Callback User's Guide

Get EWT for Callback

5/4/2025

Get EWT for Callback

Modified in 8.5.109, 8.5.205

In your Callback service, you can configure some options to specify how Callback should query the Estimated Wait Time.

- See set up an EWT method for Callback.

To retrieve the Expected Wait Time (EWT), you have three possibilities:

- Directly query URS.
- Use the EWT API for virtual queues (added in 8.5.109).
- Use the urs-stat Service.

Set up an EWT method for Callback

In your Callback service, you can configure some options to specify how Callback should query the Estimated Wait Time. The following options are used to define when a request should be submitted to the Callback Orchestration execution service. You can either:

- Set the _request_queue_time_stat parameter to provide the Stat Server statistics. For example, you can use the ExpectedWaitTime statistic to set this option: "ExpectedWaitTime;Queue;8999@SIP_Server;Environment"
- Configure the Callback service option named `_request_ewt_service` to specify the `urs-stat` service to use to retrieve EWT value from URS stats.
- Configure the option _urs_virtual_queue or _urs_ewt_vq in your Callback service to be able to use the REST API as detailed below.

Query URS

To retrieve the Expected WaitTime (EWT) for callback, you can use the lvq method of the URS Web API and directly query the following URS URL:

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

To view additional URS `lvq` input parameters and output information, open a browser with URS running and run the help method for `lvq` as follows:

```
http://<urshost>:<ursport>/urs/help/call/lvq
```

The help method is described in the Universal Routing 8.1 Reference Manual, Appendix C, "Supported Methods."

Use the Callback EWT API for Virtual Queues

Configure the URS Virtual Queue

Introduced in 8.5.109
You can query Estimated Wait Time statistics if you configure a Virtual Queue for your Callback service by using the service option _urs_virtual_queue. Then, you can use the new REST API, Query EWT, as detailed in the Stat Service API page.

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

Configure another Callback Queue

Introduced in 8.5.205

If you configure different Virtual Queues for different calls, for example, scheduled callbacks, callbacks, and regular calls, those queues may share the same agent group. In this use case, URS cannot retrieve EWT for scheduled callbacks because it has its own queue and URS doesn't have any EWT info for this queue yet.

To avoid this issue, configure the correct VQ name to query for EWT in the _urs_ewt_vq option of your Callback service. If this option is configured, the system will use this specific queue to query EWT; if not, the system uses the queue defined in _urs_virtual_queue to query EWT as usual.

> ### Important
> You need to define this option for each service.

Use the urs-stat Service

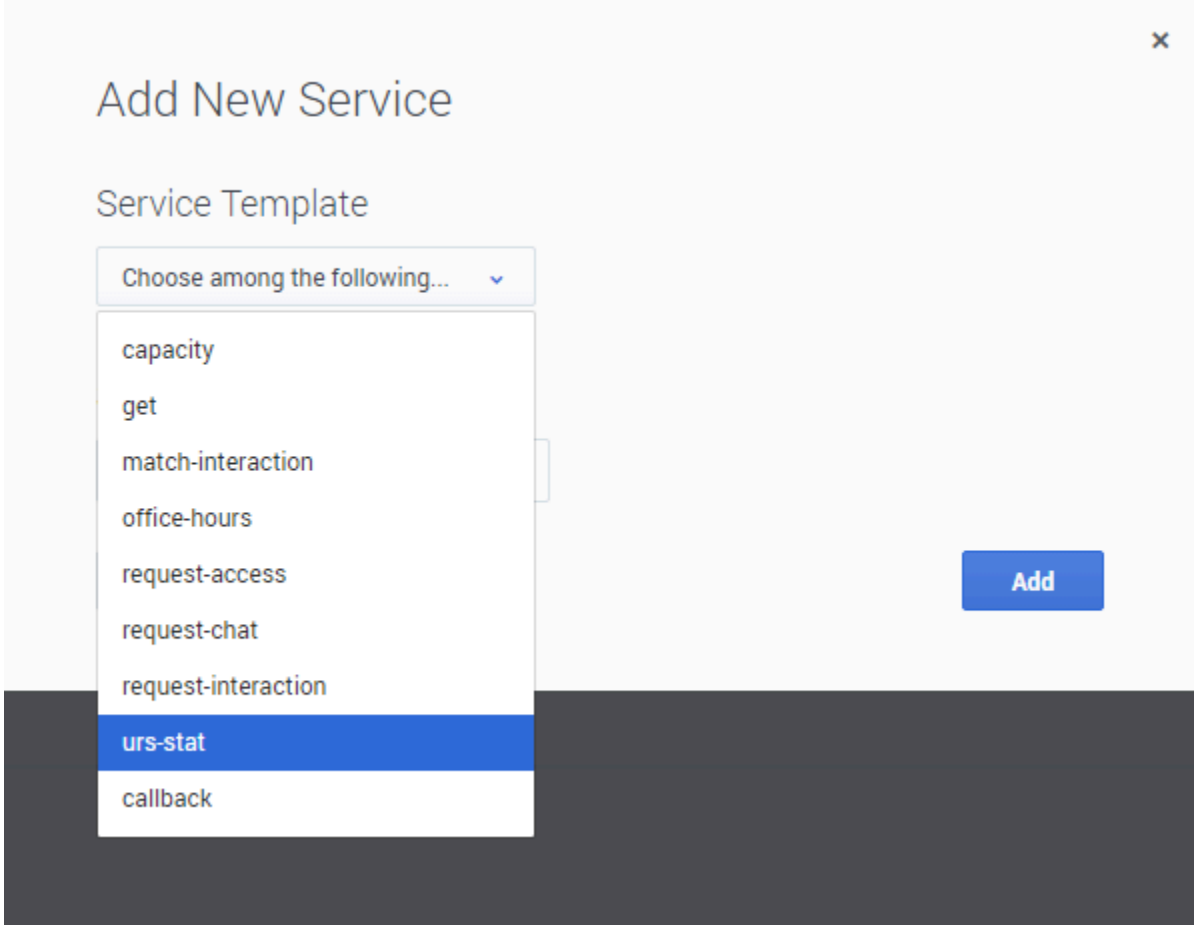
Create a GMS built-in service using the **urs-stat** template that provides the following benefits:

- Statistics caching of the statistic to reduce the load on URS. The `_caching_policy` parameter sets the cache period in seconds (see below).

- Load balancing and scaling across multiple GMS nodes.
- A single point of contact for your app.

Create a urs-stat Service

To create this GMS built-in service, select the **urs-stat** template when creating a new service.

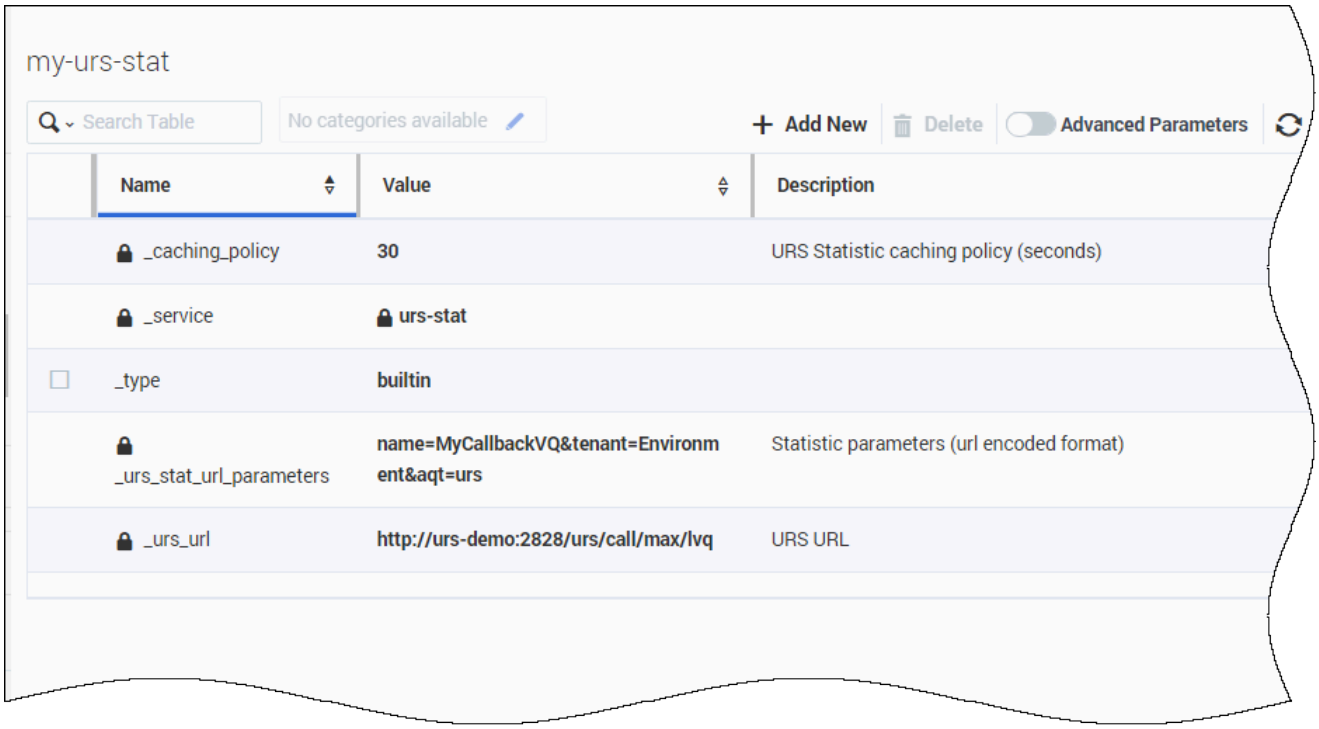


Configure urs-stat parameters

Configure the following parameters in your <name-of-urs-stat-service> service:

```
_caching_policy=30 # Cache refresh time in seconds
_service=urs-stat
_type=builtin
_urs_stat_url_parameters=name=<VQ_Name>&tenant=<Tenant_Name>&aqt=urs
_urs_url=http://<urshost>:<ursport>/urs/call/max/lvq
```

Where: `VQ_Name`, `Tenant_Name`, `urshost`, and `ursport` match the environment and Callback service's Virtual Queue (VQ). The following screenshot shows the creation and configuration of the **my-urs-stat** service.



Important

The `_urs_url` option can point to the load balancer in front of the URS that should be configured as part of the GMS provisioning steps in that scenario.

Query EWT using the urs-stat service

The following query example shows the resulting response that you get when you call the service:

```
GET http://<gmshost>:<gmsport>/genesys/1/service/<name-of-urs-stat-service>

Response:
{ "wcalls" : 20, "wpos" : 21, "time" : 1467922222, "hit" : 95, "calls" : 20,
 "wt" : 0, "ewt" : 300, "pos" : 21, "aqt" : 300 }
```

Important

- The value of interest here is `ewt`: the time unit is seconds and can be a float value.
- An empty object will be returned if there is no activity for the VQ.

You can use a single service for multiple VQs by omitting the `_urs_stat_url_parameters` option from the service and including the value for that option (for example, name of virtual queue, tenant ID, or statistical method) in the HTTP request as follows:

```
http://<gmshost>:<gmsport>/genesys/1/service/<name-of-urs-stat-service>
?name=<one-of-the-callback-VQs>&tenant=<tenant-name>&aqt=urs
```

The URS stat service will append the content of the `_urs_stat_url_parameters` option and the HTTP request parameters to the URS query. To view additional URS `lvq` input parameters and output information, open a browser with URS running and run the help method for `lvq` as follows:

```
http://<urshost>:<ursport>/urs/help/call/lvq
```

The help method is described in the Universal Routing 8.1 Reference Manual, Appendix C, "Supported Methods."

If, for example, you set the following configuration for the `<name-of-urs-stat-service>` service:

```
_caching_policy=5
_service=urs-stat
_type=builtin
_urs_stat_url_parameters=scale=true&tenant=Environment&aqt=urs
_urs_url=http://<ursloadbalancer>:<ursport>/urs/call/max/lvq
```

You can can use this service for multiple VQs by specifying only the name of a virtual queue in the HTTP request as follows:

```
http://<gmshost>:<gmsport>/genesys/1/service/<name-of-urs-stat-service>
?name=<one-of-the-callback-VQs>
```

Calculate Estimated Wait Time using AHT or Time in Queue

When Callback queries Estimated Wait Time, you can select one of the following methods:

- urs - EWT calculated based on average time in queue. To use this method, specify `aqt=urs` in the strings used to retrieve statistics, which can be specified either in options or URLs, as detailed in the previous sections.
- urs2 - EWT calculated based on average agent handle time. To use this method, specify `aqt=urs2` in the strings used to retrieve statistics, which can be specified either in options or URLs, as detailed in the previous sections.

urs method

If you specify `aqt=urs`, URS calculates the Estimated Wait Time based on the recent history of the calls distributed from the Virtual Queue. For example, for a given VQ, if a call was distributed in 60 seconds, another call in 120 seconds, and the third one in 60 seconds, then URS calculates that the average time per call is 80 seconds: (60+120+60)/3. So, if the Virtual Queue already has 6 calls waiting, then the Estimated Wait Time for a new call is 560 seconds: 80*(6+1).

urs2 method

If you specify `aqt=urs2`, URS calculates the Estimated Wait Time based on the average time spent by agents to handle calls from this Virtual Queue.

- URS calculates the average time per call for each agent. URS tracks the periods when the agent becomes ready then busy, and using this data, it gets the average time that the agent spends per call.
- URS also needs to know which agents are answering calls from the given Virtual Queue, so the precision of `aqt=urs2` depends on how precisely URS can get this list of agents.
  - If URS already has pending calls in the Virtual Queue, then it looks for which agents the calls are waiting for and uses their average time per call.
  - If the Virtual Queue is empty and has no waiting call, URS considers the agents from the agent group (or skill expression) used as a target for this Virtual Queue in the past.

When URS calculates Estimated Wait Time based on agent handle time, the result shows that agents working together handle calls faster than each agent does separately. Let's say that agent1 has an average of 60 seconds per call, agent2 of 120 seconds per call, agent3 of 60 seconds per call. URS considers that agent1 handles 1/60 part of a call per second, agent2 handles 1/120 part of a call per second, and agent3 handles 1/60 part of a call per second. So, working together, they handle 1/60+1/120+1/60 = 5/120 = 1/24 part of the call per second, which means an average of 1 call in 24 seconds.

As a result, if there are 6 calls in the queue, the Estimated Wait Time for a new call will be 168 seconds: 24*(6+1).

Note that URS also takes into count whether agents are shared between several Virtual Queues.

For example, if URS notes that agent1 is handling calls from 3 Virtual Queues, then instead of counting an average of 60 seconds for this agent, it will count 180 seconds because the agent is supposed to spend 1/3 of their time working on calls from each queue. As a result, URS will calculate that the average time per call is now 33 seconds for all of the 3 agents handling this queue.

In this scenario, if there are 6 calls in the queue, the Estimated Wait Time for a new call will be 231 seconds: 33*(6+1).