

- Messages 10070 and 10071 are recommended for operations monitoring.
- Messages 10072 and 10073 are for debugging purposes only, they are disabled by default.
- Messages 10074 and 10075 are generated when the protection configuration option changes its value (or at startup). We need this information in the standard log because the debug logging is cut, when Stat Server is in overload. These messages are for troubleshooting only.

See also Stat Server Deployment Guide for more information on LMS messages.

Performance Counters

The following table includes new performance counters:

Counter	Description
cpu	Main-thread CPU percentage (% of single processor)
pcpu	Process CPU percentage (% of total)
shc	stats hit count
shcs	stats hit count suppressed
clens	client events not sent
opc	overload periods count
opd	overload periods duration sec

Counter	Description
osn	overload stats normal
osns	overload stats not sent
osnu	overload stats not updated

Hot Standby (HA)

Genesys uses the term hot standby to describe the redundancy type in which a backup server application remains initialized, clients connect to both the primary and the backup servers at startup, and the backup server data is synchronized with the primary server. Data synchronization and existing client connections to the backup guarantee higher availability of a component.

Starting with release 8.5.109, Stat Server supports Hot Standby redundancy for Stat Servers operating in high availability (HA) mode on Windows and Linux platforms.

Introduction

In HA mode, primary and backup instances of Stat Server replicate historical statistics and their aggregates to each other. Therefore, there is no single point of failure, as if one Stat Server instance fails and is then restarted, it can restore the statistics and their aggregates from another Stat Server instance.

The communication between primary and backup Stat Server instances is performed within a session. In case of a disconnect with the subsequent reconnect, the backup instance tries to restore the previous session. If it succeeds then not the whole population of the statistics but just those that were opened during the disconnect, are replicated. If a session cannot be restored, the whole population of statistics is replicated.

Overview

At startup, both Stat Server instances "read" their configuration and establish connections to each other. When the backup Stat Server instance connects to the primary, both ensure that they are configured as HA pair. Both Stat Server instances run concurrently.

After the successful initialization, two Stat Server instances start synchronization of non-replicated stats. When a new statistic is successfully opened, it is added to a non-replicated collection. Each Stat Server instance pushes statistics to another instance along with their aggregates. This is done in chunks (see the chunk-size option) by the timer (see the chunk-timeout option). If for a receiving Stat Server a statistic is new, this statistic is opened. If a statistic is not new for a receiving Stat Server, then only aggregate synchronization is performed. Aggregates need to be synchronized, except for `JavaCategory`, where only statistic definitions are synchronized – that requires identically absolute or relative paths to the same extension on two Stat Server instances.

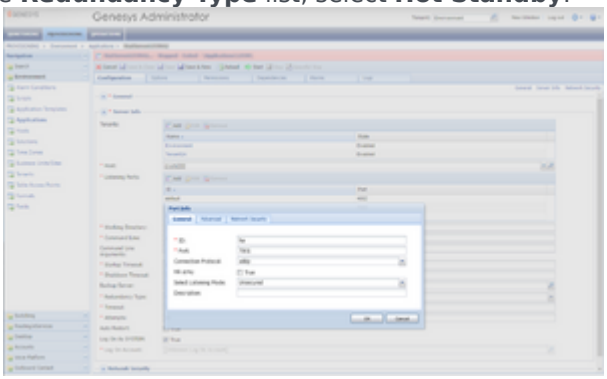
If some mandatory statistical attribute (such as object, filter, time profile, or time range) is deleted in the configuration, the associated Stat Server instance deletes this statistic but still waits for the acknowledgement from another instance.

If a connection between primary and backup Stat Servers is lost, the backup instance tries to reconnect (using frequency, configured as the value of the connect-timeout option) and restore a session. If the session is restored, only new statistics (statistics opened during the disconnect) are

replicated. Otherwise, all statistics are replicated.

Configuration

1. In Genesys Administrator, under **Provisioning** > **Environment** > **Applications**, select your primary Stat Server application.
2. In the **Server Info** section, in the **Backup Server** field, specify the backup Stat Server application.
3. In the **Redundancy Type** list, select **Hot Standby**.



Configure HA port for Stat Server instances
(Click thumbnail to enlarge.)

4. For **Listening Ports**, click **Add**. The **Port Info** dialog appears.
   In the **Port Info** dialog, configure the port for communication between the Primary and Backup Stat Server instances. The Stat Server instance configured as a backup always tries to connect to the **HA Port** that is set on the Stat Server instance configured as primary. You must:
   1. In the **ID** field, enter **ha**.
   2. Choose what connection protocol the Stat Server HA pair use; the following protocols are supported: `default`, `addp`, or `tls`.
      - For **default** connection protocol between the Primary and Backup Stat Servers:
        1. In the **Listening Mode** list, choose **Unsecured**,
        2. In the **Connection Protocol** field, enter **default**.
      - For **ADDP** connection protocol between the Primary and Backup Stat Servers:
        1. In the **Listening Mode** list, choose **Unsecured**,
        2. In the **Connection Protocol** field, enter **addp**,
        3. Configure the ADDP-specific options described in step 5.
      - For **TLS** connection protocol between the Primary and Backup Stat Servers:
        1. Ensure that trusted CA certificates are installed on the hosts where the HA Stat Servers are running. For more information about TLS connections and CA certificate generation, see the Secure Connections (TLS) section of the *Genesys Security Deployment Guide*.
        2. In the **Listening Mode** list, choose **Secured**,

3. In the **Connection Protocol** field, enter **default**.

- If the Connection Protocol for ha Listening Port is not specified, an ADDP connection protocol is established for HA Stat Servers.

Changes in ha port configuration take effect after Stat Server restart.

5. On the `Options` tab, configure relevant Hot Standby configuration options in the **[ha]** section, which contains the following options:
   - addp-remote-timeout
   - addp-timeout
   - addp-trace
   - chunk-size
   - chunk-timeout
   - connect-timeout
   - session-expiration-period
   - session-expiration-timeout

> **Important**
>
> New resources, for example: stat types, time profiles, time ranges, filters, can be dynamically added to Stat Server options. However, statistics, using those resources, should only be opened after making identical changes to **both** Stat Server instances of the HA pair. Otherwise, such statistics may not replicate properly.

Consistency

After all activity in the system is stopped, if both Stat Server instances are configured identically and connected, then they have exactly the same open historical statistics (see exceptions below), and the values for most of those statistics are closed in the HA pair.

Due to the way of how the aggregate synchronization procedure works the double-counting of statistics is possible, but it has a low probability, and even if it happens, its relative contribution is very small.

Special cases of a double-counting are possible during synchronization of `GroupBy` statistics, such as when one Stat Server instance is started later than the other one, so it did not see the user-data on a call, and created different child for the same call, and after sync, the same call counted twice for different user data keys.

Current statistics are not synchronized and can be different in two instances for the following reasons:

- One Stat Server instance "saw" a real action start, while another instance did not.

- One Stat Server instance did not "see" user data for the call, since it is not provided in the `EventRegistered`, while another instance "saw" it, as it connected to T-Server prior to when the call was started.

Historical custom formulas, having current component (for example: `SUM(CallInternal,duration)+@SUM(CallInternal;duration )`), can be out-of-sync for the same reasons, as for the current statistics above.

Only the historical component of the aggregate is replicated for the following statistics:
`Category=Formula`
`Subject=<Status>`
`Expression=<combination of current and historical component>`
As a result, such statistics might have different values in different instances of the same HA pair.

The primary and backup Stat Server instances might have an inconsistent statistic with the `Subject=DNAction` when an `Action` ends, if prior to that during the synchronization of Stat Server instances (while the `Action` was incomplete) one of the following conditions occurred:

- The last reset point of the `TimeProfile` was different in the primary and backup Stat Server instances.
- The start of the `Filter` validation was different in the primary and backup Stat Server instances.

Diagnostic

Starting with release 8.5.109, Stat Server logging is enhanced with the new `HA` value for the **[statserver]**/debug-level option. If the `HA` value is set, Stat Server logs messages related to HA functionality.

Protected and Normal Modes

In releases prior to 8.5.112.07, Stat Server reloads the entire configuration when it cannot restore the session with Configuration Server (normal mode). Both Stat Servers in a primary-backup pair can reload the configuration simultaneously and, therefore, the whole pair becomes unavailable, which adversely impacts Contact Center routing decisions.

Starting with release 8.5.112.07, in addition to normal mode Stat Server supports *protected mode* of configuration reloads for redundant pairs as a result of inability to restored the session with Configuration Server and after the `HistoryLogExpired` notification is received from Configuration Server. This functionality enables the Stat Server running as primary in the pair to delay configuration reloads and stay available until the backup Stat Server in the pair completes configuration reload. Backup or standalone instances of Stat Server behave as in normal mode, reloading the configuration unconditionally.

When the `HistoryLogExpired` notification is received from Configuration Server and the **[statserver]**/config-reload-delay-if-primary option is set to `true`, the primary Stat Server checks that the **[statserver]**/accept-clients-in-backup-mode option is set to `yes` in the backup Stat Server. If not, the primary instance reloads configuration immediately (as before 8.5.112.07 release). Otherwise, it delays configuration reload until the backup instance is up and running—it closes the connection to Configuration Server, switches to protected mode, and starts periodic checking for the backup instance readiness (by connecting to the backup instance's default client port). When the backup instance completes configuration reload and is ready to accept clients, the primary instance notifies Management Layer with the `SERVICE_UNAVAILABLE` status. In response, Management Layer initiates switchover and moves the current primary instance to backup mode. When the primary instance is switched to backup mode, it ends the protected mode, connects to Configuration Server, and reloads the configuration.

Stat Server generates log events when normal/protected mode starts and stops. Refer to Log Events (02-10084, 02-10085, 02-10086, and 02-10087) for more information.

Budget-Based Routing

Starting with release 8.5.110, Stat Server supports budget-based routing.

Overview

Budget-based routing is an optional addition to the existing capacity-based routing. Capacity-based routing allows to account for cross-media dependencies and limitations on the allowed number of interactions (such as, those imposed by Trade Unions). There are three possible routing schemas:

- **Capacity-based only** – This is the legacy approach. Budget-based routing is not enabled.
- **Hybrid** – Both capacity and budget-based routing are used. Budget-based routing must be enabled.
- **Budget-based** – Works when there are no cross-media dependencies. By specifying 255 as the maximum value in capacity rules, dependencies do not apply and budget limitations are used instead. To enabled the budget-based routing set the **[budget]**/enabled option on the **Options** tab in the **Advanced View (Annex)** of the Tenant object to `true`.

The following steps must be done prior to using budget-based routing:

- Determine all classes of interactions for a given media type.
- Assign a cost to each class of interactions.
- Assign agent budgets, based on known interaction costs.
- Add costs and budgets to the configuration.
- Modify a routing strategy to assign costs to interactions.
- Enable budget-based routing in the configuration.
- Make sure that the URS release is 8.1.400.50 or higher.

Each interaction can be assigned its unchangeable cost (using the routing strategy or Media Server) presented in the `UserData` key `InteractionCost` (or its override). The key name has to match the value of the interaction-cost-key option, recognized by URS. Stat Server notifies URS about available budgets in the extended `CurrentTargetState` statistics. URS takes available budgets into account when making a decision about the most suitable agent for the interaction with the specified cost. The agent (or place) has a particular budget assigned on the Tenant/MediaType/Agent or Place level with options in the **[budget]** section. Budgets on an agent and place are calculated independently, as well as capacity vectors.

> ### Important
>
> - Starting with release 8.5.110.20, Stat Server accepts dynamic interaction-cost related

content changes in UserData and recalculates cost-dependent statistics accordingly.

- Changes in the **default-cost-<call-type>** or interaction-cost-key options in the **[budget]** section in the Annex of the Tenant object as well as the **default-cost-<call-type>** options in the **[budget]** section in the Annex of the MediaType object do not cause immediate re-computation of interaction costs for existing interactions. However, in handling the subsequent changes to UserData of existing interactions, new values of options will be used.
- To have consistent results, Genesys recommends to use the UpdateUserData method (instead of the AttachUserDat method) in routing strategy to associate the cost with interaction.
- Explicitly assigned interaction-cost for an existing interaction has priority over the default-cost and does not change with default-cost changes.

Monitoring

CurrentTargetState Attributes

For agents only, the SCurrentTargetStateInfo attribute pExtensions contains (optionally) the embedded KV-List under the Budget key.

Starting with release 8.5.110.20, the TimeStamp attribute is added. The TimeStamp attribute specifies the moment of the last change in the corresponding media budget counters and is used by URS to select an agent. If two available agents have the same Used/Total/Avail counters for the particular media, then the one with the smaller TimeStamp is selected.

Unchangeable System Attributes

Starting with release 8.5.110.20, the numeric OrigInteractionCost system attribute is available for call actions on the following objects:

- DN
- Media Channel
- Queue
- Routing Point

When an action is started, the immutable OrigInteractionCost system attribute has the same value as the InteractionCost system attribute.

Example: The following statistic can be used to calculate the average interaction cost change, introduced by a given agent:

```
[Average_Cost_Change]
```

```
Category=Formula
Objects=Agent
Expression=AVG( CallInternal,CallInbound,CallOutbound,CallConsult,CallUnknown; abs( kvnum(
system, "InteractionCost")-kvnum( system, "OrigInteractionCost")) )
Formula=GetNumber( System, "budget_avail", -1 )
```

Changeable System Attributes

Starting with Stat Server release 8.5.110.20, the value of the `InteractionCost` system attribute can change. This numeric attribute was introduced in Stat Server release 8.5.110.03 and is available for call actions on the following objects:

- `DN`
- `Media Channel`
- `Queue`
- `Routing Point`

Example: The following statistic can be used to calculate the budget needed to handle all voice calls waiting in a given `Virtual Queue`:

```
[Required_Budget]
Category=CurrentCustomValue
Objects=Queue
MainMask=CallWait
Subject=DNAction
MediaType=voice
Formula=GetNumber( System, "InteractionCost", -1 )
```

The following numeric system attributes on `Routable/NonRoutable` actions are available for `Agents/Places/AgentGroups/PlaceGroups`:

- budget_total
- budget_used
- budget_avail
- budget_timestamp

Example: The following statistic can be used to calculate an available budget for a chat on a group of agents:

```
[Available_Chat_Budget]
Category=CurrentCustomValue
Objects=GroupAgents
MainMask=Routable
Subject=DNAction
MediaType=chat
Formula=GetNumber( System, "budget_avail", -1 )
```

Logging

The following LMS messages are available for the budget-based routing nitifications:

- 10077|STANDARD|GCTI_SS_BUDGET_MODEL_ENABLED|Budget model is enabled on tenant '%s'
- 10078|STANDARD|GCTI_SS_BUDGET_MODEL_DISABLED|Budget model is disabled on tenant '%s'

When changed, budget related parameters are printed in the debug log for the following configuration objects:

- Tenant
- Media-type
- Agent
- Place

For agents and places, the budget statistic is printed in the debug log, along with the capacity information.

Appendix A: Statistical Formula Definition

Parameter	Possible Value
formula	• <aggregate>
aggregate	• <aggr_boolean> ? <aggregate> : <aggregate> • <aggregate> ^ <aggregate> • <aggregate> % <aggregate> • <aggregate> + <aggregate> • <aggregate> - <aggregate> • <aggregate> * <aggregate> • <aggregate> / <aggregate> • - <aggregate> • (<aggregate>) • <system_proc1> (<aggregate>) • <system_proc2> (<aggregate> , <aggregate>) • greatest (<aggregate_list>) • least (<aggregate_list>) • <aggregate_func> (<action_types_list> ; <number> ; <condition>) • <aggregate_func> (<action_types_list> ; <number>) • COUNT (<action_types_list> ; <condition>) • COUNT (<action_types_list>) • @COUNT (<action_types_list> ; <condition>) • @COUNT (<action_types_list>) • <float literal> * <int literal> * <sys_var>
aggr_boolean	• <aggr_boolean> & <aggr_boolean> • <aggr_boolean> | <aggr_boolean>

Parameter	Possible Value
	• ~ <aggr_boolean> • (<aggr_boolean>) • <aggregate> = <aggregate> • <aggregate> != <aggregate> • <aggregate> < <aggregate> • <aggregate> <= <aggregate> • <aggregate> > <aggregate> • <aggregate> >= <aggregate>
aggregate_list	• <aggregate> • <aggregate> , <aggregate_list>
action_operand	• <action type> • ~ <action_type> • *
action_types_list	• <action_operand> • <action_types_list> , <action_operand>
condition	• <boolean>
boolean	• <boolean> & <boolean> • <boolean> | <boolean> • ~ <boolean> • (<boolean>) • <kv_boolean> • <number> = <number> • <number> != <number> • <number> < <number> • <number> <= <number> • <number> > <number> • <number> >= <number>

Parameter	Possible Value
	• <string> = <string> • <string> != <string> • rank_lt (<number> , <int literal>) • rank_gt (<number> , <int literal>)
number	• <boolean> ? <number> : <number> • greatest (<number>_list) • least (<number>_list) • <number> ^ <number> • <number> % <number> • <number> + <number> • <number> - <number> • <number> * <number> • <number> / <number> • - <number> • (<number>) • <system_proc1> (<number>) • <system_proc2> (<number> , <number>) • <num_attr> • <kv_number> • <float literal> • <int literal> • convert (<string>) • <sys_var>
number_list	• <number> • <number> , <number_list>
num_attr	• duration • start • end

Parameter	Possible Value
string	• <string> + <string> • <kv_string> • sub (<string> , <int literal> , <int literal>) • upper (<string>) • lower (<string>) • <string_literal>
kv_string	• kvstr (<kv_attrib> , <string>) • kvstr (<kv_attrib> , <string> , <int>)
kv_number	• kvnum (<kv_attrib> , <string> , <int>) • kvnum (<kv_attrib> , <string>) • kvsum (<kv_attrib> , <string>) • kvmin (<kv_attrib> , <string>) • kvmax (<kv_attrib> , <string>) • kvavg (<kv_attrib> , <string>)
kv_boolean	• kvexists (<kv_attrib> , <string> , <string>)
kv_attrib	• <kv_atomic_attrib> • kvlist (<kv_attrib> , <string>) • kvlist (<kv_attrib> , <string> , <int>)
kv_atomic_attrib	• udata • global_udata • reasons • extensions • system
int	• <int literal> • - <int literal>

Parameter	Possible Value
system_proc1	• log • exp • sqrt • sin • cos • abs • ceil • floor
system_proc2	• pow
aggregate_func	• SUM • MAX • MIN • AVG • @SUM • @MAX • @MIN • @AVG
sys_var	• $istart