Genesys Mobile Services API Reference

Stat Service API

5/8/2025

Stat Service API

Modified in 8.5.114

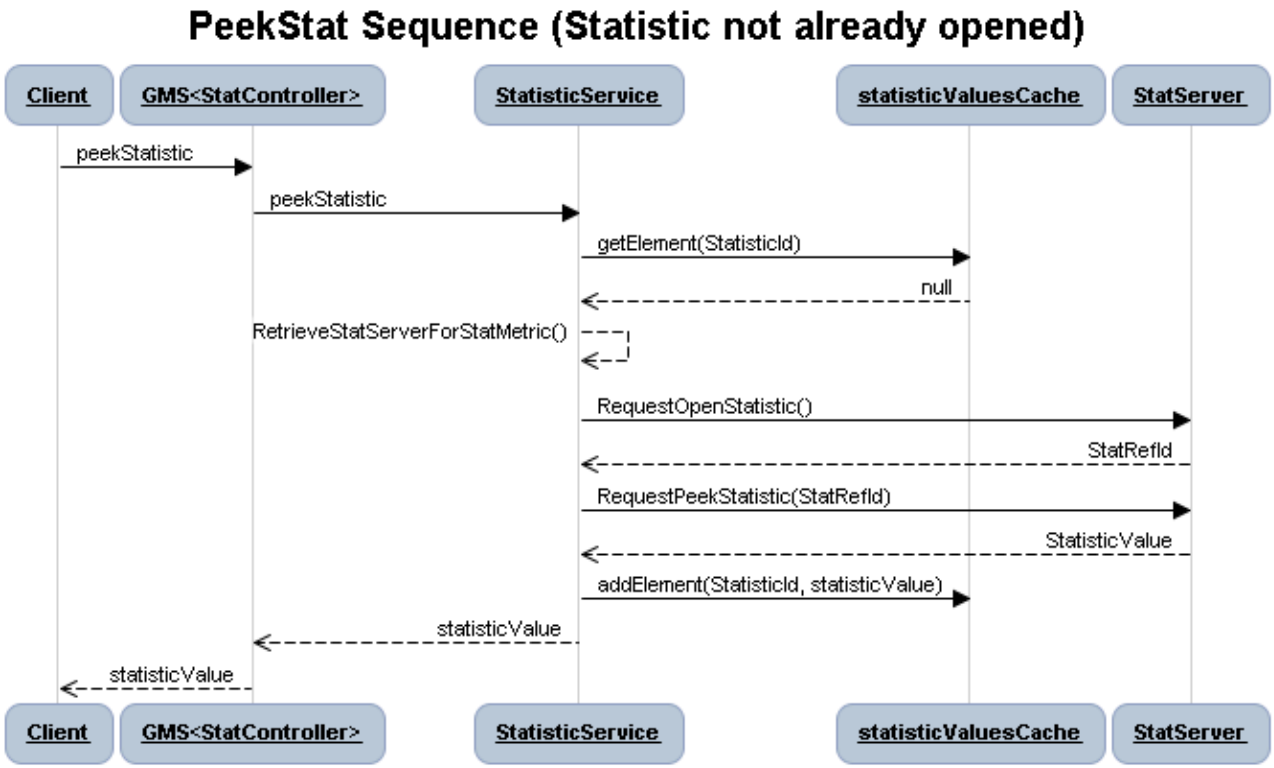
This API retrieves statistics from the Genesys Statistics Server (Stat Server) and URS.

> **Important**
> You need an Authentication header for this service.

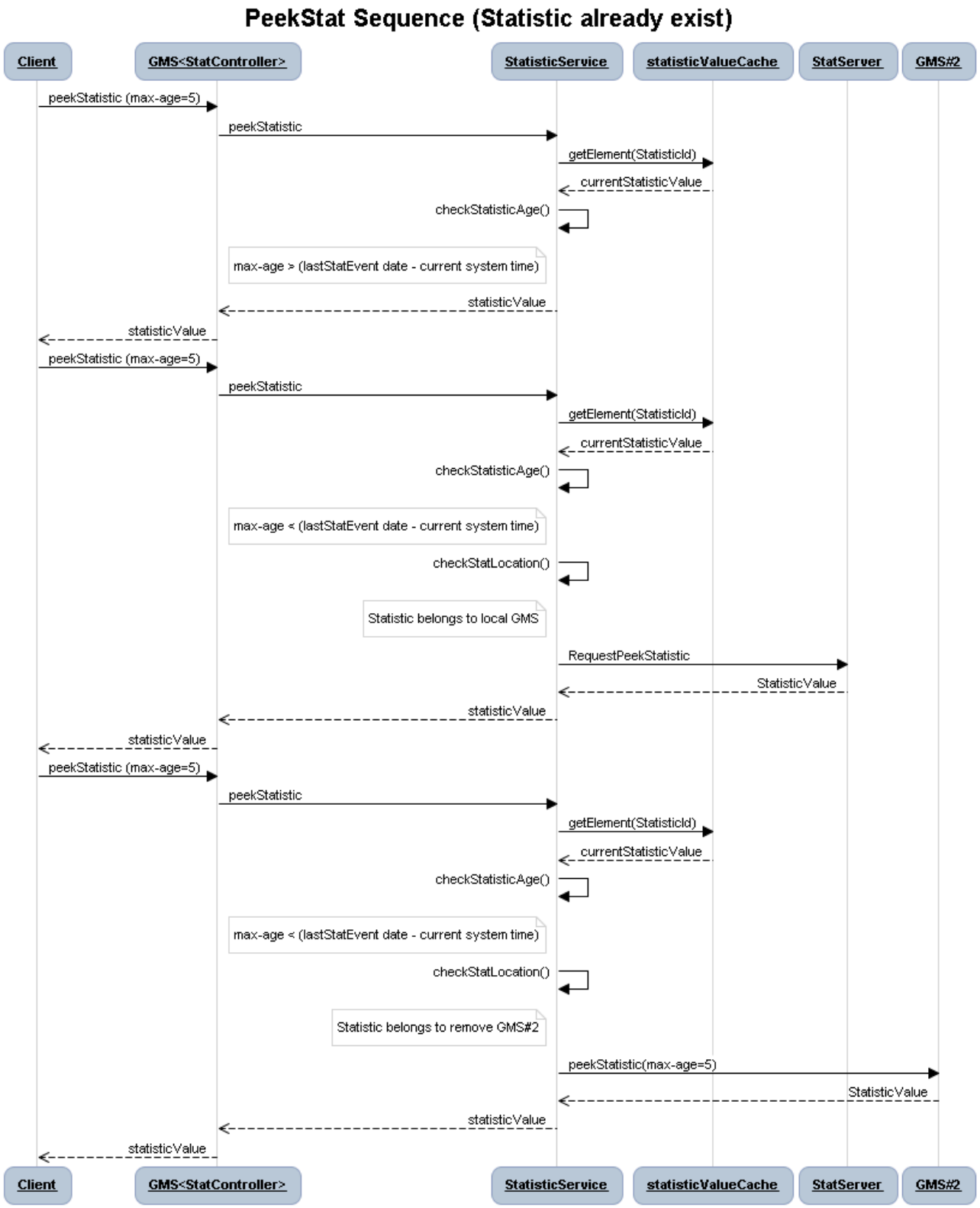
Stat Request APIs

Sequence Diagrams

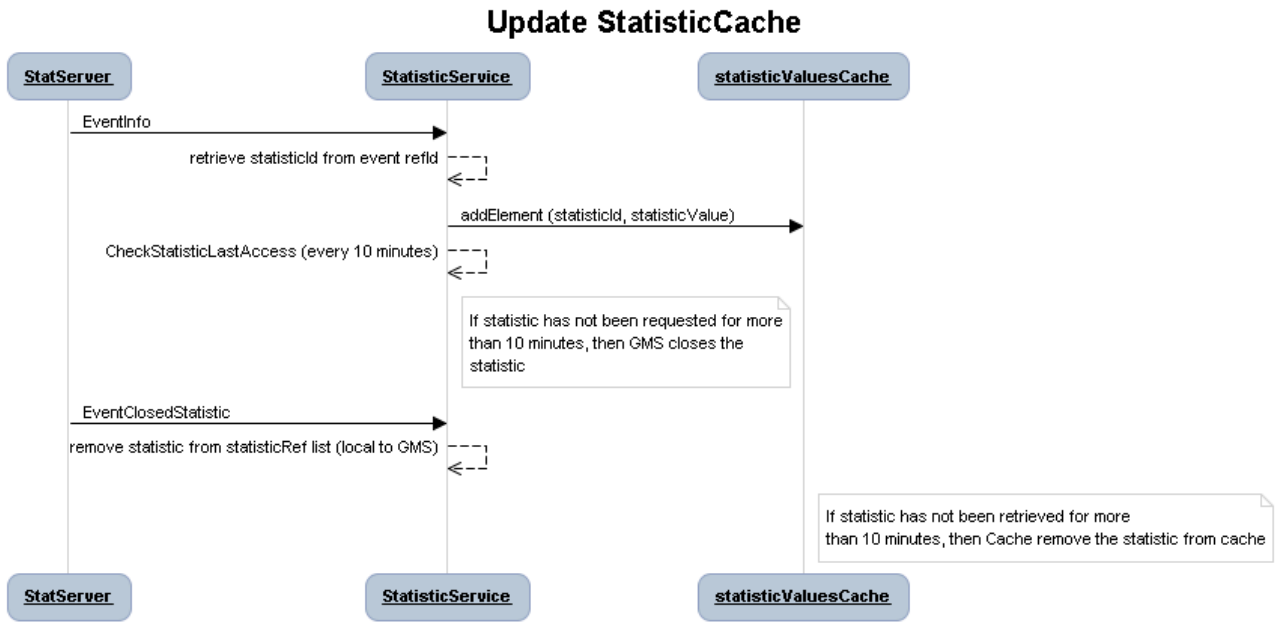
[+] Show Peek Stat Sequence (Statistic is not already opened)



[+] Show Peek Stat Sequence (Statistic already exists)


PeekStat Sequence (Statistic already exist)
Client
GMS<StatController>
StatisticService
statisticValueCache
StatServer
GMS#2
peekStatistic (max-age=5)
peekStatistic
getElement(StatisticId)
currentStatisticValue
checkStatisticAge()
max-age > (lastStatEvent date - current system time)
statisticValue
statisticValue
peekStatistic (max-age=5)
peekStatistic
getElement(StatisticId)
currentStatisticValue
checkStatisticAge()
max-age < (lastStatEvent date - current system time)
checkStatLocation()
Statistic belongs to local GMS
RequestPeekStatistic
StatisticValue
statisticValue
statisticValue
peekStatistic (max-age=5)
peekStatistic
getElement(StatisticId)
currentStatisticValue
checkStatisticAge()
max-age < (lastStatEvent date - current system time)
checkStatLocation()
Statistic belongs to remove GMS#2
peekStatistic(max-age=5)
StatisticValue
statisticValue
statisticValue
Client
GMS<StatController>
StatisticService
statisticValueCache
StatServer
GMS#2

[+] Show Peek Stat Sequence (Update Cache)



Accessing the APIs

The Stat Server API exposes two interfaces:

- "**genesys/1/internal_statistic**" — for internal access with no authentication control.
- "**genesys/1/statistic**" — for external use, which uses Basic Access Authentication (BA) to authenticate users.

As a standard protocol, the username and password for BA can be passed in the URL, as shown here:

```
http://username:password@127.0.0.1:8080/genesys/1/statistic
```

This API provides the following queries:

- PeekStat
- URS Stat

> **Important**

> See the Extended API to query multiple statistics.

PeekStat Request

This request fetches one statistic value.

Operation

Method	POST		
URL	/genesys/1/statistic		
Parameter	**Type**	**Mandatory**	**Description**
Header Parameters			
Cache-Control : max-age=XX	A number of seconds	no	The max-age value used to check if GMS has to recalculate the statistic. If the peek statistic time window is greater than max-age, GMS recalculates the statistic value. If the value is not present, it returns the latest value in cache.
Body			
objectId, objectType, tenant, metric (or statisticType), filter, notificationMode, timeProfile, timeRange, timeRange2		yes	The body can be either a `MultiPart` form or an `x-www-form-urlencoded` form consisting of different items representing the key/value pairs associated with the statistic (`objectId`, `objectType`, `tenant`, `tenantPassword`, `metric`, `notificationMode`, `filter`, `timeProfile`, `timeRange`, `timeRange2`). • `NotificationMode` can be `NoNotification`, `Reset`, or `Immediate`. • `filter` is the business attribute value to use to filter the results.

Response

HTTP code	200
HTTP Message	OK
HTTP Header	Content-Type: application/json;charset=UTF-8
Body	A JSON object representing a structure with value

If a problem occurs during operation, the following status codes are returned:

HTTP code	HTTP Message	HTTP Body
400	Bad Request	message: {"message":"Notification Mode is unknown","exception":"com.genesyslab.gsg.serv
401	Unauthorized	Access to API refused
403	Forbidden	message: {"message":"Object type not valid","exception":"com.genesyslab.gsg.services
403	Forbidden	message: {"message":"Agent 'Kippola' (Tenant 'Environment') not found","exception":"com.genesyslab.gsg.service
403	Forbidden	message: {"message":"Wrong parameter","exception":null}
500	Server Error	message: {"message":"Metric (TotalLoginTime;) not found on running StatServer","exception":"com.genesyslab.gsg.se
500	Server Error	message: {"message":"Statistic Service bad parameter for tenant","exception":"com.genesyslab.gsg.servic

> **Important**
>
> You can make this request without using authentication by replacing **/genesys/1/statistic** with **/genesys/1/internal_statistic**.

Example

Request

```
$ curl -v -u default:password -X POST --data
"metric=TotalLoginTime&objectType=Agent&objectId=KSippola&tenant=Environment"
http://localhost:8080/genesys/1/statistic
```

```
Content-Type: application/x-www-form-urlencoded
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
{"value":4963}
```

URS Stat Request

This API allows you to query URS statistics with GMS.

To use this request, you must add the following configuration information:

- URS version must be 8.1.400.12 or higher.
- In your URS application, allow HTTP requests:
  1. Open your URS application in Genesys Administrator or Configuration Manager.
  2. In the `Server Info` tab, create an HTTP listening port.
- In the reporting section of your GMS application, create the `_urs_url` option and set it to the URL of URS.

You can also create a **builtin** service to retrieve URS statistics with the urs-stat template. When you create your service, select the "urs-stat" type in the Admin UI service panel. See the Admin UI Help for further details. Then, to retrieve the statistics, you can make the following request:

```
GET http://<gms_home>:<gms_port>/genesys/1/service/<urs_service>
```

Operation

Method	GET		
URL	/genesys/1/statistic/urs/***parameters***		
Parameter	**Type**	**Mandatory**	**Description**
Header Parameters			
Cache-Control : max-age=XX	A number of seconds	no	The max-age value used to check if GMS has to recalculate the statistic. If the peek statistic time window is greater than max-age, GMS recalculates the statistic value. If the value is not present, it returns the latest value in cache.
URL Parameters			
parameters		no	For information on these parameters, either refer

Method	GET		
			to the URS documentation or send the following HTTP request to URS: http://URS_host:URS_HTTP_port/urs/help/

Response

HTTP code	200
HTTP Message	OK
HTTP Header	Content-Type: application/json;charset=UTF-8
Body	A JSON object representing a structure with value

If a problem occurs during operation, the following status codes are returned:

HTTP code	HTTP Message	HTTP Body
401	Unauthorized	message: "Authentication failed: username and/or password is incorrect"
500	Server error	message: {"message":"Wrong request","exception":"org.apache.http.HttpExcep
500	Server error	message: {"message":"Call not found","exception":"org.apache.http.HttpExcepti

Note: You can make this request without using authentication by replacing **/genesys/1/statistic/urs/** with **/genesys/1/internal_statistic/urs/**.

Example

Request

```
$ curl -v -u default:password "http://localhost:8080/genesys/1/statistic/urs/stat/targetstate?tenant=Environment=KSippola.A&json=&ext="
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json
{
    "dnsall":6,
    "voice":1,
    "status":4,
    "loading":0,
    "dns":
    [
        {
            "type":38,
            "number":"KSippola",
            "switch":"",
            "status":4,
```

```
            "loading":0,
            "ready":1,
            "media":1,
            "loggedin":0
        },
        {
            "type":36,
            "number":"KSippola",
            "switch":"",
            "status":2,
            "loading":0,
            "ready":0,
            "media":1,
            "loggedin":0
        },
        {
            "type":61,
            "number":"KSippola",
            "switch":"",
            "status":4,
            "loading":0,
            "ready":1,
            "media":1,
            "loggedin":0
        },
        {
            "type":1,
            "number":"7001",
            "switch":"SIP_Switch",
            "status":4,
            "loading":0,
            "ready":1,
            "media":0,
            "loggedin":0
        },
        {
            "type":1,
            "number":"7001_IM",
            "switch":"SIP_Switch",
            "status":8,
            "loading":0,
            "ready":0,
            "media":0,
            "loggedin":0
        },
        {
            "type":38,
            "number":"7001_IM",
            "switch":"SIP_Switch",
            "status":8,
            "loading":0,
            "ready":0,
            "media":0,
            "loggedin":0
        }
    ],
    "login":"SIP1",
    "place":"SIP_Server_Place1",
    "ready":1,
    "agent":"KSippola",
    "dnsrdy":3
}
```

Extended Stat Request API

- PeekStat Request: Retrieve configured statistics
- PeekStat Request: Querying Multiple Statistic Values in a Single Request
- PeekStat Request: Filtering Multiple Statistic Values in a Single Request

PeekStat Request: Retrieve configured statistics

Starting with 8.5.101, you can provision statistics in the `stat` section of your configuration options instead of passing properties as parameters of the statistics requests.

For instance, you can define `stat1` in your configuration as follows:

```
[stat.stat1]
_type=builtin
_service=statistic
metric=TotalLoginTime
notificationMode=Immediate
objectId=KSippola
objectType=Agent
tenant=Environment
filter=Bronze
```

Then, you can retrieve the statistics for `stat1` with a simple statistics query:

```
POST http://localhost:8080/genesys/1/statistic/stat1
```

Response:

```
{"value":41889}
```

Operation

Method	POST		
URL	/genesys/1/statistic/**stat_name**		
Parameter	**Type**	**Mandatory**	**Description**
Header			
Cache-Control: max-age	Integer	no	The `max-age` value in seconds of the statistic. If the peek statistic time window is greater than `max-age`, GMS recalculates the statistic value. If `max-age` is not

Method	POST		
			specified, GMS returns the latest value in the cache.
URL parameters			
stat_name	String	yes	Name of the statistic defined in the `stat` section of your configuration.
Body			
objectId, objectType, tenant, metric (or statisticType), filter, notificationMode, timeProfile, timeRange, timeRange2		yes	The body can be either a `MultiPart` form or an `x-www-form-urlencoded` form consisting of different items representing the key/value pairs associated with the statistic (`objectId`, `objectType`, `tenant`, `tenantPassword`, `metric`, `notificationMode`, `filter`, `timeProfile`, `timeRange`, `timeRange2`). • `NotificationMode` can be `NoNotification`, `Reset`, or `Immediate`. • `filter` is the business attribute value to use to filter the results.

Response

HTTP code	200
HTTP Message	OK
HTTP Header	Content-Type: application/json;charset=UTF-8
Body	A JSON object representing the statistic.

If a problem occurs during operation the following status codes are returned:

HTTP code	HTTP message	Body
400	Bad request	`message:`

HTTP code	HTTP message	Body
		{"message":"stat.total-time not found","exception":"com.genesyslab.gsg.se check GMS options
401	Unauthorized	Access to API refused

PeekStat Request: Querying Multiple Statistic Values in a Single Request

This request gets the current values of several peek statistic objects.

Operation

Method	POST		
URL	/genesys/1/statistics		
Parameter	**Type**	**Mandatory**	**Description**
Body			
objectId, objectType, tenant, metric (or statisticType), filter, notificationMode, timeProfile, timeRange, timeRange2		yes	The body can be either a `MultiPart` form, an `x-www-form-urlencoded` form, or a JSON object consisting of different items representing the key/value pairs associated with the statistic (`objectId`, `objectType`, `tenant`, `tenantPassword`, `metric`, `notificationMode`, `filter`, `timeProfile`, `timeRange`, `timeRange2`). • `NotificationMode` can be `NoNotification`, `Reset`, or `Immediate`. • `filter` is the business attribute value to use to filter the results.

Response

HTTP code	200
HTTP Message	OK
HTTP Header	Content-Type: application/json;charset=UTF-8
Body	A JSON object representing a structure with value

If a problem occurs during operation the following status codes are returned:

HTTP code	HTTP message	Body
200	OK	You get an error message instead of one of the expected values if the requested statistic is incorrect; for example: `message: {"stat1":8940,"stat2":"do not have 5 semicolon delimiters"}` This means that, instead of getting a 400 Bad request response, you are able to retrieve statistics, even if one of them is incorrect.
400	Bad request	`message: message: {"message":"Error for stat1: Agent 'Kippola' (Tenant 'Environment') not found","exception":"com.genesyslab.gsg.se`
401	Unauthorized	`Access to API refused`
403	Forbidden	`message: {"message":"Wrong parameter","exception":null}`
500	Server Error	`message: {"message":"Metric (CurrentGrouargetState) not found on running StatServer","exception":"com.genesyslab.g`

> **Important**
>
> You can make this request without using authentication by replacing **/genesys/1/statistics** with **/genesys/1/internal_statistics**.

Examples with Multiple Statistics

The following example uses `application/x-www-form-urlencoded`:

Request

```
$ curl -v -u default:password -X POST --data
"stat1=TotalLoginTime;Agent;KSippola;Environment;Immediate;&stat2=CurrentGroupTargetState;GroupAgents;Billing;E
http://localhost:8080/genesys/1/statistics
```

```
Content-Type: application/x-www-form-urlencoded
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
{
    "stat1":248,
    "stat2":
    [
        {
            "agentDbId":102,
            "placeDbId":102,
            "agentId":"KSippola",
            "placeId":"SIP_Server_Place1",
            "extensions":
            [
                {"VOICE_MEDIA_STATUS":4},
                {"AGENT_VOICE_MEDIA_STATUS":4},

{"DEVCAP":"AAIAAAAAAAEABXZvaWNlAAAAAAAAAAEAAAABVxTnEgAAAAAAQAAAAIABXZvaWNlAAAAAAAAAAAAAAAAAVxTnEgAEY2hhdAAAAAAA
AAAAAFcU5xIAAAAA"}
            ],
            "mediaCapacityList":
            [
                {"mediaType":"chat",
                    "currentInteractions":0,
                    "maxInteractions":1,
                    "currentMargin":1,
                    "timestamp":1460961964},
                {"mediaType":"vmail",
                    "currentInteractions":0,
                    "maxInteractions":0,
                    "currentMargin":0,
                    "timestamp":1460554482},
                {"mediaType":"voice",
                    "currentInteractions":0,
                    "maxInteractions":1,
                    "currentMargin":1,
                    "timestamp":1460961964},
                {"mediaType":"webcallback",
                    "currentInteractions":0,
                    "maxInteractions":1,
                    "currentMargin":1,
                    "timestamp":1460961964}
            ],
                "dnstatusExList":
            [
                {"dnId":"7001",
                    "gswDnTypes":1,
                    "mediaCapacityList":
                    [
                        {"mediaType":"voice",
                            "currentInteractions":0,
                            "maxInteractions":1,
                            "currentMargin":1,
                            "timestamp":1460987666}
                    ],
                    "multimedia":0,
                    "status":4,
                    "switchId":"SIP_Switch",
```

```
                    "time":1460554482},
                {"dnId":"7001_IM",
                    "gswDnTypes":1,
                    "mediaCapacityList":
                    [
                        {"mediaType":"voice",
                            "currentInteractions":0,
                            "maxInteractions":0,
                            "currentMargin":0,
                            "timestamp":1460987666},
                        {"mediaType":"chat",
                            "currentInteractions":0,
                            "maxInteractions":255,
                            "currentMargin":0,
                            "timestamp":1460987666}
                    ],
                    "multimedia":1,
                    "status":8,
                    "switchId":"SIP_Switch",
                    "time":1460553961}
            ]
        }
    ]
}
```

The following example uses JSON:

Request

```
$ curl -v -u default:password -H "Content-Type: application/json" -X POST --data
'{"stat1":{"objectId":"KSippola","objectType":"Agent","tenant":"Environment","tenantPassword":"","metric":"Tota
http://localhost:8080/genesys/1/statistics

Content-Type: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8

{
    "stat1":3426,
    "stat3":
    [
        {"agentDbId":102,
            "placeDbId":102,
            "agentId":"KSippola",
            "placeId":"SIP_Server_Place1",
            "extensions":
            [
                {"VOICE_MEDIA_STATUS":4},
                {"AGENT_VOICE_MEDIA_STATUS":4},

{"DEVCAP":"AAIAAAAAAAEABXZvaWNlAAAAAAAAAAEAAAABVxTzkwAAAAAAQAAAAIABXZvaWNlAAAAAAAAAAAAAAAAAVxTzkwAEY2hhdAAAAAAA
AAAAAFcU85MAAAAA"}
            ],
            "mediaCapacityList":
            [
                {"mediaType":"chat",
                    "currentInteractions":0,
```

```
                    "maxInteractions":1,
                    "currentMargin":1,
                    "timestamp":1460961964},
                {"mediaType":"vmail",
                    "currentInteractions":0,
                    "maxInteractions":0,
                    "currentMargin":0,
                    "timestamp":1460554482},
                {"mediaType":"voice",
                    "currentInteractions":0,
                    "maxInteractions":1,
                    "currentMargin":1,
                    "timestamp":1460961964},
                {"mediaType":"webcallback",
                    "currentInteractions":0,
                    "maxInteractions":1,
                    "currentMargin":1,
                    "timestamp":1460961964}
            ],
            "dnstatusExList":
            [
                {"dnId":"7001",
                    "gswDnTypes":1,
                    "mediaCapacityList":
                    [
                        {"mediaType":"voice",
                            "currentInteractions":0,
                            "maxInteractions":1,
                            "currentMargin":1,
                            "timestamp":1460990867}
                    ],
                    "multimedia":0,
                    "status":4,
                    "switchId":"SIP_Switch",
                    "time":1460554482},
                {"dnId":"7001_IM",
                    "gswDnTypes":1,
                    "mediaCapacityList":
                    [
                        {"mediaType":"voice",
                            "currentInteractions":0,
                            "maxInteractions":0,
                            "currentMargin":0,
                            "timestamp":1460990867},
                        {"mediaType":"chat",
                            "currentInteractions":0,
                            "maxInteractions":255,
                            "currentMargin":0,
                            "timestamp":1460990867}
                    ],
                "multimedia":1,
                "status":8,
                "switchId":"SIP_Switch",
                "time":1460553961}
            ]
        }
    ]
}
```

PeekStat Request: Filtering Multiple Statistic Values in a Single Request

> **Important**
> This is a JSON query. This JSON query is available since 8.5.004.05.

Operation

Method	POST		
URL	/genesys/1/statistics		
Parameter	**Type**	**Mandatory**	**Description**
Body: A JSON item representing the key/value pairs associated with statistics (objectId, objectType, tenant, tenantPassword, metric, notificationMode, filter). • filter is the business attribute value to use to filter the results. • notificationMode can be set to `NoNotification`, `Reset`, or `Immediate`.			

Response

HTTP code	200
HTTP Message	OK
HTTP Header	Age: containing the age (in seconds) of the statistic value.
Body	A JSON array of key/value pairs where key is the key stat (see request) and value is the statistic value.

If a problem occurs during subscription, the following status codes are returned:

HTTP code	HTTP Message	HTTP Body
400	Bad Request	The provided filter is incorrect; for example: `{"message":"Filter name is not found","exception":"com.genesyslab.gsg.se`
403	Forbidden	Part of the submitted data is incorrect, for example: `{"message":"Place 'SIP_Server_Place' (Tenant 'Environment') not found","exception":"com.genesyslab.gsg.se`

HTTP code	HTTP Message	HTTP Body
500	Internal server error	If Stat Server is not connected, it returns, for example, `{"message":"Statistic Service unavailable","exception":"com.genesyslab`

Example with Filter

Query

```
http://localhost:8080/genesys/1/statistics
```

Input: JSON File

```
{
"stat1": {
        "objectId":"KSippola",
        "objectType":"Agent",
        "tenant":"Environment",
        "tenantPassword":"",
        "metric":"TotalLoginTime",
        "filter":"Bronze"
 },
 "stat2": {
        "objectId":"9002@SIP_Switch",
        "objectType":"Queue",
        "tenant":"Environment",
        "metric":"ExpectedWaitTime"
        }
}
```

Response

```
 HTTP/1.1 200 OK
 Date: Thu, 19 Jun 2014 09:06:43 GMT
 Content-Type: application/json;charset=UTF-8
 Content-Type: application/json
 Content-Length: 25

{"stat1":0,"stat2":10000}
```

EWT APIs

Starting in 8.5.112, new Estimated Waiting Time (EWT) APIs are available to query EWT. They use different calculation types to provide the Estimated Waiting Time. These APIs return either a JSON object presentation that describes the information in the virtual queue or a collection of such objects (one per every matched virtual queue the call is in). GMS retrieves all of the results from URS.

> **Important**
> In your GMS configuration, add a connection to an active URS to enable this service.

To provide Estimated Waiting Time, three methods are available:

- ewt—URS checks how fast interactions go through the virtual queue and how many interactions are still pending. URS ignores the current agent availability and does not immediately adjust the EWT if, for example, all of the agents handling the queue suddenly logout.
- ewt2—Similarly to the first method, URS checks how fast interactions go through the virtual queue and how many interactions are pending. Additionally, URS takes into account the agents who have historically been handling interactions of the Virtual Queue. If all of these agents logout, URS notices and adjust the EWT value to a very high number like 10000 for example.
- ewt3—Query EWT from Stat Server.

Query-EWT for a Virtual Queue

Introduced in: 8.5.109
Modified in: 8.5.112

You can query Estimated Wait Time statistics if you have configured a Virtual Queue for your Callback service by using the _urs_virtual_queue option. Three calculation types are available and use a URS query to retrieve the Estimated Wait Time in seconds.

- ewt

```
http://<urshost>:<ursport>/urs/call/max/lvq?name=VQ_Name&aqt=urs&tenant=TenantName
```

- ewt2 (added in 8.5.112)

```
http://<urshost>:<ursport>/urs/call/max/lvq?name=VQ_Name&aqt=urs2&tenant=TenantName
```

- ewt3 (added in 8.5.112)

```
http://<urshost>:<ursport>/urs/call/max/lvq?name=VQ_Name&aqt=stat&tenant=TenantName
```

Operations

Name	Type	Mandatory	Description
• **GET /genesys/1/ewt/{VQ-Name}** • **GET /genesys/1/ewt2/{VQ-Name}** • **GET /genesys/1/ewt3/{VQ-Name}**			
URI Parameters			

Name	Type	Mandatory	Description
{VQ-Name}	string	Yes	Alias name of a virtual queue configured in a service. If you are using a version older than 8.5.114, the VQ alias name must not include spaces. If your VQ alias name includes spaces, you should replace spaces with underscores, which renames the alias. For example, `GMS Chat VQ Multimedia Switch` should be renamed `GMS_Chat_VQ_Multimedia_Switch`

Response

HTTP code	200
HTTP message	OK
Response Body (JSON content)	
Name	**Description**
See object description below.	JSON-formatted string of information per virtual queue. The `ewt` parameter provides the Estimated Waiting Time in seconds. If you did

The JSON objects describing information in the virtual queue have the following properties:

Attribute	Description
time	UTC timestamp when the call entered in the virtual queue.
wt	Time in seconds that the call is in the virtual queue (effectively CurrentTime-time).
calls	Current number of all calls in the virtual queue.
wcalls	Number of waiting calls in the virtual queue.
pos	Position of the call in the virtual queue.
priority	Calls priority in the virtual queue. Absent if scaling is used.
aqt	Average quitting rate of calls from the virtual queue. May be skipped if unknown.
ewt	Expected waiting time for the call.
hit	Percentage of calls distributed into the virtual queue. Its value is -1 if no calls were distributed yet.

In case of error, you will receive:

HTTP code	400
HTTP message	Not found

HTTP code	204
HTTP message	OK, but the record is empty.

Examples

```
GET http://localhost:8080/genesys/1/ewt/VQ_GMS_Callback

200 OK
{
    "hit": 0,
    "ewt": 28.4,
    "calls": 9,
    "pos": 1,
    "aqt": 10000,
    "wpos": 10,
    "time": 1506021690,
    "wt": 0,
    "wcalls": 9
}


GET http://localhost:8080/genesys/1/ewt2/VQ_GMS_Callback

200 OK
{
    "hit": 0,
    "ewt": 126.8,
    "calls": 9,
    "pos": 3,
    "aqt": 10000,
    "wpos": 10,
    "time": 1506021690,
    "wt": 0,
    "wcalls": 9
}


GET http://localhost:8080/genesys/1/ewt3/VQ_GMS_Callback

200 OK
{
    "hit": 0,
    "ewt": 35.2,
    "calls": 9,
    "pos": 10,
    "aqt": 10000,
    "wpos": 10,
    "time": 1506021690,
    "wt": 0,
    "wcalls": 9
}
```

Query-EWT for All Virtual Queues

Introduced in: 8.5.112

You can query Estimated Wait Time statistics if you have configured at least a Virtual Queue for one of your Callback service by using the _urs_virtual_queue option. Three calculation types are available and use a URS query to retrieve the Estimated Waiting Time in seconds.

- ewt

```
http://<urshost>:<ursport>/urs/call/max/lvq?aqt=urs&tenant=TenantName
```

- ewt2 (added in 8.5.112)

```
http://<urshost>:<ursport>/urs/call/max/lvq?aqt=urs2&tenant=TenantName
```

- ewt3 (added in 8.5.112)

```
http://<urshost>:<ursport>/urs/call/max/lvq?aqt=stat&tenant=TenantName
```

Operations

- **GET /genesys/2/ewt**
- **GET /genesys/2/ewt2**
- **GET /genesys/2/ewt3**

Response

HTTP code	200
HTTP message	OK
Response Body (JSON content)	
Name	**Description**
JSON array	JSON array of objects. Each object describes the information per virtual queue. See below for details.

The JSON objects describing information in the virtual queue have the following properties:

Attribute	Description
time	UTC timestamp when the call entered in the virtual queue.
wt	Time in seconds that the call is in the virtual queue (effectively CurrentTime-time).
calls	Current number of all calls in the virtual queue.
wcalls	Number of waiting calls in the virtual queue.

Attribute	Description
pos	Position of the call in the virtual queue.
priority	Calls priority in the virtual queue. Absent if scaling is used.
aqt	Average quitting rate of calls from the virtual queue. May be skipped if unknown.
ewt	Expected waiting time.
hit	Percentage of calls distributed into the virtual queue. Its value is -1 if no calls were distributed yet.

In case of error, you will receive:

HTTP code	400
HTTP message	Not found

HTTP code	204
HTTP message	OK, but the record is empty.

Examples

```
GET http://localhost:8080/genesys/2/ewt
```

```
200 OK
{
    "VQ_GMS_Callback_Out": {
        "time": 1506021728,
        "wt": 0,
        "calls": 0,
        "wcalls": 0,
        "pos": 1,
        "wpos": 1,
        "aqt": 9.999,
        "ewt": 9.999,
        "hit": 0
    },
    "VQ_GMS_Callback": {
        "time": 1506021728,
        "wt": 0,
        "calls": 9,
        "wcalls": 9,
        "pos": 10,
        "wpos": 10,
        "aqt": 10000,
        "ewt": 100000,
        "hit": 0
    },
    "VQ_CB": {
        "time": 1506021728,
        "wt": 0,
        "calls": 5,
        "wcalls": 5,
        "pos": 6,
        "wpos": 6,
        "aqt": 10000,
```

```
        "ewt": 60000,
        "hit": 0
    }
}

GET http://localhost:8080/genesys/2/ewt2

200 OK
{
    "VQ_GMS_Callback_Out": {
        "time": 1506021728,
        "wt": 0,
        "calls": 0,
        "wcalls": 0,
        "pos": 1,
        "wpos": 1,
        "aqt": 9.999,
        "ewt": 9.999,
        "hit": 0
    },
    "VQ_GMS_Callback": {
        "time": 1506021728,
        "wt": 0,
        "calls": 9,
        "wcalls": 9,
        "pos": 10,
        "wpos": 10,
        "aqt": 10000,
        "ewt": 100000,
        "hit": 0
    },
    "VQ_CB": {
        "time": 1506021728,
        "wt": 0,
        "calls": 5,
        "wcalls": 5,
        "pos": 6,
        "wpos": 6,
        "aqt": 10000,
        "ewt": 60000,
        "hit": 0
    }
}

GET http://localhost:8080/genesys/2/ewt3

200 OK
{
    "VQ_GMS_Callback_Out": {
        "time": 1506021728,
        "wt": 0,
        "calls": 0,
        "wcalls": 0,
        "pos": 1,
        "wpos": 1,
        "aqt": 9.999,
        "ewt": 9.999,
        "hit": 0
    },
    "VQ_GMS_Callback": {
        "time": 1506021728,
        "wt": 0,
        "calls": 9,
```

```
        "wcalls": 9,
        "pos": 10,
        "wpos": 10,
        "aqt": 10000,
        "ewt": 100000,
        "hit": 0
    },
    "VQ_CB": {
        "time": 1506021728,
        "wt": 0,
        "calls": 5,
        "wcalls": 5,
        "pos": 6,
        "wpos": 6,
        "aqt": 10000,
        "ewt": 60000,
        "hit": 0
    }
}
```


Configuration

Stat Server Connection

In order to use the Stat Server API, there must be a connection to Stat Server in the GMS Application. You can create the connection in Configuration Manager on the *Connections* tab. See Creating and Configuring the GMS Application Cluster Object.

You can add several Stat Servers in the *Connection* tab; this feature is used in case of different statistic definitions in the Stat Servers. GMS will open the statistic on the Stat Server to which the statistic belongs to. In the case of the same statistic definition in Stat Servers, GMS will take the first Stat Server that contains the statistic definition.

ADDP Setting

Open Stat Server in the GMS *Connection* tab and set the connection protocol to *addp*. Set the values for the Local Timeout and Remote Timeout, and then select the Trace mode. See Implementing ADDP for more information.

> **Important**
> The ADDP traces are only visible on the Stat Server side.

The following example shows an ADDPtrace in Stat Server logs:

```
-AP[10]-<-2168 @15:37:30.4540

-Ap[10]->-2168 @15:37:30.4560
```

High Availability

If Stat Servers (defined in GMS connections) are configured to use High Availability (HA) (Primary/Backup Stat Server), in the case of a lost connection with the primary Stat Server, the Stat Server API will switch to the backup Stat Server.

Subscribe to Statistic Server Event Notifications

You can receive event notifications sent by the Statistics Server by using the CometD channel. To do so:

1. Register for GMS CometD channel.
2. Subscribe for a list of statistics through the new API: **POST /genesys/2/statistics**.

> **Important**
> This Statistics API currently supports polling mode.

Register for GMS CometD Channel

Create a client that will listen to CometD events sent by GMS, as shown in the following Node JS example:

```
var faye = require("faye");
var request = require('request');
// subscribe GMS comet
var referenceId;
var client = new faye.Client("http://localhost:8080/genesys/cometd");
client.setHeader("gms_user", "123456");
var subscr = client.subscribe("/_genesys", function (message) {
    var stats = message["message"]
    var statReferenceId = stats["statReferenceId"]
    ///...
    client.disconnect();
});
```

Subscribe to Statistics through the Statistics API

Use the API to subscribe to a list of statistics and retrieve for each of them the reference ID of the Stat Server. Then, you will receive CometD event notifications associated with these reference IDs.

> **Important**

> You must subscribe with the same `gms_user` ID as the one used for the CometD client.

Operation

POST /genesys/2/statistics			
Parameter	**Type**	**Mandatory**	**Description**
Header Parameters			
Content-type		Yes	`application/json;charset=UTF-8`
gms_user	String (UUID)	Yes	The user ID that has been used in CometD at registration time.

Body: The body is a JSON structure that lists the statistics to get notifications for.

For example:

```
{
    "stat1": {
        "objectId": "kmilburn",
        "objectType": "Agent",
        "tenant": "Environment",
        "tenantPassword": "",
        "metric": "TotalLoginTime",
        "filter": ""
    },
    "stat2": {
        "objectId": "KSippola",
        "objectType": "Agent",
        "tenant": "Environment",
        "tenantPassword": "",
        "metric": "TotalLoginTime",
        "filter": "Bronze"
    }
}
```

Response

The response contains the reference IDs from the Stat Server subscribed statistics.

HTTP code	200
HTTP Message	OK
Body	A JSON array of key/value pairs, where: • The key is the name of the subscribed statistics. • The value is the reference ID of the Stat Server subscribed statistics.

For example:

```
200 OK
{"stat1":1958339548,"stat2":1329710246}
```

Errors

HTTP Code	500
HTTP Message	Server Error
Body	• {"message":"Could not read JSON: Unexpected character ('\"' (code 34)): [...]"}. • {"message":"Error for stat1: Supplied Object type does not belong to specified Stat Type.", "exception":"com.genesyslab.gsg.services.statistic.StatServerEventErrorException"}.

Example

The following node JS sample shows how to subscribe statistics with this API.

```
var request = require('request');
// send request to subscribe stat using gms_user of comet subscription
request({
    url: 'http://localhost:8080/genesys/2/statistics', //URL to hit
    headers: {
        'Content-Type': 'application/json',
        'gms_user': '123456'
    },
    method: 'POST',
    json: true, // output data is JSON
    json: {  // input data is JSON too
        "stat2": {
            "objectId": "KSippola",
            "objectType": "Agent",
            "tenant": "Environment",
            "tenantPassword": "",
            "metric": "TotalLoginTime",
            "filter": ""
        }
    }
}, function (error, response, body) {
    if (error) {
        ///...
    } else {
        referenceId = body["stat2"];
        ///...
    }
});
```

As a result of the subscription, you would receive the following reference IDs:

```
200 OK
{"stat1":1958339548,"stat2":1329710246}
```

As a result, CometD Polling responses include these reference IDs too:

```
[
  {"data":
  {"id":"8f8dc3a0d00311e5845a0de0f4fa8277",
  "message":
  {"statValue":"0","statReferenceId":"1958339548"},
  "tag":"service.statistic.push.1958339548"},
  "channel":"/_genesys"},
  {"id":"55","successful":true,"channel":"/meta/connect"}
]
[
  {"data":
  {"id":"9560da60d00311e5845a0de0f4fa8277",
  "message":
  {"statValue":"0","statReferenceId":"1329710246"},
  "tag":"service.statistic.push.1329710246"},
  "channel":"/_genesys"},
  {"id":"56","successful":true,"channel":"/meta/connect"}
]
```


Limitations

- CometD requests TTL is 120 seconds (by default).
- The statistics filter subscription is set to 10000 seconds, then the subscription is removed and you must send back the same stat request before the end of TTL subscription.
- You cannot define a statistic in Configuration Server for the GMS application that will be automatically subscribed at GMS startup. You must subscribe to the statistics as detailed above.